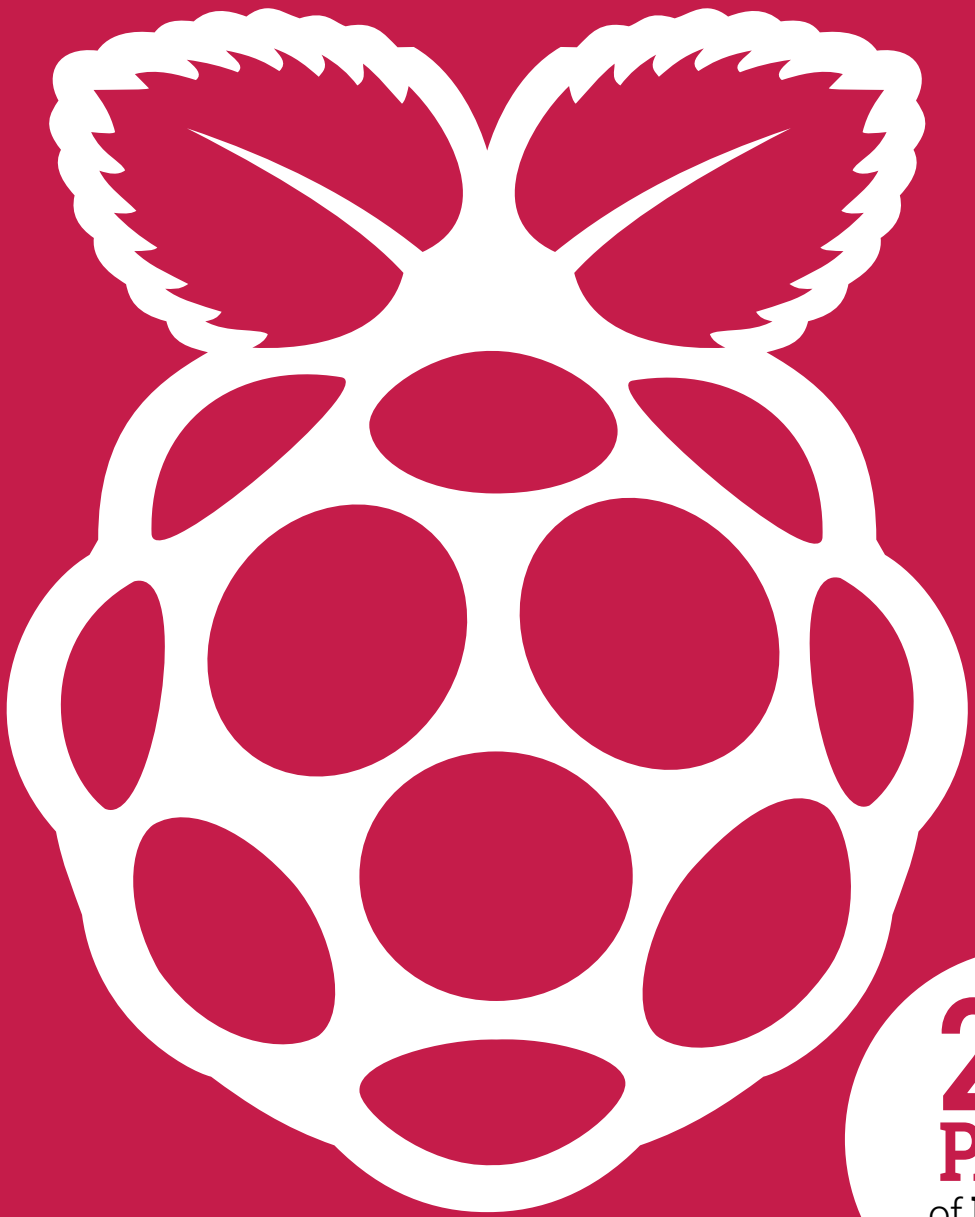
THE *Official*

RASPBERRY PI PROJECTS BOOK

VOLUME 3



200 PAGES of **hacking** & **making**

FROM THE MAKERS OF *The MagPi* THE OFFICIAL RASPBERRY PI MAGAZINE

SUBSCRIBE TODAY FROM ONLY £5

SAVE UP TO 35%

Subscriber Benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 35% compared to stores

Rolling Monthly Subscription

- **Low monthly cost** (from £5)
- **Cancel at any time**
- **Free delivery to your door**
- **Available worldwide**

Subscribe for 12 Months

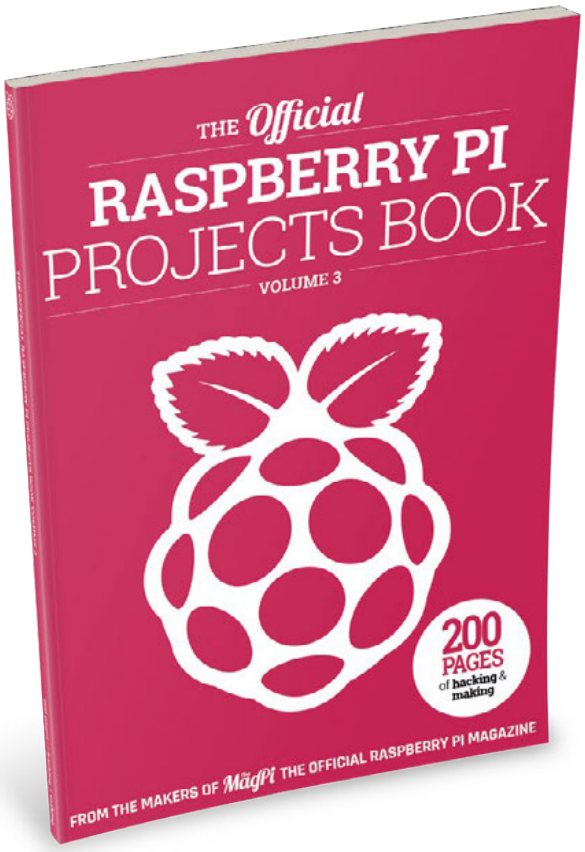
£55 (UK)	£90 (USA)
£80 (EU)	£90 (Rest of World)

Free Raspberry Pi Zero W Kit with 12 Month upfront subscription only (no Raspberry Pi Zero W Kit with Rolling Monthly Subscription)

Subscribe by phone: **01293 312193**

Subscribe online: **magpi.cc/subscribe**

Email: **magpi@subscriptionhelpline.co.uk**



HELLO!

Since the last Official Projects book came out, the Raspberry Pi has become the third best-selling computer of all time, behind only PCs and Macs. It's safe to say the Raspberry Pi is becoming a household name and every day we greet new eager coders and makers into the community.

The aim of these project books is to showcase some of the very best things you can do with the Raspberry Pi. To that end the book's 200 pages are packed with amazing creations from around the Pi world along with guides and step-by-step tutorials to make your own projects. Whether you're new to Pi and want to learn coding basics or a grizzled maker veteran wanting to get some new ideas, there's something for everyone.

I look forward to seeing what you're inspired to do after reading this book!

Rob Zwetsloot

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
Publishing Director: **Russell Barnes**
Production Editor: **Rob Zwetsloot**
Sub Editors: **Phil King, Rachel Churcher, and Jem Roberts**

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

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

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

PUBLISHING
For advertising & licensing:
russell@raspberrypi.org
Comms Director: **Liz Upton**
CEO: **Eben Upton**

CONTRIBUTORS
Wesley Archer, Ioana Culic, Lucy Hattersley, Richard Hayler & Sons, Phil King, Alan McCullagh, K.G. Orphanides, Dave Prochnow, Lucy Rogers, Richard Smedley, Francesco Vannini, Clive Webster, & many more Pi-loving people!

recycle
When you have finished with this magazine, please recycle it.

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

 ISBN: 978-1-912047-70-3.

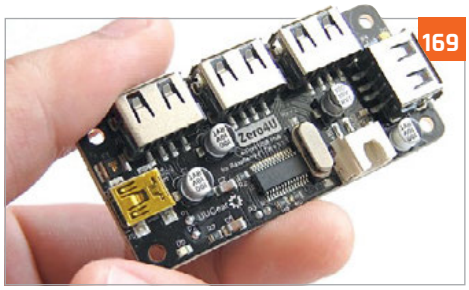
Contents

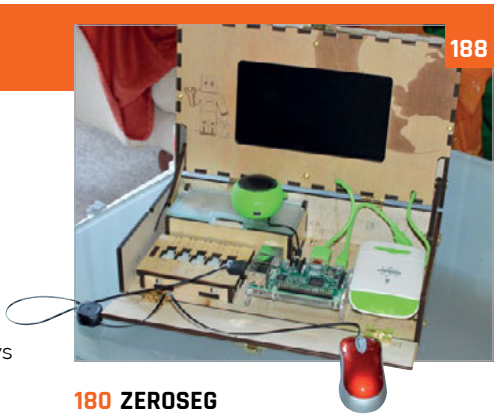
BEGINNERS GUIDE TO CODING
Learn the basics of coding and start your journey into programming
PAGE 06

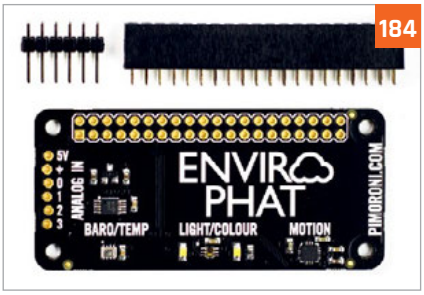
Projects

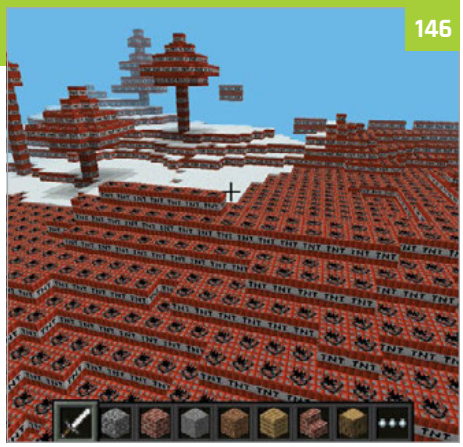
Tutorials

Reviews









Beginner's Guide to CODING

Discover the joy and art of computer programming with your Raspberry Pi

Learning to code is one of the most profoundly life-changing things you can do. This has always been true, but learning to code is increasingly important in the modern world.

The reason the Raspberry Pi was created was to challenge a drop in computer science applications at Cambridge University. Modern computers, and especially games consoles, were fun and powerful, but not easily programmable.

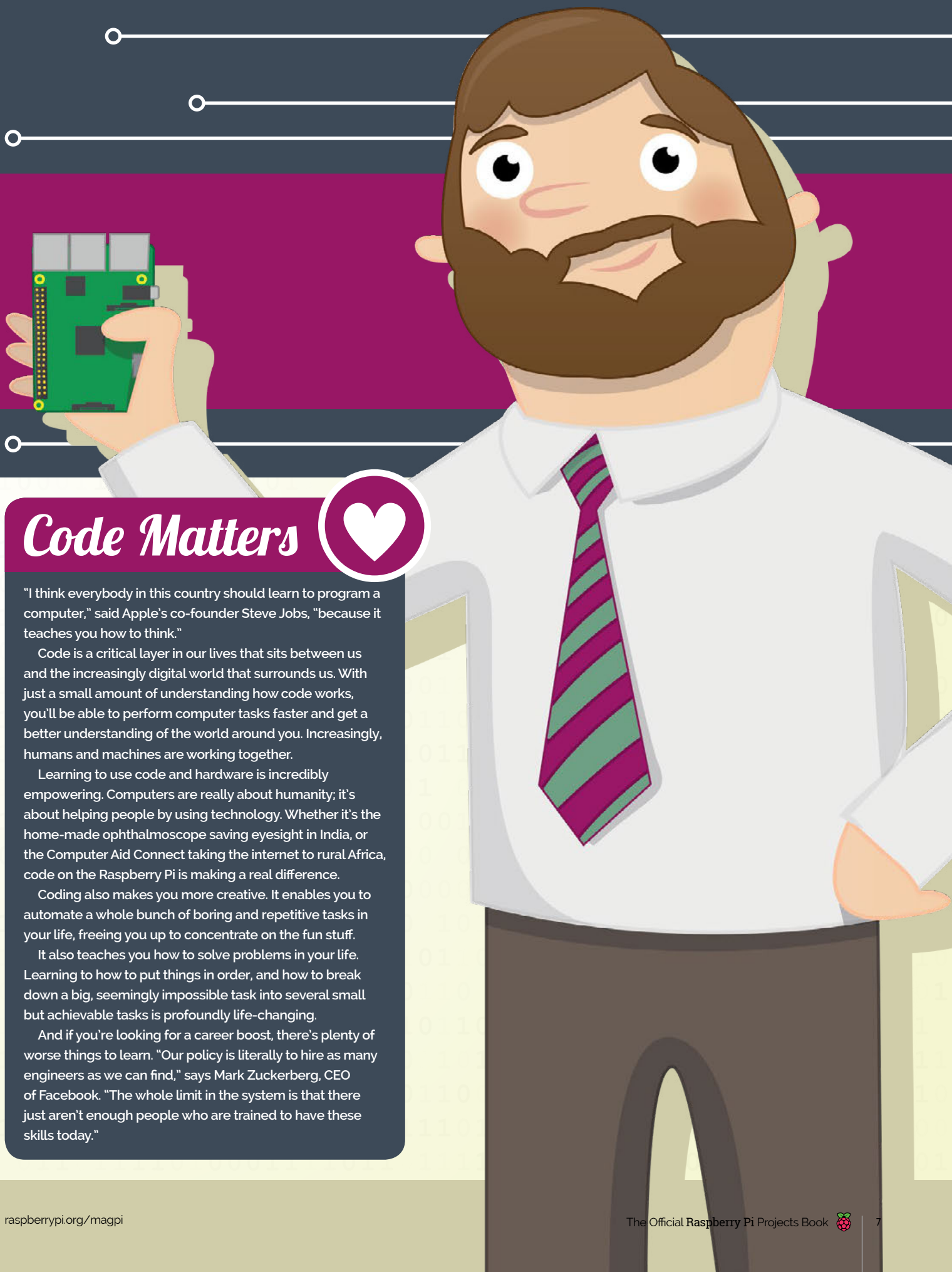
The maker community fell in love with the Raspberry Pi, thanks to its cheap and hackable nature. Building and tinkering are the primary reasons we love Raspberry Pi. Great projects use a combination of hardware and software together.

So, whether you're a hacker learning to make better projects, or a would-be coder looking for a better career, this feature is set to help you on your way.

The good news is that you don't need to be a genius to know coding, just as you don't have to be a genius to read and write. It's actually pretty simple once you learn a few simple concepts like variables, branching, and loops.

Perhaps you're brand-new to coding. Maybe you did a little BASIC in school, or used old languages like Pascal and Fortran. Or maybe you're already knee-deep in projects and just want to learn the language that controls them.

Wherever you're coming from, we're here to walk you through the basic concepts of computer programming. We'll demystify the whole process of code, so you can get a better understanding of what's going on inside your Raspberry Pi.



Code Matters

"I think everybody in this country should learn to program a computer," said Apple's co-founder Steve Jobs, "because it teaches you how to think."

Code is a critical layer in our lives that sits between us and the increasingly digital world that surrounds us. With just a small amount of understanding how code works, you'll be able to perform computer tasks faster and get a better understanding of the world around you. Increasingly, humans and machines are working together.

Learning to use code and hardware is incredibly empowering. Computers are really about humanity; it's about helping people by using technology. Whether it's the home-made ophthalmoscope saving eyesight in India, or the Computer Aid Connect taking the internet to rural Africa, code on the Raspberry Pi is making a real difference.

Coding also makes you more creative. It enables you to automate a whole bunch of boring and repetitive tasks in your life, freeing you up to concentrate on the fun stuff.

It also teaches you how to solve problems in your life. Learning to how to put things in order, and how to break down a big, seemingly impossible task into several small but achievable tasks is profoundly life-changing.

And if you're looking for a career boost, there's plenty of worse things to learn. "Our policy is literally to hire as many engineers as we can find," says Mark Zuckerberg, CEO of Facebook. "The whole limit in the system is that there just aren't enough people who are trained to have these skills today."

What is a Program?

Discover the building blocks of software and learn what goes on inside a program

Which Python?

Python 2 and Python 3 are both commonly used. Python 3 is the future, so we're going with it. Lots of courses still teach Python 2, and it's not a bad idea to take a closer look at the differences between the two: **magpi.cc/2gP6zX3**

Before you go any further, let's look at what a program actually is. The dictionary definition is a "set of instructions that makes a computer do a particular thing."

A computer program is a lot like a recipe. It has a list of ingredients, called 'variables', and a list of instructions, known as 'statements' or 'functions'. You follow the instructions from the recipe one line at a time and end up with a tasty cake.

The real miracle of computers, however, is that they can do the same thing repeatedly. So you can get a machine to bake a thousand cakes without ever getting tired. A program may contain loops that make it do the same thing over and over again.

Programs also make decisions, and different paths through a program can be taken. Your recipe could make a scrummy chocolate cake or a delightful batch of doughnuts, depending on the variables (the ingredients) it has.

Compiled vs Interpreted

Python is an 'interpreted language'. You write the code and then run the program. Under the hood, it's being translated and runs on the fly. Some languages, such as C and Java, are compiled. You write the program, then compile it to get a build file (written in machine code), then you run the build. It's a faff you can do without for now.

One thing that may surprise you when you begin programming is just how little you need to know to get started. With just a few variables, a smattering of flow, and some functions, you can get a computer doing all the hard work for you.

Inside your Pi

At the heart of your Raspberry Pi are billions of voltage switches known as binary digits (or 'bits' for short). There are 8,589,934,592 of them in its 1GB of RAM, to be exact. All these switches can be set to high or low, which is typically represented as 0 (for low or off) and 1 (for high or on). Everything you see on the screen, hear from the speakers, and type on the keyboard is billions of switches being turned on and off.

Obviously, it's not that easy for humans to talk directly to computers. It's possible to use machine language and send binary instructions directly to a computer, but this isn't where any sane person starts (or ends if they want to remain sane).

Instead, we use a coding language to program. This is written using easy-to-understand functions like `print()`. These are then interpreted into machine language, which the computer understands.

We're going to use Python to learn to code. Python is a truly great programming language. It has a rich syntax that's free from clutter; you don't have to worry about things like curly braces and static typing that crop up in more complicated languages like Java.

With Python, you can just create code, run it, and get things done. Python is one of the languages found most commonly inside *The MagPi*, so learning it here will help you understand lots of the code used in projects.

IDE and IDLE

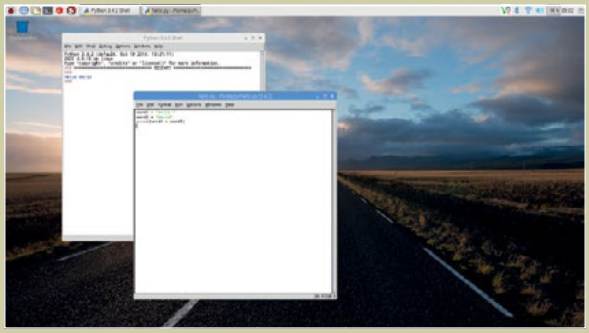
You don't have to write Python programs using a text editor like Leafpad and run them in the terminal. Instead, you can use a neat all-in-one solution, known as an 'IDE' (integrated development environment).

IDEs combine a text editor with program-running functionality. Often, they'll include fancy features like debugging and text completion.

Click **Menu > Programming > Python 3 (IDLE)**, and you'll get a new window called the Python Shell. This Shell works just like Python on the command line. Enter `print("Hello World")` to see the message.

You can also create programs in a built-in file editor. Choose **File > New File**. Enter this program in the window marked 'Untitled':

```
word1 = "Hello "
word2 = "World"
print(word1 + word2)
```



Above Python IDLE makes it easy to create programs and run them without having to use the command line

Don't forget to include the space after 'Hello'. Choose **File > Save As** and save it as **hello.py**. Now press **F5** on your keyboard to run the program. (Or choose **Run > Run Module**). It'll display 'Hello World' in the Shell.

The advantage of using Python IDLE is that you can inspect the program in the Shell. Enter `word1`, and you'll see 'Hello '. Enter `word2` and you'll see 'World'. This ability to inspect and use the variables in your program makes it a lot easier to experiment with programming and detect bugs (problems in your code).

Why Python?

There are a lot of programming languages out there, and they all offer something special. Python is a great option for beginners. Its syntax (the use of words and symbols) is easy to read. And it scales all the way up to industrial, medical, and scientific purposes, so it's ideal for beginners and experts alike.

Python in the terminal

You don't need to do very much to set up Python on your Raspberry Pi. Open a terminal in Raspbian and enter `python --version`. It will display the installed version of Python 2. Enter `python3 --version` to see your version of Python 3.

We're going to use Python 3 in this feature (see 'Which Python?' boxout). You can open Python 3 in the terminal by just typing `python3`.

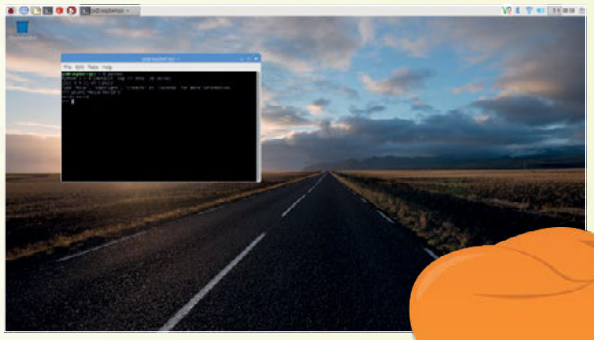
The '$' command-line prompt will be replaced with '>>>'. Here you can enter Python commands directly, just as you would terminal commands.

It's tradition to christen any new language by displaying 'Hello World'. Enter `print("Hello World")` and press **RETURN**. You'll see 'Hello World' outputted on the line below.

Using the Shell is known as Interactive Mode. You can interact directly with the code. It's handy for doing maths; enter `1920 * 1080` to get the answer: 2073600.

Mostly, you create Python programs using a regular text editor and save the files with a '.py' extension. Don't use a word processor like LibreOffice Writer, though – it'll add formatting and mess up the code.

Use a plain text editor like Leafpad (**Menu > Accessories > Text Editor**). Here you can enter your code, save it as a program, and then run the file in the terminal. Enter `python3 yourprogram.py` at the command line to run a program.



Left Python comes pre-installed in the Raspbian operating system and you can use it at the command line

Variables

Variables are all-purpose containers that you use to store data and objects

Python types

Python has five standard data types:

- Numbers
- String
- List
- Tuple
- Dictionary

Foo bar?

You'll come across 'foo' and 'bar' a lot when learning to code. These are dummy placeholders and don't mean anything. They could be zig and zag or bim and bam. Nobody's quite sure, but it might be related to the expression 'fubar' from the Vietnamese war.

If you've created a science project or experiment, you may have come across variables. In science, a variable is any factor that you can control, change or measure.

In computer programming, variables are used to store things in your program. They could be names, numbers, labels, and tags: all the stuff your program needs.

In Python, you write the name of a variable then a single equals sign and the word, number or object you want to put in it.

Enter this code directly into the Shell:

```
foo = 1
bar = 2
```

Remember: the variable name is on the left, and the thing it contains is on the right. Imagine you've got two plastic cups, and you've scrawled 'foo' on the first and 'bar' on the second. You put a number 1 in foo and a number 2 in bar.

If you ever want to get the number again, you just look in the cup. You do this in Python by just using the variable name:

```
foo
bar
```

You can also print out variables by passing them into a **print** function:

```
print(foo)
print(bar)
```

Variables can also be used to contain 'strings'. These are groups of letters (and other characters) that form words, phrases or other text.

Creating a string variable in Python is pretty much the same as creating an integer, except you surround the text with single (**' '**) or double (**" "**) quotes.

Using double quotes makes it easier to include apostrophes, such as **print("Don't worry. Be Happy")**. This line would break after 'Don' if you used single quotes – **print('Don't worry, be happy')** – so use double quotes for now.

Why variables count

Variables make it much easier to change parts of your code. Say you've got an excellent coding job at Nursery Rhymes Inc and you've written a classic:

```
print("Polly put the kettle on")
print("Polly put the kettle on")
print("Polly put the kettle on")
print("We'll all have tea")
```

The head of marketing comes in and says "our data shows that Polly isn't trending with the millennial demographic." You say "Huh!" and he barks "Change Polly to Dolly."

You now have to go through and change the variable in all three lines. What a downer! But what if you'd written thousands of lines of code and they all needed to change? You'd be there all week.

With variables, you define the variable once and then use it in your code. Then it's ready for changing

at any time:

```
name = "Polly"

print(name + " put the kettle on")
print(name + " put the kettle on")
print(name + " put the kettle on")
print("We'll all have tea")
```

This code prints out the same classic nursery rhyme. But if you want to change the name of our character, you only have to change it in one place:

```
name = "Dolly"
```

…and the poem will update on every line.

What's your type?

When you create a variable in Python, it's automatically assigned a type based on what it is. You can check this using the **type()** function. In the shell interface, enter:

```
foo = "Ten"
bar = 10
```

Now use the **type()** function to check the type of each variable:

```
type(foo)
type(bar)
```

It will say **<class 'str'>** for **foo**, and **<class 'int'>** for **bar**. This concept is important, because different types work together in a variety of ways, and they don't always play nicely together.

For example, if you add together two strings they are combined:

```
name = "Harry"
job = "Wizard"
print("Yer a " + job + " " + name)
```

This prints the message "Yer a Wizard Harry". The strings are concatenated (that's a fancy programming term for 'joined up'). Numbers, though, work completely differently. Let's try a bit of maths:

```
number1 = 6
number2 = 9

print(number1 + number2)
```

Instead of concatenating 6 and 9 together to give 69, Python performs a bit of maths, and you get the answer '15'.

Type casting

So what happens when you want to add a string and an integer together?

```
name = "Ben"
number = 10
print(name + number)
```

You'll get an error message: 'TypeError: Can't convert 'int' object to str implicitly'. This error is because Python can't add together a string and an integer, because they work differently. Ah, but not so fast! You can multiply strings and integers:

```
print(name * number)
```

It'll print 'Ben' ten times: you'll get 'BenBenBenBenBenBenBenBenBenBen'.

If you want to print out 'Ben10', you'll need to convert the integer to a string. You do this using a **str()** function and putting the integer inside the brackets. Here we do that, and store the result in a new variable called **number_as_string**:

```
number_as_string = str(number)
print(name + number_as_string)
```

This code will print out the name 'Ben10'. This concept is known as 'type casting': converting a variable from one type to another.

You can also cast strings into integers using the **int()** function. This is particularly useful when you use **input()** to get a number from the user; the input is stored as a string. Let's create a program that asks for a number and exponent and raises the number to the power of the exponent (using the '**' symbol):

```
number = input("Enter a number: ")
exponent = input("Enter an exponent: ")
result = int(number) ** int(exponent)
```

Our first two variables, **number** and **exponent**, are strings, while our third, **result**, is an integer. We could just print out the result:

```
print(result)
```

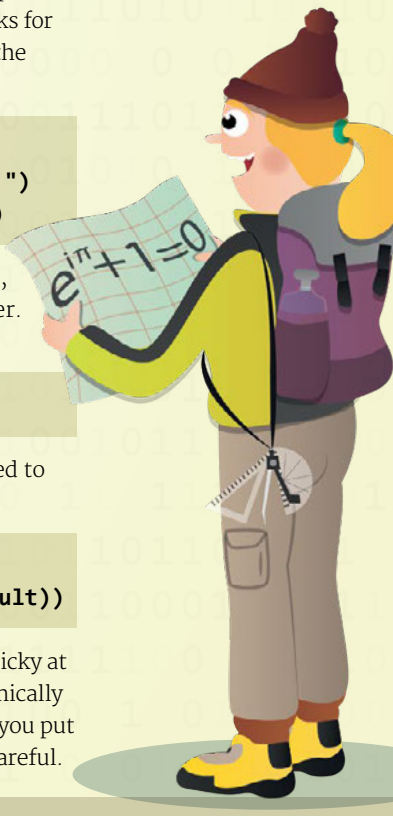
But if we wanted to include a message, we need to type cast **result** to a string:

```
print(number + " raised to the power
      " + exponent + " is " + str(result))
```

Variables, types, and type casting can be a bit tricky at first. Python is a lot easier to use because it dynamically changes the type of a variable to match the thing you put in it. However, it does mean you have to be a bit careful.

What to call a variable?

Variable names should be lower-case words separated by an underscore '_'. They can include numbers, but must start with a letter. You can call variables pretty much anything, but there's a small list of reserved keywords you should avoid (**magpi.cc/2h7MH1y**). It's a good idea to call them something that will be obvious when you use them in your program, like 'student_name' or 'person_age'.



Controlling flow with While & For

Get your program to do all the hard work with while and for loops

Comparison operators

These comparison operators are commonly used in conditions to determine if something is True or False:

==	equal
!=	not equal
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
<>	less than or greater than

Computers are great because they don't mind doing the same stuff over and over again. Their hard-working nature makes computers ideal for doing grunt work.

When looking at variables earlier, we printed out this nursery rhyme:

```
print("Polly put the kettle on")
print("Polly put the kettle on")
print("Polly put the kettle on")
print("We'll all have tea")
```

We didn't like the repetition of Polly, so we replaced it with a variable. But this code is foolish in another way: you have to write out the same **print** line three times.

We're going to use a loop to get rid of the repetition. The first loop we're going to look at is a 'while loop'. In Python 3 IDLE, create a new file and save it as **polly.py**; enter the code from the top of the next page.

We start with two variables:

```
name = "Polly"
counter = 0
```

Then we use the **while** statement followed by a condition: **counter < 3**.

On the next line down, you press the space bar four times to indent the code. Don't press the **TAB** key (see 'Tabs or spaces?' boxout).

```
while counter < 3:
    print(name + " put the kettle on")
    counter = counter + 1
```

The **<** symbol stands for 'less than'. It checks if the item on the left is less than the item on the right. In this case, it sees if the variable counter (which starts at 0) is less than 3. This condition is known as 'True'; if it wasn't, it'd be known as 'False'.

Finally, enter the last line of code:

```
print("We'll all have tea")
```

Save and run the program (press **F5**). It will print 'Polly put the kettle on' three times and then 'We'll all have tea'.

While, condition and indent

There are three things here: the while statement, the condition, and the indented text, organised like this:

```
while condition:
    indent
```

Imagine a three-way chat between all three items in our **polly.py** program:

Tabs or spaces?

There's a massive nerd debate about whether to use spaces or tabs when indenting code. There are valid arguments on both sides but use spaces for now. When you're a hardcore coder, you can make the argument for tabs.

While: "Hey Condition! What's your status?"
Condition: "True! The counter is 0. It's less than 3."
Indent: "OK, guys. I'll print out 'Polly put the kettle on' and increase the counter by 1. What's next?"

While: "Hey Condition. What's your status?"
Condition: "True! The counter is now 1."
Indent: "OK. I'm printing out another 'Polly put the kettle on' and increasing the counter by 1."

This goes on till the counter hits 3.

While: "Hey Condition. What's your status?"
Condition: "False! The counter is now 3, which isn't less than 3."
While: "OK guys. We're done!"

The program doesn't run the indented code, but moves to the single **print** at the end: 'We'll all have tea'.

For and lists

The next type of loop is known as 'for'. This is designed to work with lists.

Lists are a type of variable that contain multiple items (strings, numbers, or even other variables). Create a list by putting items inside square brackets:

```
banana_splits = ["Bingo", "Fleegle",
                 "Drooper", "Snorky"]
```

Now enter **banana_splits** in the Shell to view the list. It will display the four names inside the square brackets. You can access each item individually using the variable name and square brackets. Enter:

```
banana_splits[0]
```

...and you'll get 'Bingo'. Lists in Python are zero-indexed; that means the first item in the list is [0]. Here are each of the items. Type them into the Shell to get the names returned:

Polly.py

```
name = "Polly"
counter = 0

while counter < 3:
    print(name + " put the kettle on")
    counter = counter + 1

print("We'll all have tea")
```

```
banana_splits[0] # "Bingo"
banana_splits[1] # "Fleegle"
banana_splits[2] # "Drooper"
banana_splits[3] # "Snorky"
```

Zero-indexed lists can be confusing at first. Just remember that you're counting from 0. A for loop makes it easy to iterate over items in a list. Create this program and save it as **splits.py**:

```
banana_splits = ["Bingo", "Fleegle",
                 "Drooper", "Snorky"]

for banana_split in banana_splits:
    print(banana_split)
```

It doesn't matter what you use as the variable in a for loop, as long as you remember to use it in your indented code. You could put:

```
for dude in banana_splits:
    print(dude)
```

It's common to name the list as something plural (such as 'names', 'pages', and 'items') and use the singular version without the 's' for the 'in' variable: 'for name in names', 'for page in pages', and so on.

Infinite loops

You must be careful to change the counter in a while loop, or you'll get an infinite loop. If you delete the line **counter = counter + 1** from our while loop, it will run forever: it never goes above 0, so the indented code runs over and over again. This bug is known as an 'infinite loop' and is a bad thing to have in your programs.

Conditional Branching

Give your programs some brains with conditional branching

Logical operators

You can combine conditions together using logical operators.

and Both operands are true: (a and b) is True

or Any operator is true: (a or b) is True

not Checks if something is false: not (a and b) is True if both a and b are False.

Your programs have been slowly getting more powerful. We've learned to run instructions in procedural order, replaced parts of our program with variables, and looped over the code.

But another important part of programming is called 'conditional branching'. Branching is where a program decides whether to do something or not.

Of course, a program doesn't just decide whether or not to do things on a whim: we use the sturdy world of logic here.

The start of all this is the powerful 'if' statement. It looks similar to a loop, but runs just once. The if statement asks if a condition is True. If it is, then it runs the indented code:

```
if True:
    print("Hello World")
```

Run this program, and it'll display 'Hello World'. Now change the if statement to False:

```
if False:
    print("Hello World")
```

...and nothing will happen.

Of course, you can't just write True and False. Instead, you create a condition which evaluates to True or False; a common one is the equals sign (==). This checks whether both items on either side are the same. Create a new file and enter the code from **password1.py**. This code is a simple program that asks you to enter a password; if you enter the correct password, 'qwerty', it displays 'Welcome'.

Be careful not to confuse the equals logic operator == with the single equals sign =. While the double equals sign checks that both sides are the same, the single equals sign makes both sides the same. Getting == and = mixed up is a common mistake for rookie coders.

What else

After if, the next conditional branch control you need to learn is 'else'. This command is a companion to if and runs as an alternative version. When the if branch is True, it runs; when the if branch is False, the else branch runs.

```
if True:
   print("The first branch ran")
else:
   print("The second branch ran")
```

Run this program and you'll see 'The first branch ran'. But change True to False:

```
if False:
   print("The first branch ran")
else:
   print("The second branch ran")
```

...and you'll see 'The second branch ran'. Let's use this to expand our password program. Enter the code from **password2.py**.

Run the program again. If you get the password correct now, you'll get a welcome message. Otherwise, you'll get an 'incorrect password' message.

Elif

The third branching statement you need to know is 'elif'. This statement stands for 'else if', and sits between the if and else statements. Let's look at an elif statement. Enter this code:

```
if False:
    print("The first block of code ran")
elif True:
    print("The second block of code ran")
else:
    print("The third block of code ran")
```

Run this program and you'll find it skips the first if statement, but runs the elif statement. You'll get 'The second block of code ran'.

The else statement doesn't have a True or False condition; it runs so long as neither the if or elif statements are True. (Note that the else statement here, as always, is optional; you can just have if and elif.)

But what happens if you change both the if and elif conditions to True? Give it a try and see whether just if runs, or elif, or both. Experiment with removing the else statement and play around. It'll help you get the hang of the if, elif, and else statements.

FizzBuzz

We're going to show you a common program used in computer programming interviews. It's a classic called 'FizzBuzz', and it shows that you understand if, else, and elif statements.

First, you need to know about the modulo operator (**%**). This is used to get the remainder from a division and is similar to a divide operator. Take this function:

```
10 / 4 == 2.5
```

If we use a modulo instead, we get this:

```
10 % 4 == 2
```

Modulo turns out to be handy in lots of ways. You can use **% 2** to figure out if a number is odd or even:

```
10 % 2 == 0 # this is even
11 % 2 == 1 # this is odd
```

This program works out if a number is odd or even:

```
number = 10

if number % 2 == 0:
    print("The number is even")
else:
    print("The number is odd")
```

OK – let's move on to FizzBuzz.

Password.py

```
password = "qwerty"
attempt = input("Enter password: ")

if attempt == password:
    print("Welcome")
```

Password2.py

```
password = "qwerty"
attempt = input("Enter password: ")

if attempt == password:
    print("Welcome")
else:
    print("Incorrect password!")
```

Writing FizzBuzz

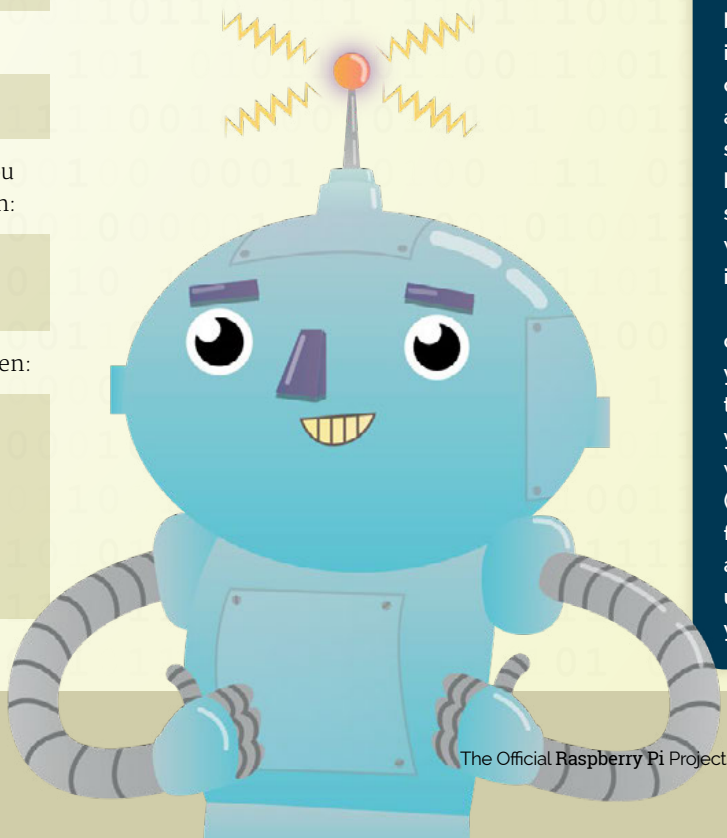
The brief for our FizzBuzz is to print the numbers up to 100. If a number is divisible by three (such as 3, 6, and 9), then you print 'Fizz' instead of the number; if the number is divisible by five, you print 'Buzz' instead.

But if a number is divisible by both 3 and 5, such as the number 15, then you print 'FizzBuzz'.

We're also introducing a new element in FizzBuzz: the 'and' statement. This checks if two conditions are both True: that the number can be divided by both 3 and 5. It only returns True if both conditions are True.

There are three main logical operators: and, or, and not. The first two are relatively straightforward, but the 'not' operator can be more confusing at first. Don't worry about it too much; you'll get the hang of it with practice.

Enter the **fizzbuzz.py** code from page 17 to practise using if, else, and elif elements and logical operators.



Comments

A mark of a good programmer is to use comments in your programs. Comments are used to explain bits of your program to humans. They are completely ignored by the computer.

In Python, you start a comment line with a hash symbol (#). It can be on a line on it own, or it can come right after a line of code. As soon as Python hits the #, it'll stop translating whatever follows into machine code.

Comments help other users to read your program, but they will also help you understand what you're doing (long after you've forgotten). It's a good habit to use comments in your programs.

Creating Functions

Create the building blocks of code and make more robust programs

You've come a long way since your first 'Hello World'. Your programs now check for conditions and loop over themselves.

You're now writing programs that are known as 'Turing complete', named after Alan Turing, the father of computer science and artificial intelligence, who hacked the German Enigma code in WWII.

Now we're going to take things a little further. We're going to introduce you to a form of modularity called functions.

Functions are blocks of code that you write once and can repeat anywhere. It's a little like being able to write a block of text once, and then paste it whenever you need it.

Spotting a function

Python is packed with built-in functions, and you've already been using them in your programs. Commands like **`print()`**, **`len()`**, and **`type()`** are all functions. They're easy to spot: a small command starting with a lower-case letter and followed by a pair of parentheses '()'.

Python documentation

You can browse or download a copy of the Python documentation directly from the Python website at **python.org/doc**. Python has a whole bunch of built-in functions. You can view a list of all the built-in functions on the Python documentation website (**magpi.cc/2gPsGK3**).

Using functions

Let's take a look at a function called **`abs()`**. It stands for 'absolute', and returns the absolute value of any number you pass into it (the bit you pass in is called the 'argument').

An absolute number is the positive of any number, so if you write **`abs(-2)`** you get 2 back. Try this in the Shell:

```
abs(2) # returns 2
abs(-2) # returns 2
```

You can store the returned result as a variable:

```
positive_number = abs(-10)
```

We find it easier to read a function backwards, from right to left. The value is passed into the parentheses, then the function cranks it and returns a new value. This is passed left and stored into the variable.

Defining a function

The great thing about Python is that you don't just use the built-in functions: you get to make your own. These are called 'user-defined functions'.

You create a function using the **`def`** keyword, followed by the function name and parentheses. Inside the parentheses, you list the parameters. These are the same as the arguments, only inside the definition they are called 'parameters'.

```
def function(parameter):
    return parameter
```

Our function here doesn't do anything; it simply accepts a parameter and returns it.

At the end of the function definition is a colon (**:**). The function code is indented by four spaces, just like a loop or if/else branch.

The code inside the indentation runs when you call the function. Functions typically include a **return** statement which passes back an expression.

Working functions

We're going to create a function that prints the lyrics to Happy Birthday.

Type out the **happy_birthday.py** code from the listing, then run it. In the Shell, enter:

```
happy_birthday("Lucy")
```

This function call uses the string 'Lucy' as the argument. This string is passed into the function as the parameter and is then available for use in the indented code inside the function.

Return statements

Many functions don't just run a block of code; they also return something to the function call.

We saw this in **abs()**, which returned the absolute value of a number. This can be stored in a variable.

In fact, we're going to recreate the **abs()** function, so you can see how it's working behind the scenes.

In maths, you invert a positive/negative value by multiplying a negative number by -1, like this:

```
10 * -1 = -10
-10 * -1 = 10
```

We need to create a function that takes a number as a parameter and checks if it's negative. If so, it multiplies it by -1; if it's positive, it simply returns the number. We're going to call our function **absolute()**.

Enter the code in **absolute.py**. When the function hits either of the **return** statements, it returns the value of the number (either on its own or multiplied by -1). It then exits the function.

Run the **absolute.py** code and enter the following in the Shell:

```
absolute(10)
absolute(-10)
```

Our last program listing is a classic known as 'FizzBuzz'; as mentioned on page 23, it will help you to understand if, else, and elif.

You also need to know the modulo operator (**%**) for FizzBuzz. This operator returns the remainder from a division. If you don't know how modulo works, watch this video (**magpi.cc/2h5XNRO**).

Now work through the code in **fizzbuzz.py**.

Happy_birthday.py

```
def happy_birthday(name):
    count = 0
    while count < 4:
        if count != 2:
            print("Happy birthday to you")
        else:
            print("Happy birthday dear " + name)
        count += 1
```

Absolute.py

```
def absolute(number):
    if number < 0:
        return number * -1
    else:
        return number
```

Fizzbuzz.py

```
count = 0
end = 101

while count < end:
    if count % 5 == 0 and count % 3 == 0:
        print("FizzBuzz")
    elif count % 3 == 0:
        print("Fizz")
    elif count % 5 == 0:
        print("Buzz")
    else:
        print(count)

    count += 1
```

Going further

Here are some resources you will find useful.

GPIO Zero Essentials – magpi.cc/2bA3ZP7
This Essentials guide book explains how the GPIO Zero Python module provides access to a bunch of features. These are used to hook up electronics to your Raspberry Pi via the GPIO pins.

FutureLearn – magpi.cc/2h5Sthf
The Raspberry Pi Foundation has two new online training courses: Teaching Physical Computing with Raspberry Pi and Python, and Teaching Programming in Primary Schools.

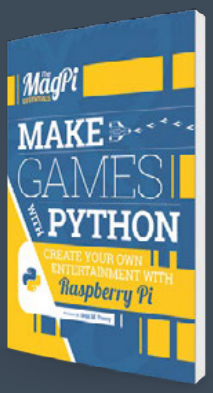
Learning Python – magpi.cc/2h2opWC
This tutorial provided by The Raspberry Pi Foundation has files you can download. You download the file, called **intro.py**, using this command in a Terminal: `wget http://goo.gl/0ZDOdX -O intro.py --no-check-certificate`. Open the **intro.py** file in IDLE; all the instructions are in the file.

Importing Code

Stand on the shoulders of giants by importing other programmers' code

Pygame

If you want to learn more about Pygame, check out *Make Games With Python*, our free Essentials Guide to the Pygame module. **magpi.cc/2h2m0vh**



This being the modern world, you're not supposed to do all the work on your own. Instead, you will often stand on the shoulders of other programmers who have done the groundwork for you.

Your programs can import code created by other people using the **import** statement. This enables you to import modules and use their functions – only they're now known as 'methods'.

You import the module at the command line, and then access the functions using dot notation. This is where you list the module, followed by a dot (**.**), then the method.

A common module to use is **math**. This allows you to access lots of maths methods. Open a Python Shell and enter:

```
import math
```

You now have access to all the methods in **math**. You won't notice any difference, but if you type:

```
type(math)
```

...it will say '<class 'module'>'. Let's try out dot notation now. Type **math** followed by a dot and the name of the method (function) you want to use:

```
math.sqrt(16)
```

This gives the square root of 16, which is 4.

Some methods have more than one argument. The **math.pow()** method raises a number to an exponent:

```
math.pow(64,3)
```

This returns 262144.0.

You can also access constant values from a module, which are fixed variables contained in the module. These are like functions/methods, but without the parentheses.

```
math.pi
```

This returns pi to 15 decimal spaces: 3.141592653589793.

```
math.e
```

This returns Euler's number to 15 decimal spaces: 2.718281828459045.

It's also possible to import methods and constants from modules using **from**. This enables you to use them inside your programs without dot notation (like regular functions). For example:

```
from math import pi
from math import e
from math import pow
```

Now, whenever you type **pi** or **e**, you'll get pi and Euler's number. You can also use **pow()** just like a regular function. You can change the name of the function as you import it with **as**:

```
from math import pi as p
```

Now when you enter **p** you'll get pi to 15 decimal spaces. Don't go crazy renaming functions with **as**, but it's common to see some methods and constants imported as single letters.

By creating your own functions, and importing those created by other people in modules, you can vastly improve the capabilities of your programs.

We're going to take everything we've learnt and use it to create a game of Pong; this is one of the world's first videogames.

Write out the code carefully in **pong.py**. Here you'll find variables, functions, loops, and conditional branching: all the stuff we've talked about. Hopefully, you'll now be able to decipher most of this code.

If you're interested in taking Pong further, this program is similar to a version of a Pygame program by Trever Appleton (**magpi.cc/2hgkOUX**). His version has a scorecard and more advanced code. We've kept ours simple so it's easier to start with.

Hopefully this isn't the end of your Python, or programming, journey. There are lots of places you can learn programming from. And we'll have more programming resources for you in every issue of *The MagPi*.

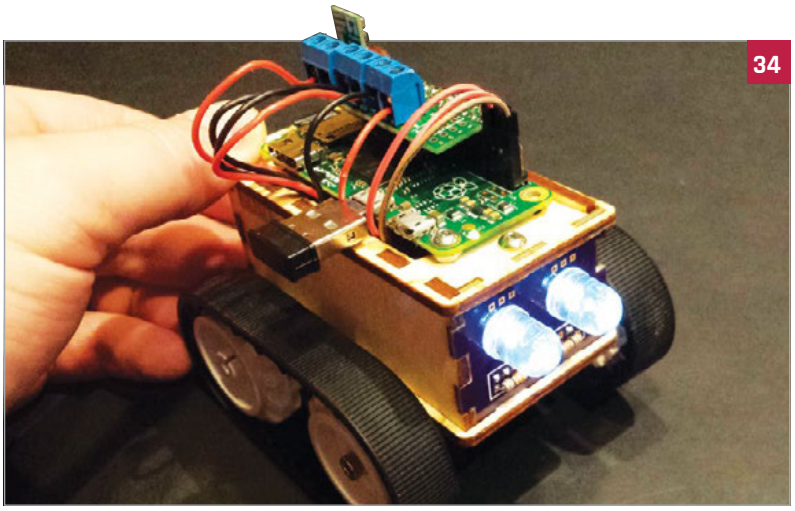
Pong.py

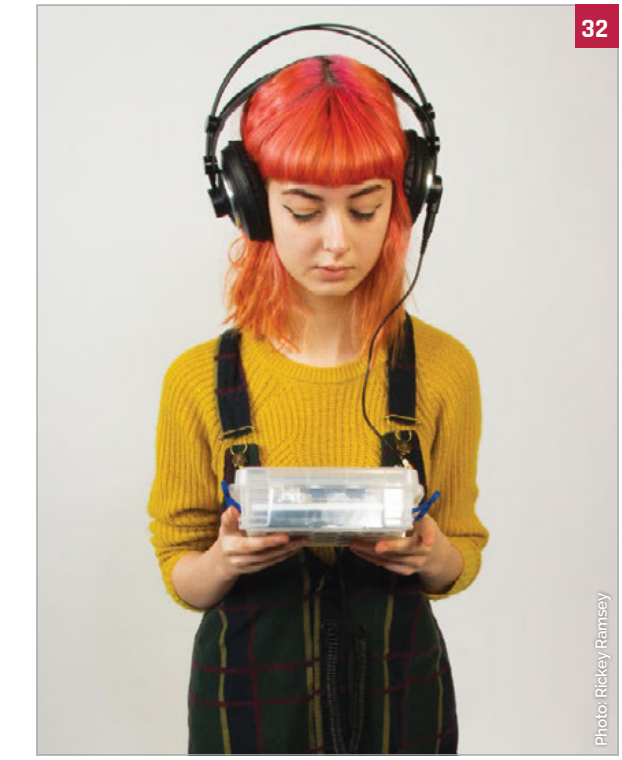
```
import pygame, sys
from pygame.locals import *

# Set up game variables
window_width = 400
window_height = 300
line_thickness = 10
paddle_size = 50 # try making this smaller for a harder game
paddle_offset = 20

# Set up colour variables
black = (0  ,0  ,0  ) # variables inside brackets are 'tuples'
white = (255,255,255) # tuples are like lists but the values don't change

# Ball variables (x, y Cartesian coordinates)
# Start position middle of horizontal and vertical arena
ballX = window_width/2 - line_thickness/2
ballY = window_height/2 - line_thickness/2

# Variables to track ball direction
ballDirX = -1 ## -1 = left 1 = right
ballDirY = -1 ## -1 = up 1 = down

# Starting position in middle of game arena
playerOnePosition = (window_height - paddle_size) /2
playerTwoPosition = (window_height - paddle_size) /2

# Create rectangles for ball and paddles
paddle1 = pygame.Rect(paddle_offset,playerOnePosition, line_thickness,paddle_size)
paddle2 = pygame.Rect(window_width - paddle_offset - line_thickness, playerTwoPosition, line_thickness,paddle_size)
ball = pygame.Rect(ballX, ballY, line_thickness, line_thickness)

# Function to draw the arena
def drawArena():
    screen.fill((0,0,0))
    # Draw outline of arena
    pygame.draw.rect(screen, white, ((0,0),(window_width,window_height)), line_thickness*2)
    # Draw centre line
    pygame.draw.line(screen, white, (int(window_width/2)),0),((int(window_width/2)),window_height), (int(line_thickness/4)))

# Function to draw the paddles
def drawPaddle(paddle):
    # Stop the paddle moving too low
    if paddle.bottom > window_height - line_thickness:
        paddle.bottom = window_height- line_thickness
    # Stop the paddle moving too high
    elif paddle.top < line_thickness:
        paddle.top = line_thickness
    # Draws paddle
    pygame.draw.rect(screen, white, paddle)

# Function to draw the ball
def drawBall(ball):
    pygame.draw.rect(screen, white, ball)

# Function to move the ball
def moveBall(ball, ballDirX, ballDirY):
    ball.x += ballDirX
    ball.y += ballDirY
    return ball # returns new position

# Function checks for collision with wall and changes ball direction
def checkEdgeCollision(ball, ballDirX, ballDirY):
    if ball.top == (line_thickness) or ball.bottom == (window_height - line_thickness):
        ballDirY = ballDirY * -1
    if ball.left == (line_thickness) or ball.right == (window_width - line_thickness):
        ballDirX = ballDirX * -1
    return ballDirX, ballDirY # return new direction

# Function checks if ball has hit paddle
def checkHitBall(ball, paddle1, paddle2, ballDirX):
    if ballDirX == -1 and paddle1.right == ball.left and paddle1.top < ball.top and paddle1.bottom > ball.bottom:
        return -1 # return new direction (right)
    elif ballDirX == 1 and paddle2.left == ball.right and paddle2.top < ball.top and paddle2.bottom > ball.bottom:
        return -1 # return new direction (right)
    else:
        return 1 # return new direction (left)

# Function for AI of computer player
def artificialIntelligence(ball, ballDirX, paddle2):
    # Ball is moving away from paddle, move bat to centre
    if ballDirX == -1:
        if paddle2.centery < (window_height/2):
            paddle2.y += 1
        elif paddle2.centery > (window_height/2):
            paddle2.y -= 1
    # Ball moving towards bat, track its movement
    elif ballDirX == 1:
        if paddle2.centery < ball.centery:
            paddle2.y += 1
        else:
            paddle2.y -=1
    return paddle2

# Initialise the window
screen = pygame.display.set_mode((window_width,window_height))
pygame.display.set_caption('Pong') # Displays in the window

# Draw the arena and paddles
drawArena()
drawPaddle(paddle1)
drawPaddle(paddle2)
drawBall(ball)

# Make cursor invisible
pygame.mouse.set_visible(0)

# Main game runs in this loop
while True: # infinite loop. Press Ctrl-C to quit game
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        # Mouse movement
        elif event.type == MOUSEMOTION:
            mousex, mousey = event.pos
            paddle1.y = mousey

    drawArena()
    drawPaddle(paddle1)
    drawPaddle(paddle2)
    drawBall(ball)

    ball = moveBall(ball, ballDirX, ballDirY)
    ballDirX, ballDirY = checkEdgeCollision(ball, ballDirX, ballDirY)
    ballDirX = ballDirX * checkHitBall(ball, paddle1, paddle2, ballDirX)
    paddle2 = artificialIntelligence (ball, ballDirX, paddle2)
    pygame.display.update()

```

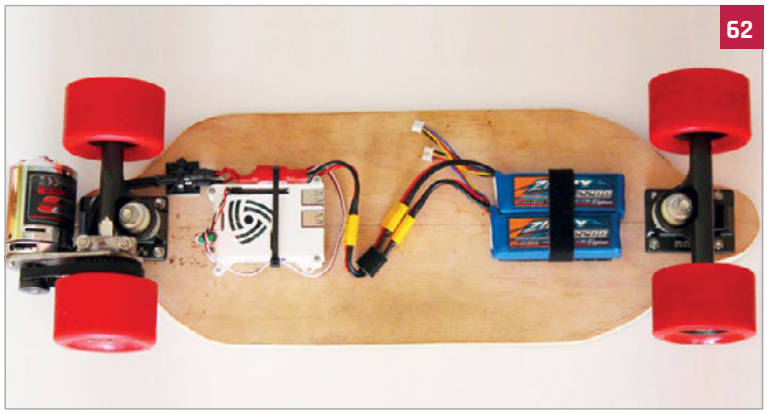
You should try writing this code out yourself, however you can always check it against the code file here: **magpi.cc/2zjxbEC**

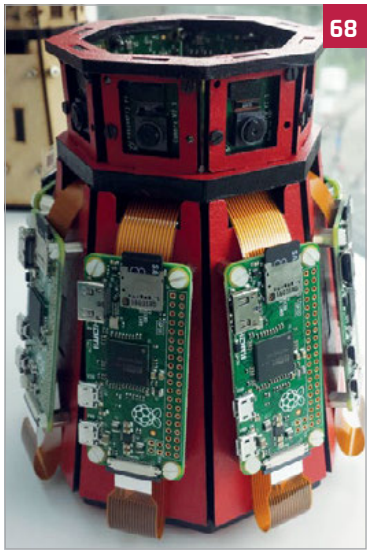
PROJECTS SHOWCASE

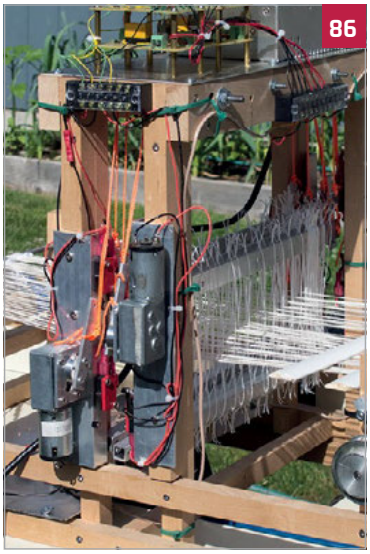
Here are just some of the amazing projects that the Raspberry Pi community make every day. Hopefully they'll give you some amazing ideas of your own











Projects

JAMES BOND SPY PROJECTS

Come in, Bond, and please be careful. We have the very latest in spy technology ready for your next mission

Right, now pay attention 007: we have gathered some special projects, all made using the best of British microcomputer technology – our beloved Raspberry Pi.

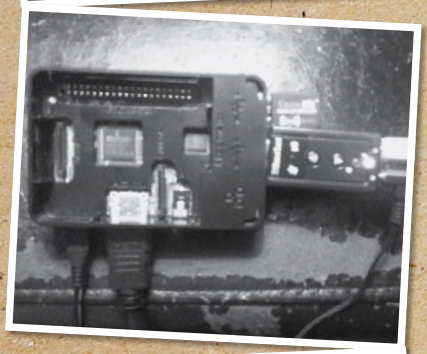
Our agents in the field have been working with the Raspberry Pi components, and have been creating experimental spy projects that can foil even the craftiest double agents.

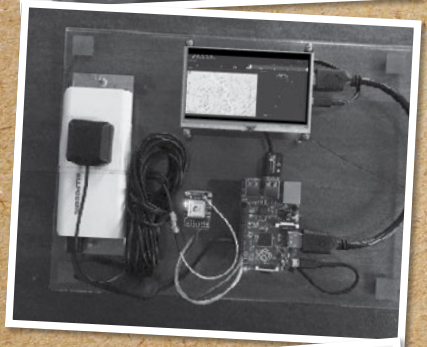
If you want to eavesdrop on a conversation, well, we've got just the thing: a light bulb that can surreptitiously transcribe spoken dialogue. On the other hand, if you're worried about your voice being detected, our voice distortion box will help you out.

Or if you prefer the visual approach, why not build a classic spy camera? Our pinhole cameras can be hidden in any location, or you can wear our bow tie spy camera. No, I never joke about my work, 007.

If you want to find out where somebody is heading, make sure you set up our GPS tracker. We even have motion detection cameras for long-range spying, and a Raspberry Pi-powered Geiger counter for those trickier missions. Self-respecting secret agents should keep their equipment out of sight, so we've got a computer that hides inside your lunch box.

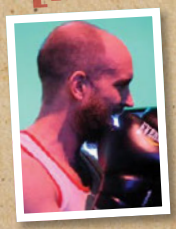
Need I remind you, 007, that you have a licence to kill, not to break the law? Do not use this equipment for nefarious purposes. This is a training mission only, so please remember to spy on operatives who are part of your assignment and not random members of the public. Good luck out there in the field, Mr Bond, and do please try to return the equipment in one piece this time.





[TOP SECRET]

[MAKER PROFILE]



BRIAN HOUSE
& KYLE MCDONALD

Brian House is a fellow at the Tow Center for Digital Journalism at Columbia University; Kyle McDonald is an artist who works in the open with code, and an adjunct professor at NYU's ITP (Interactive Telecommunications Program).
magpi.cc/1QRSep8

CONVERSNITCH

This Raspberry Pi device plugs into a light socket and listens to conversations. These are then transcribed and shared on Twitter

You'll Need

- Raspberry Pi
- Microphone
- WiFi dongle
- Light socket adapter

Conversnitch is one of the coolest spy devices we've seen created using a Raspberry Pi. Created by two Brooklyn-based artists, it impersonates a light bulb and records nearby conversations. These audio recordings are then sent to Amazon's Mechanical Turk program and are cheaply transcribed by hand.

Kyle McDonald and Brian House created the device to raise questions about the nature of public spaces, but it's certainly one of the most impressive spy devices we've seen.

Conversnitch costs less than $100 to build. The parts are simple: it uses a Raspberry Pi board, microphone, and a WiFi dongle. An Edison screw light bulb fitting is hacked to provide power to the Raspberry Pi, and the whole unit is housed in a plastic circular case.

Bryan House explains: "The device continually records ten-second snippets of audio, analyses them for potential voice content, and sends promising files to Mechanical Turk for transcription. The system then posts these transcriptions to Twitter."

A video showing the build process is available on Vimeo (**vimeo.com/87564506**) and all of the source code is on GitHub (**magpi.cc/1QRRSin**).

Conversnitch is a frighteningly easy project to put together. Perhaps the hardest part would be hacking a light bulb connection to provide power to a Raspberry Pi, although devices like the GE Socket Adapter or Leviton 2-Outlet Socket make life much easier.

Getting people to be more aware about the laws that are in place to protect us is the aim, reveals Kyle. "Some artists, like Brian and myself, see it as their responsibility to make work that's about culture right now. Security and surveillance are one of the main topics we're dealing with.

"I hope that what happens is that people become scared and frustrated, and more aware of the laws around privacy and wiretapping, just by having something to look at.

"Conversnitch automatically tweets overheard conversations, bridging the gap between (presumed)

VOICE CHANGER

Your voice is a dead giveaway, so why not add to your disguise with a voice changer? We love Dave Shevett's Raspberry Pi Realtime Voice Changer (**magpi.cc/1pvwGlD**). Dave built it as part of his Technomancer costume for Halloween. "One thing I've always wanted was a voice changer that would let me have a nice sepulchral voice to go with the creepy visage," says Dave. The project combines a Raspberry Pi with a USB sound adapter and Pyle Audio portable PA (**pyleaudio.com**). A script uses ALSA (**alsa-project.org**) to record and play back audio in real-time. There is a slight delay of around a tenth of a second, but some hacks can reduce the delay.

private physical space and public space online," Kyle explains.

Once assembled, the small light bulb plugs into any standard Edison screw fixture, and will connect to a local WiFi network – in demonstrations, it was set up in coffee shops. The audio from the microphone is streamed to Amazon's Mechanical Turk (**mturk.com**) service, where it is transcribed by anonymous workers. Mechanical Turk specialises in HIT (Human Intelligence Tasks); these are low-cost, short tasks that are performed by humans around the world. In this case, they quickly translate the recorded snippets of audio to text.

Once the hardware is assembled, the Raspberry Pi continuously records ten-second samples and analyses them for interesting audio. If it finds some, it uploads the audio to an Amazon S3 bucket and reports to the server. The server then creates an MTurk HIT task with a link to the audio. A cron process then checks the HITs for completion and, if it finds one, posts the result to Twitter.

According to the project's README.md document, "Conversnitch complicates the divisions between the physical and virtual, illegal and playful, human and machine, spoken and textual, exposure and illumination."

SPY CAMERA

There are lots of projects around for creating surreptitious spy cameras, and it really couldn't be easier. The Raspberry Pi Camera Module is small enough to go undetected in many situations, and with a bit of concealment can be hidden in all kinds of ways. We like Raymond Wong's Spy Bow Tie (**dai.ly/x2pjw8u**) and Tetranitrate's Spy Shirt (**magpi.cc/1QRSh4o**). If you want something truly inauspicious, then Adafruit's Spy Camera (**magpi.cc/1pvuXwM**) is much smaller than the regular Camera Module. It can be hidden inside clothing, so long as there's a pinhole for the camera to see out. They'll never spot you recording.

You'll Need

- Custom lunch box
- 10,000 mAh USB battery
- Raspberry Pi 2 with micro SD card
- Clear Raspberry Pi case
- 5-inch LCD
- Bluetooth keyboard
- 6,800 mAh 2v rechargeable battery
- USB cables

LUNCH BOX COMPUTER

Hide your Raspberry Pi where nobody will think to look… inside your lunch box

No secret agent outfit is complete without a briefcase, and no spy's briefcase is complete without a stash of secrets.

There are lots of Raspberry Pi computer projects around, but we think you'll admire this Lunch Box Computer by the cryptically named hacker, D10D3. It has all the components you need to run Raspberry Pi code on the move, and hides your Pi safely inside an inconspicuous box. It's ideal for places where computers are not allowed, and impromptu hacking projects.

[MAKER PROFILE]

D10D3

A maker, a hardware and software hacker, an artist, and general dreamer. He has an insatiable need to build things and modify them. He's a lover of science fiction, fantasy, cyberpunk, comic books, computers, and role-playing games. In short, he's a geek and a jack of all trades.

magpi.cc/1QRSkwR

"It's a good first Raspberry Pi project," says D10D3, "since it requires no coding, soldering, or tooling of any kind. All you have to do is acquire the parts, plug everything in, and secure it in the lunch box.

"The Raspberry Pi isn't a very fast machine, but it's extremely versatile and easy to use. This rig will have all the functionality of a WiFi netbook, albeit a slow one, with a subtle screen."

D10D3 uses a Raspberry Pi Model B in his tutorial, but we'd advise using the newer Raspberry Pi 2 (or Pi 3) with its faster processor. Two chargers are required: an RCA for the screen, and the other for the Raspberry Pi. We'd be tempted to swap out the 5-inch display used here for a Raspberry Pi Touch Display, which connects directly to the Raspberry Pi's DSI port using a ribbon cable.

Start by charging up the batteries and installing Raspbian Jessie on the SD card. Then plug the WiFi dongle into the Pi (if not using a Pi 3) and attach the Bluetooth dongle. Connect the Raspberry Pi to the display, and attach power to the Pi and display. "Make sure everything works," advises D10D3, "You might need to configure the WiFi dongle or change the screen size. If it defaults to a resolution that's too high, it'll be hard to read the text, so you can always plug an HDMI monitor into it while you configure it."

Once you know everything works, you can use some foam tape to mount everything in the lunch box. "I only used tape on the front edge of the keyboard, so I could swing it up to turn it on and change its batteries. Make sure that you arrange things so you have room to get into it to charge stuff later. Also, make sure there's room to unplug the Pi when you're done using it, since the Pi doesn't have an on/off switch."

There are lots of ways to do a project like this, so feel free to deviate from D10D3's plans: "All of the parts are modular, and you can change its abilities by using different parts."

The Lunch Box Computer is a great project for budding spies and sleuths. It's a quick hack for hiding a Raspberry Pi, and allows you to carry your portable hacking device into places with tight security. Just be careful not to share your sandwiches, 007.

VIDEO CAPTURE UNIT

Long-term video recording is a great idea for any spy. After all, you can't be at a stakeout all day and night. It's pretty easy to capture video on the Raspberry Pi with any webcam or the Raspberry Pi Camera Module, but we admire Matt Hawkins's Video Capture Unit (**magpi.cc/1pvwEdI**).

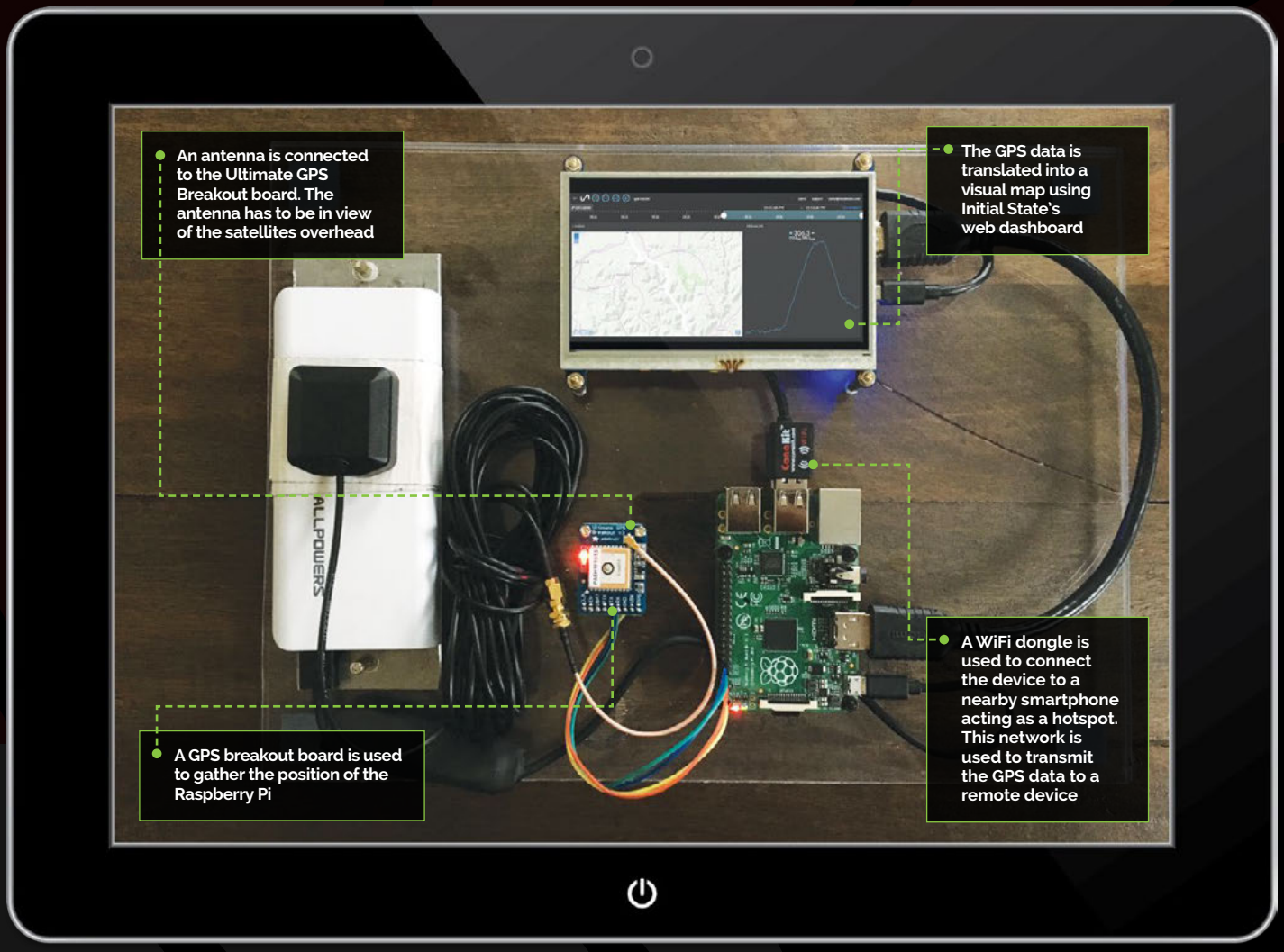
This project is a simple video capture unit that records video in a loop, with the minimum amount of hardware and setting up. "I wanted a standard setup I could quickly deploy around the house, garden, car, or bike," explains Matt. "The software needed to be easy to set up so I could use it at short notice."

USB DEAD DROPS

Spies need to share information – and with hackers everywhere, the time-honoured means of sharing secret documents is the 'dead drop'. This espionage trick is used to pass items between two people, without them ever having to meet each other. The USB dead drop is a modern take on the classic trick, which uses USB flash drives embedded (physically) into a wall. Greg Horton is a web developer from San Francisco, and his USB Dead Drops project (**magpi.cc/1QRSqEO**) is an excellent primer. You'll need a USB flash drive, plumber's tape, a drill, and some patching cement. Check out Dead Drops (**deaddrops.com**) for further information.

MOTION DETECTION ALARM SYSTEM

Spies don't just need to gather information – they also need to keep their location secure. When you're out in the field, what about all your files back at HQ? The answer is to create an alarm like Anne Nevin's Motion Detection Alarm System (**magpi.cc/1QJeXxf**). We like this one because it uses Twilio (**twilio.com**) to send you alerts via SMS, so they come straight through on your mobile phone. You can set it up with any webcam, and it uses Reactive Blocks (**bitreactive.com**) to program the alarm system.



GPS TRACKER

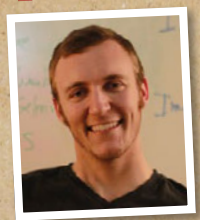
Keep track of movements with this hand-built GPS tracker

You'll Need

- Raspberry Pi
- Adafruit Ultimate GPS Breakout
- Battery charger
- Smartphone or WiFi hotspot device
- Adafruit SMA-to-uFL adapter
- Antenna
- WiFi dongle
- Female-to-female jumper wires

Tracker devices are a classic spy staple, and building a GPS tracker from a Raspberry Pi is entirely possible – as David Sulpy, co-founder and CTO of Initial State, shows. His GPS Tracker combines a Raspberry Pi tethered to a mobile phone with an Adafruit GPS module (**magpi.cc/1Ufpyre**).

[MAKER PROFILE]



DAVID SULPY

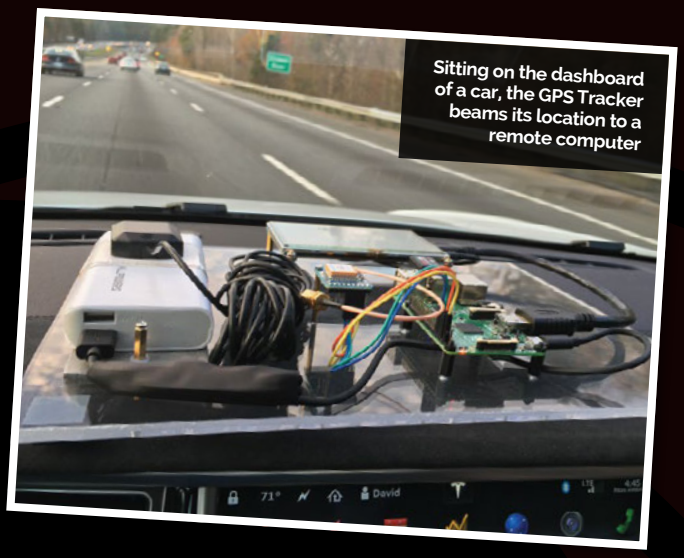
David is a computer scientist, software and security engineer, and founder/CTO of initialstate.com, a data analytics service for Internet of Things devices. **thegoodbits.sulpy.com**

The combined project tracks the location of the device and streams it over the mobile data connection to you. This GPS data is parsed through Initial State (**initialstate.com**), a data visualisation web dashboard enabling you to view the GPS Tracker's location in a real-time map view.

"I researched many ways to accomplish this task without having to utilise superfluous hardware," says David. "The Raspberry Pi's proliferation as a mobile computing platform makes it the perfect candidate for projects like this."

This GPS Tracker is a project for a future member of Q Branch to get their teeth into. It combines a variety of quirky parts with a lot of interesting code, and gathers a lot of data.

The Adafruit Ultimate GPS Breakout board comes with the board, some header pins, and a CR1220


Sitting on the dashboard of a car, the GPS Tracker beams its location to a remote computer

battery holder. Wires are soldered to the board and then connected to the GPIO pins on the Raspberry Pi. "If you've never done any soldering before, I recommend you get some spare header pins, a breadboard, and a perma-proto board to practise on," says David. A uFL-to-SMA connector is then used to connect the Ultimate GPS Breakout board to an antenna.

"Make sure the antenna is placed in an area where it has a clear line of sight to some sky," advises David. "GPS requires at least three satellite fixes to triangulate a geographic position, and at least four to get an altitude."

With the GPS hardware established, you'll need to get to grips with configuring the device and testing that it works. All of the code is available on Initial State's GitHub page (**magpi.cc/1QJf4sC**).

The project uses the pynmea2 module (**magpi.cc/1QRSy7b**) for parsing NMEA (National Marine Electronics Association) sentences, whose output looks like this: '$GPGGA,183345.000,3606.9007'. This obviously isn't very readable, so it's parsed through a dashboard account at Initial State. "We turn sensor and event data into information that matters, by making it easy to visualise and interact with data from internet-connected devices," explains the Initial State website.

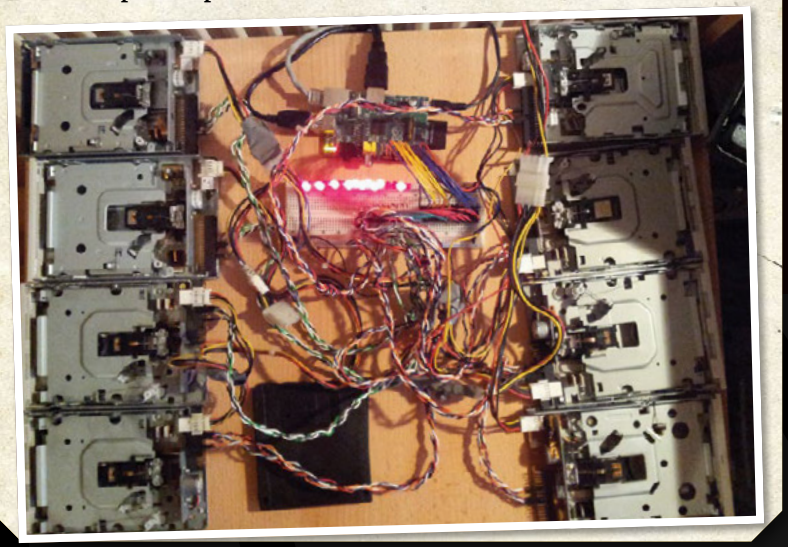
"When you log into your Initial State account," says David, "you should see a 'GPS Tracker' bucket. When you select this bucket and choose the 'Tiles' application, you should see a map with the GPS coordinates."

The final part is to make the whole thing mobile, and for that you'll need a mobile phone to provide a WiFi hotspot. You can create WiFi hotspots with most smartphones, including iPhone and Android devices. The Raspberry Pi is powered by a battery and connected to the smartphone via a WiFi dongle. The result is a clever project that broadcasts its position, no matter where it goes.

JAMES BOND HARD DRIVE

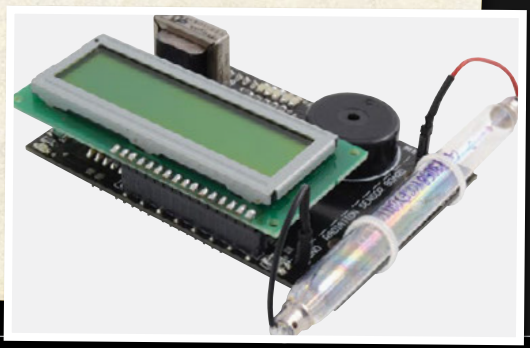
No spy is complete without a theme tune, and James Bond has the best theme tune of them all. But who wants to hear it played out of ordinary speakers, when you can get eight old floppy drives and belt it out using a Raspberry Pi to control them? That's right – old floppy drives have been hacked into musical instruments!

Daniel Kukiela's James Bond Theme On Eight Floppy Drives (**youtu.be/P3jOitAgcCI**) does what it says. "I used the Raspberry Pi and proprietary software," explains Daniel. "This makes it possible to move the head with a specific frequency, thus issuing the correct head sounds. Properly selected frequencies make it possible to 'play' music on diskettes." You'll find more information (in Polish) on PCCode (**pccode.pl**).



RASPBERRY PI GEIGER COUNTER

Radiation is deadly, and all spies should be able to get out of a dangerous spot in a hurry. That's why we think Cooking Hacks' Radiation Sensor Board for Arduino and Raspberry Pi (**magpi.cc/1Ufr7oU**) is a great piece of kit. The board is connected to a Raspberry Pi using an Arduino Shield Connection Bridge. The device uses a J305 Geiger tube, along with a piezo speaker and LED display, to provide information on radiation levels. It can detect alpha, beta, and gamma radiation levels, as well as background radiation. Geiger tubes measure the number of pulses generated, and you'll need to convert them into sieverts to obtain workable readings. This project is an excellent learning programme for all secret agents, and you'll get to know what levels of radiation different areas and activities have (and what safe levels are).



DR PAUL BUCKLEY

Dr Paul is a retired doctor who has taken up small-scale farming. His North Yorkshire farm has four Shire horses trained to work a flock of around 150 sheep.
magpi.cc/1TeDkc0

Quick Facts

- The enclosure was constructed from 6mm plywood and scrap timber
- The total cost was around £60
- The EU has introduced compulsory tagging for all sheep
- Seventy of Dr Paul's sheep are a rare endangered breed called Soay
- He uses four Shire horses to herd his sheep

EIDSHEPHERD: SHEEP TAGGER

The Raspberry Pi might have been to space, but it's also proving its worth back on Earth, where in a muddy field an RFID hacked Pi is being used to tag sheep

We see all kinds of cool gizmos built with the Raspberry Pi, but what really thrills us is to see our favourite computer being used to create practical projects that are useful in the real world. You don't get more real than sheep herding in North Yorkshire.

A few years ago, Dr Paul Buckley swapped intensive care for small-scale farming in Scarborough. He now has a flock of around 150 sheep.

Like all farmers, he keeps an eye on his sheep, but from last year it has been compulsory for farmers to tag sheep with eID (electronic identification) chips. "All sheep in the UK have to have ear tags," Paul tells us, "one of which is yellow and contains a transducer holding the details of the animal.

"It's similar to the microchip identification that's common in dogs and cats," he explains.

The specification for the sheep tags are tightly controlled, and the reader has to operate at 134kHz. "Sadly, this doesn't allow the extremely cheap and easily available dog and cat chip readers to work, because they operate at a different frequency," says Paul. "The commercially available readers aren't cheap [£700 – £5,500], and this is prohibitively expensive."

The breakthrough arrived when Paul investigated commercial RFID readers. "I came across a supplier in Australia [**priority1design.com.au**] who makes an RFID reader card for the princely sum of about £25," says Paul. "Primarily designed to be used with a laptop PC, the card has a serial data output through a USB connector and in that mode works exceptionally well. I decided that a fully portable reader that could be

The sheep have a small electronic tag fitted behind the ear that needs frequent scanning

> "The biggest issue was getting the device to emit an audible beep on successful read of the tag"

easily carried and available for tag reading in the field would be very useful. The project was born."

The Sheep Tagger is officially called the eIDShepherd – an eID ear tag reader for sheep. "My wife calls it the 'cricket bat'," laughs Paul. The heart of the Raspberry Pi is a RFIDRW-E-USB module, Adafruit 16×2 LCD display, a DS1307 clock module, and a piezo buzzer.

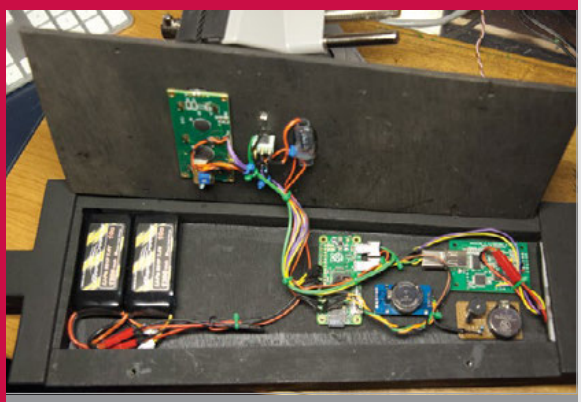
"I found, to my surprise and delight, that numbers appeared on the screen when I wafted an old sheep tag over the aerial of the reader module," recalls Paul. "My wife failed to see why I was excited. I persevered and learnt how to manipulate the data string to produce CSV-compatible output, and then how to create and append it to a text file on the SD card."

There were some challenges in turning the equipment into a fully working sheep tagger. "I discovered I needed a clock when, for the first time, I ran the Raspberry Pi off batteries and disconnected from the internet," says Paul. "The biggest issue was getting the device to emit an audible beep on successful read of the tag." The workaround was buzzer hardware attached to a GPIO pin of the Pi Zero. This is momentarily set high when a data string detects the serial input.

The total cost for the final project was less than £60, much less than a commercial device. "We collect the sheep in a pen and put the tip of the Sheep Reader close to the yellow tag in the sheep's ear. A bright blue LED on the front of the device blinks momentarily and an audible beep confirms the data acquisition. When I get home, the data is downloaded from the SD card as a text file.

"We've had no problems in the field," continues Paul. "We've been using it alongside a laptop to ensure that there are no bad reads or missing data, but all the reads have been correct and the two systems have correlated perfectly. I am now confident to use the Sheep Reader as a standalone device."

EIDSHEPHERD



>STEP-01

The cricket bat

The components are housed inside a wooden box, with two batteries used to power the Raspberry Pi and LED display. The RFID scanner is fitted at the bottom of the 'cricket bat'.

>STEP-02

Ready to tag

The LCD display provides information on the sheep being tagged. A piezo buzzer inside the bat provides feedback when a tag has been successful.

>STEP-03

Tag... you're it

The Sheep Tagger in action. It's held next to the yellow tag attached to the sheep's ear. The information about that sheep is saved to a file on the device, and is examined later.

LICHEN BEACON TEAM

The team is made up of Barry Byford, an electronic design automation specialist; Tom Hall, a music technology lecturer at Anglia Ruskin University; and Drew Milne, a poetry lecturer at the University of Cambridge. **magpi.cc/2yTQPdJ**

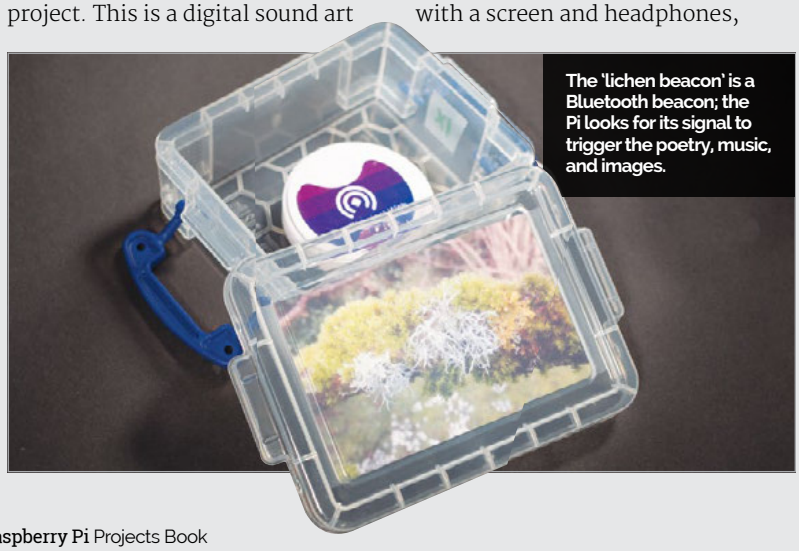
LICHEN BEACONS

An interactive sound art installation which shows that science and engineering aren't the only applications for the Raspberry Pi

Quick Facts

- The project took about six months to complete
- At the time of writing, it's only been shown in two places
- While the beacons aren't Pi-powered, it could be done easily
- The music was created in SuperCollider, which is also used by Sonic Pi
- More than one work might be hosted on the platform in the future

Lichens, apparently, are dual organisms. Fungi and algae living together mutually, they're a good indication of air pollution and the subject of the fascinating Lichen Beacons project. This is a digital sound art installation, involving someone walking around a large room with a portable Raspberry Pi (Pi-in-a-box) and uncovering Bluetooth beacons that activate different responses. The portable Pi comes with a screen and headphones, and the Bluetooth responses to the Eddystone Bluetooth beacons come in the form of music, pictures, and poems.



The 'lichen beacon' is a Bluetooth beacon; the Pi looks for its signal to trigger the poetry, music, and images.

"The idea with this platform is to make it possible to slow down and take in a digital environment, at a very different pace from the usual hectic screen-hopping and social media hot-desking that seems to define most digital environments," says the team that created the installation. Tom Hall made the music, Drew Milne wrote and read the poetry, and Barry Byford brought it all together with code.

"One of the great things about the Pi-in-a-box we created was that people needed no technical skills to use it," Barry explains. "We had a very wide range of

The humble Pi-in-a-box seems like a very simple affair, but it does exactly what needs to be done

Photo: Krisztián Hofstädter

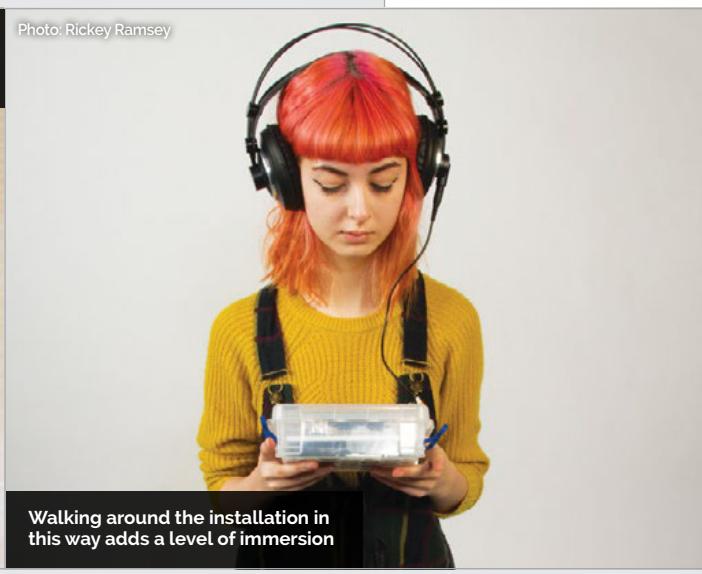


Photo: Rickey Ramsey

Walking around the installation in this way adds a level of immersion

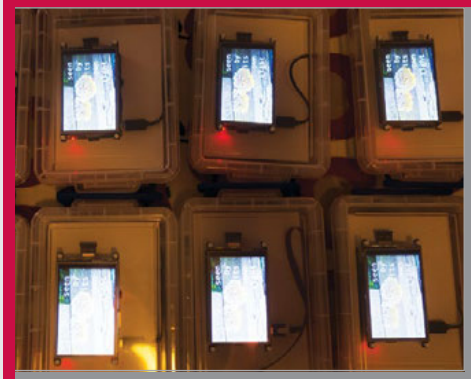
people, including some that were self-declared technophobes, and because all they had to do was walk and explore the location looking for 'Lichen Beacons', there were no technology usage issues with the equipment. This was a very pleasing result and made the event much more inclusive."

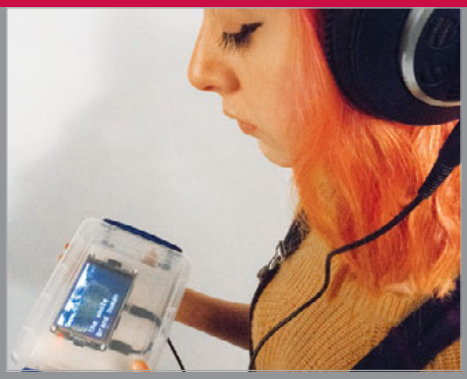
"From my perspective, there's a special affinity between lichens, digital photography, and small screens," Drew says. "The challenge is to find a new grammar of thinking and writing that can echo the world-making symbiosis of lichen life. Our installation offers the perfect platform for thinking about the poetics of digital environments, and how such environments relate to the world's fragile ecology… There's politics in the poetics, too: a way of thinking about how sound art can respond to the sites in which it is installed, while also opening up the larger questions of our environmental crisis. Our installation is a model for using technology in ways that are both home-made and also at the sharp end of what contemporary technology makes possible."

The sound design is binaural, with music wrapping around the sequential poems to create an immersive experience. Part of the future plans to improve the installation involve this sound, according to Tom: "Just as the audience can experience the 18 parts of the installation in any order, I'd like to create a musical environment that responded differently to the order in which people visited the beacons."

The installation should be turning up in more places around the UK and Europe, so keep an eye out for information on where you might be able to experience it; the full event schedule can be found on the Ludions website: **ludions.com/events**.

SEARCH FOR BEACONS





>STEP-01
Get your gear

The Pi-in-a-box is a container that has a battery, screen, and headphones attached. All you need to do is pick it up, put it on, and carry it around.

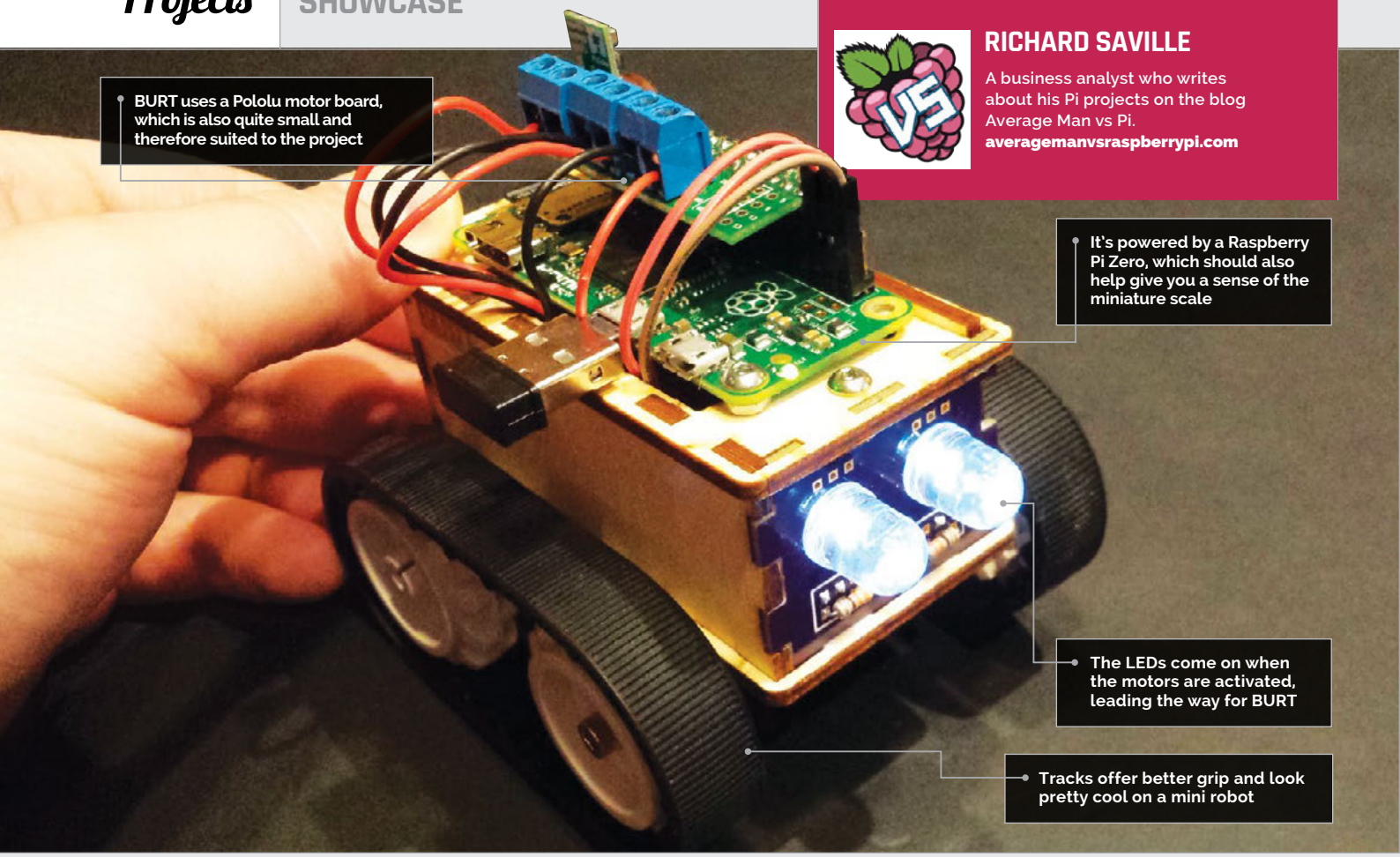
>STEP-02
Find the beacons

Bluetooth beacons are placed around the room, their signal strength activating different parts of the installation. You and the beacons dictate the pace.

>STEP-03
Learn

Listen to the music, hear the poem, and take in the information. It's not just designed to look pretty: it's also trying to impart a message.

RICHARD SAVILLE
A business analyst who writes about his Pi projects on the blog Average Man vs Pi.
averagemanvsraspberrypi.com



Quick Facts

- Work started after Pi Wars in December
- It's coded in Pygame, the module for Python
- The wood was cut at RazorLab
- A Pololu motor controller powers the Pololu motors
- The PCB face saves on wiring space

BURT BOT

A Raspberry Pi Zero robot, **BURT** is one of the smallest and cutest robots you'll see

Raspberry Pi Zero robots are hardly new – in fact, in *The MagPi* Pi Zero issue (#40) we featured a fully functional Raspberry Pi Zero robot before the board was even out! The thing we like best about Pi Zero robots is that they're always quite inventive and different (such as the Matchbot), and BURT is no different.

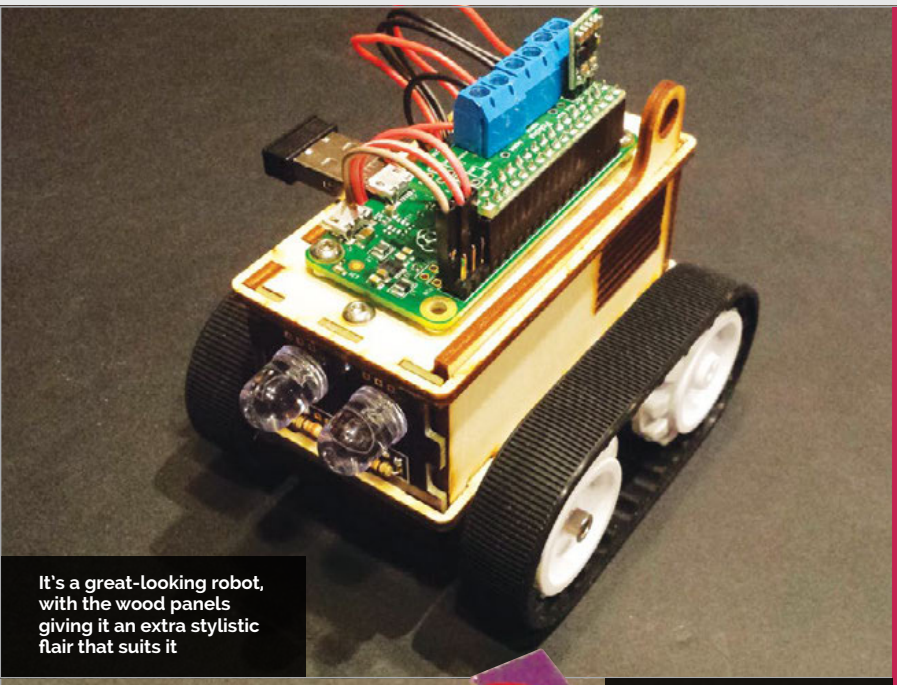
Created by Average Man himself Richard Saville, BURT stands for Boxey Unintelligent Robot with Tracks. "I had attended Pi Wars back in December with my other robot, 'AverageBot'," Richard tells us. "When the new Pi Zero came out, I wanted to try my hand at making a mini robot using the things I had learnt through Pi Wars."

It's a remote-controlled device, so not truly autonomous, but it's still an excellent little project. BURT comprises a custom-designed plywood chassis that can be rapidly revised and remade, a custom PCB 'face' with a couple of LEDs that react to the movement of the robot, and a series of motors. Motor drivers and remote controls finish off the robot to make it work.

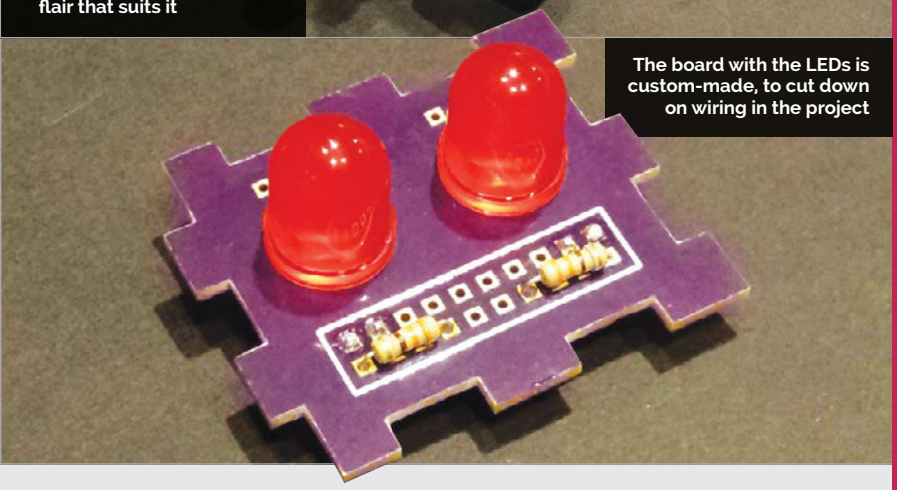
"It's not complex at all in terms of features," Richard points out. "BURT has no sensors or anything clever – simply two motors and basic controls (hence the 'Unintelligent' part of the BURT name). The complex part is putting it all together and working out where everything can go, whilst trying to maintain a small footprint. Everything is compact and fiddly, but that was always the aim."

BURT is still a work in progress, but Richard seems happy with the way it has turned out so far: "I aimed to make a small basic robot that could move around; that part works as intended, and the media centre remote control does the job well. BURT seems to be able to negotiate a range of different terrains with ease."

Some of Richard's plans to upgrade BURT should help to make it more autonomous. First



It's a great-looking robot, with the wood panels giving it an extra stylistic flair that suits it



The board with the LEDs is custom-made, to cut down on wiring in the project

on the agenda is to try to add a line sensor, much like the CamJam EduKit robot. The current power source, a series of AAA batteries, doesn't last very long either, so he's considering replacing it with a rechargeable LiPo one: "LiPo batteries still scare me a bit after seeing some videos of them 'going bad', but I may give it a try."

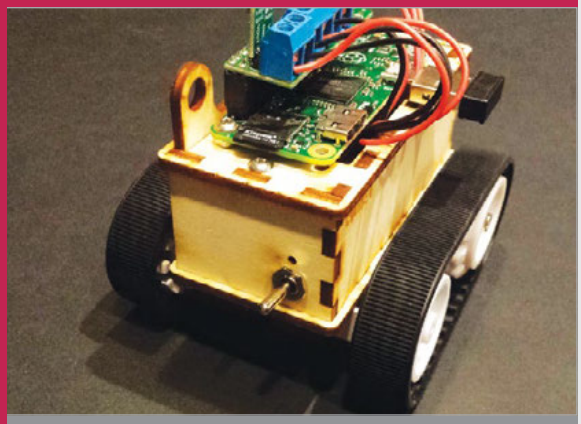
> "I wanted to try my hand at making a mini robot using the things I had learnt through Pi Wars"

Asides from BURT, Richard does have other Pi projects planned.

"I've got a really cool project partially completed for Father's Day," Richard reveals. "My dad is a huge Pink Floyd fan and has never see them play, but of course they don't play live as Pink Floyd anymore. To become #1 son, I'm making him a Pi screen unit, coded with Pygame, that will let him choose a country and see all the live gig videos. I can't get him a ticket, but I can take him back in time (kind of)!"

Robots and time machines with the Raspberry Pi, then. Richard does have a major tip for anyone wanting to give a robot like his a go, though: "If anyone wants to try something similar, remember the old rule and don't make the same mistakes as I did: 'measure twice, cut once'."

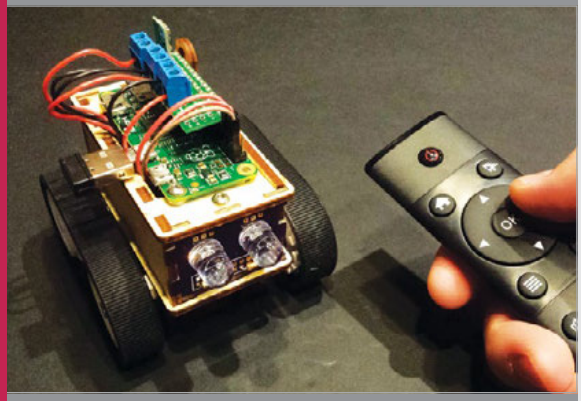
MOVING BURT



>STEP-01

Turn it on!

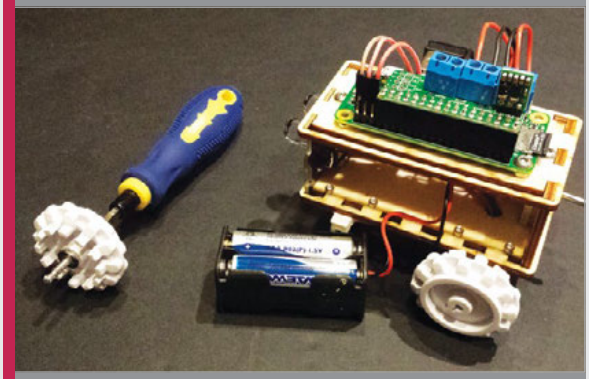
You'll need to add some batteries and get the Pi started up. This will provide power for the motors, then you can start the script to use it.



>STEP-02

Controlling BURT

BURT is currently controlled with a media centre remote, which is picked up by a USB sensor. The LEDs on the front react to BURT's movement.



>STEP-03

Recharge

Currently, BURT uses four AAA batteries to power itself, but they don't last too long, so you'll need to replace them every so often.

JAMES MITCHELL

A software QA engineer at Oracle, James is also a Pi enthusiast who organises the regular Raspberry Jam Berlin event. **raspberryjamberlin.de / twitter.com/monkeymademe**

Quick Facts

- This particular adapter only works with Canon lenses
- It was 3D printed at Dimension Alley in Berlin
- You need to move the lens to keep the moon in frame
- James is hoping to use a longer lens for better photos
- He's also shot a time-lapse video with the adapter

PI MOON CAMERA

With the aid of a 3D printed lens adapter, it's possible to take detailed photos of the moon using the Raspberry Pi Camera Module!

Shooting the moon has been an obsession of James Mitchell's for a very long time. After several failed attempts to get a clear photo from various cameras and lenses over the years, he decided "on a bit of a whim" to give it a go with a Raspberry Pi Camera Module (**magpi.cc/1Nd44Dx**).


The pink tinge is caused by a reflection of light on the camera sensor, which James has since fixed

"I have been working on pushing the limits of the Pi camera for a while," James tells us. One of his first pictures was a long exposure of a Lego figure, which won him the runner-up prize in an Adafruit competition. "I wanted to use my lenses to improve on that."

After discovering a design for a Canon lens adapter for the Pi by Charmlee (**magpi.cc/1MhDNJl**), James got a 3D print made at his local Berlin printing cafe and was impressed: "Everything simply slots together. Actually quite a nice design, to be honest, but it only works with Canon EF (EOS) lenses." He soon realised that using the adapter with the Pi Camera Module requires the removal of the latter's tiny stock lens. "It really is simple… it just screws off. What might be considered difficult is that it has a little glue inside to stop it from moving. So I used a couple of pairs of pliers to remove it." While this did buff the plastic slightly, the lens can go back on to protect the sensor when storing the camera.

The Pi camera sensor can then be inserted into the 3D printed adapter, enabling it to be exposed directly to light coming through the attached Canon lens. Since the sensor only

The final image, post-Photoshop. The pink tinge on the original unedited image (bottom-left) was caused by a layer on the Pi camera sensor

Above All the components James used for the project, including the red 3D printed lens adapter in three parts, which he later wrapped in duct tape

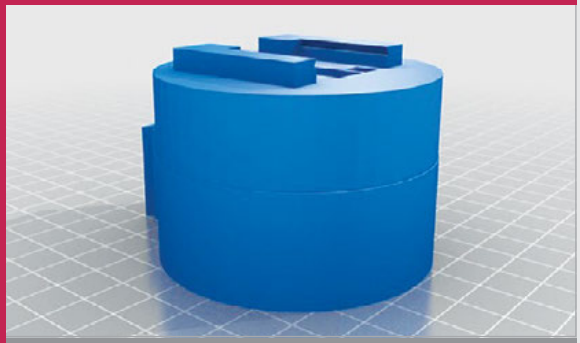
captures a small section of what the lens can see, it does so with 5 megapixels' worth of detail. This led James to try shooting a full moon from his Berlin balcony, using a 78–300mm lens, which resulted in a surprisingly detailed image of the cratered lunar surface.

Asked what camera settings he used, James replies, "I'm embarrassed to say I left it on auto. In fact, the command I used had no modifiers at all – **`raspistill -o moon.jpg`**. What I got really was a happy accident. Could I improve the picture with changing some settings? Absolutely!... I think I need to learn more about how the optics work. Maybe I could adjust the distance between the lens and the sensor to improve the sharpness."

James admits that a DSLR would be better for taking a picture like this, "but only with a longer lens, something larger than 300mm, or even a telescope." One peculiarity in his moon photo is the pink vignette around its edges, caused by light reflecting on a layer of the Pi camera sensor. He has since managed to reduce the effect by wrapping the camera adapter in black duct tape.

As well as being useful for astronomy images, James's setup is ideal for any type of photo or video requiring a high level of zoomed-in detail. "You could use [it] to monitor birds' nests that might be quite a distance away [or for] macro photography with an assortment of lenses and adapters."

BUILDING A MOON CAMERA



>STEP-01
3D print the adapter

The Canon lens adapter was 3D printed from a design found on Thingiverse.com, by Charmlee (**magpi.cc/1MhDNJl**). Its three parts – the main body, base, and rear panel – simply slot together.

>STEP-02
Remove Pi camera lens

To enable the Pi Camera Module's sensor to be exposed directly, its tiny stock lens needs to be removed. It can be unscrewed using small pliers – do so at your own risk!

>STEP-03
Fit it all together

The Canon EOS lens screws onto the front of the adapter body, while the Pi camera sensor slots into the rear panel. You're now ready to shoot the moon, or any other subject!

WIMBORNE MODEL TOWN

An attraction in Wimborne, Dorset. A small team worked on the lights, with Terry Coles doing the coding.
magpi.cc/26eCNfu

Quick Facts

- The town is refurbished in the winter while it's closed
- Terry is a veteran coder
- The model railway itself is full of Thomas the Tank Engine figures
- More upgrades can be added in future
- Parts of the track are interactive for visitors

WIMBORNE MODEL RAILWAY

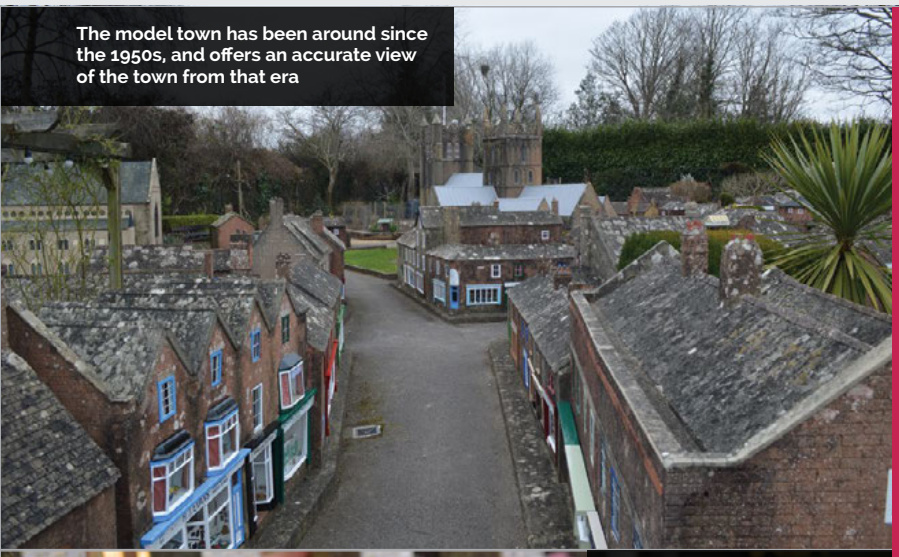
A model town in Wimborne has a separate model railway attraction. When it came time to upgrade parts of it, a Raspberry Pi was the natural choice

In the beautiful town of Wimborne, very close to where *The MagPi* is made, is a superb model town that has existed for almost 70 years. It's a lovely tourist spot and a wonderful attraction, of a type that's fairly rare these days. As part of the model town, there's also a great little model railway. During the off season just past, the lights that make up part of this miniature railway were upgraded using a Raspberry Pi, by a team led by Terry Coles, an engineering manager who helps out at Wimborne Model Town.

"The fundamental requirement for the lighting is simple," Terry tells us. "Simulate daylight, dim the lighting to simulate night-time, and then bring on lights in the trackside houses and streets. However, these were not the difficult goals; a bit more work was necessary because the finished system needed to be reliable, easy to use and maintain, and capable of being upgraded readily in the future."

After a lot of discussion, the team settled on a Raspberry Pi Zero for the task: "This allowed for future upgrades and was considered more accessible to the average volunteer than, say, using an Arduino."

The system isn't just made up of a Raspberry Pi, though: there's



The model town has been around since the 1950s, and offers an accurate view of the town from that era

Although Thomas the Tank Engine-themed, there are some other pop-culture cameos on the railway

much more to it. The lights being used weren't just a few LEDs stuck to a breadboard, as Terry explains:

"The WiringPi2 library was used to allow programming of a pulse width modulation (PWM) signal. This dims and brightens a 48W LED strip on GPIO pin 18 (the hardware PWM pin) to simulate the 'Day' lights. A MOSFET driver was used to carry the current needed. The lighting in the houses and elsewhere was switched by four other GPIO pins via a relay board."

The system itself was developed on a Raspberry Pi with Raspbian, to make sure it was all working. Originally written in Python with the RPi.GPIO library, it doesn't quite support hardware PWM so was then changed to the aforementioned WiringPi2. The final code was then moved over to a piCore image for actual deployment. Terry says that as the volunteers will just turn the Pi on/off rather than do a software shutdown – it's a bit tricky with the way the system is built – piCore was the obvious choice, as the OS is loaded into memory and so there's no reading or writing going on from/to the SD card.

> "Reliable, easy to use and maintain, and capable of being upgraded readily in the future"

All the software written for the project is also open-source, with Terry maintaining very thorough documentation and sources that you can find here: **magpi.cc/1Nl7nhg**. If you ever find yourself on the south coast in or near Dorset, go give the model town and railway a look!

DAY & NIGHT CYCLE

>STEP-01
Daytime

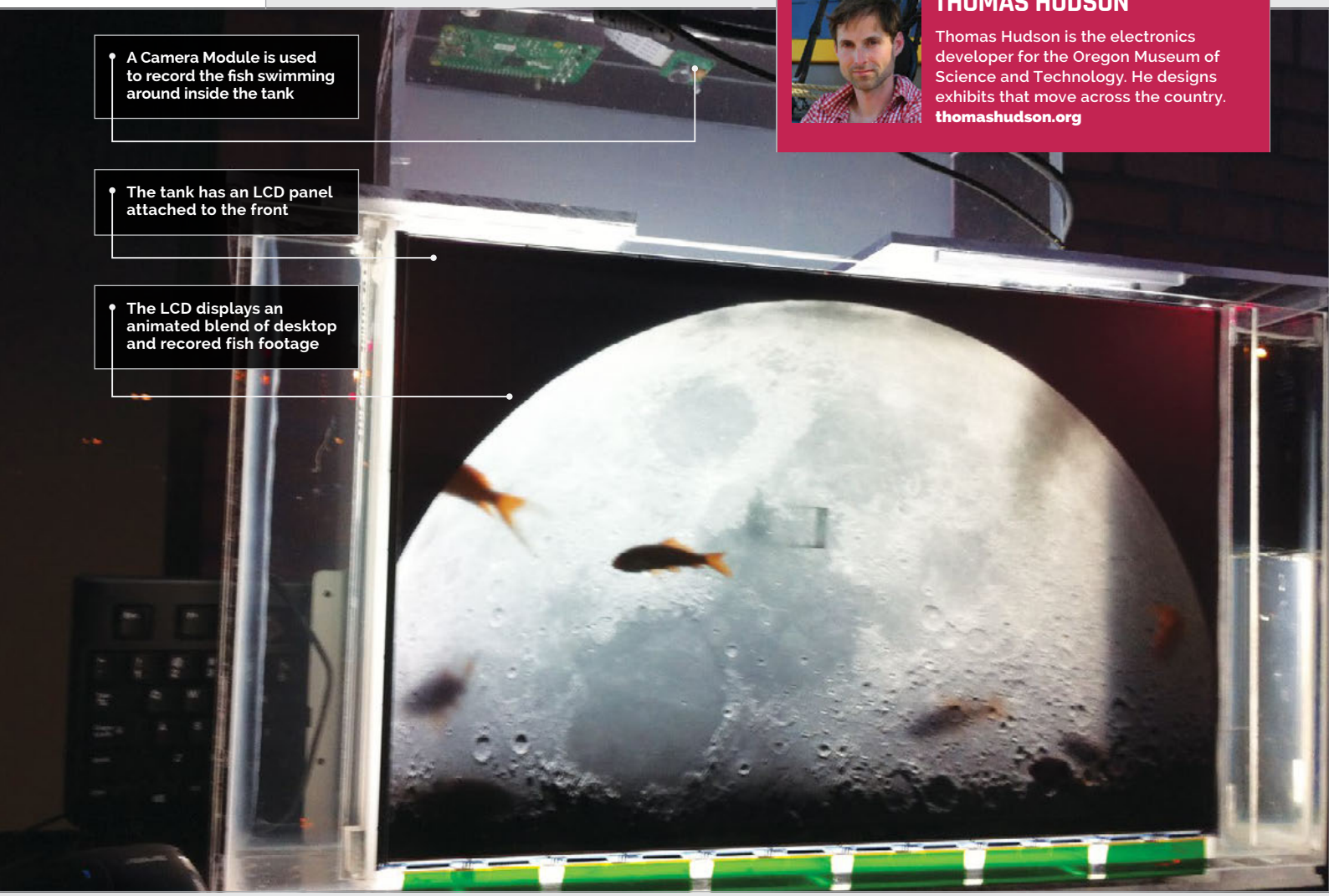
During the 'day' portion of the cycle, a 48W LED strip is lit by the Pi (seen at the top of this photo) to light up the model railway. It's very bright, and illuminates the entire display for five minutes.

>STEP-02
Night-time

As the lights dim to signify the end of the day, lights in the village start to come on, one by one; first a few houses, then some streetlights.

>STEP-03
When dawn comes

As the fake sun rises after three minutes, the lights in the town slowly start to turn off in order, as the town's diminutive population gets ready for the day.



THOMAS HUDSON

Thomas Hudson is the electronics developer for the Oregon Museum of Science and Technology. He designs exhibits that move across the country.
thomashudson.org

FISH-EYE

There's something fishy about **Thomas Hudson**'s new monitor. We investigate the display that's also an aquarium

Quick Facts

- The Fish-eye was an art installation in Oregon
- The tank was made from acrylic and acrylic glue
- Five common goldfish lived in the tank
- The tank cost around $50 (£35) to build
- It took 12 hours for the fish tank glue to hold

We adore this hybridised fish-tank and LCD screen. Known as the 'Fish-eye', the project was build by Thomas Hudson, a developer at the Oregon Museum of Science & Technology.

The Fish-eye is a fascinating creature. Unexpectedly, the LCD monitor is situated at the front of the tank, and the Raspberry Pi software superimposes the fish onto the display. "A Raspberry Pi Camera Module sits on top of the fish tank, looking down," explains Thomas.

Fish have a tranquil, calming effect, and watching them glide serenely around a tank while you work is awesome. And if you want to watch space fish, then the Earth or Moon make great fishy backdrops.

"I built the tank out of acrylic using acrylic glue," Thomas tells us. "It is amazing stuff: fairly toxic, but it welds the acrylic together so it is watertight.

"The monitors were free," he adds. "One from a free box on the sidewalk and another from a friend. Everyone is getting rid of 19-inch monitors right now."

With the custom tank constructed, Thomas stripped down the LCD screen and fixed it to the front of the fish tank.

"The camera captures live video of the fish from the top view and streams it live to the LCD on the front of the tank. So when you're looking at the front of the tank, you are looking at both live video of the fish from the top view while watching the 'real' fish as seen through the LCD screen."

You don't actually see the fish. The camera records them and superimposes them on your desktop

> "This intersection of moving life and still pixels gave the project its artistic message"

The Fish-eye uses quite a bit of bit of software to achieve its unique effect. It runs a picture program called Feh (**feh.finalrewind.org**) to flash full-screen images of space fish on the Moon. Oxmplayer (**elinux.org/Omxplayer**) is used to show the live stream and also recorded video of the fish. Finally, the Raspberry Pi runs a surveillance script based on Motion MMAL (**magpi.cc/23MBxfH**) to identify moving fish and attach bubbles to those fish displayed on the live video feed.

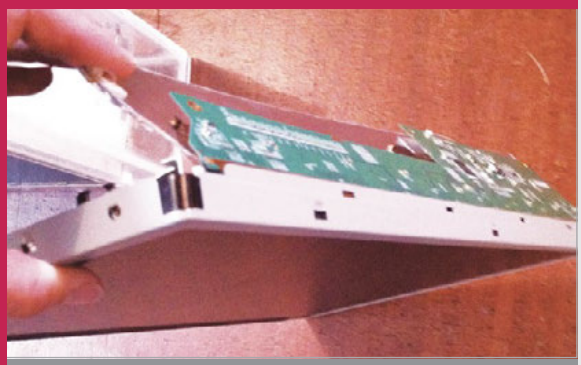
"I worked on the project for about a month," reveals Thomas. "Two shows were happening, so I made two Fish-eyes. One [was] for an annual electronics art exhibition called ByteMe! 5.0; the second was for the Portland Winter Light Festival."

This intersection of moving life and still pixels gave the project its artistic message. "It was important for something to [be] living in the box," explains Thomas. "This is juxtaposed against how much time we spend staring at a 'box' that is not living, and is very much dead, with the exception of all those electrons buzzing through them and the cold emitting light."

The Fish-eye was retired after the Portland Winter Light Festival was over, and the goldfish returned to a regular tank. "It was a sculptural piece," says Thomas. "I feel that kids got it right away. They loved the idea of 'fish on the Moon' and 'fish in space'.

"I think there is something very beautiful about having depth to your monitor. You can still focus on your work, as the screen is crystal clear." Indeed, we think the Fish-eye is a fabulous display.

BUILDING A FISH-EYE

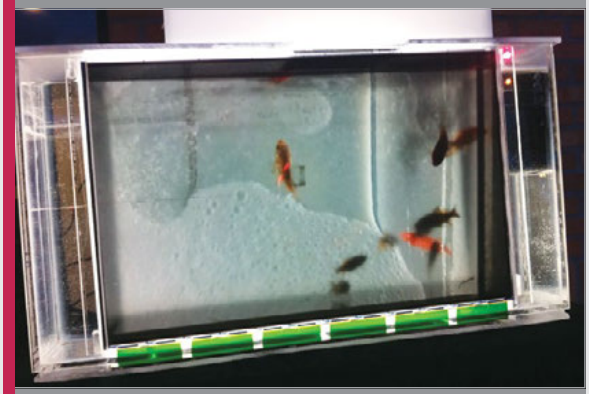


>STEP-01
Stripped display

A 19-inch LCD is recycled from an unloved monitor. Once stripped of its surrounding case, the display will form the front of the Fish-eye screen.

>STEP-02
Tanks a lot

A container is built from sheets of clear acrylic glued together. After the glue has finished drying, the display is attached to the front of the tank.



>STEP-03
Fish Pi

Water fills the tank, and the fish swim inside. The Raspberry Pi and Camera Module blend the display and fish together.

MARK SANDERS

A software developer who loves great coffee, so much so he built his own roaster.
magpi.cc/228QtW9

Quick Facts

- Mark likes to brew coffee in a cafetière, also known as a French press
- Colombian, Ethiopian, and Mexican beans have roasted well
- This is Mark's first Raspberry Pi project
- The whole thing took between two and three months to build
- The roaster is controlled by a custom web interface

RASPBERRY PI COFFEE ROASTER

Want to up your coffee-making game even further? Make a cheap yet accurate coffee roaster, like **Mark Sanders** did, for a better cup

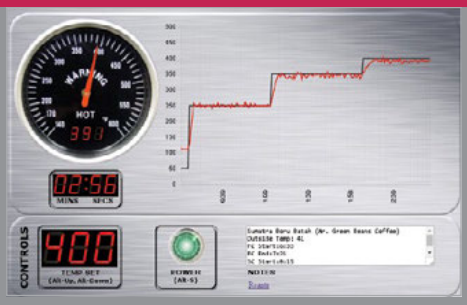
Right The build effects on the case add an extra level of class to this custom coffee roaster

Making a truly good cup of coffee takes a lot of effort. Essential oils evaporate from ground coffee within minutes, so pre-ground coffee is out. Boiling water will burn the coffee as it's brewed, so you need it at about 80–85°C. Grind size, exact weights, precise timing, brewing method: all of it is important when making coffee. Possibly the most extreme step you can take is to roast your own beans to make sure they're fresh and to your own requirements, and this is what Mark Sanders has found himself doing.

"While in search of quality coffee, I stumbled upon a webpage

GETTING THE PERFECT ROAST

>STEP-01
Temperature set
The temperature is set in 25°F (14°C) increments, starting at about 300-350°F (150-175°C) and is manually increased over 5-6 minutes to 450°F (230°C).



>STEP-02
Maintaining temperature
The Raspberry Pi checks the temperature every 0.1 seconds and turns the heating element on or off, depending on whether the roaster needs to warm up or cool down respectively.

>STEP-03
Manual inspection
The roaster won't know when the beans are done, so Mark keeps an eye on the coffee and listens for any cracking in the beans to determine whether or not they're ready.

that suggested home roasting as a source of tasty coffee," Mark tells us. "Getting started is as simple as purchasing a used popcorn popper from the local thrift store for $4-$8."

Popcorn poppers are popular among home-roasters, as they stir the beans while very hot air blows on them – very similar to commercial coffee roasters. There was a small problem, though, according to Mark:

"It roasted the coffee too fast, as I frequently finished roasts in under five minutes [when they should take 7-12]. In order to slow the roast down, I would unplug the popper for 30-second intervals. This grew tiresome and I started to ponder how I could add temperature control to the popper. This was the beginning of my Raspberry Pi project."

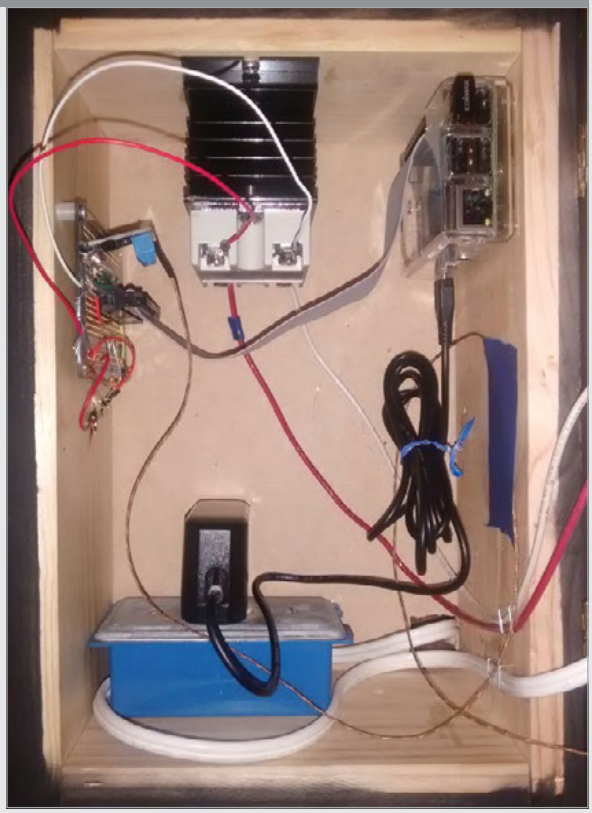
The solution is familiar to those who have tried (or at least seen) sous vide projects: controlling the heating apparatus. In this case, Mark cannibalised the parts from the popper and used them in another vessel.

"I took the heater and fan from a popper and added several other components to create a temperature-controlled coffee roaster," Mark explains. "A thermocouple was added to measure the temperature inside the roasting chamber. A GPIO pin was connected to an AC relay that allowed the Raspberry Pi to control the popper's heater. I developed a web interface to set the temperature and save roast data for archiving. The web interface shows the current temperature using an analogue dial gauge, as well as a chart that graphs the temperature for the entire roast. For some extra flair, four LEDs were added to the front of the roaster to simulate a flame when the heater is turned on."

> "I took the heater and fan from a popper and added several other components"

A full list of parts and instructions to build a roaster yourself is available online: **magpi.cc/228QtW9**. The build is quite fiddly, with a lot of soldering and patience required, according to Mark. However, the results speak for themselves:

"[The roaster] has completed over 20 roasts and it has worked very well. I'm able to control the temperature throughout the entire roast process, log notes about the roast, and save the roast data for reference later. It's much better than unplugging and plugging in the popper by hand."



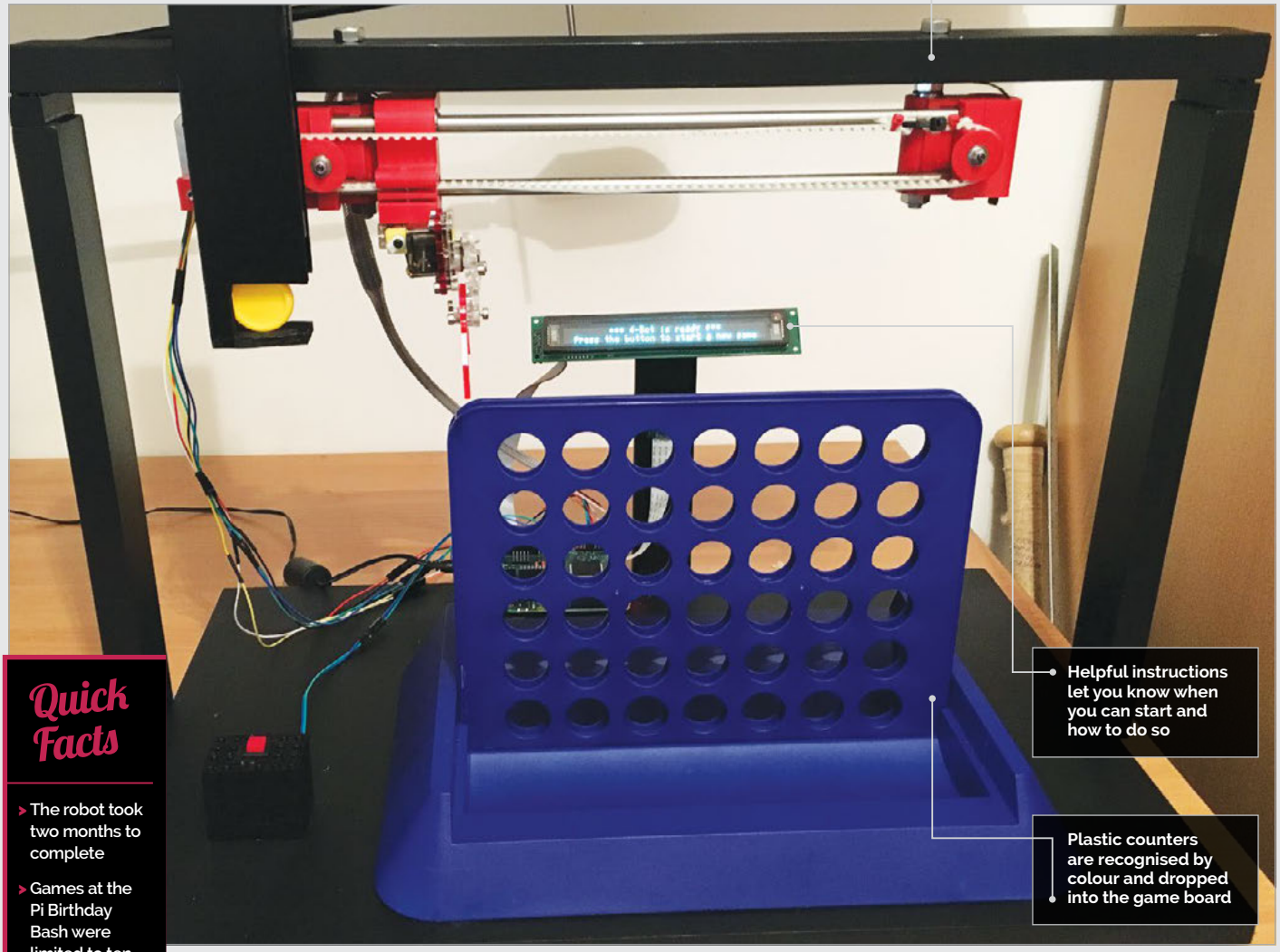
Above The coffee roaster is controlled by a number of electronic components to get it working just right

DAVID PRIDE

Returning to school 30 years later, David became an MSc student thanks to taking up the Pi as a hobby.
magpi.cc/1XrC3zU

4BOT

A robot that plays Connect 4 against a human opponent thanks to a bit of maths and code



Quick Facts

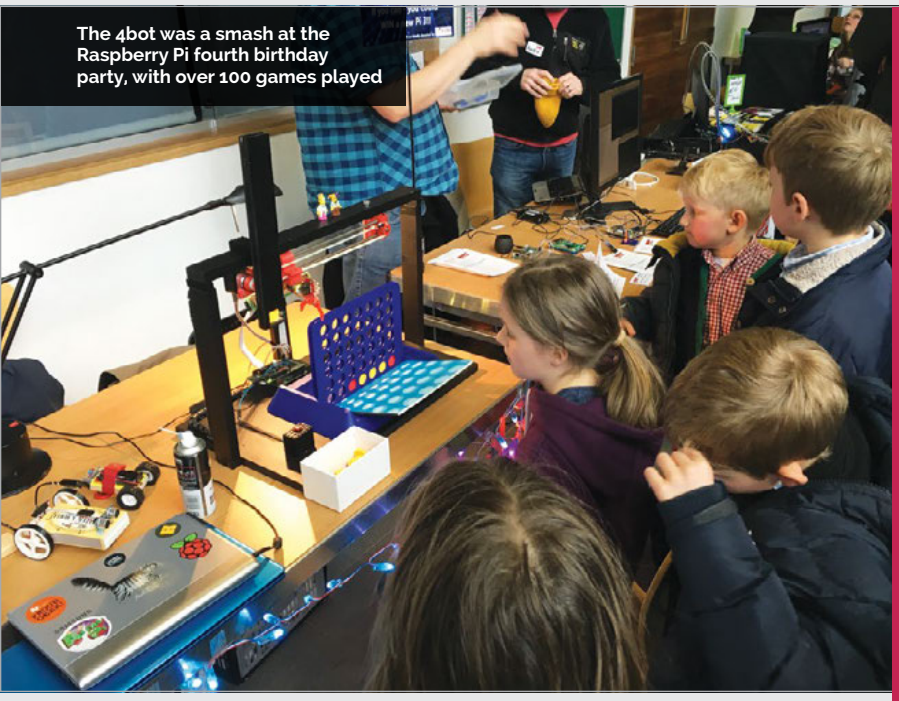
- The robot took two months to complete
- Games at the Pi Birthday Bash were limited to ten moves each
- Only a few people won the game in ten moves
- The add-on board that works the servos will soon be available as a HAT
- David is currently converting an Ohbot to run on Raspberry Pi

You may have seen computers that can defeat grandmasters at chess, win at trivia game shows or, more recently, beat a human at Go, but have you ever played Connect 4 against a robot? Humour us by answering no (statistically speaking, you probably haven't) and then prepare to be amazed at the 4bot created by David Pride.

The 4bot has humble beginnings, as David explains its origins to us: "My wife bought me the brilliant MeArm kit and I used it to build a Lego block sorter, as you do. This used the Pi Camera Module and a colour recognition script I wrote in Python to identify the different coloured blocks, and then used the arm to drop them in the correct 'buckets'." A video of this can be seen here: **youtu.be/FJ8WV1uLhFA**.

"Based on this, I was then looking for other uses for the colour-capture code," David continues. "Connect 4 seemed like a really good choice. Research soon led me to find that the game, and the logic behind it, is far from simple. There is good information online; however, whilst I found many versions of Connect 4 for Python, few of them ran successfully on the Pi."

The robot works by taking a picture of the game board, processing the colours, and then giving the game program the state of the board to calculate the next move.



The 4bot was a smash at the Raspberry Pi fourth birthday party, with over 100 games played

"In terms of how well it plays, Connect 4 is a 'perfect' game in mathematical terms," David reveals. "There are a huge but finite number of solutions, and they can all be calculated with enough processing power. The trade-off is in the depth of search and therefore the time taken to calculate each move... If you increase the search depth, this massively increases the calculation time. So I selected a middle ground where the bot plays a pretty mean game, but the total time per move is still acceptable. With capturing and processing the image, calculating the next move, and delivering the counter, the total time per turn is around 25 seconds."

> "In terms of how well it plays, Connect 4 is a 'perfect' game in mathematical terms"

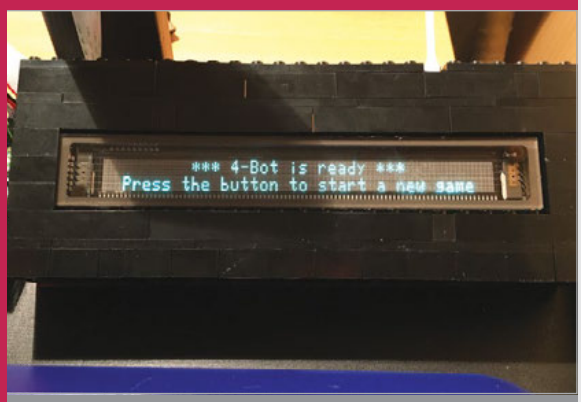
Recognising the colours currently in play is not easy, though, and took a bit of trial and error to get right.

"The trickiest part was undoubtedly capturing the game board accurately every time," David tells us. "It is extremely light-dependent, as the Python module works by capturing the RGB value of the 42 spaces on the board. These values, however, do change dramatically depending on the lighting. I wrote a 'testcard' script that can be run with counters in known locations. This script then reports back what it thinks it sees, and the tolerances for the RGB components can then be adjusted until the result matches what is actually there on the real game board. This made the game more portable, as it can be adjusted to its surroundings each time."

Current upgrades for the robot involve a stronger frame, as it came in for heavy use at the fourth Pi birthday party. Maybe it's time to bring robot Connect 4 battles to the next Pi Wars?...

Above The Pi Factory Lego block sorter

MAKING A MOVE



>STEP-01
Begin game

The 4bot lets you know when it's ready to go. Read the instructions on the display and press the button to start the match.

>STEP-02
Red goes first

It's time to make your first move. Once you're done, the Pi camera takes a photo of the game area and processes it to find out the state of the board.

>STEP-03
Beat the computer

The Raspberry Pi calculates its next move, although it's given a time limit on how much it can think about it. The arm then places the counter in the desired column.

ALAIN MAUER

Alain lives with his daughter Stacy and son Scott in a small village in the north of Luxembourg.
awallelectronic.blogspot.com

The interface designed by Alain displays six cartoons. Scott uses the buttons to choose which video to play

SCOTT TV

One MagPi reader has built a customised television that his autistic son can use unaided

Quick Facts

- The build took two weeks to complete
- Arch Linux is used for the interface
- Videos are played using Omxplayer
- The scripts were written using Python 3.5
- Scott has Kanner syndrome, a severe form of autism

Having a child with autism isn't easy. Alain Mauer's son, Scott, can't make eye contact and doesn't talk. He also requires constant supervision.

"Communication with Scott is very difficult," says Alain. "He understands us, but can't tell us what he wants. You can't leave him alone for a single second.

"But from time to time he has to stay in his room," adds Alain. The answer was to make Scott's bedroom "Scottcompatible! You can lock everything from the wardrobe to the window, and we've installed a camera in the wardrobe too."

But this isn't much fun for Scott. He doesn't like staying alone in his room and finds it boring. "He doesn't play with toys or use his imagination to play. But we know that he likes cartoons with music," reveals Alain.

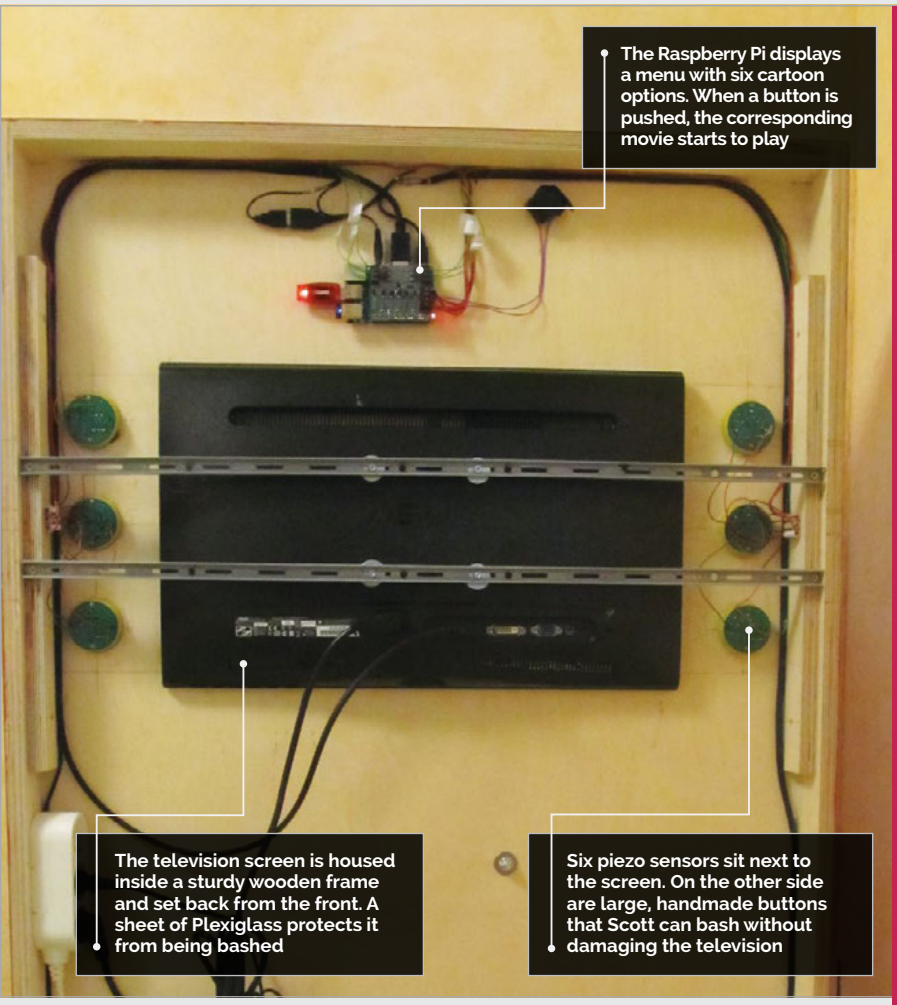
Right Scott enjoying his new TV. He can bash the buttons (and screen) all day long without damaging it

"Scott had a 32-inch TV in his room because he loves to watch cartoons, but one day he destroyed it. So we tried another one with a Plexiglass sheet in front of it, but he tried to destroy it too."

Alain started to wonder if the problem wasn't the television, but what was playing. "He has no ability to tell us, or to stop it, on his own," explains Alain. "So he gets frustrated, and tapping against the noisy thing is his only way of stopping it."

The answer was to build a television that was Scott-proof, and the result is Scott TV: an unbreakable television with the screen hidden inside. Six large, easily bashable buttons start and stop cartoons playing. A Raspberry Pi – tucked safely inside the wooden case – powers the whole project.

Built out of 18mm multiplex wood, the Scott TV case houses the television, but it still needs protection. A sheet of 8mm



Plexiglass in front of the display is the solution. "It wasn't complicated to build," says Alain. "With a jigsaw, a drill machine, and a little table saw, all is possible.

"The Raspberry Pi is the main part of the project. Normally, I tell everybody that the Pi is optimised for controlling stuff and not as a media player. But I was wrong: it plays full HD videos."

The Scott TV has six large buttons, handbuilt by Alain with piezo switches on the inside. "The buttons were the challenge," says Alain, "but you can use any kind of buttons."

The menu displays previews from six cartoons (one next to each button). If Scott pushes a button, the cartoon starts playing. If he pushes any button when a movie is playing, it returns to the menu screen.

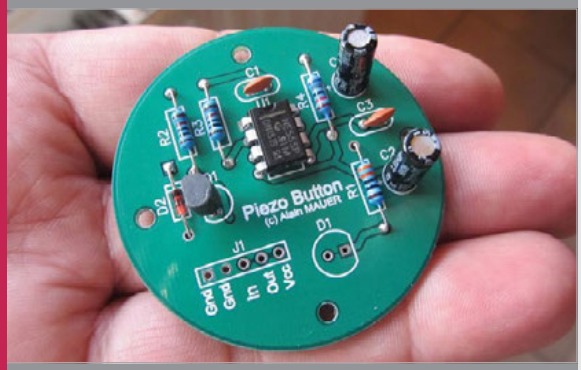
"I used NOOBS to install Arch Linux. For [a] console-based media player, I used Omxplayer. I then used Python 3.5 to write the scripts. I'm not a Linux expert or a programmer, so Google was my friend. The scripts are available on GitHub (**magpi.cc/22amFIG**).

> "To see him so happy was the biggest thank you from him to me"

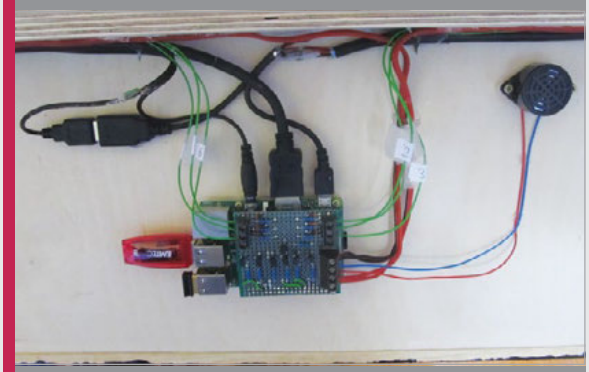
"Since it was set up in his room, Scott likes to stay longer and push the buttons," Alain tells us. "Sometimes he sits in front of the media player and just looks at the animated menu, or he plays the Twinkle Twinkle Little Star video ten times and laughs. To see him so happy was the biggest thank you from him to me."

CREATING SCOTT TV

>STEP-01
Building the case
Multiplex wood was cut, using a jigsaw, to form the case. The large rectangle is for the screen, while the circles form holes for the buttons.



>STEP-02
Button bashing
Scott TV requires six large and robust buttons. These were handmade and proved the most challenging part of the build. Piezo switches are used inside the circular button case to detect pushes.



>STEP-03
Playing media
The buttons and display are wired up to a Raspberry Pi. A Python script detects button pushes, and plays (or stops) one of six cartoons.

MARTIN MANDER

Martin is a business intelligence analyst with a passion for converting vintage technology into new creations using modern components, especially the Raspberry Pi.

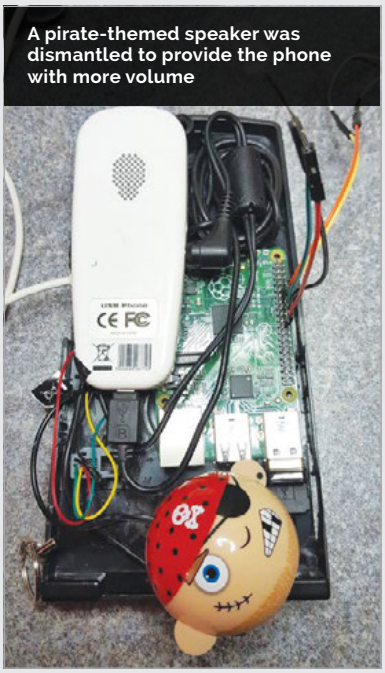
magpi.cc/1U61EvA

ALEXAPHONE

The Raspberry Pi inside this 1970s telephone calls up Amazon's Alexa voice assistant and gets the answers to almost any question

Quick Facts

- The AlexaPhone took two weeks to build
- Alexa uses natural language processing AI to answer questions
- AlexaPhone only takes two seconds to respond
- The Trimphone's dial contains a tiny amount of radioactive tritium
- Around 1.6 million Trimphones were sold in the 1970s



A pirate-themed speaker was dismantled to provide the phone with more volume

Martin Manders has a passion for upcycling vintage technology. He's well-known for using the Raspberry Pi to add 'smarts' to classic VCR and radio technology, but his latest project, the AlexaPhone, makes an old telephone ultra-smart with a connection to Amazon's Alexa voice search system.

Martin has stripped out a classic 1970s Trimphone and fitted a Raspberry Pi inside. "You lift the handset, speak your query, and the response from Alexa is read out via a built-in speaker," explains Martin.

Alexa provides users with information on the web via voice search, including weather, news, and vital facts like 'how old is Graham Norton?' or 'why is the sky blue?'.

"She has a fun side too," Martin tells us, "with a seemingly bottomless selection of dad jokes and preprogrammed responses to odd questions like 'would you like to build a snowman?'." Beyond the jokey façade, Alexa also sets timers and reminders, plays music, and reads audiobooks.

"I have a real weakness for retro design," says Martin, "especially telephones and televisions." The Trimphone was the height of technology in the 1970s, replacing the bell ring of classic phones with an electronic warbler. "I think I picked it up at a car boot sale in Brighton about 15 years ago," he

The finished project is a stylish 1970s Trimphone that calls Amazon's Alexa service

recalls. "It proved perfect for the AlexaPhone project, as the internal wiring has convenient modern ribbon cables for connection to the Raspberry Pi."

Martin used an old USB internet phone to connect the Trimphone's microphone to the Raspberry Pi. A cheap portable speaker is stripped down and fitted inside to play back the responses.

"After some digging around, I came across Sam Machin's excellent AlexaPi code on GitHub (**magpi.cc/1U61O6r**)," reveals Martin. "It offers Alexa integration for the Raspberry Pi with a physical button connected to the GPIO pins.

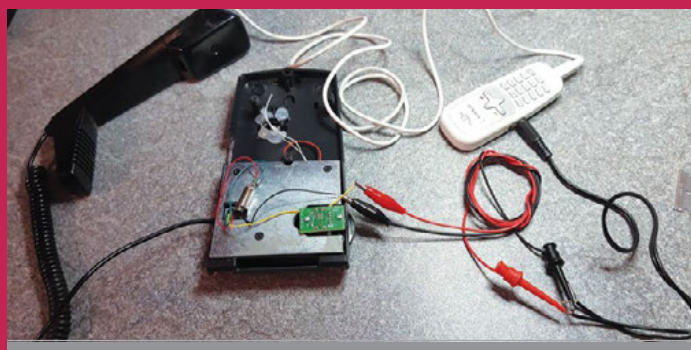
"The AlexaPhone started out as a quick distraction," he continues, "but was so much fun to build it just took over. "I got the AlexaPi software fully working on my Raspberry Pi 3 in the workshop, then repeated all the steps, removing cables as it gradually became wireless-enabled and headless. Getting all of the components to fit inside the phone body was a bit of a tight squeeze, but with some liberal plastic trimming it came together in the end."

The only thing Martin updated was the name of an MP3 file in the script. This was a straightforward change so that the AlexaPhone would sound its signature Trimphone ringtone on boot instead of Alexa's usual chirpy 'Hello!'.

> "She has a fun side too, with a seemingly bottomless selection of dad jokes"

"I've found it accurate, the voice recognition is good, and even when I've stumbled over words, Alexa usually figures out what was said. I have it on my office desk and use it nearly every day, sometimes to get information, but often just asking questions out of curiosity to see whether Alexa will understand them. She's particularly good at maths as well, so the AlexaPhone comes in handy for double-checking the kids' homework answers."

BUILDING A RETRO VOICE ASSISTANT



>STEP-01

Hooking up a mic

A USB internet phone is hooked up to the Trimphone using a 3.5mm audio cable taped to the connections of the receiver. This acts as a digital microphone.

>STEP-02

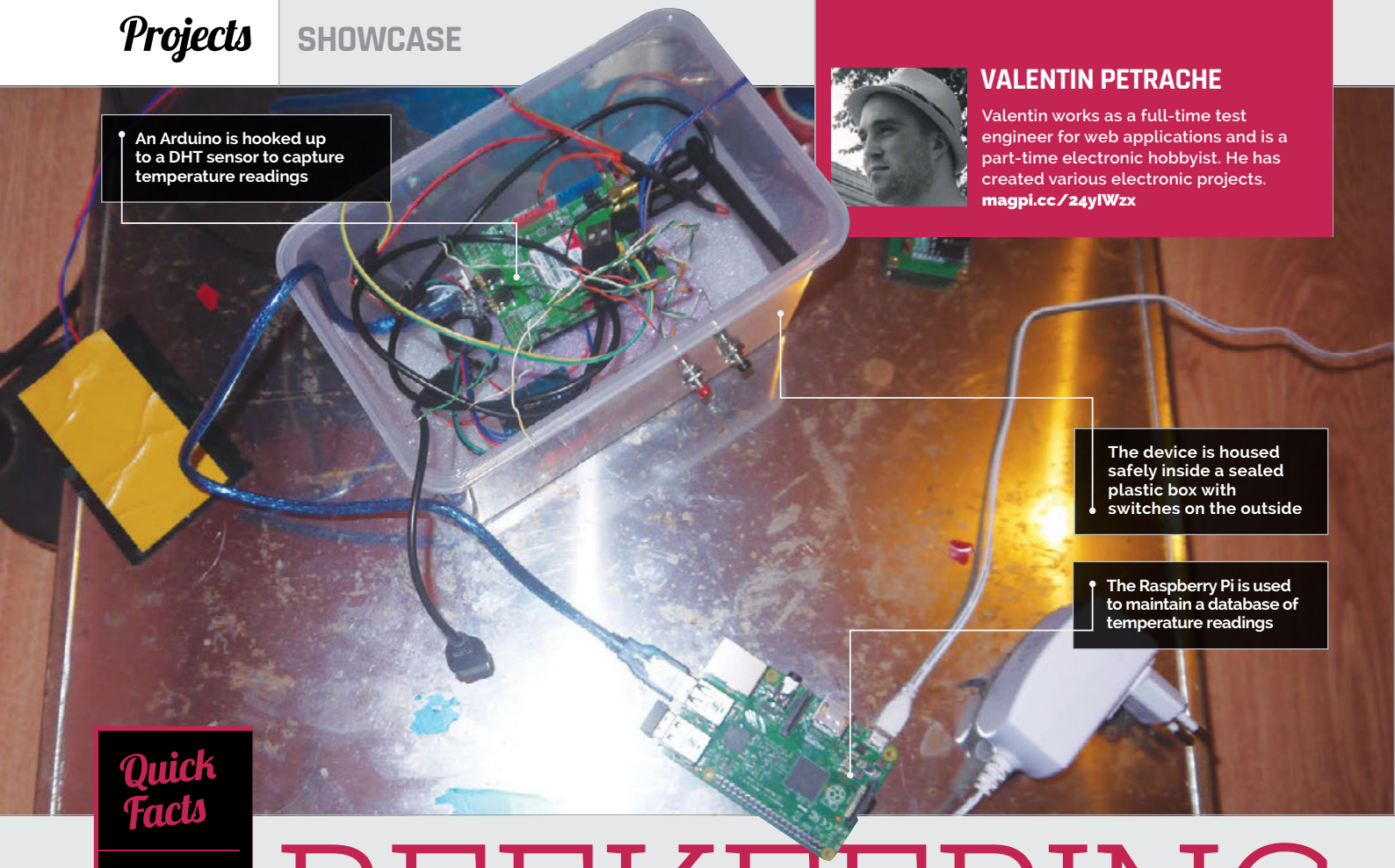
Speaker

The AlexaPhone speaks its response after you hang up the receiver. A cheap toy USB speaker is stripped apart and wired up inside the AlexaPhone.

>STEP-03

Alexa calling

The body of AlexaPhone is Dremeled out to fit all the components inside. The end result is a neat Trimphone that calls Alexa when you pick up the handset.



VALENTIN PETRACHE

Valentin works as a full-time test engineer for web applications and is a part-time electronic hobbyist. He has created various electronic projects.
magpi.cc/24yIWzx

Quick Facts

- Each hive has around 30,000 bees
- Valentin owns 30 hives in total
- It took about a month to build
- It measures the temperature outside the hive
- The battery lasts around three days

BEEKEEPING SERVER

Home-grown honey made with Raspberry Pi sounds tasty to us. Have a slice of this home-built beehive with smart scales and sensors

The Beehive Server lets Valentin know the ideal time to collect his honey

Bees are amazing! They build hives, waggle dance for each other, and make tasty honey, which we then steal and turn into Crunchie bars.

It's a good job bees make three times as much honey as they need, given how tasty it is. In summer months the bees are busy making honey, which they then use as food during the colder winter months.

Beekeepers aim to snaffle the extra honey at just the right time. One smart *MagPi* reader, Valentin Petrache, got rid of the guesswork and built a smarter beehive. The Beekeeping Server measures the weight and temperature of a hive. Valentin then uses the data to figure out the exact time to harvest his honey.

"When I first thought of the idea," says Valentin, "I made a checklist of important information on beekeeping, and how to harvest as much honey as possible.

"I measure the outside temperature and humidity," he tells us. "It's important to know when the bees are active, and to know the hive's weight.

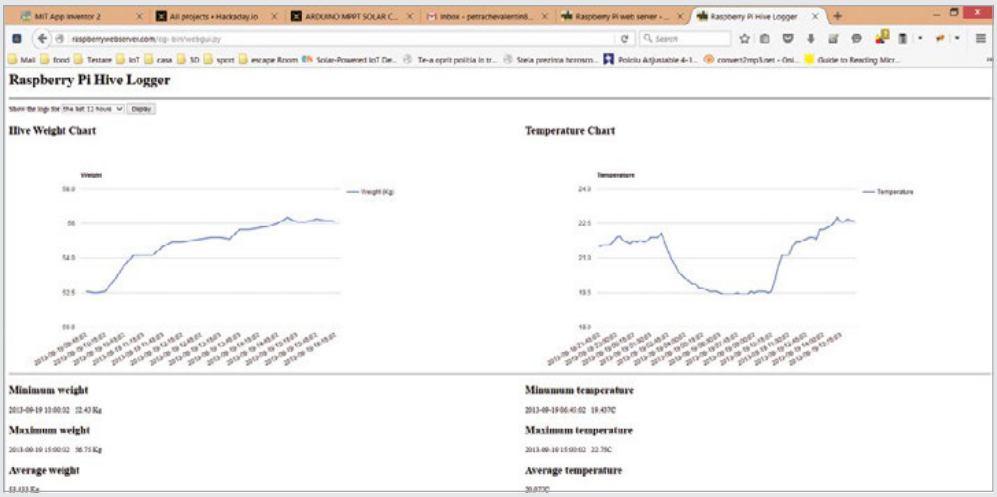
"In beekeeping, it's important to keep your hives close to a lawn of flowers where bees extract

pollen and produce honey. If the flowers don't have enough pollen, the bees start eating the honey from the hive. Therefore, you have to move the hives to a better area. So I thought, "What a great idea to have a weight scale under a hive."

The Beekeeping Server merges different devices and sensors. The project contains a Raspberry Pi, Arduino board, DHT23 sensor, HX-711 scale amplifier board, load scale (rated at 300kg), small LCD, WiFi dongle, and lithium polymer battery.

"The Arduino board is the core of the project," explains Valentin. "It reads the temperature, humidity and weight, and prints it in a human-friendly format. The HX711 amplifier sensor reads raw data from the load scale and sends it to Arduino in kilograms. The DHT sensor reads outdoor temperature and humidity, and the LCD [displays] weight, temperature, and humidity.

"The Raspberry Pi has the role of the server for wireless communication," he continues. An Apache server and SQL database are set up in Raspbian. The Raspberry Pi accesses the data from the Arduino and hosts a webpage displaying the results. Alternatively, the small LCD on the beehive provides data on-site.



Above The temperature and weight of the hive are accessed remotely, letting the owner know when the honey is ready to be collected

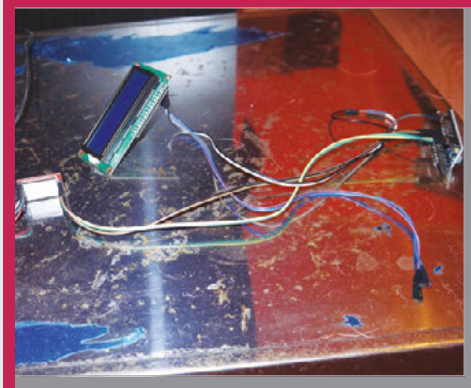
"It took about a month to build," says Valentin, "and of course there were problems." From library issues all the way to frying a board, Valentin has had his work cut out.

"At first, I had problems with getting a reading from the HX711 sensor. My scale had different colour codes for the four wires it has, and the HX711 was getting no data. Currently, I have problems with the weight I get. The scale is very sensitive to temperature variation. I'm currently trying to resolve that problem by creating a thermal barrier between the sensor and outside temperature."

Despite the gremlins, the Beekeeping Server is in constant use. Valentin is ironing out the issues and looking to add a solar panel soon (the battery needs changing every three days).

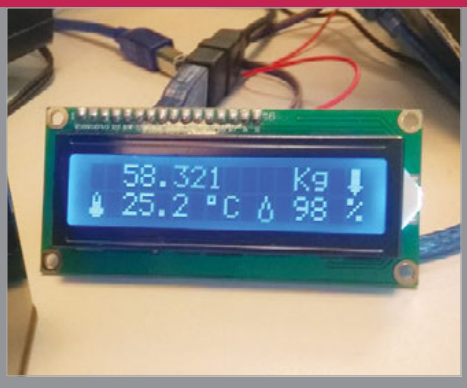
"It's fun building one," he tells us. "The process of creating something, and the information you learn during the build process, it's all very rewarding."

INSIDE THE HIVE



>STEP-01
Arduino and DHT sensor

A DHT (digital humidity and temperature) sensor is connected to the Arduino on port 10. An LCD is hooked to the Arduino, so Valentin can check the readings locally at the beehive.



>STEP-02
Box and buttons

A plastic tub is used as an enclosure to protect the parts from the bees. A button is connected to pin 12. Pressing it activates the backlight so the LCD can be read in low light conditions.

>STEP-03
Bee safe

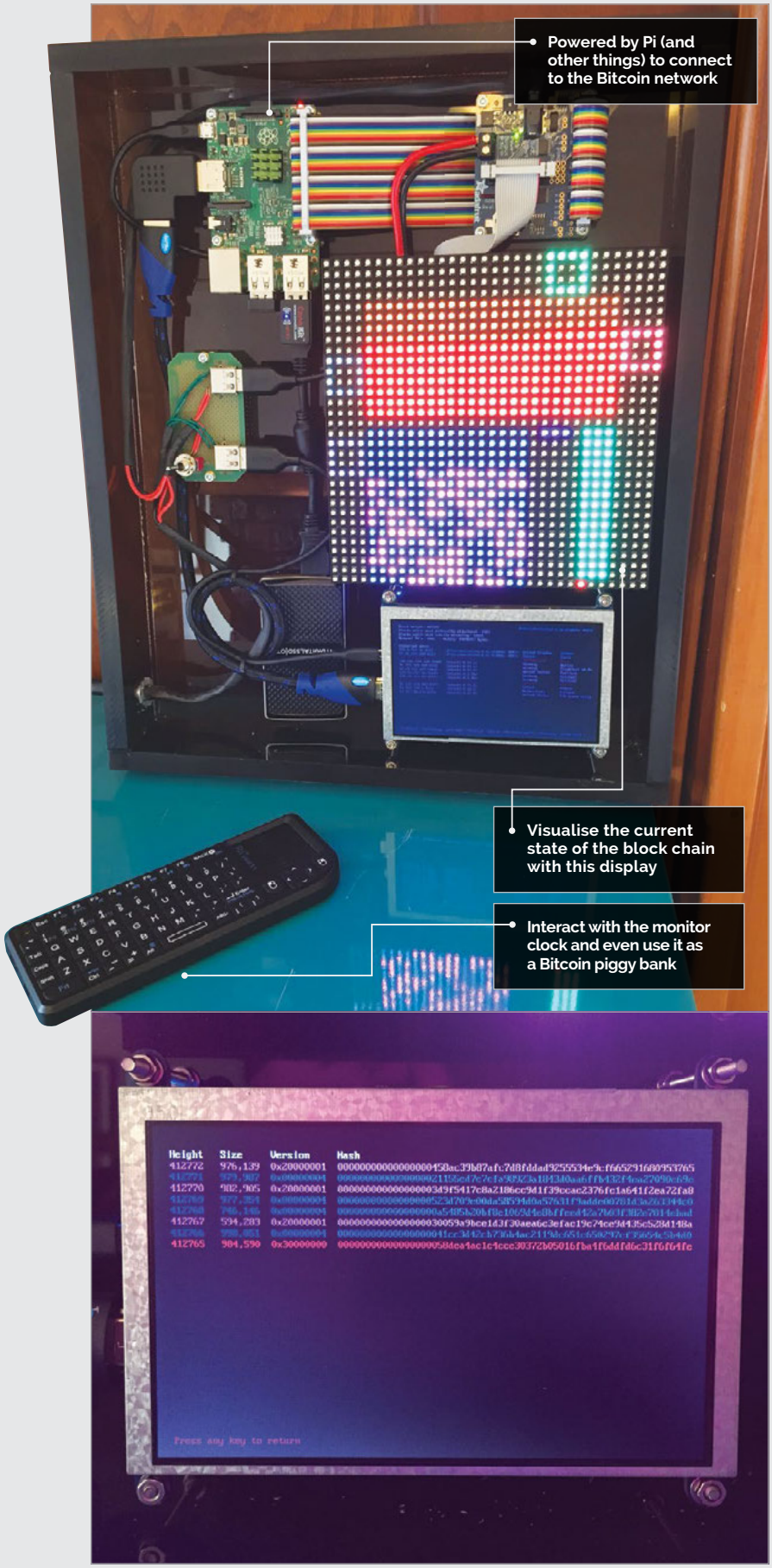
The Raspberry Pi and Arduino devices are placed inside the sealed box, with a switch used to turn it on and off. This unit is then connected to a weight scale, mounted at the bottom of the beehive.

MATTHEW ZIPKIN

A lifelong programmer, Matthew is a professional audio engineer working on TV, films, and music.
youtu.be/78u8EQtIXMY

BITCOIN BLOCK CLOCK

Visualise a Bitcoin node with a great light display that tells you how well the current block is going in your block chain



Left More information is provided on the screen below the display

Quick Facts

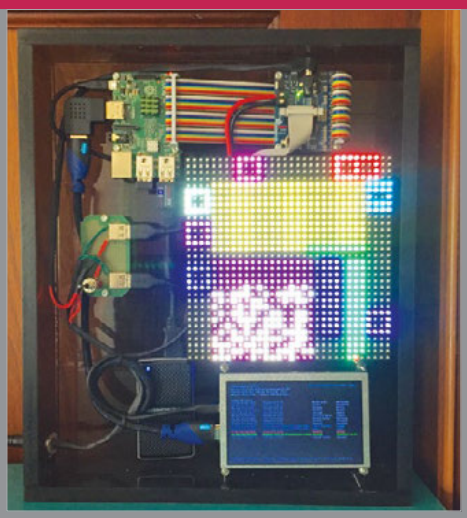
- The build took about four months
- A full list of components can be found here: **magpi.cc/1PnglXo**
- The enclosure build was fairly new ground for Matthew, but turned out well
- Matthew chose the Pi due to the strength of the community
- Matthew is currently building a second, better version

If you work or pay attention to the technology industry, you'll know how ubiquitous Bitcoin is. The digital currency is big in the tech world, and considering how much Bitcoin is currently worth, it can be very advantageous to help mine some of it – it's basically free money! Well, minus the expense of the electricity required to help with the mining. Raspberry Pis are quite popular for Bitcoin uses, but Matthew Zipkin has found a fairly novel way of connecting his Pi to Bitcoin.

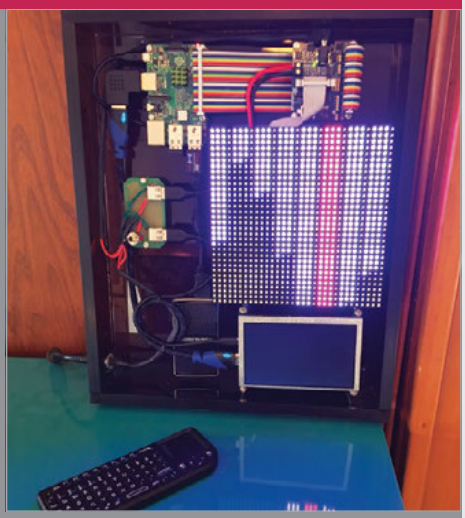
"The Bitcoin Block Clock is a Bitcoin full node connected to a 32×32 RGB LED grid, which visualises certain network properties in a fun, colourful clock-like display," Matthew explains. "So that's two things: a Bitcoin full node is a computer that runs the Bitcoin software. It connects to 12 peers on the network who also run the same software, and together as part of the ~7,000-node Bitcoin network establish the global consensus on the block chain, the ledger of all Bitcoin transactions. That software

HOW TO VISUALISE BITCOIN

>STEP-01
Start it up
Plugging in the system is only the first step, as you need to start the specific script as well to make sure everything is running.



>STEP-02
Clock display
You can just sit and watch the Bitcoin interactions, powered by a specific API and code that translates the data in real time for the display.



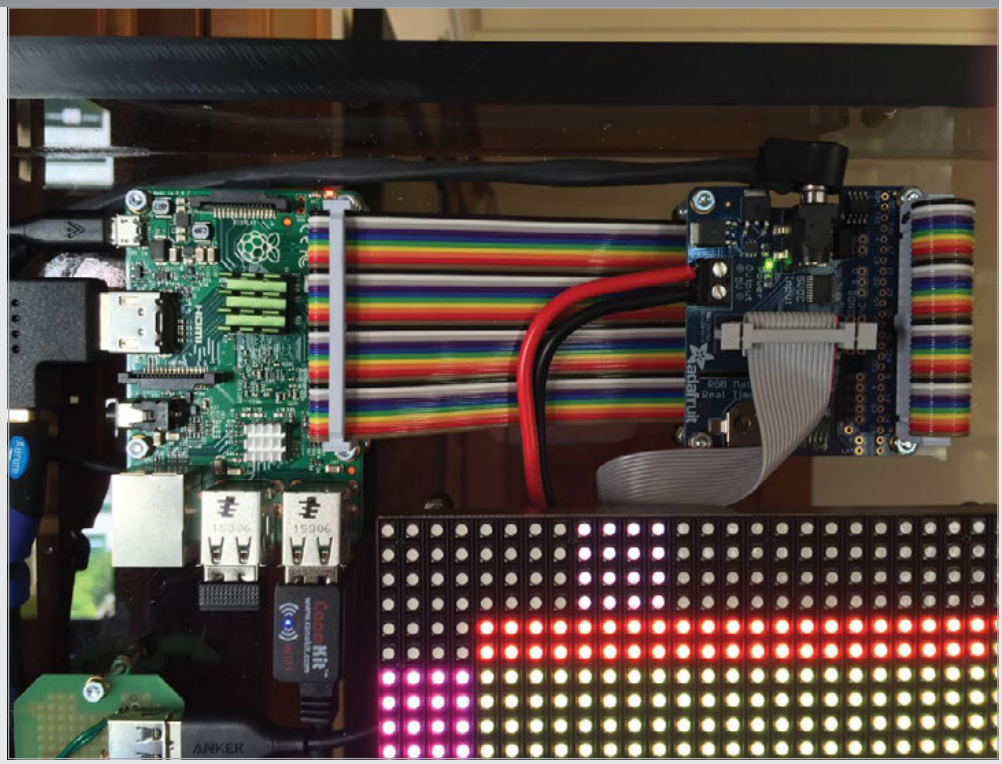
>STEP-03
Different displays
You can also change the displays and interact with the clock using a wireless keyboard. You can then deposit and withdraw money from it.

can also send and receive Bitcoin transactions, giving my project a 'piggy bank' functionality. You can send money to the clock!"

The LED grid lights up and shows a lot of data for the user, including current block progress, how difficult it is to mine Bitcoins, and how much you can mine. It may look a little complicated, but to those in the know it offers a lot of useful info.

"There are a few companies that sell compact Bitcoin full nodes to users and hobbyists," Matthew tells us as he explains why he made it. "But they're boring! They don't even have a screen. You just plug it into the wall like an air freshener and let it do its thing. You can SSH into them and do technical things, but they really provide very little utility to the user. There was one company that used to make them with a screen that at least showed some network statistics, and I wanted one, but they went out of business! So I decided to make my own."

The final product looks great and, according to Matthew, works well: "It's been running non-stop ever since I finished it. The SSD was a huge help. Lots of compact full nodes just use a USB flash drive, and that works OK just for the Bitcoin stuff. But the other functions I wanted my clock to do were so slow, and I discovered the bottleneck was the flash drive. It looks awesome and everyone who comes into my living room is mesmerised by it. It looks like some crazy alien clock! You don't need to know anything about Bitcoin or computers to appreciate that it's cool."



Above The whole system is open for all to see: no tricks here

INGMAR STAPEL

Ingmar has been building Pi-powered robot cars since 2012. He's also working on a security robot for the home.
custom-build-robots.com

DISCOVERER METAL-DETECTOR ROBOT

This smart Raspberry Pi robot is equipped with a metal detector, along with GPS tracking and a pan-and-tilt camera

Quick Facts

- The Discoverer took nearly six months to build
- There's a 350m range for remote control
- It uses a Navilock NL-602U GPS receiver
- The original chassis was a smaller IKEA box
- The pan-and-tilt kit features two mini-servos

We've seen all sorts of Pi-powered robots in our time, but the Discoverer is the first with a built-in metal detector. Mounted on a PVC pipe arm in front of the four-wheeled robot, the detector emits a beep whenever it passes over a metallic object. Prolific robot maker Ingmar Stapel, from Munich, came up with the idea after watching a TV show about people trying to find gold with a sophisticated metal detector. "I was immediately inspired to build my own affordable robot-car with a metal detector in order to discover some treasure in my garden," he tells us.

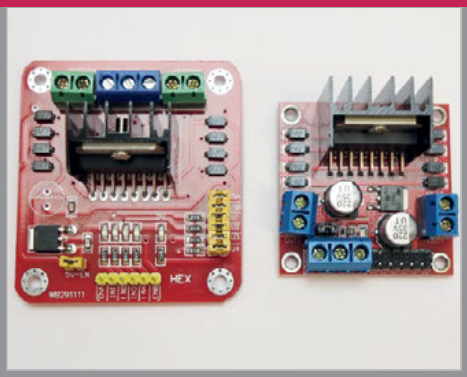
The basis for the new robot was his previous cardboard car, which required some major adaptations. "First, I had to look for a different chassis that is also suitable for outdoor usage. Second, I needed a metal detector that fits to the robot-car and the Raspberry Pi to ensure remote-controlled treasure hunting."

For the chassis, he used a plastic storage box to contain all the electronics, adding PVC piping around the exterior to hold a pan-and-tilt camera and the metal detector. At first, Ingmar tried using a cheap electric-cable detector from a DIY store, but its coil was too small to detect metal in the ground. He then came across the Gary's Pulse-AV metal detector (**magpi.cc/1XdKBeK**). "With support from Gary, I was able connect the metal detector to the Raspberry Pi and to get everything working." This involved using a step-down converter to change the detector's 12V output to 3.3V for a GPIO pin on the Pi.

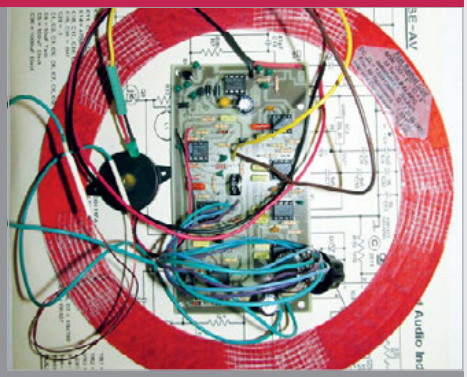
As Ingmar wanted the Discoverer to stream live video to a laptop, it would need a camera. After mounting a Pi Camera Module on the front of the chassis, he found the angle of view was too limited. "I bought the pan-and-tilt kit

BUILDING A METAL-DETECTING ROBOT

>STEP-01
Chassis and wheels
The chassis is a standard plastic storage box bought from a DIY store. Four motors are attached to the bottom of the box using wall mounts for PVC pipes.



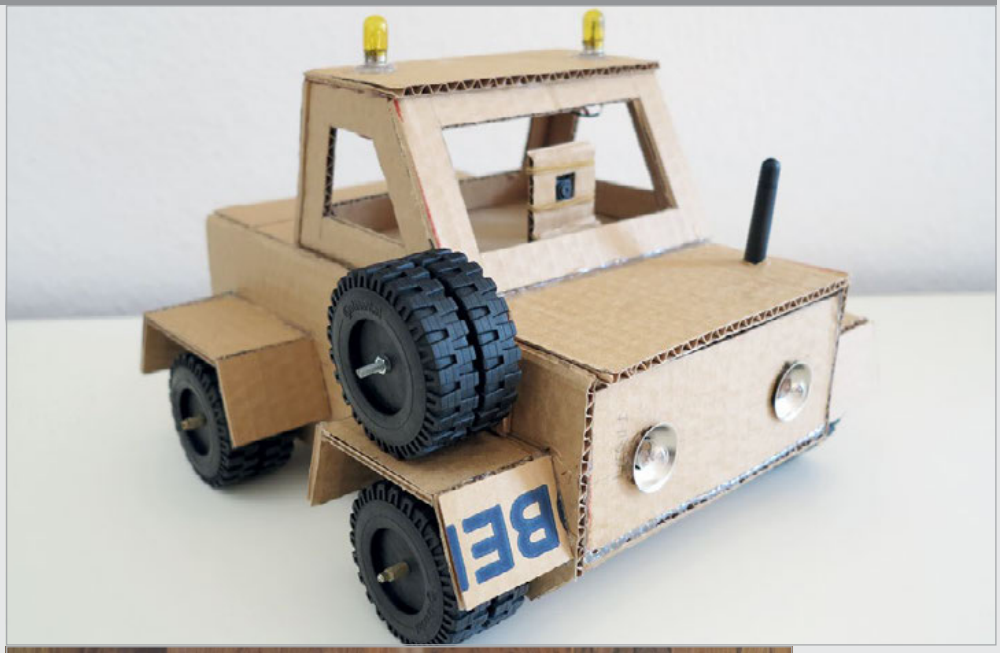
>STEP-02
Motor drivers
Standard L298N H-bridge motor controllers are used to trigger the DC motors. Ingmar plans to replace the latter with worm-gear motors and also use larger wheels.

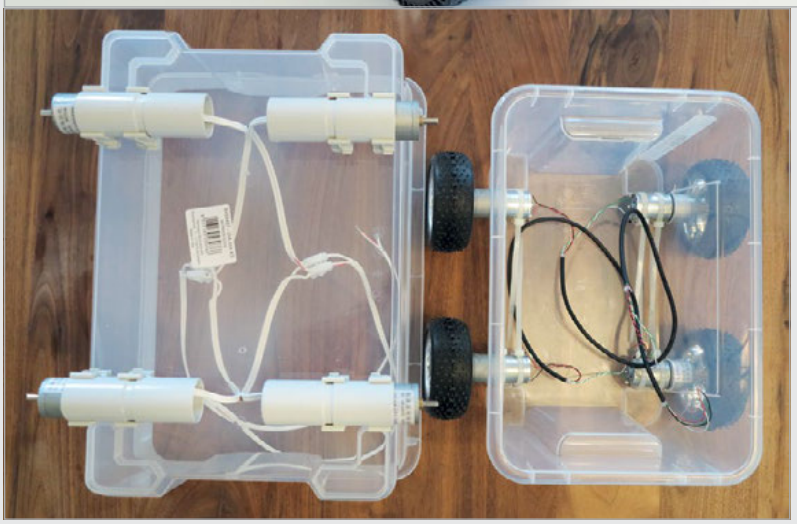


>STEP-03
Metal detector kit
A Gary's Pulse-AV PI detector kit (**magpi.cc/1XdKBeK**) is used for the metal detector part, which is mounted fairly low to the ground on a PVC pipe arm.

and mounted it in the middle of the robot and I got a much better overview." Some trial and error was involved in getting the live video streaming to work with very low latency, since even the tiniest delay would make remote control difficult. In the end, he used MJPG-streamer (**magpi.cc/24F96k6**) with some patches to get it to run on the Pi 3: "The video is very fast with a very low latency… it is even possible to make some robot-car races in our apartment."

While originally just remote-controlled, the Discoverer is now able to move autonomously using a GPS receiver, Sense HAT (for the compass), and a Python program Ingmar had already developed. "It is still in a beta version, but it is already able to import a KML file with GPS waypoints generated via Google Earth. Imported into my Python program, the robot-car is able to drive from one GPS waypoint to the next." When it's finalised, he plans to include a manual in his first book, due to be published (in German) this autumn. While the Discoverer is mainly for fun treasure hunting, Ingmar thinks it could also have some serious applications, such as detecting mines in war zones.





Above The Discoverer is loosely based on Ingmar's earlier cardboard car (**magpi.cc/1VQv8QJ**), which has live video streaming

Left Unable to fit all the components into the original IKEA plastic storage box, Ingmar replaced it with a larger one

10 AMAZING ARCADE MACHINE PROJECTS

Building your arcade machine is an amazing idea, and there are projects galore from other makers. Follow in the footsteps of these builds to have your own coin-op at home

The Raspberry Pi has long been the beating heart of many an arcade machine project, and the souped-up Raspberry Pi 3 emulates coin-ops better than ever.

So if you've ever schemed and dreamed of your own personal arcade machine, now's the time to put your plan into action.

You might be pondering what type of arcade machine you'd like to build: a full-sized cabinet, a bartop device, or a cool tabletop machine? Whatever you choose, the Raspberry Pi community has your back. There are designs and concepts for every conceivable shape and size of arcade machine. Some are incredibly detailed, others purely practical, and most look plain fun to build (and play).

Most arcade projects run RetroPie (**magpi.cc/Retro-Pie**) or PiPlay (**piplay.org**), although there are other options like Lakka (**lakka.tv**).

Building an arcade machine is an incredibly rewarding project, and no matter what type of machine you end up with, you're bound to enjoy using it and showing it off to your friends.

MATT SHAW

Matt Shaw, is a keen gamer who lives in Adelaide, Australia. He was formerly a boilermaker and welder, and has always had a penchant for making things.

imgur.com/a/wzua5

WINE BARREL ARCADE

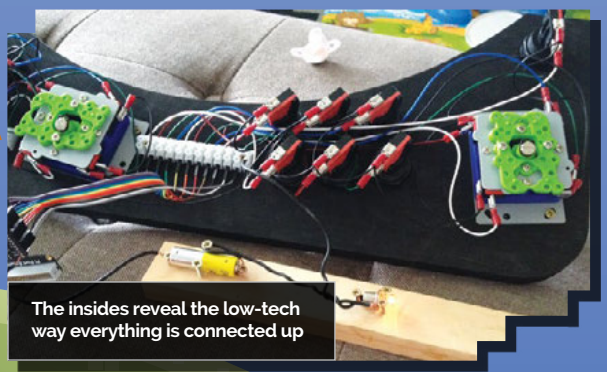
Drain a barrel of wine and turn it into a Donkey Kong-style arcade table

We adore this grand *Donkey Kong*-style barrel transformed into a tabletop arcade machine.

Built by Matt Smith, a gamer from Adelaide, Australia, the Wine Barrel Arcade (**magpi.cc/2gSiEto**) is a surprisingly low-tech project. There's no soldering, and it uses crimps and block connectors to hold everything in place. All you'd need is a ludicrously oversized barrel and space to store it.

Matt based the idea on an earlier project called A Barrel of Kong (**magpi.cc/1VwrzPK**), although that project featured a JAMMA board (to accept original arcade machine boards). "With a mind to do something like that, I asked for a wine barrel for Christmas," he says. Sensibly, Matt switched the JAMMA arcade board approach of that project to a Raspberry Pi running PiPlay (**piplay.org**).

With a wine barrel rolled into his workshop, Matt sourced an unloved 4:3 monitor and table glass mounted on non-slip rubber. The only real cost was for the buttons, joysticks, and wiring. These are all held together with an MDF panel, cut to fit using an angle grinder, circular saw, and jigsaw. According to Matt, the whole build came in at around £90.



The insides reveal the low-tech way everything is connected up

"The guys at the PiPlay forum were amazing, and still help me out (and anyone else that asks, it seems) today," Matt told Australian Kotaku (**kotaku.com.au**)."I love *Wonder Boy*, as it's such fun and a long game, but the *Street Fighter II* series [is] the best." Matt enjoys epic *Street Fighter* battles with his friends over beers.

It's visually impressive, but the best thing about the Wine Barrel Arcade machine is that despite the size, it's a relatively basic construction. We think this is one of the most impressive projects you can make – and if you've got the space, it's sure to impress.

YOU'LL NEED

- Wine barrel
- 4:3 TFT monitor
- Joystick and buttons
- MDF board

MICRO PI

This desktop cabinet is a fully functioning arcade machine with a Raspberry Pi inside

A unique approach to building an arcade machine is this Micro Pi, designed by Marco Tan. Roughly the size of a can of Coke, it's still a perfectly formed arcade machine.

"I always wanted a tiny arcade machine for my desk," explains Marco, "[so] I decided to build it as small as possible. It may look simple from the outside, but a lot of engineering went into making it. After months of development and eight printed prototypes later, it's finally ready."

The cabinet is 3D printed with nylon SLS (Selective Laser Sintering). "It needs to be printed with nylon SLS, or else it won't work: it won't be strong enough," says Marco. You can buy a case directly from Shapeways (**magpi.cc/1S4FqXV**).

The controls are a GH7455-ND mini joystick and nine 679-2431-ND tactile push buttons. Epoxy is used to hold everything in place, and there's quite a bit of soldering in the project. A speaker is recycled from an old MP3 player to provide audio, and a 2.5-inch TFT forms the display.

It's a fiddly project, but it uses widely available parts, and the case can be pre-bought. There are full details on Marco's Instructables page (**magpi.cc/1S4Fw1I**).

We think this is one of the cutest projects we've ever seen, and it's a superb way to creating a dinky arcade machine that's fully functional.

The Micro Pi packs a fully functioning arcade machine into a Coke can-sized cabinet

The NaCade is a neat tabletop arcade machine made from a clear acrylic sheet so you can view the insides

NACADE

Don't be shy! Show off your Raspberry Pi with pride inside this see-through arcade machine

"Who doesn't like gaming?" asks Krimmy, creator of the NaCade. "Having grown up playing arcade machines, as a kid you could only dream of owning one. Now, with advances in technology, gaming is available to everybody."

NaCade (**magpi.cc/1S4HSgV**) is a naked arcade machine case, displaying the Raspberry Pi powered innards in all their glory. The display is a 7-inch LCD monitor recycled from a car reversing system. "[The display] is sufficient enough for these low-resolution games," says Krimmy, "although the menus can be a bit too small to read. I had to use a larger monitor when I set it up.

"I used an arcade-quality joystick and illuminated buttons, which are wired directly into the Pi," he adds.

Just when you think the NaCade couldn't get any cooler, Krimmy drops the solar panel on you. "The solar controller regulates voltage from an external source, which in this case is a solar panel. Yes, it's powered by the sun!

"Sure, there [are] plenty of consoles and handheld units, even smartphones to choose from, but what I wanted was the nostalgic feel of a stand-up arcade without the need for a large room to put it in. The portability is pretty handy, too."

TOM ROLFE

Tom Rolfe is a writer for TapSmart and *Swipe* magazine.
tapsmart.com

GALACTIC STARCADE

With clear instructions and a solid parts list, this superb arcade machine is a rewarding build

YOU'LL NEED

- MDF board
- LCD TFT monitor
- Joystick and buttons
- Heatsink for Raspberry Pi
- Plexiglas and flexible LED strip kit

The Galactic Starcade is a great bartop project that keeps the weight (and cost) down while providing two sets of controls and a large 19-inch TFT monitor.

Built by Bristolian techie Tom Rolfe, the Starcade (**magpi.cc/1qOxaVh**) is our tip for a solid arcade cabinet with clear instructions. It's a challenging project, but you won't get lost during the build.

"I've always wanted an arcade machine for authentic retro gaming," reveals Tom, "but they take up a lot of space and cost a lot of money. Making a custom bartop cabinet like this one solves both of those problems. It also lets you play potentially thousands of games on a single machine. This project costs under £200 ($280) to make, whereas a pre-built custom cabinet can set you back four or five times that amount."

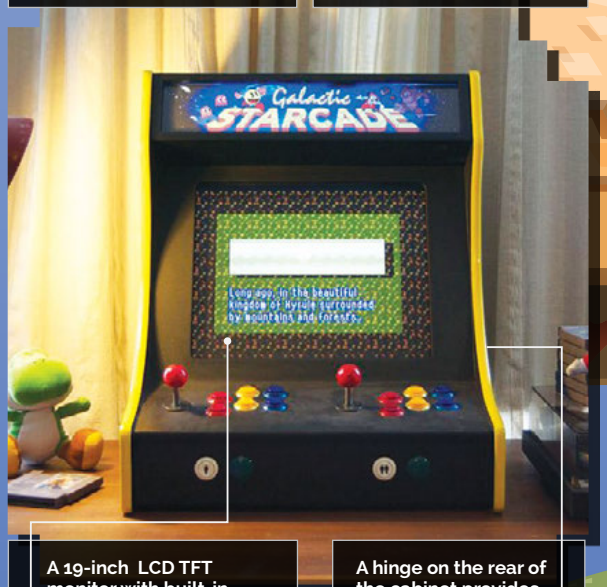
What we like most about the Galactic Starcade is the amount of detail in Tom's instructions. "I've knocked up full 1:1 scale printable guides for the side panels and the control deck, plus a reference sheet with dimensions and angles for the rest of the panels," says Tom. You'll find this project easier to follow than many others.

The cabinet is made from painted MDF, and the marquee is Plexiglas and a flexible LED strip kit. A Raspberry Pi is used along with a heatsink to keep things cool, and the controls are from Ultracabs (**magpi.cc/2gRjwOq**). It uses RetroPie as the software.

"There's a few things I would do differently, but overall I'm very happy with how this turned out," says Tom. "It proves that a little thing like the Raspberry Pi can happily power a near full-size arcade machine."

The cabinet is made from regular MDF (medium-density fibreboard) cut to shape and glued together

The controls are an all-in-one unit with a USB interface and pre-crimped wires



A 19-inch LCD TFT monitor with built-in speakers provides both video and audio output

A hinge on the rear of the cabinet provides quick access to the Raspberry Pi inside

The cabinet is a repurposed Ikea Lack table, which saves you the effort of making a cabinet from scratch

PIK3A

Some ideas are pure genius, like this Ikea coffee table turned into an arcade machine

YOU'LL NEED

- Ikea Lack coffee table
- Raspberry Pi and Arduino Leonardo
- Four-way ball-top joystick and buttons
- 17-inch LCD monitor
- USB computer speakers

We love getting ideas in Ikea, but clearly not as much as Element 14's community manager, Spanner Spencer, who had the genius to take apart an Ikea coffee table and turn it into an arcade table.

"It's an IKEA Lack coffee table with an LCD monitor cut into the top, arcade controls next to the monitor, and a Raspberry Pi 3 and accessories buried inside," explains Spanner.

He describes it as a "minimalist, contemporary interpretation of the classic coin-op cocktail cabinet that uses an IKEA coffee table and a Raspberry Pi 3."

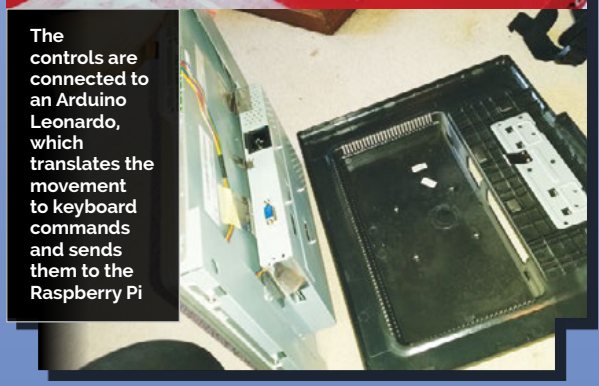
The display is an old 17-inch LCD monitor with a 4:3 ratio (this shape is better to match the square table). The chassis is taken out of the plastic casing, and the screen inside the shielding is the same depth as the Lack coffee table. "This means that once you've got the screen, all you need to do is drop it into the hole [you cut]," says Spanner.

SPANNER SPENCER

Spanner is the community manager for Element14, an online community for engineers.
magpi.cc/1q0Dxb9

A square in the table is cut to contain the 17-inch display. The wood is then prised out to reveal the structural filling inside

Holes drilled into the surface house the joystick and buttons. The controls are inserted from underneath



The controls are connected to an Arduino Leonardo, which translates the movement to keyboard commands and sends them to the Raspberry Pi

There are clear instructions for joysticks and buttons on Element 14's website (**magpi.cc/1q0xwLG**). "Drill 28mm holes for each one," says Spanner. "This is the standard size for arcade buttons, and also gives the joystick plenty of room to move without the hole being visible around the round, flat cover that comes with the joystick."

The PIK3A uses an Arduino Leonardo (**magpi.cc/1q0xyTQ**) to interface the controls with the Raspberry Pi. It's an interesting way to hook up the controls, and a lot easier than other implementations we've seen.

MINI ARCADE

Looking for the perfect scale arcade machine? Look no further

We featured Tiburcio de la Carcova's Galaga Pi project in *The MagPi* 44, but we couldn't do a feature on our favourite arcade projects and not mention the Mini Arcade.

Tiburcio's miniature arcade machine reproductions (of which *Galaga* is just the latest) remain the gold standard for scale reproductions (they even have small coin slots). Hand-built from plywood and acrylic, and with as many 3D-printed parts as possible, they are a labour of love. Reproduction is probably beyond all but the most dedicated of makers, but Tiburcio takes us through the build of *Galaga* on his YouTube channel (**magpi.cc/1V8XEvY**)

So far he's built perfect reproductions of *Space Invaders*, *Pac-Man*, and *Galaga*, with more to come. They're the most inspirational builds around.

ARCADE PI

This bare-bones arcade project is space-saving and great value

Arcade Pi is alone in our list of favourite arcade machines in that it isn't a complete build. Instead, it packs the Raspberry Pi inside a wooden base containing the arcade stick and buttons.

We think the Arcade Pi (**magpi.cc/1Q5gGw8**) is a great option for those looking to build an arcade machine, but not having the space, or money, for a full cabinet build. "I've always dreamed of having an arcade machine," says creator Sacha. "Man, it takes a lot of space, and it's expensive. I arrived at the conclusion that I had to build one myself."

This project enables you to experiment with arcade joysticks and buttons and build a working arcade console in a much smaller space, and for a much lower price.

Like all great arcade sticks, you should definitely customise the face plate yourself

LEGEND OF ZELDA BARTOP

There are lots of bartop projects around, but this *Legend Of Zelda* (**magpi.cc/1qOD3lx**) is one of the prettiest. "I've always wanted my arcade cabinet," says Phrazelle. "I'm glad I did because the final build exceeded my expectations by far."

COFFEE TABLE PI

Graham Gelding's Coffee Table Pi (**magpi.cc/1qODcFy**) is one of the neatest arcade tables around. It also has a huge 24-inch LCD screen and is sturdy and child-friendly (with rounded corners and Perspex mounted over the screen).

BUBBLE BOBBLE

We just adore Christopher Sadler's *Bubble Bobble* bartop arcade machine (**magpi.cc/1VwzpsM**), if for no other reason than its lovely decals. "All my plans were based around wanting to have artwork from my favourite game ever: *Bubble Bobble*," says Christopher.

TIM MAIER

Tim is currently studying for a Bachelor of Information Technology degree at Queensland University of Technology, with plans to major in Computer Science.
magpi.cc/29Burv1

Quick Facts

- The project cost $500 AUD (£298) to build
- It currently covers 2-3km per charge
- The board can currently reach up to 15km/h
- Tim plans a battery and ESC upgrade
- Build recipe is available on GitHub

MOTORISED SKATEBOARD

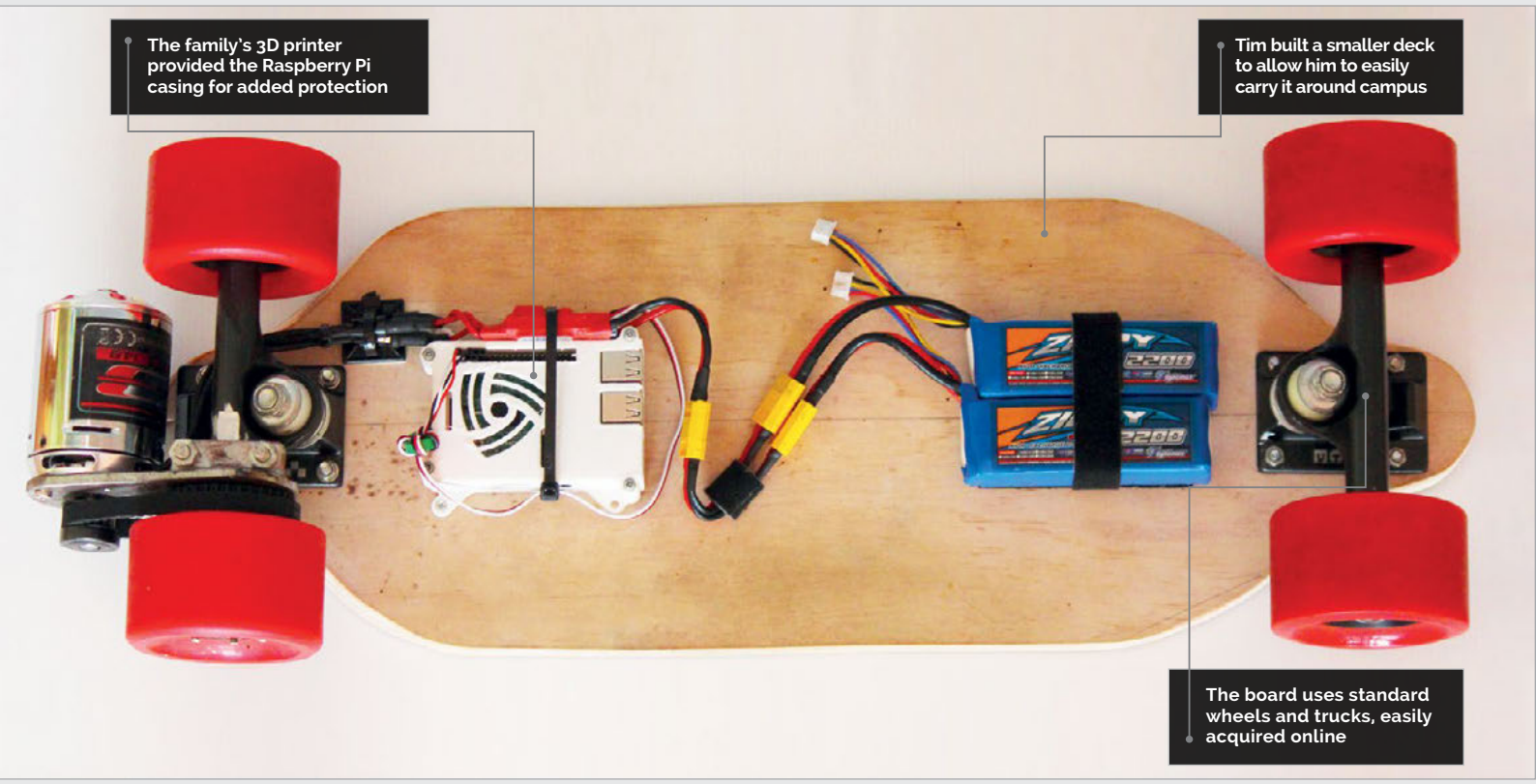
A university assignment allowed one student the chance to realise his dream of building the latest in whizzy commuter gadgets

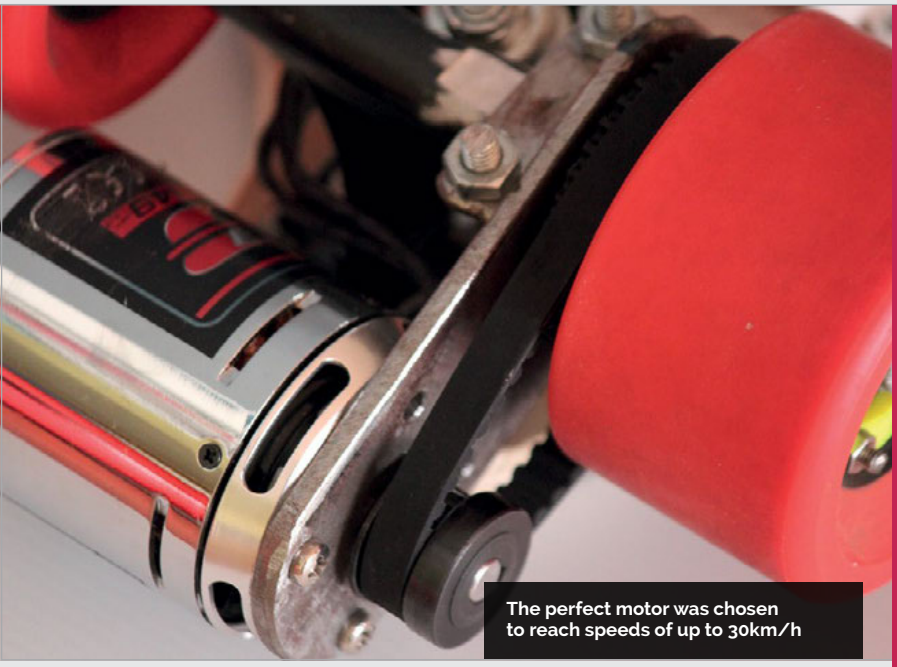
By now we're sure you're aware that e-boards have officially become a 'thing'. From knee-driven mini-Segways and two-wheeled 'hoverboards' to standard motorised decks, the streets are filled with wheeled commuters. And while Marty McFly may have failed to deliver on the true hoverboard of our dreams, search online for an electric skateboard and you'll find the next best thing, albeit with a hefty price tag.

So when Queensland University of Technology student Tim Maier was assigned with the task of 'building something with a Raspberry Pi', he already knew what he planned to create.

"Building an electric skateboard had been something on my mind for some time, as buying one was not a viable option," Tim explains, when we ask him if he had considered any other directions for his project. "So when we were told about the task, it all kind of linked up and I started to do my research on what to buy."

With a few requirements in mind, Tim started researching the perfect motor. He wanted to achieve an average speed of 30km/h to aid his commute, and knew the motor would easily





The perfect motor was chosen to reach speeds of up to 30km/h

be one of the most expensive components of the build. Finally, he decided upon a Turnigy Aerodrive SK3, matching it with two 2200mAh lithium polymer (LiPo) batteries and a basic electronic speed control (ESC).

Despite having to rely on YouTube and assorted literature to educate him on how to utilise Python, the biggest hurdle for Tim turned out to be the drive system. "Finding a way to attach the motor mount to the skateboard truck was a huge fiddle." He ended up creating a makeshift U-bolt system, though he plans to upgrade the mounting layout when attaching a new ESC.

> "The motor itself is controlled by a Pi and Wii Remote (a Wiimote to those in the know)"

The motor itself is controlled by a Pi and Wii Remote (a Wiimote to those in the know). Holding the '1' and '2' buttons will connect the Wiimote to the Pi. The 'B' button activates the motor, while 'up' and 'down' on the D-pad control speed.

Upon completing the build, Tim has been met with thousands of YouTube views and calls for how-to guides and board sales. Spurred by the positive response, he's provided the code and kit list on GitHub (**magpi.cc/29Burv1**), and plans to also create an instructional video of an upgraded design for anyone wanting to make their own. The Pi Skate 2.0 will house more batteries for longevity, a higher-quality ESC for greater speed and the ability to brake, and possibly LED lights because, well, why not? And as for the Wiimote? Tim hopes to move the board's control system to a mobile phone or smartwatch, thereby reducing the bulk of the console controller.

Of his future in the digital making industry, Tim suggests, "I will most probably venture into a career within the computing and engineering field, but haven't really thought too much about what area I will specialise in. I'm keeping my options open as it's such a broad field." In the meantime, his continued experimentation with Raspberry Pi will lead to further YouTube videos as and when he builds projects.

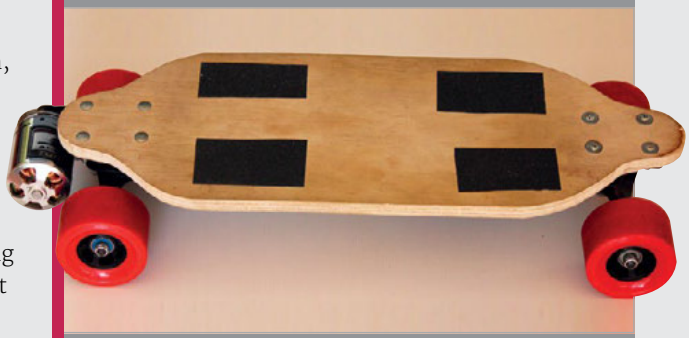
TAKE TO THE STREETS

>STEP-01
The power

Two LiPo batteries are connected in parallel to the ESC, taking approximately an hour each to charge. They power both the motors and the Raspberry Pi.

>STEP-02
The control

The ESC runs 5V power to the GPIO pins, which powers the Raspberry Pi. This is where control from the Wiimote converts to a pulse-width modulation signal, sending it back to the ESC.



>STEP-03
The launch

The ESC then signals to the motor to go-go-go, and you're on your way. Make sure to keep balanced as you're zooming along!

CORY GUYNN

Cory Guynn is the creator of the Internet of LEGO and is a systems engineer at Cisco Meraki. He has a degree in Computer Electrical Engineering & Technology from Purdue University.
InternetOfLEGO.com

Quick Facts

- Around 20,000 LEGO bricks were used
- It took one-and-a-half years to build
- Over 46,000 people have interacted with the blog
- 17 LEGO kits were used
- Most LEGO kits were from the Creator Expert series

THE INTERNET OF LEGO

One maker is taking a love of LEGO to a whole new level with Raspberry Pi. Take a tour of this incredible internet-connected cityscape being built brick by brick

We love LEGO and the internet, so what could be finer than this 'Internet of LEGO' project? Well, discovering the Raspberry Pi serving as its brain, and catching up with its maker to learn all about this amazing connected city.

"The Internet of LEGO is a living project where I set out to learn everything about the Internet of Things," says Cory Guynn. Packed with sensors, the city reports to a Raspberry Pi that acts as its brain. Cory has used this to create the Internet of LEGO blog (**internetoflego.com**).

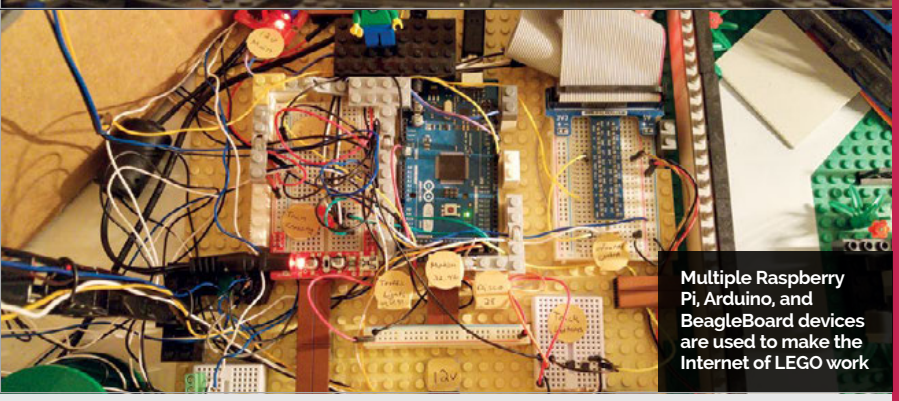
"I grew up playing with LEGO bricks and model trains, which taught me about construction and electronics, and allowed me to be creative. The use of LEGO also allows me to represent a city or build prototype systems easily. Plus, it gives me an excuse to buy a bunch of LEGO bricks in my thirties," laughs Cory.

"A Raspberry Pi Model B+ is the heart of my city," he reveals, "and that was the starting point of the project." The Raspberry Pi is attached to Arduino boards that control most of the GPIO operations. Cory also uses Cactus Micro Rev2, BlueDuino, WeMos and NodeMCU boards, along with a Wio Link and BeagleBone Green.

The city itself is complex, with many structures and buildings hooked up to a huge range of sensors: RFID, ultrasonic proximity, infrared, and magnetic reed switches are used to keep track of the city environment.

The train system is Cory's favourite part of the city. "I love seeing things in motion," he reveals. "There are several things that I've been able to do that make for a dynamic environment." Cory has built a train scheduling

We love the detail of the Internet of LEGO city, with citizens going about their daily lives



Multiple Raspberry Pi, Arduino, and BeagleBoard devices are used to make the Internet of LEGO work

system using the Transport for London API. This system displays the schedule on an OLED screen and switches to the train track to match the destination.

The trains are controlled by WiFi and an infrared transmitter attached to a tower. Infrared sensors are used to detect incoming trains and trigger a crossing signal (with a servo controlling the arm, and LEDs for lights).

"Everything is connected to an Arduino Mega, which is then USB-tethered to the Raspberry Pi," explains Cory.

He has developed a huge amount of software to control all the structures in the Internet of LEGO. "The Johnny-five.io robotics framework made programming sensors and servos easy. I wrote a few projects, including a train crossing, train track switch, elevator, motion detector, and city lights."

Most of the orchestration of the city and its many sensors is handled by Node-RED. "This system allows me to add inputs and outputs from any of my projects that can interact with one another," says Cory. "For example, when somebody visits my blog, a REST call to Node-RED triggers an MQTT message to my disco lights, which turns on strobing LEDs in a LEGO club and also triggers the Palace Cinema marquee lights for five seconds."

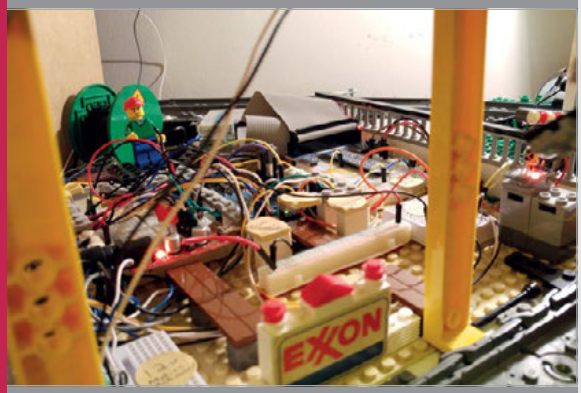
> "Cory has developed a huge amount of software to control all the structures in the Internet of LEGO"

Cory tells us he is indebted to the open-source community, which has allowed him to learn so much about programming and hardware. "I hope to give back to the community through my projects and experience along the way."

CITY PLANNING

>STEP-01
Regular LEGO

The Internet of LEGO is a Raspberry Pi-powered city built from regular LEGO bricks and kits. These are then given Internet of Things-style smarts, thanks to various electronic components.



>STEP-02
Raspberry and Arduino

A Raspberry Pi controls two Arduino boards using an MQTT broker. The Arduinos are assigned to GPIO operations, while a second Pi acts as an MQTT cluster member and runs Node.js automation scripts.

>STEP-03
A working city

The Internet of LEGO is fully automated, with reed sensors used to detect train presence. Timetable information is based on London API data. When people visit the blog, the Palace Cinema lights up and the disco starts working.

DJORDJE UNGAR

Djordje Ungar is a programmer by day and a digital alchemist by night. He is a hobby artist, animator, musician, game maker, hacker, and tinkerer.
djordjeungar.com

HAL 9000

"Open the pod bay doors, HAL" are chilling words to anybody who has watched *2001: A Space Odyssey*, apart from one maker who decided it would be a good idea to build HAL 9000 for real

Quick Facts

- HAL stands for 'Heuristically programmed ALgorithmic computer'
- HAL is rumoured to be IBM with each letter shifted one backward
- The total cost for the build was $99 (including the Raspberry Pi)
- All the parts are off-the-shelf computer components
- The case is laser-cut acrylic covered in black paint

The case is made from 3mm black acrylic laser-cut into the right shape for HAL. It is spray-painted for a professional finish

A stripped-down web camera is fitted with a camera lens. This completes the HAL 9000 look, and the web camera also provides the device with a microphone

A USB speaker is fitted inside the device near the grille at the bottom. Jasper software provides HAL with a voice

"I had this Raspberry PI Model B waiting to become something great," says Djordje Ungar, "and what greater thing can a computer hope for than to become the iconic computer from Stanley Kubrick's *2001: A Space Odyssey*? I mean, come on!

"The first time I heard synthesized speech, I thought to myself: 'Wow, how cool would it be if that was a voice of HAL 9000?' It was when I stumbled upon Jasper, this amazing open-source project that allows you to control a computer with your voice, that I knew I had to make HAL."

Aside from the Raspberry Pi, all the other parts are off-the-shelf computer components that you can buy online. "I used a 3mm thick black acrylic, and I've painted some parts to look metallic," Djordje reveals. "The box is 300 × 96 × 62mm, which is a bit larger than a tall carton of milk.

"I examined a dozen movie stills from the film," he continues. "I was only able to guess the actual size of the original HAL, so I based it off the lens. Some things like the number of holes on the speaker panel and the logo are spot on."

To allow HAL 9000 to see, Djordje added a webcam and camera lens. "I wanted to find a super convex lens like the one from the movie, but lenses of that calibre are anything but cheap. Even the used ones were way too expensive. So I had to settle for

A bargain camera lens is fitted inside the laser-cut acrylic case to form HAL 9000's famous single eye. Red LEDs complete the effect

a cheaper, non-convex lens." Luckily, he found a used one that he liked, via local classified advertisements: a Titanium Super Wide Lens 0.42x AF.

"The lens is mostly for style, to give the whole build a more polished look, but it does also expand the field of view of the webcam a little.

the case," he tells us. The brain is the Raspberry Pi running the Jasper project (**magpi.cc/29iKyxS**).

"The Jasper project offers a couple of TTS (Text-To-Speech) engines," explains Djordje. "I had a couple of options, but I chose eSpeak (**magpi.cc/29iKtdt**) since I'm most familiar with it."

> "It works as a voice assistant, but it can play any WAV files, including the classic lines from the movie"

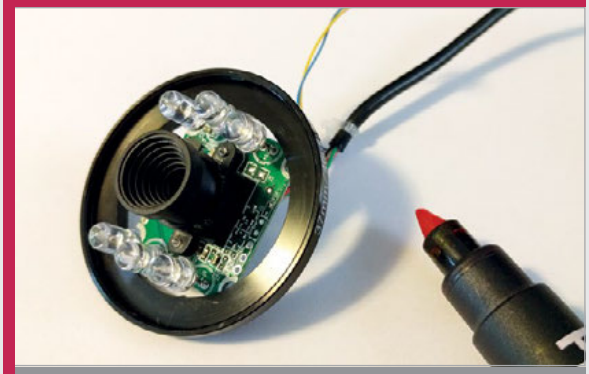
"The truth is: I didn't need the webcam. I only needed the microphone. But since they come bundled together, having a webcam was a plus. I removed the casing and left the circuit board exposed, to make it as compact as possible. Then I glued it onto the ring that goes on the back of the lens."

For HAL 9000's voice and brain, Djordje fitted a speaker and a Raspberry Pi inside the case. "My only criterion for the speaker was that it should be small and fit inside

It works as a voice assistant, but it can play any WAV files, including the classic lines from the movie. "I wrote a simple Jasper module that will play random WAV files."

Djordje's friends are impressed by the project: "In a world where you can talk to your phone's digital assistants, or ask Google a question with your voice, no one is blown away by the fact that HAL responds to voice commands, but everyone loved the case and wanted one."

BUILDING A HAL 9000



>STEP-01
Stripping the webcam

A recycled Insten USB digital six-LED webcam is stripped down. A red marker is used to paint the LEDs red, giving the device's eye the same ominous glow HAL has in the movie.

>STEP-02
Stuffing the case

The case is laser-cut from an acrylic sheet and glued together. The lens is fitted through a round hole in the front face and the webcam fitted inside. This provides HAL 9000 with a camera and, more importantly, a microphone.

>STEP-03
Assembled HAL

All the components are fitted inside, and the Raspberry Pi provides the software for human interaction. A speaker is fitted to the bottom of the device to provide the voice.

ZERO360

Take 360-degree panoramas with some clever Pi Camera Module placement and programming

JAMES MITCHELL

James is a software quality assurance engineer based in Berlin. He also organises the Raspberry Jam Berlin.
magpi.cc/2bgxXri

Quick Facts

- There are eight Pi Zeros and cameras
- The build took a few months
- It currently only sees 52 degrees of vertical space
- The Pi 3s actually power the Pi Zeros
- James has also taken pictures of the moon with a Pi camera

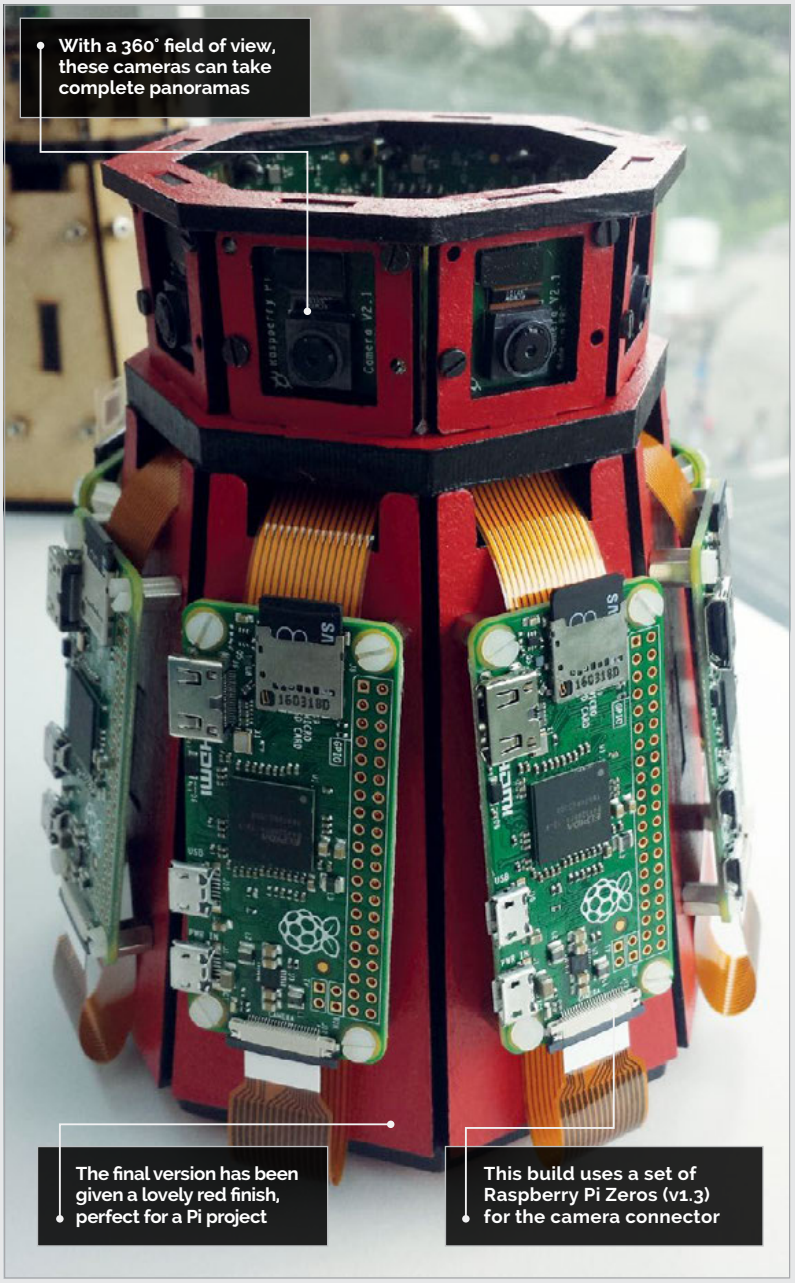
There's always some new visual technology trying to break into the mainstream, whether it's to try to improve the way we experience things or make a bit of money. The quality, however, varies wildly. At the moment, we're entering a new age of virtual reality (VR); this has created an interesting new set of visual experiences that has inspired James Mitchell.

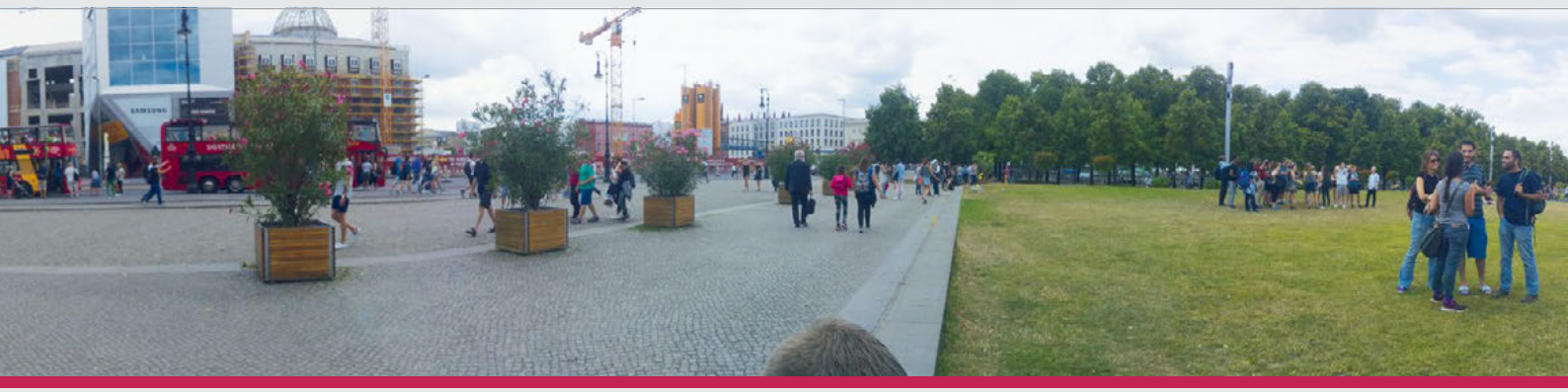
"Recently, there has been a rush of 360-degree VR videos online," James tells us. "They're really impressive. Loving the technical side of photography and the Raspberry Pi, it seemed only logical that I would try and build something that would allow me to recreate those videos using the Raspberry Pi."

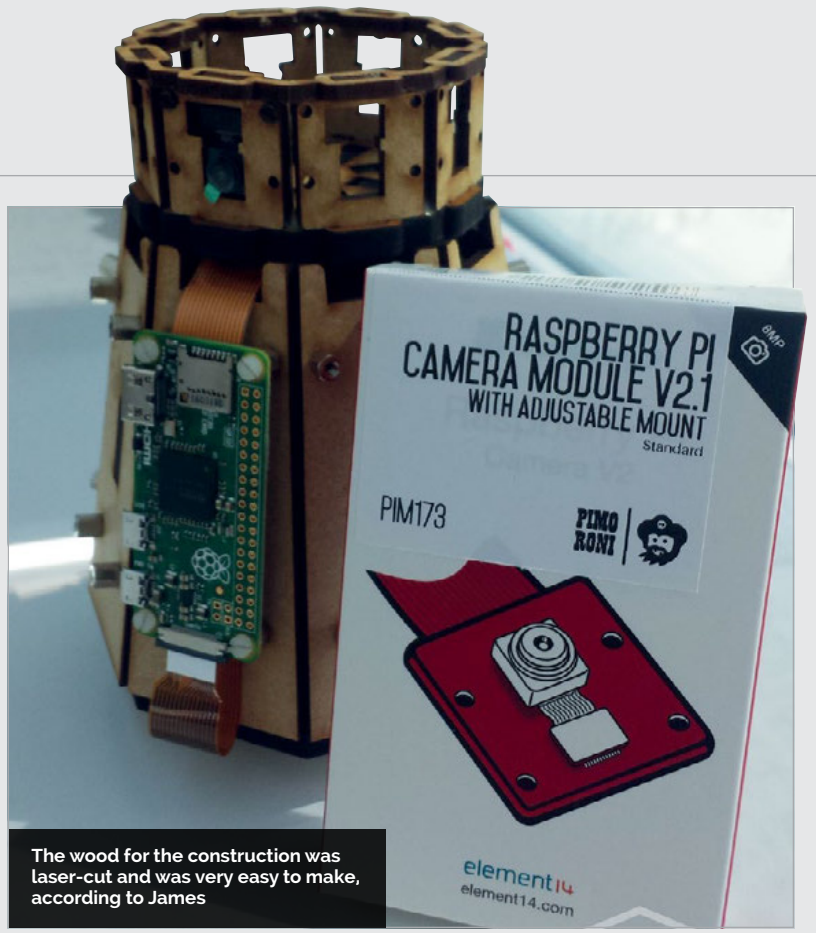
And so he did with the Zero360: a bank of Raspberry Pi Camera Modules arranged in a circle, connected to Pi Zeros. They can all take a photo at once; these are then stitched together to make a 360-degree panorama.

Why make it out of Pi Zeros, though? James explains that cost was a big factor:

"The issue is that the equipment for making 360-degree videos is extremely expensive. Using the Raspberry Pi, it's a fraction of the







The wood for the construction was laser-cut and was very easy to make, according to James

cost. You could argue that the Zero360 is not really that cheap when you could use your mobile phone or even a DSLR camera, but those would only take a single still image and need a user to move the camera around, whereas the Zero360 can take stills from all angles at the same time and repeatedly. Those stills can be made into a time-lapse. Also, video is an option! These features don't normally come that cheap!"

The housing for the system was quick to make, once James had managed to procure enough Raspberry Pi Zeros; however, the code took a few weeks on and off to get working. Two Raspberry Pi 3s are also used in the project to stitch the image together, and the build is otherwise just made up of Pi Zeros, Camera Modules, and power cables. "I'm using Raspbian Lite on all the Pis, with the raspistill and picamera Python libraries," James explains. "I also managed to stitch the images on the Pi 3 using Hugin."

Aside from some issues with getting the networking going, the whole project is pretty straightforward.

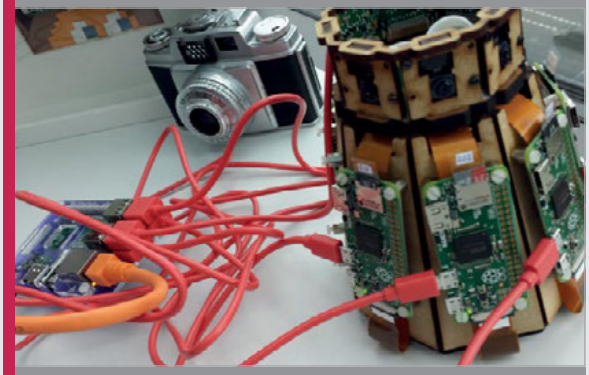
"Code-wise, there's still a lot of work to do, so I can't claim it's doing what it does efficiently," admits James. "But the final results are amazing! It's especially cool that the images are stitched together on the Pi itself!"

James has plenty of plans to improve the Zero360 in the future, so it can make even better panoramas.

MAKING A PANORAMA

>STEP-01
Relay the command

The setup has the Raspberry Pi 3s command the Pi Zeros to take their photos, rather than controlling them directly from a separate computer.



>STEP-02
Gather the photos

The photos from each individual Pi Zero are then sent over the network to one of the connected Pi 3s, rather than both of them.

>STEP-03
Stitch in time

Hugin is used on the Raspberry Pi 3 to stitch all the images together. The Pi 3 is chosen for this as it has a bit more power than the Pi Zeros.

RUSSELL GROKETT

Retired programmer/engineer Russell belongs to astronomy, amateur radio, Linux, and cloud computing clubs. In his spare time he travels worldwide, and is also an amateur caver and scuba diver.
grokett.org

EARTHQUAKE PI

This clever box of tricks rumbles to alert you to earthquakes anywhere in the world

Quick Facts

- Earthquake Pi took three days to build and program
- It took longer to write the documentation!
- The device checks for quakes every 15 minutes
- An electric toothbrush motor provides the rattle
- Earthquake sounds are played through a speaker

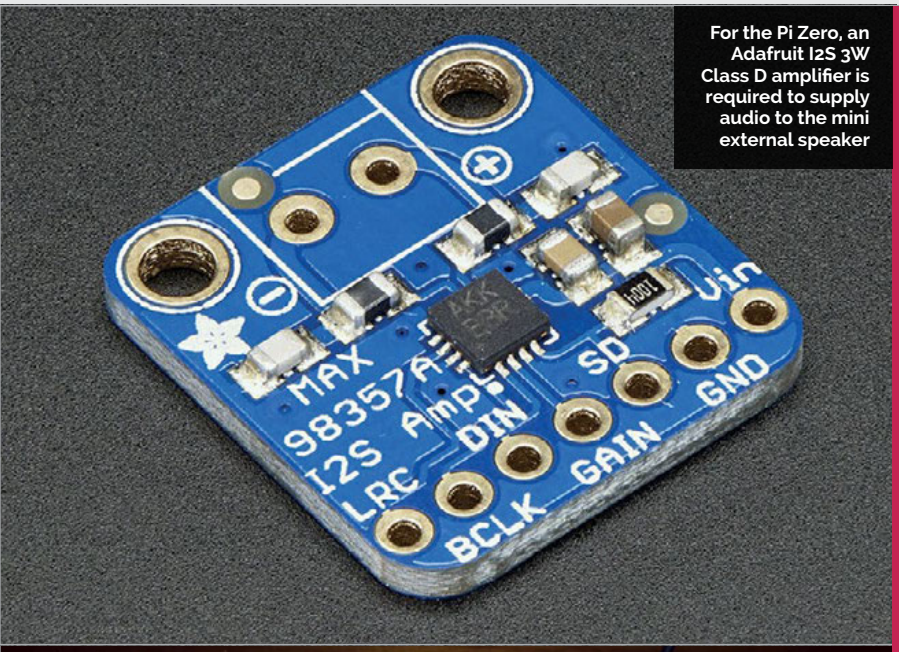
Russell Grokett has been fascinated by earthquakes and geology ever since he was a child, when his father built him a simple swinging beam seismograph. However, since Russell now lives in Florida, known for hurricanes but not quakes, he's created the Earthquake Pi (**magpi.cc/2aPNa62**) to satisfy his interest. Rather than acting as a detector of local tremors, like some Pi-powered projects, it's a neat alert system that uses real-time open data from the United States Geological Survey (USGS) to detect earthquakes around the globe.

"I had seen fancy maps and graphs of their data," explains Russell, "[but] I wanted to 'feel' (safely!) when an earthquake occurs. So I came up with the idea of taking their data and building a device that rattles and rumbles when an earthquake occurs. This is different from the typical detection on a chart or graph."

The Earthquake Pi comprises a wooden box containing the electrical components, including a Raspberry Pi Zero and a vibrating motor recycled from an old battery toothbrush to make the box rattle during an alert: "I found that just loosely taping the motor down worked best, as it bounces around a bit while running." To complete the effect, an external speaker plays earthquake sounds, while a strip of NeoPixels light up and an LCD display shows details of the seismic event.

"By default, the vibrating motor alerts run for a few seconds per magnitude: about two seconds for mag. 1 and up to about ten seconds for a mag. 9 (never heard that, luckily!)." The LCD display and NeoPixel bar graph then come on, displaying the quake location and magnitude. Lastly, the earthquake audio sound effect plays for a few seconds more. "You just set the box on your desk or table where is sits quietly... until boom! The first few times it goes off will probably scare you, as it's completely unpredictable!"

Russell's Python program includes a variable that can be set to the minimum magnitude



For the Pi Zero, an Adafruit I2S 3W Class D amplifier is required to supply audio to the mini external speaker

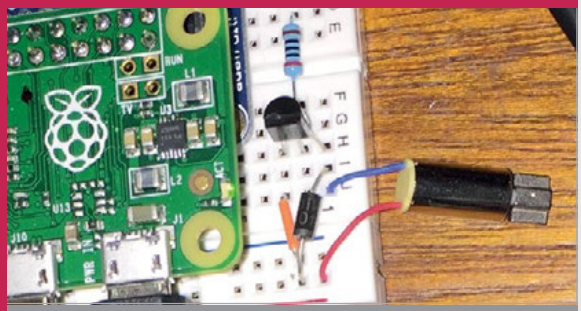
The Earthquake Pi – without optional NeoPixel bar graph – showing an alert for a small quake in California

for alerts. "If you set it to alert on even the smallest (magnitude 1.0 or greater) earthquakes, then it will be going off almost every hour or so." He tells us his is set to magnitude 3.0 and higher and goes off a few times per day. "When I hear it rattle, I run in to see where the earthquake is located. I especially perk up if I hear it rattle for many seconds, as that means a big one occurred somewhere. After watching it for a while, you do see a pattern of where many earthquakes are occurring. It's also a good geography lesson, as I've never heard of many of the cities or islands where they are, and so I look them up on a map."

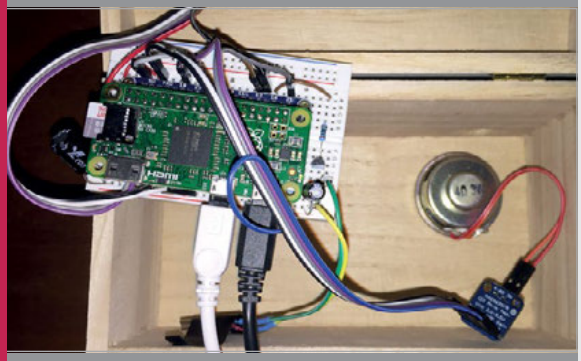
> "The first few times it goes off will probably scare you, as it's completely unpredictable!"

To ensure the Earthquake Pi doesn't wake him during the night, Russell uses a cron job to only run the program between 8am and 11pm: "You DO NOT want to run it while trying to sleep. It would probably scare everyone in the house!"

BUILDING AN EARTHQUAKE ALERT SYSTEM



>STEP-01
Vibrating motor
Taken from an old electric toothbrush, the vibrating motor is connected to the Pi Zero via a breadboard circuit, including a transistor and rectifier diode to limit the current.



>STEP-02
Internal connections
Inside the wooden box, the Pi Zero's GPIO pins are wired up to various components, including a vibrating motor, audio speaker, and LCD display (on the lid), via a cobbler kit and breadboard.

>STEP-03
LCD display
A test script is used to check the LCD display is working correctly. The Earthquake Pi requires a 20×4 screen to show all the details of the earthquake during each alert.

JONATHAN TYLER-MOORE

When he's not building awesome projects, Jonathan is a dab hand at photography, tweeting like a pro, and winning things.
twitter.com/piboyuk

THE TABLET OCARINA PROJECT

Ocarina players Robert and James sought the help of teenager Jonathan to build an interactive touch tablet for reading music

Quick Facts

- The Rebel Makers Club runs once a month
- Jonathan coded with the Adafruit Python MPR121 library
- Jonathan used his mobile phone to record notes
- The project was showcased in July
- The Blue Flash Music Trust backed the project

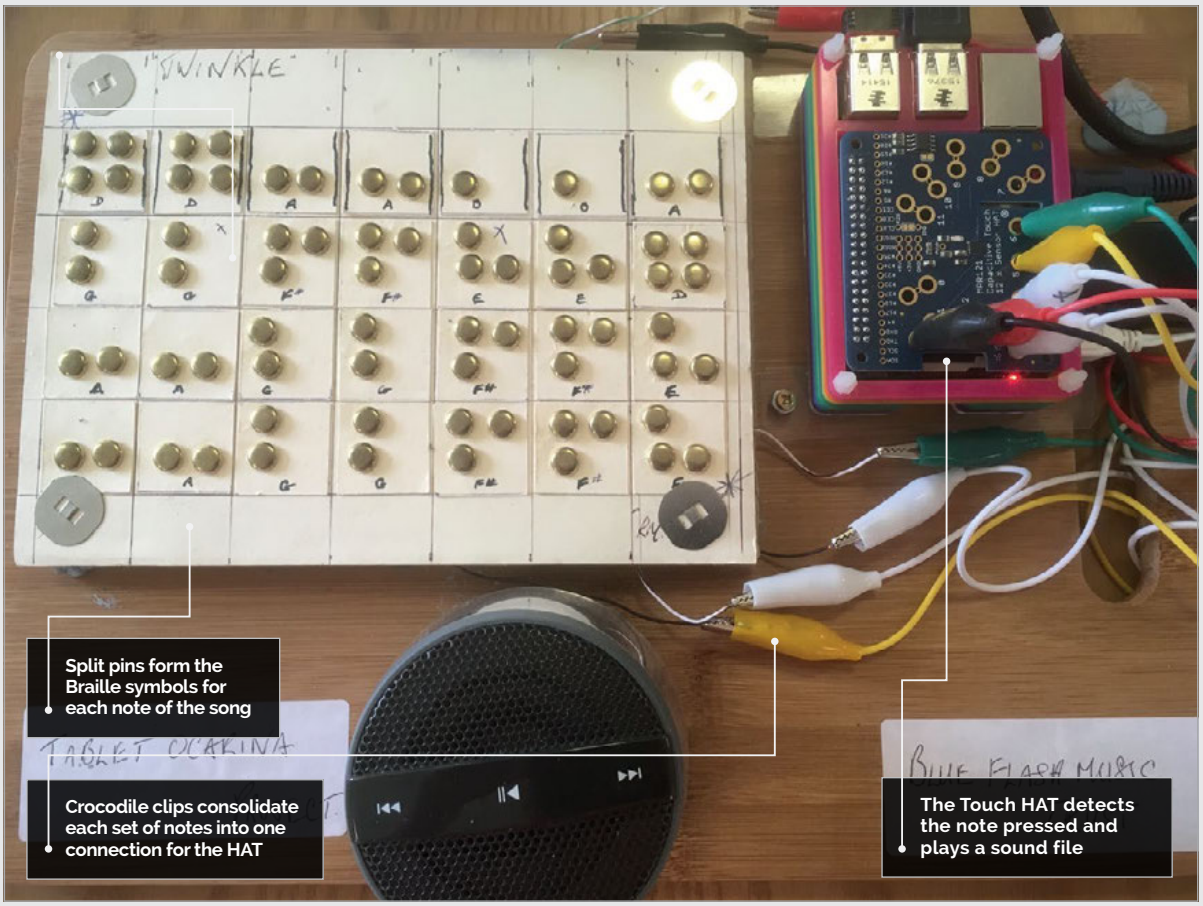
When Robert Mayfair met eight-year-old James at a party in 1994, he gave him the gift of an ocarina. James was blind, and so thankful for the gift that he later contacted Robert and asked for lessons. A new bond was instantly formed between the two.

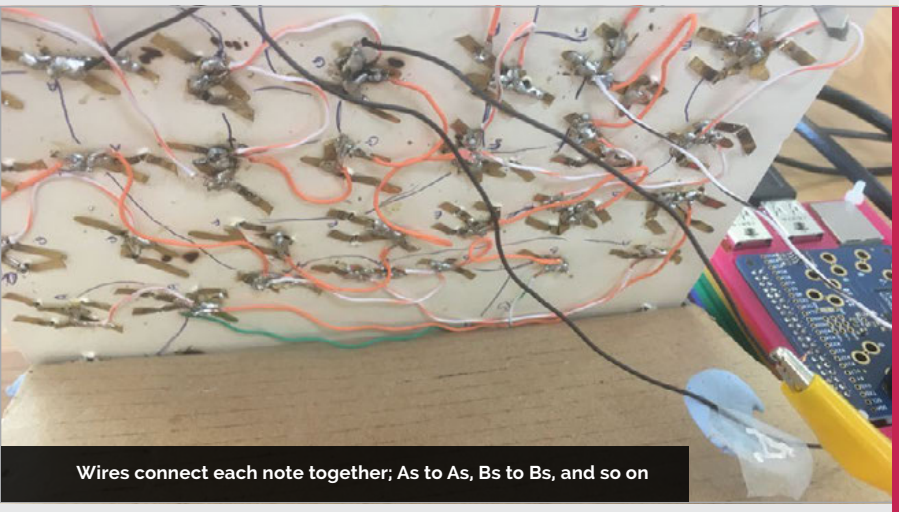
Over the years of friendship, Robert and James have collected nearly 30 different instruments, with James's love for music ever-growing, especially toward the ocarina. The joy of learning music together, however, is often clouded by the inability to truly share the experience; resources are limited for the visually impaired.

Recently, Robert discovered that the Royal National Institute of Blind People (RNIB) had published a Braille book of ocarina music, and though this was a wonderful advancement in accessibility for the visually impaired, Robert realised that sighted people were unable to interact with the content:

"On buying the book I realised that the Braille book was of no use to the sighted person, as it was like looking at a landscape covered with snow."

Aiming to find a solution, Robert found his answer far quicker than anticipated when he came across a HackHorsham display in a shopping centre last November. The display, using pieces of fruit to produce music via conductivity, gave him the inspiration he needed to change the way he and James read music together.





Wires connect each note together; As to As, Bs to Bs, and so on

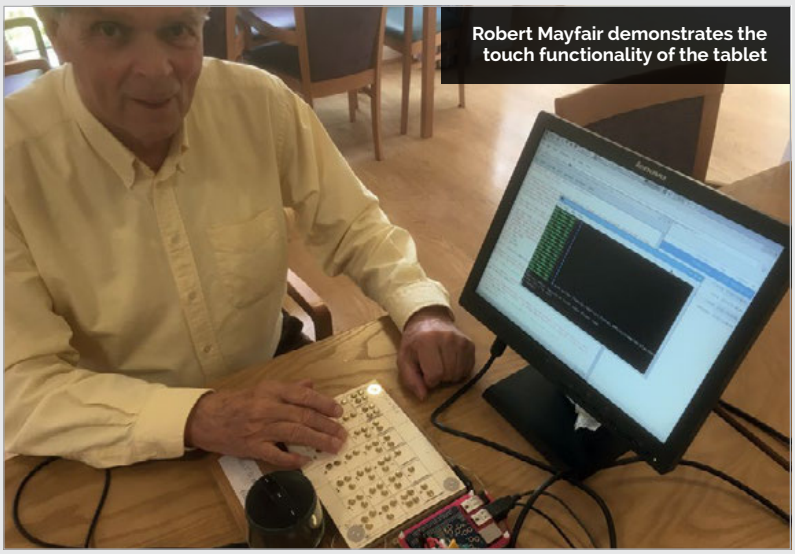
Robert produced a prototype of plastic and cardboard, and later brad nails, that James was able to interact with, recognising *Twinkle Twinkle Little Star* via touch. After a few alterations, a tablet was produced where nails formed the notes of the song in Braille, James reading them with one finger.

In April this year, Robert attended The Rebel Maker Club, a monthly event hosted by HackHorsham, and met Jonathan Tyler-Moore. Jonathan already had experience of building with Pi and finding solutions for issues using tech. So it was no surprise when the 13-year-old quickly introduced a Pi and speaker to the setup, allowing the appropriate note to be played aloud as split pins were touched on the tablet build.

Each split pin is wired, with sets of notes connected together. All A notes, B notes and so on are then connected via crocodile clips to an Adafruit Capacitive Touch HAT. Touch an A note on the board, and the HAT recognises the connection and tells the Pi to play the appropriate sound. Jonathan used an ocarina to record each note onto his mobile phone, later copying them to the Pi as OGG files.

The build was a success; James and Robert now have access to the technology that will allow them to learn music together, through both touch and sound.

The project was finally showcased at the HackHorsham event at the Capitol Theatre in July, receiving praise from musicians and educators alike. Backed by the Blue Flash Music Trust, a community-based music charity within Horsham, the Tablet Ocarina Project is still a work in progress and a promising starting point for a broader scope of builds.



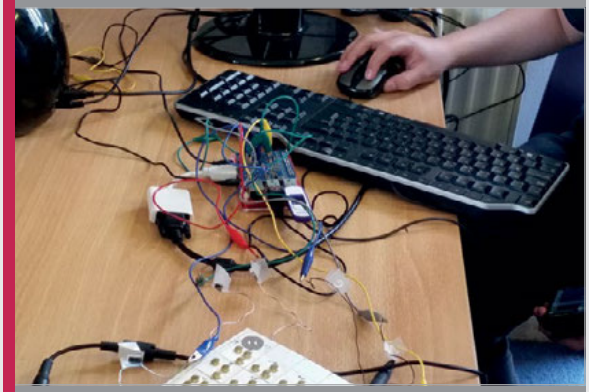
Robert Mayfair demonstrates the touch functionality of the tablet

LEARNING THROUGH TOUCH AND SOUND

>STEP-01

Building the song

The song is set out on the board, note by note, using split pins. Each set of notes is linked together into an Adafruit Capacitive Touch HAT.



>STEP-02

Recording the notes

Each note is recorded using an ocarina, then loaded onto the Pi as an OGG file, which is a type of audio/music file like an MP3.

>STEP-03

Learning the tune

When touching a set of pins, the HAT recognises the note and the appropriate sound is played through a speaker.

RIDWAN NASRUDDIN & GUUST VAN UDEN

Ridwan (pictured) and Guust are VJs working in The Netherlands. They create interactive digital artworks for club nights in Amsterdam.

magpi.cc/2aWdy9e

Quick Facts

- When assembled, it measures two metres in diameter
- In geometry, a torus is a circle rotated around an axis
- TORUS's Fadecandy controls eight NeoPixel LED strips
- The TORUS features 400 LEDs in total
- TORUS is painted white to reflect projected film

TORUS

VISUAL MUSIC INSTALLATION

Amsterdam club nights look incredible thanks to this TORUS, a Raspberry Pi-controlled visual art sculpture

Amsterdam is famous for its party scene, but The Netherlands is also a high-tech hub with lots of creative people working in science and computer technology.

At the end of the week the Dutch like to party, and Amsterdam's clubs are full of high-tech audiovisual treats.

TORUS is a music installation piece created by Dutch visual artists Ridwan Nasruddin and Guust van Uden. It's a large sculpture based on the torus geometric shape, covered in hundreds of LEDs all controlled by a Raspberry Pi.

"TORUS started as a research project," explains Ridwan. "We were already doing visuals during club nights on a flat white screen, but we wanted to create a sculpture.

"We are interested in origami shapes and modular forms," he continues. "We stumbled upon the paper art of Yoshinobu Miyamoto [a Japanese architect]. Inspired by his art, we created the TORUS."

Built from 6mm MDF plywood, TORUS comprises 18 blades assembled in a circular pattern. The blades are covered in NeoPixel LED strips (**magpi.cc/2yNoovV**), and the whole unit mounted in a dance club alongside a projector.

The LEDs are controlled using an Adafruit board called Fadecandy. This is a NeoPixel driver with built-in dithering, that can be controlled over USB. "We tried different ways to control the LEDs," says Ridwan, "and found out Fadecandy was the best way to send the signal

TORUS lights up Amsterdam's De Marktkantine club with its blend of film projection, light, and sculpture

from Resolume (**resolume.com**) to the Raspberry Pi and then to the LEDs."

The Raspberry Pi is connected by a 50-metre Ethernet cable, used to send the signal from Resolume (running on a laptop) to the Pi.

"Because we didn't knew a lot about how to connect LEDs to Resolume, we researched and experimented a with different boards and types of LEDs. We thought we could learn it quickly, but when we couldn't figure it out we asked around and found Nathan Marcus, a local programmer." Nathan wrote the major part of the code and with his help, they learned how to create the image they had originally envisaged.

The Raspberry Pi was added to get the LED data over a long distance, of about 50 metres.

"We had already settled on using a Fadecandy because of it capabilities and ease of use," explains Nathan. The Raspberry Pi functions like a server connected over Ethernet with the laptop. It interprets the messages from the laptop and sends them to the Fadecandy.

"We had a lot of help from the guys of the club [De Marktkantine in Amsterdam] where we showed the TORUS." To hang the TORUS where everyone could see it, they used steel cables to make a hanging truss above the stage.

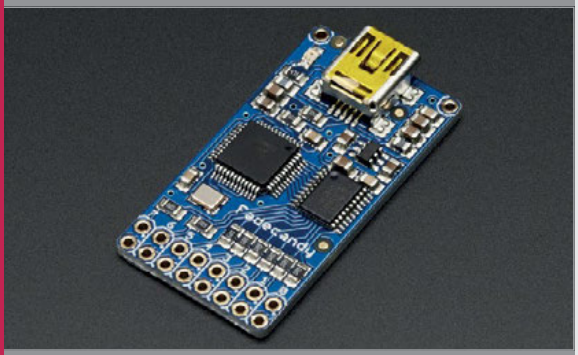
"The reactions of the crowd are great. We always wanted to make the experience of the music as intense as possible and it works. By creating one focus point, people really get into the vibe of the club night."

> "We always wanted to make the experience of the music as intense as possible"

BUILDING A TORUS

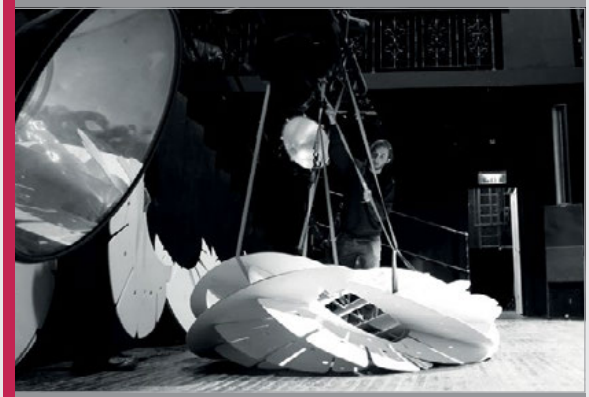
>STEP-01
Making the pattern

The TORUS is made from MDF cut into 19 interlocking blades. These can be assembled and disassembled, making it easy to transport to and from a club venue.



>STEP-02
Using NeoPixels

The TORUS blades are covered in AdaFruit NeoPixel LED strips. These are then controlled using Fadecandy (a custom board for controlling NeoPixel strips). A Raspberry Pi is connected to control the Fadecandy board.



>STEP-03
Assemble TORUS

The TORUS is assembled on location and the Raspberry Pi is connected to the DJ booth using a 50-metre Ethernet cable.

BETHANIE FENTIMAN

19-year-old Bethanie plans to visit every Disney park in the world… which is an amazing bucket list goal if ever we've seen one.
magpi.cc/2cnzMDO

WIZARD CHESS

Quick Facts

- Watch the set in action at **youtu.be/Z7xdFn5bVrA**
- The Wizard Chess Tour started in Harlow
- After Harlow, it visited the Covent Garden Raspberry Jam
- Inspiration came from Instructables user maxjus at **magpi.cc/2il6A71**
- It's wingardium leviOsa, not leviosAH

19-year-old Bethanie Fentiman shocked her A-level classmates when she rocked up with a fully working Harry Potter Wizard's Chess set as her final coursework assignment

Bethanie Fentiman can't play chess, but when her imagination sparked and the opportunity presented itself, she brought the iconic game of Wizard's Chess from Harry Potter to life using a Raspberry Pi, stepper motors, and possibly a little magic.

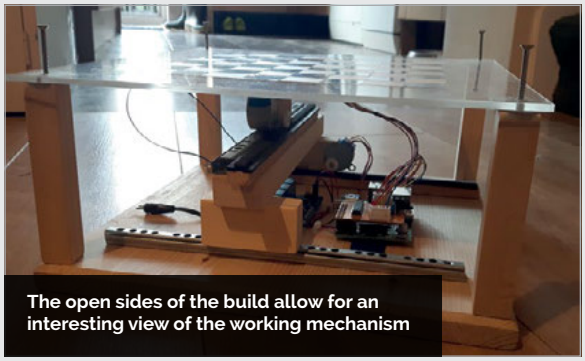
For her A-level computing coursework, Bethanie took an idea that had been nestling in the back of her mind, and turned it into a reality. Well, as much of a reality one can create when the literary version includes battling chess pieces that leave their opponents crushed to rubble on the board.

Luckily for Bethanie, she's a self-proclaimed Jambassador, actively participating in the Raspberry Pi scene via the Kent Raspberry Jam. With a community of makers to support her, Bethanie knew that she could complete the build and got to work, researching similar projects online that used magnets and motors to 'magically' move chess pieces on a board.

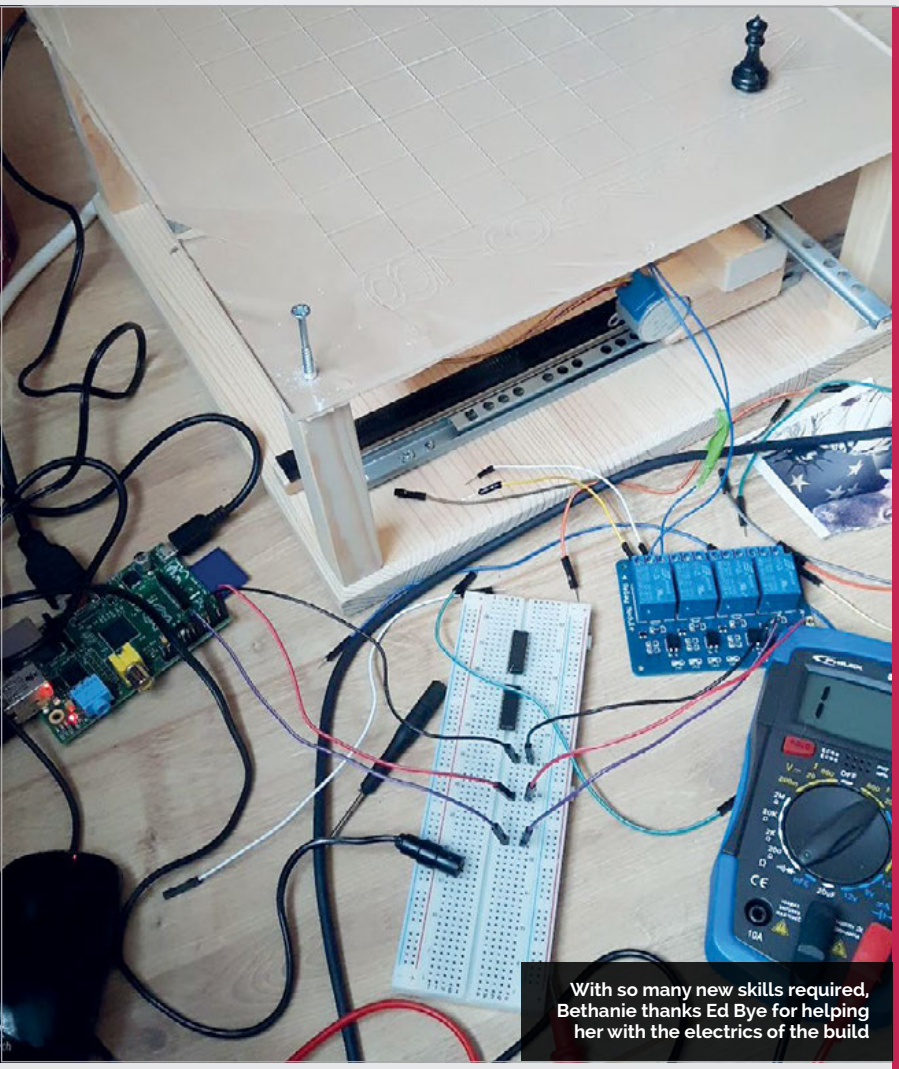
After an internet search for inspiration, she came across an Instructables build for an Arduino-powered chess-playing robot by user maxjus, and used the main concept as the basis for her build. The guide provided all the information Bethanie needed to build the physical structure of the board, allowing for drawer runners, gears and, of course, the electromagnet that would move each piece when required. A 4tronix PiStep board, along with two 28BYJ-48 stepper motors, took up the job of moving the runners and electromagnet into place, linked through to the Raspberry Pi.

As mentioned previously, Bethanie didn't actually know how to play chess. So when it came to inputting the legal movements of each piece, she had two options: learn fast, or cheat a bit. Opting for the latter due to the time constraints of her coursework deadlines, Bethanie researched all the possible moves of each chess piece and worked them into the code. She could always learn to play the game later on.

A second issue, and one far more associated with the original material from which she was taking her inspiration, was what



The open sides of the build allow for an interesting view of the working mechanism



With so many new skills required, Bethanie thanks Ed Bye for helping her with the electrics of the build

> When I turned up with the fully moving and playable board at school, they were shocked

the pieces would do as they 'took' an opponent. In the book, each piece defeats its foes through 'barbaric' means. In reality, Bethanie plans on an upgrade to allow for movement around pieces… though once she gets her belated invitation to Hogwarts, we're sure she'll incorporate the expected level of brutality.

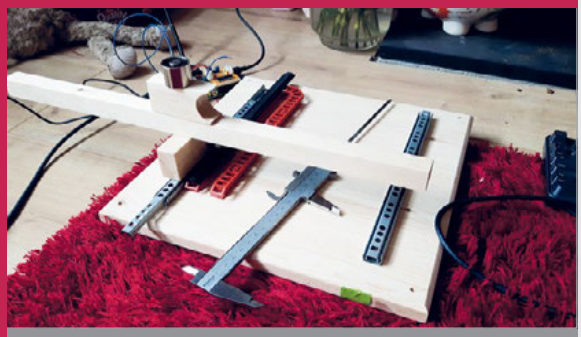
With the build complete and presented to her computing A-level class, Bethanie's Wizard Chess was met with amazement.

"When I said I was going to make it, they just thought I was going to write the code and come up with designs for the board. So when I turned up with the board at school, fully moving and playable, they were slightly shocked."

And they weren't the only ones. Upon finishing her board, Bethanie took it to the Kent Raspberry Jam, where Twitter soon exploded with praise. From the Jam, 'The Wizard Chess Tour' was born as Bethany and fellow Jam members took to the road and presented the project at Jams in both Harlow and Covent Garden.

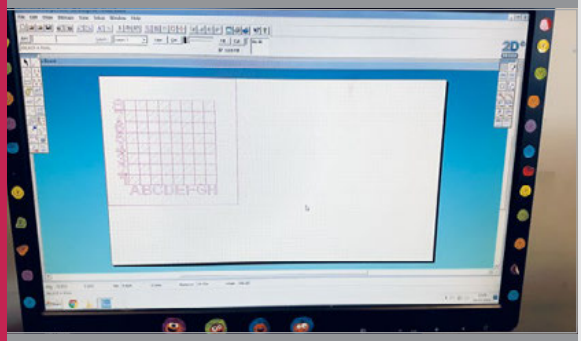
Now actively seeking an apprentice in the field, Bethanie plans on upgrading the build while continuing the Wizard Chess Tour at more Jams in the future.

YOU'RE A (CHESS) WIZARD, HARRY



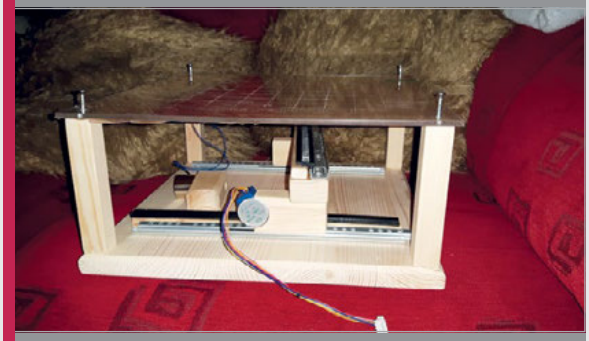
>STEP-01
Setting up the runners

Runners allow for the motors to move the electromagnet, and code dictates which pieces to shift across the board. Here Bethanie could put her newly discovered soldering skills to the test.



>STEP-02
Etching the acrylic

Bethanie was fortunate enough to have access to various pieces of equipment, although she admits that any future build would omit the added vinyl that made movement less fluid.



>STEP-03
Building the board

The entire build was a learning curve for Bethanie, allowing her to expand her knowledge of new skills and to call on a number of Raspberry Pi community members for support.

JOON GUILLEN

When not busy being a dad, Joon works for an online retailer as a Linux sysadmin. And when not busy doing that, he makes music under the moniker modulogeek and dabbles in geeky projects from time to time.
modulogeek.com

Quick Facts

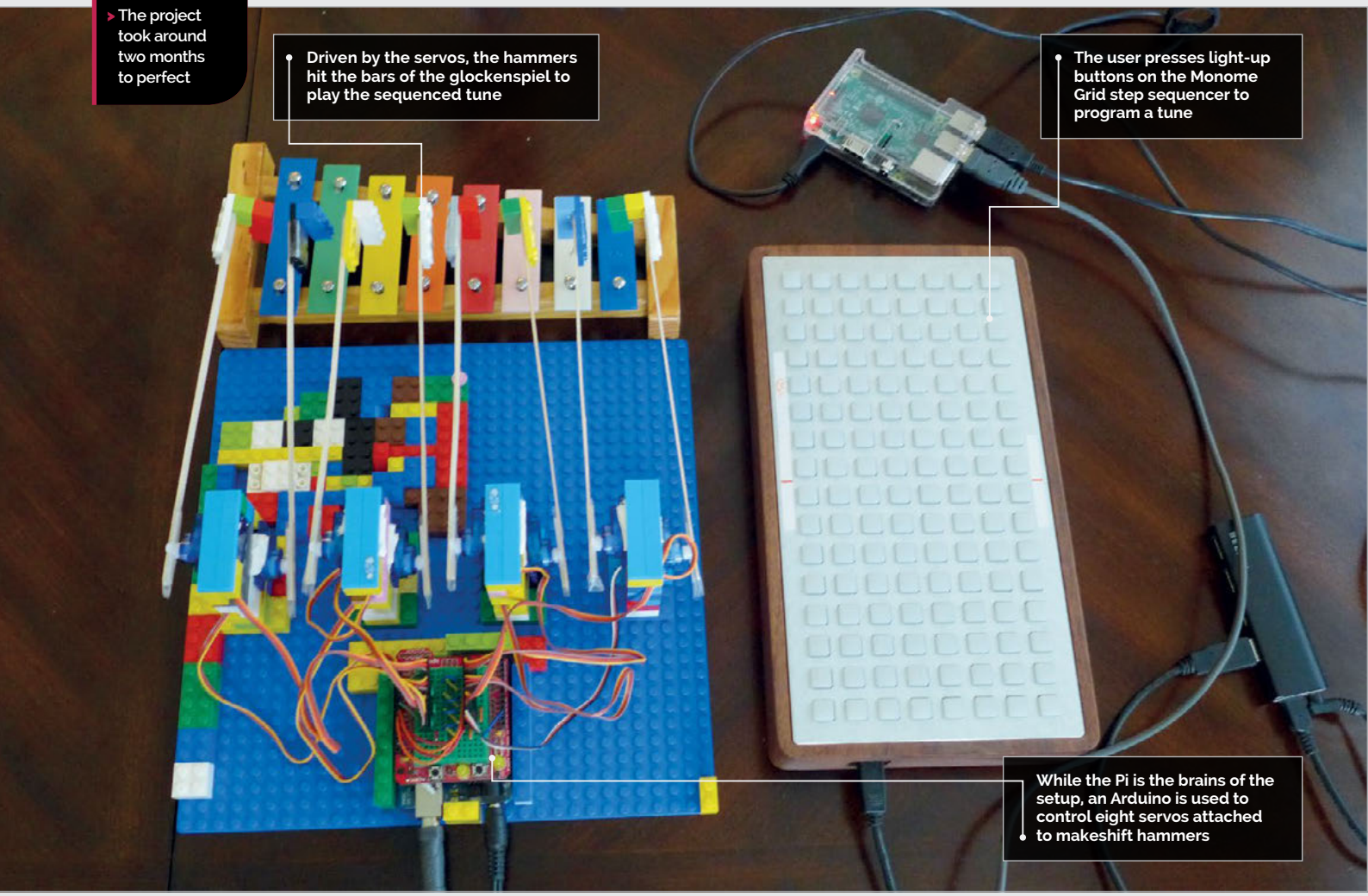
- The LEGO contraption took around three evenings to build
- Joon's young daughter added some extra blocks
- The hammers are made from coffee stirrers and LEGO
- A Pi 3 runs the Python sequencing software
- The project took around two months to perfect

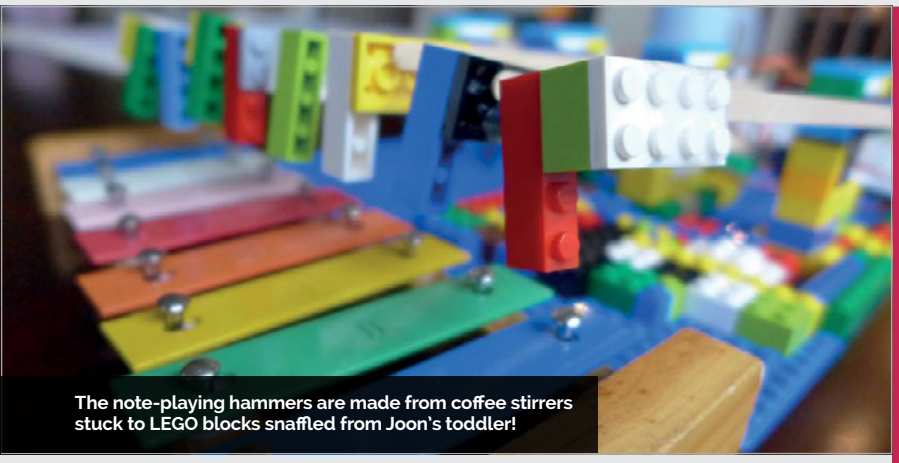
MONOMEPI

A music box featuring old and new technology in perfect harmony

With hammers hitting the bars of a toy glockenspiel to play a tune, the Monomepi sounds just like an old-fashioned music box, but this Pi-powered contraption is based on new technology… and on quite a lot of LEGO. "It was just my luck that the components fitted with the LEGO bricks almost perfectly!" reveals its creator, Joon Guillen.

He got the idea after seeing a couple of videos of Arduino-based music boxes a few years ago, while working on a Conway's Game of Life Pi project using a Monome Grid, a versatile piece of hardware that can be used to control music and more. For the Monomepi, the Monome is connected to a Raspberry Pi 3 running a step sequencer program, which registers the user's button presses on the Monome and lights them up accordingly. The Pi then sends serial commands to an Arduino Uno connected via a ProtoShield kit to eight servo motors, which move makeshift hammers to play glockenspiel notes to match the pattern shown on the Monome. On the latter, the user can switch buttons on and off to alter the sequence as it plays.





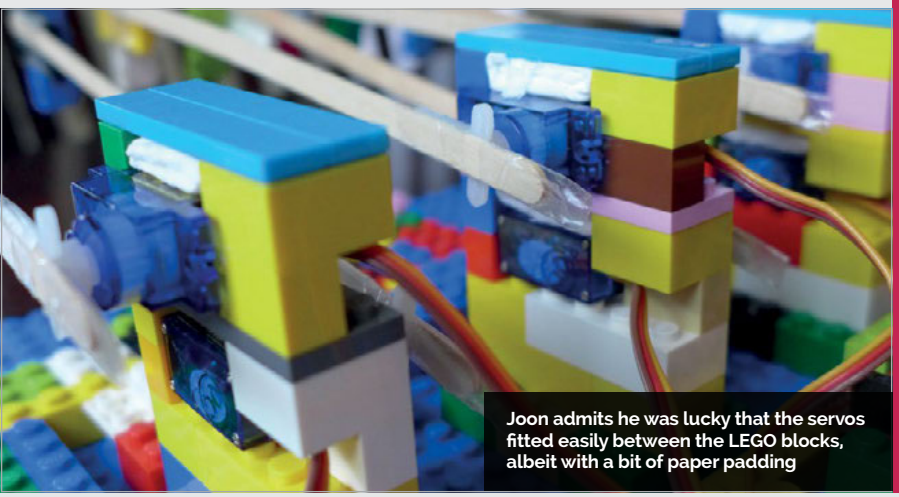
The note-playing hammers are made from coffee stirrers stuck to LEGO blocks snaffled from Joon's toddler!

"The contraption itself took only two or three evenings to build," Joon tells us. "I focused most of my energy on the software side, so the physical construction was almost an afterthought." To build it, he borrowed a bunch of LEGO blocks from his young daughter. "They were the first things I thought of using. I haven't the talent for crafts, and so LEGO was the quickest way to build the contraption. My daughter even added some blocks of her own in there!"

While the construction was quick, the project as a whole took around two months, with Joon working casually over the course of several evenings and weekends. "Most of it was figuring out the step sequencer logic, Arduino code, and optimising performance." The main Python program running on the Pi is based on a Monome library Joon had created for his previous project. "That took a very long time, as I had zero Python knowledge when I started out. The library has since undergone several improvements through the years."

While Joon opted to control his servos via an Arduino, he says there's no reason why anyone creating a similar project couldn't trigger them from the Pi itself, using a suitable motor driver board. And if you're lacking a Monome (quite an expensive piece of kit), a touchscreen could be used instead: "A web-based UI should work, too. Or, if one isn't necessarily trying to make a step sequencer, push buttons or [a computer] keyboard are viable control alternatives."

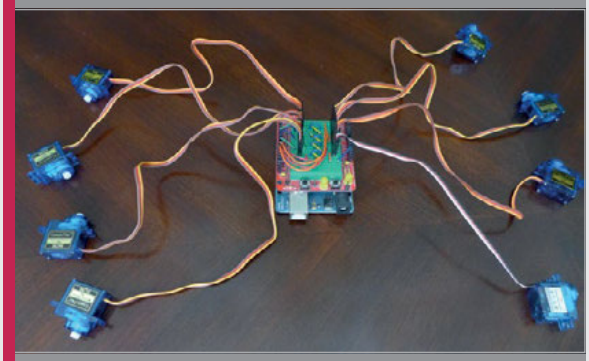
As a part-time musician, Joon plans to sample the Monomepi to use in at least one of his tracks. He's also looking to improve the project by "adding features to the step sequencer program, such as having more than 16 steps, and the ability to use multiple velocities. Other than that, I am trying to think of more ways to use my servos with the Pi!"



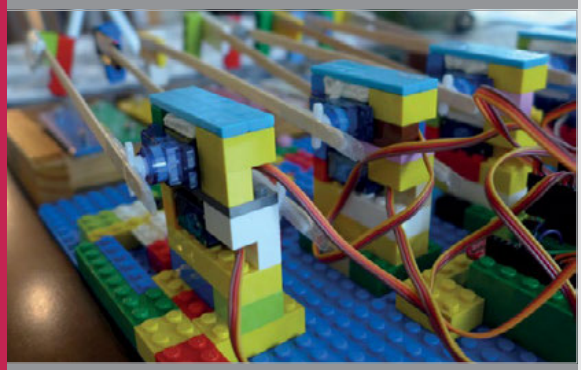
Joon admits he was lucky that the servos fitted easily between the LEGO blocks, albeit with a bit of paper padding

BUILDING A MODERN-DAY MUSIC BOX

>STEP-01
Glockenspiel hammers
To play the notes on a toy glockenspiel, the hammers are made from coffee stirrers, sticky tape, and LEGO blocks borrowed from Joon's young daughter.



>STEP-02
Arduino servos
Eight servo motors are connected to an Arduino Uno R3 and ProtoShield kit with a mini-breadboard. This is controlled by the Raspberry Pi and Monome Grid.



>STEP-03
LEGO construction
With the wiring complete, it's time to connect the hammers and add more LEGO blocks around the servos to keep everything firmly in place.

TEEFAX

PETER KWAN

Having worked as an engineer with teletext equipment for the last 12 years it was transmitted in the UK, Peter is an expert in the field. When not recreating teletext, he's out riding his bike through the valleys of Stroud.
teastop.co.uk/teletext

Teletext makes a comeback with the help of the Raspberry Pi

Quick Facts

- Peter built a text service for the Stroud Fringe festival
- New Teefax contributors are always welcome
- You can use Peter's wxTED page editor **(magpi.cc/2dsEZfG)**
- Dave Honess has made a teletext QR code generator
- Peter is developing a Muttlee multi-user live editing system

Before the dawn of the world wide web, teletext was the best way of keeping up to date with the latest news, sports scores and other information. The BBC's Ceefax teletext service continued in the UK right up until analogue TV transmissions ceased in October 2012. We still miss its no-nonsense approach and blocky graphics, so we're delighted that teletext has been revived by the Teefax project. Users can install the free software (**magpi.cc/2dssVeo**) on a Raspberry Pi, connect its 3.5mm video output to a TV (via the SCART socket), then hit the teletext button on the remote control.

Project founder Peter Kwan is a former teletext engineer who carried on working in the field as a hobby. "As the analogue TV network was being shut down, I was thinking about how I could generate my own teletext," he recalls. With the idea of making a low-cost basic teletext inserter, Peter manufactured his own VBIT hardware and managed to get a full teletext service running on it. Initially, there was a practical use for the system. "There is a lot of hidden signalling in the teletext signal," Peter reveals. "The BBC uses a system called Presfax which hides schedule information in databroadcast packets. They also have special signals that let London take over the whole

Above Along with news, Teefax pages include teletext art, quizzes, and some humorous articles from the likes of Mr Biffo

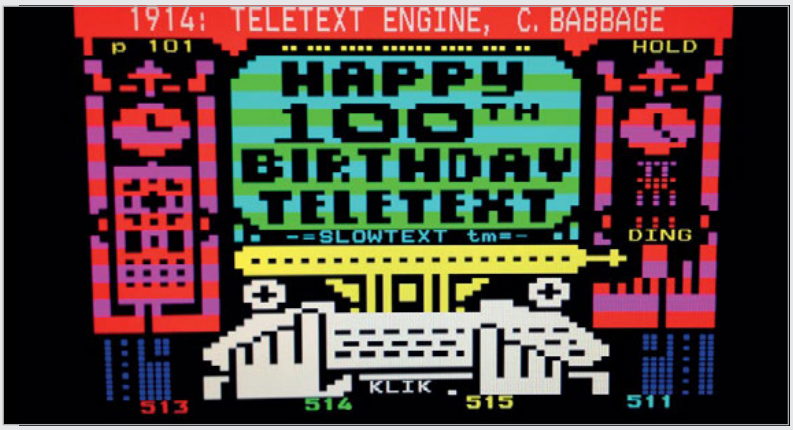
network in an emergency." In addition, betting chains use control signals to switch TV channels or mute audio in their shops, while European broadcasters use opt-out signals to insert local adverts. "These all need testing and VBIT was a low-cost and flexible way of generating these signals."

When the Raspberry Pi was launched in 2012, Peter soon realised it could be used instead of his bespoke hardware. "It had I²C, SPI, and GPIOs to drive and it was cheaper than the AVR boards that I was using, so I hooked one up and it worked. I made a second spin of the board and called it VBIT-Pi."

The next big breakthrough came when Alistair Buxton managed to create a teletext signal direct from the video output of the Pi. "I bypassed the teletext stream from my hardware to Alistair's software and instantly halved the cost of a teletext system," says Peter. "The only thing that the Pi can't do is overlaying from video, so things like subtitles and newsflash need an original hardware VBIT."

The Teefax server is actually an original Pi Model B running Subversion. "Apache handles user authentication. PHP scripts triggered by Cron scrape the BBC News website and update the news pages every day." Currently, there are seven authorised contributors to Teefax. "The real number is more because people are welcome to submit their own pages and designs and we will put them into Teefax for them." To do so, you can use Peter's wxTED page editor on a PC.

Meanwhile, Peter is currently working on a more flexible version of the VBIT system with a much faster update speed. "This actually has a commercial application in the betting industry where a small delay in reporting the 'off' in a horse race can be costly."



GETTING TELETEXT BACK ON THE TELLY

>STEP-01

Teefax server

The Teefax server is an original Raspberry Pi Model B running Subversion and Apache web server. PHP scripts scrape the BBC News website and convert stories to teletext pages.

>STEP-02

Pi client

With a client Pi's composite video output connected to a TV, teletext data is transmitted in normally unseen VBI (vertical blanking interval) lines of the video signal.

>STEP-03

Teletext pages

Hit the teletext button on your TV remote to start viewing the pages as normal. Page numbers can be entered, or coloured buttons pressed to switch sections.

BRUCE SHAPIRO

Bruce grew up enchanted by music, electronics, and making things. He has devoted himself to using motion control for making art and education tools.
magpi.cc/2fUUaRN

Quick Facts

- Each table is made in the US
- Bruce has been building Sisyphus tables for 20 years
- Sisyphus tables are displayed in museums and art galleries
- Components were picked to run quietly; it's virtually silent
- Sisyphus is demoed at Northeast Minneapolis makerspace **magpi.cc/2eWKiEH**

SISYPHUS

The incredible table that's also a work of art (and a Raspberry Pi robot)

Bruce Shapiro is a maker and an artist. Unlike Picasso or Rembrandt, Bruce doesn't paint with oil and brushes. "My medium is motion control," he tells us.

We caught up with Bruce to chat about his latest project, Sisyphus. Hot on the heels of a phenomenally successful Kickstarter campaign, Bruce has a lot to talk about.

"Sisyphus is a computer-controlled machine that moves a magnet beneath a field of sand," Bruce explains. "On the sand, a steel ball follows the magnet's changing position, creating dune patterns in its wake."

In Greek mythology, Sisyphus was condemned to roll a boulder up a mountain for all eternity. "In my art, Sisyphus is a kinetic sculpture that rolls a ball through sand, forever creating and erasing beautiful patterns." Bruce says that watching Sisyphus evokes a meditative feeling.

"Initially, I viewed Sisyphus as a kinetic sculpture. I still do! But over the years, I began to see a parallel with the relationship between making musical instruments and writing songs. As different as these creative skills are, both are absolutely integral to the final art."

Bruce has been creating Sisyphus sculptures for nearly 20 years, and has permanent installations in Switzerland, Germany, and Australia.

The heart of the project is the Sisbot, a robot that controls the metal balls which create the artwork in the sand.

"Sisyphus is a CNC machine," reveals Bruce. "It doesn't use G-code for its file format, but the principle is the same: a toolpath determines where the ball moves and its speed. With Sisbot being a polar machine, these moves end up producing spiral arcs, but this still works since small arcs connected together can emulate any path.

"My patterns tend to be algorithmic since I never learned to draw," Bruce continues. "But anyone can create paths for Sisyphus – just draw something without picking up your pen. If you can record the positions of your pen as you draw, you can compose for Sisyphus."

Above The Sisyphus table can contain more than one ball; they work together to create the artwork

Controlling Sisbot

A Raspberry Pi is the perfect computer to control the Sisbot and create the works of art, but it wasn't always that way. "For a very long time, all my motion-control artworks were controlled by Windows PCs running DOS," says Bruce. "In fact, three still do, running every day in their museums.

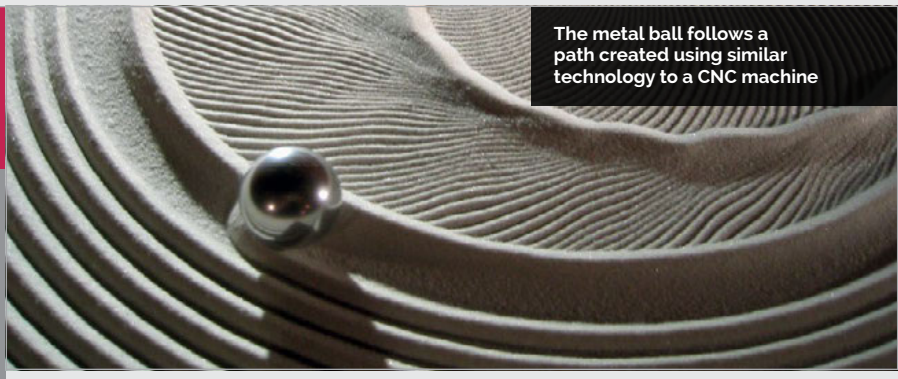
"I don't like change when it comes to something that works," Bruce admits. "Through many trials, and sometimes painful dead ends, I've learned that community matters."

> "I've been creating Sisyphus sculptures for nearly 20 years"

It was the Raspberry Pi community that convinced Bruce to switch over to a low-cost microcomputer. This was "more important than form factor and low cost," he tells us.

"The choice to use Raspberry Pi in the new home versions of

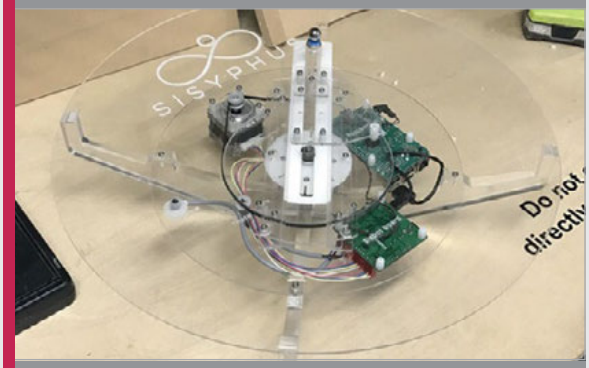
PRECISION BALL CONTROL ROBOT



The metal ball follows a path created using similar technology to a CNC machine

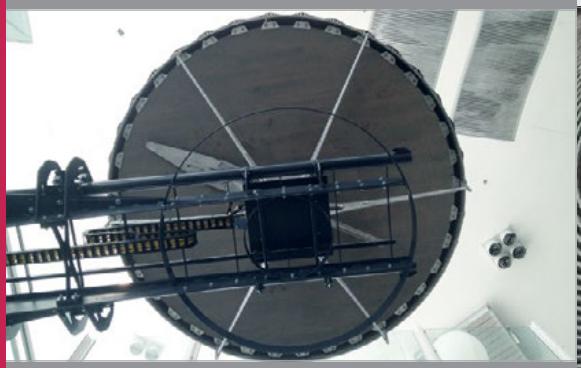
>STEP-01
The Sisbot

Under the table is a two-motor robot called Sisbot. This moves a magnet which pulls the steel ball (sitting above the sand).



>STEP-02
Playing motion

The motors of the Sisyphus are controlled by a Raspberry Pi. This plays a set of path files, much like a music player plays an MP3 file.



>STEP-03
Always on

Sisyphus has no on/off switch. When it's plugged in, it automatically calibrates and starts playing. You connect to it with WiFi from a laptop or an iPhone app. From the app, you can control the speed of the Sisbot and the lighting of the table.

Sisyphus was driven by all the 30-somethings that I listen to," says Bruce. The community informed him that JavaScript is essential and Node.js runs well on a Raspberry Pi.

"The size of the community and its willingness to share makes the Raspberry Pi unique," adds Bruce.

The requirements of the Sisyphus table are complex. "Three levels are involved," he explains. At the lowest level is "firmware written in C, running on the SisBotBoard". This was created by Brian Schmalz (**magpi.cc/2fXxWhl**) for an earlier project Bruce created, called EggBot. "Brian started many years ago with the initial crude C code I wrote for running my steppers," says Bruce.

The second level is higher-level motion control, recently ported to JavaScript by Bruce and cleaned up considerably by Alex Wayne (**magpi.cc/2fXDqs7**). This code runs in Node on the Raspberry Pi.

The third level is algorithmic path generation. "I originally did this with AutoLISP routines running in vintage AutoCAD," says Bruce, "but now I use the Grasshopper Rhino 3D plug-in."

History matters

"I became fascinated with the challenge of connecting motors to my computer about 25 years ago," Bruce recalls. "The easiest and most compelling way to demonstrate its potential is to create a machine that can draw, using just two motors."

EggBot was Bruce's first art machine. "[EggBot] rescued me from being locked up as a madman and for being so obsessed with, and talking incessantly about, the potential of hooking up motors to a computer," he laughs.

"[EggBot] is actually pretty cool," he says, "and people I showed it to, for the most part, got that. I spent many years building successively larger drawing machines that eventually were capable of moving cutting tools like plasma torches to enable me to cut out intricate shapes, mostly in metal, to make sculptures.

"I fell in love with watching them move," continues Bruce. "As I got better at designing new machines, the components that made them possible – stepper motors and the electronics that run them – were getting easier and cheaper to get hold of. In 1998, Sisyphus became my first CNC machine to escape the studio – transitioning from a tool used for making sculpture, to being the sculpture itself."

After 20 years of making, and refining, the Sisyphus project, Bruce was wary of making it available to buy. "Probably the biggest reason was not knowing how many people would respond."

He needn't have worried. What started out as a project to raise $50,000 eventually ended up getting close to two million dollars.

"When it comes to planning to make stuff for others, it helps to know how many will want it," says Bruce. "Kickstarter is a very public space to show what you want to do and find out how many people are supportive, not just with words, but with their hard-earned cash. It's an amazing leap of faith on their part, something we're still coming to grips with and take very seriously."

Above Users can control the lighting of the table, as well as the speed of the metal balls

FRED HOEFLER

An award-winning landscape photographer from Spokane WA, USA, Fred enjoys weaving in his spare time, as well as tinkering with electronics and the Raspberry Pi.
photographic-perspectives.com

Quick Facts

- The loom took Fred a year to build
- One Pi was accidentally fried in the process
- The total component cost is around $150
- Limit switches prevent the harnesses moving too far
- The user needs to send the weft threads through by hand

PI LOOM

A weaving wonder controlled entirely by Raspberry Pi

When Fred Hoefler sent back a computer-controlled tabletop loom after it stopped working with his MacBook Pro, his wife Gina suggested that he should be able to build one with "one of those Raspberry Pi things of yours." Fred recalls, "The most reasonable answer I could come up with was: 'as a matter of fact, I think I can.'" A project was born (**magpi.cc/2fDminE**).

At the outset, Fred set out some ground rules for building the new four-harness tabletop loom. As well as being controllable by a Raspberry Pi via the Bash console, no part of the hand-built loom was to cost more than the Pi itself. It would also need to be quiet enough to carry on a conversation in the same room with it.

Following a year of work, with many stops and starts, the end result is a working Pi-powered loom that meets all Fred's goals. "Basically, the Raspberry Pi drives four 12-volt DC motors turning in opposite directions, like a robot that rolls forward and backward," he explains. "These motors are attached to actuator arms that are in turn attached to a cord that runs through a pulley system to individual loom harnesses."

Fortunately, Fred had some previous electronics experience from his earlier career as an aircraft mechanic and electrician. "I was quite used to working with wiring, relays, switches, and motors. Working with aircraft aluminium, making brackets and such is also second nature to me." The project wasn't without the odd mishap, however. As Fred discovered the hard way, "A Raspberry Pi cannot directly control common motor power. You must use some kind of motor control circuit. I thought I would try to use a half-bridge MOSFET controller (a TI SN754410) that claimed to be able to handle up to 1 amp, and have all sorts of protection between the power circuit and the GPIO circuit." After one working motor test cycle, the result was a fried Pi: "We held an appropriate memorial."

Fred Hoefler shows off his hand-built, Pi-powered loom – see a video of it working at **youtu.be/QjqJOdjmbAY**

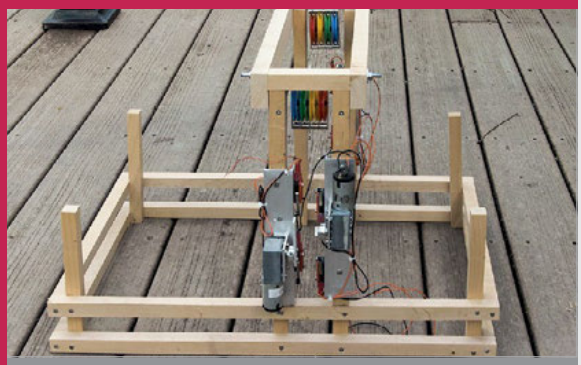
Each 12V motor moves an actuator arm with a cord attached via a pulley to a loom harness

To his surprise, Fred had less trouble programming a Python script (which can be found on his website) to control the loom. "Python works like the old BASIC interpreters we used back in the early 1970s, only easier. There are lots of Python examples out there explaining how to turn on and off GPIO pins... Once I set up how to control one harness with a module, all I had to do was copy-paste a duplicate module and change the calling parameters to control the next harness."

> "Four 12-volt DC motors turn in opposite directions, like a robot that rolls forward and backward"

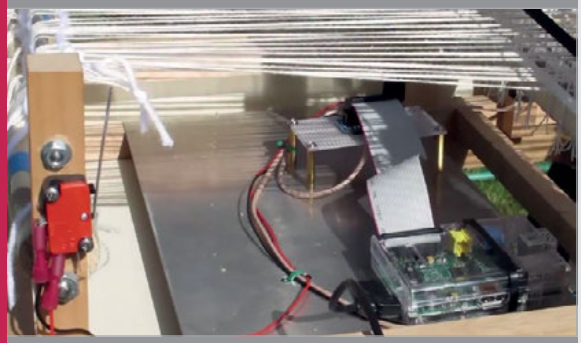
While Fred admits the loom is too slow to use for production purposes and requires an operator to send the weft threads through the shed by hand, he is looking to improve it by experimenting with different 12V actuators and possibly solar power. "Finally, I found that the manufacturer of the 'cute little loom' that started this whole project has halted production due a supplier not being able to produce a reliable control unit. If we could just send him a copy of this article and offer to upgrade the control unit..."

BUILD A PI-POWERED LOOM



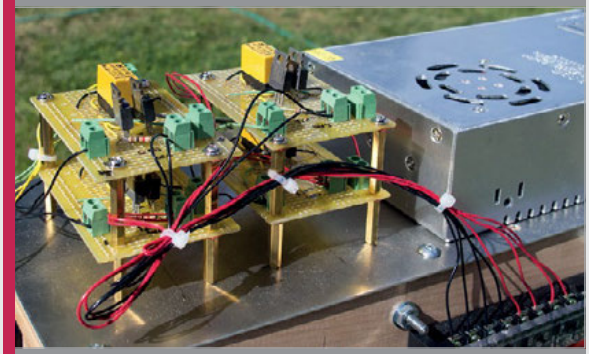
>STEP-01
Frame and motors

Attached to the hand-built wooden frame with metal brackets, the four 12V motors are linked to actuator arms, with cords running via a pulley system to lift and lower the harnesses.



>STEP-02
Pi powered

A Raspberry Pi is connected via a breakout board to the rest of the circuitry in the loom. Since the Pi cannot directly control 12V DC current, motor control units are required...



>STEP-03
Motor control units

Fred opted to build his own motor control units using 12V relays and TIP 120 Darlington transistors. An alternative would be to use a commercially available Pi relay control board.

TOP TEN

PET TECH PROJECTS

Create something for your critter with these amazing Raspberry Pi projects for pets

We love to see cool projects that combine technology and cute critters here at Raspberry Pi Towers. After all, pets are family, and we love Raspberry Pi projects designed around our four-legged (or more) friends.

There have been some great examples over the last few months, so we thought it'd be a great idea to herd them all into one giant feature to show you our favourite pet-themed projects.

Raspberry Pi builders are a quirky bunch, and we've got pet projects for all kinds of pets. Cat owners will love Jasper's Cat Exercise Wheel, David Bryan's Pi-Powered Cat Feeder, and John Shovic's MouseAir. Dogs will go all waggy-tailed for Matt Reed's Sniffur or David Hunt's Pi-Rex.

Some of the best projects here put animals on social media: Kate Bevan's Tweeting Cat Flap and Henry Conklin's Twitter for Dogs both allow pets to interact online (by moving or barking).

We've also included a cool project for smaller creatures. Will McGugan's Beetlecam monitors the nocturnal movements of elephant beetles.

There are some super projects here, so put some food in your pet's bowl and settle down to figure out what to make.

JOHN SHOVIC

Dr John C Shovic is the chief technical officer and co-founder of SwitchDoc Labs. He has also served as a Professor of Computer Science at Eastern Washington University, Washington State University, and the University of Idaho.

Left There have been two versions of the MouseAir. The second edition uses the Raspberry Pi Camera to detect nearby cats

MOUSEAIR

When it sees a cat, it flings a toy mouse! That's the basic premise behind this brilliant cat-detecting mouse launcher

YOU'LL NEED

- Pi Camera Module
- Servo motors
- DC motors

Cats love chasing mice, which is why so many cat owners wake up with unwanted presents on their doorsteps.

If your cat is a hunter, then check out John Shovic's brilliant MouseAir project (**magpi.cc/22MIwq3**). This Raspberry Pi build keeps a cat's urge to chase after mice in check by flinging toy mice for it. "The goal of the project," says John, "is to be able to detect a cat walking by and fire a mouse. It worked!"

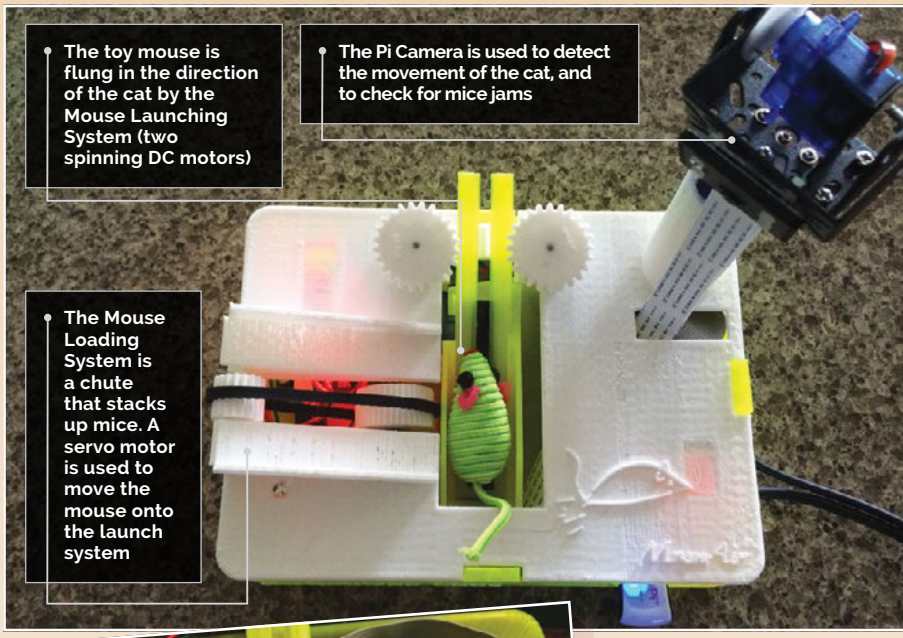
It may not surprise you to learn that the idea was conceived in a bar. The MouseAir was dreamt up in the Fedora in Coeur d'Alene, Idaho. "It was named by Sarah, the most excellent bartender there, and some Blue Moon beer," John discloses.

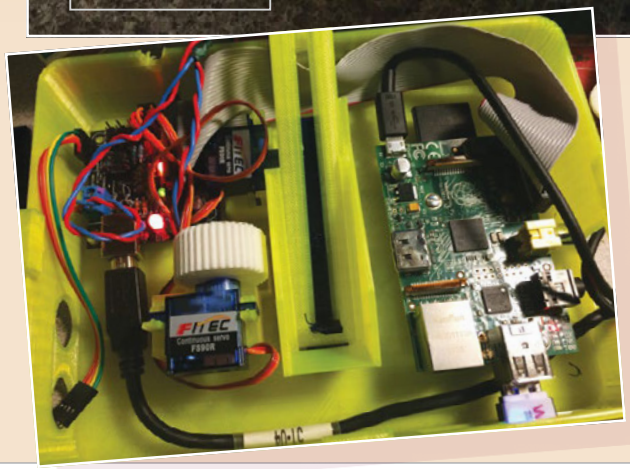
MouseAir combines a Pi Camera Module with a Mouse Loading System and Mouse Launching System (which flings toy mice into the air). "The Mouse Loading System is a conveyor belt, driven by servo motors, that drops the mice into the Mouse Launching System," explains John.

The Mouse Launching System is two rapidly spinning DC motors. These fling the mouse for the cat to chase.

"We're using a Pi Camera on a pan-and-tilt structure to capture the cat events, examine the launching mechanism for jams, motion detection, and stream video," says John. "The conveyor belt is the last thing that needs a little work.

"I had so much response to the project (including the response from my cat) that I decided to do a redesign of MouseAir, incorporating what I had learned from the project. I aggressively redesigned to eliminate unneeded hardware, and drive down the cost and size."

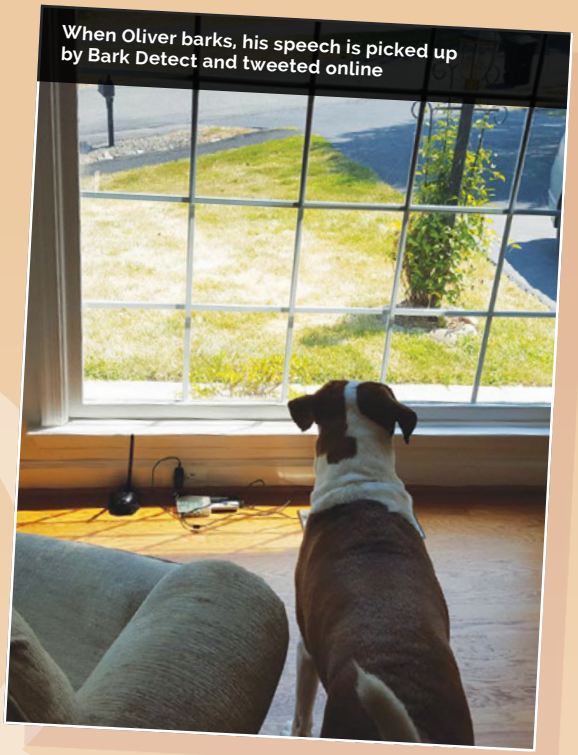




Left The MouseAir's 3D printed case houses the Raspberry Pi and servo motor

BARK DETECT

Wuf! Bark! Grrrr... this Raspberry Pi takes a dog's barking and tweets it online



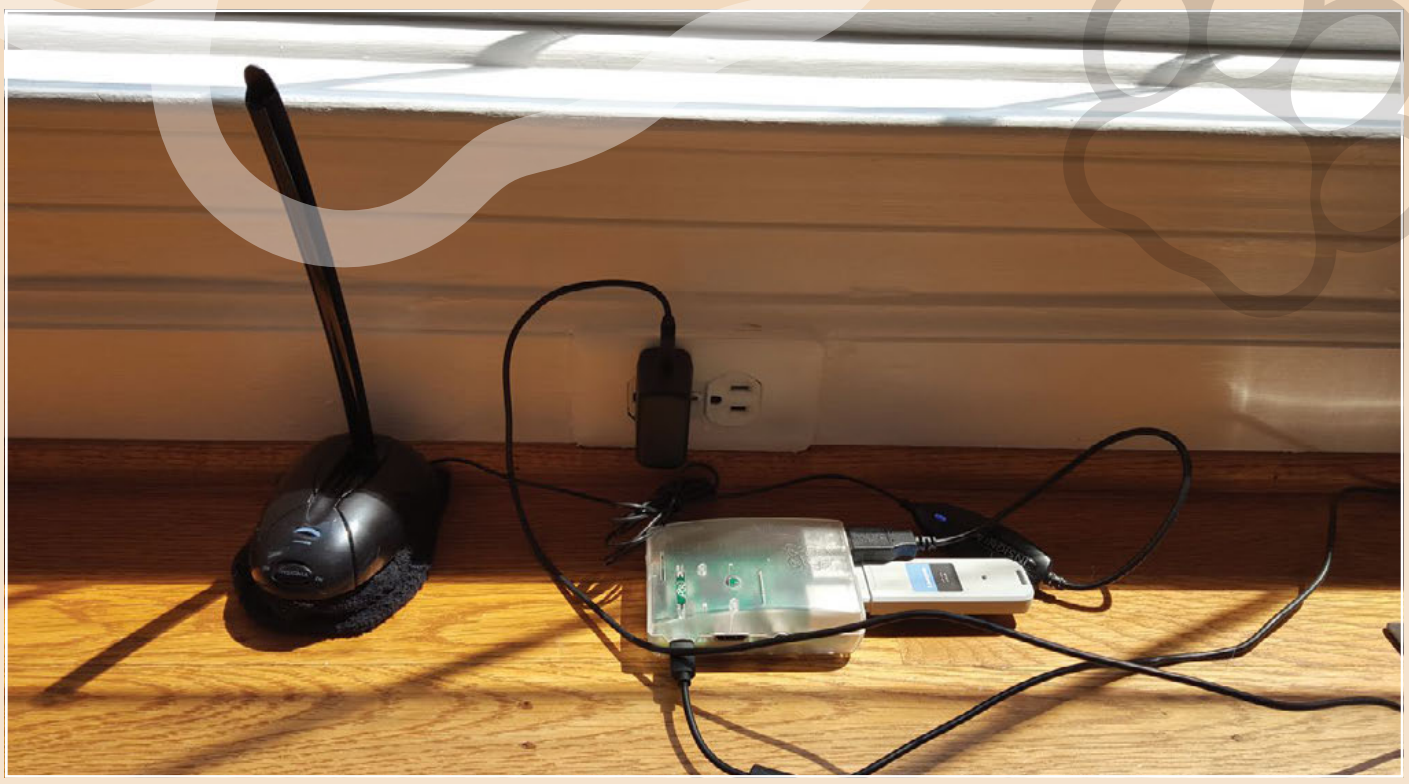
When Oliver barks, his speech is picked up by Bark Detect and tweeted online

Henry Conklin's dog, Oliver, can share his barks using Twitter thanks to the Bark Detect project (**magpi.cc/28KCBxd**). "My dog Oliver has always been quite vocal," explains Henry, "and recently I decided that his thoughts and comments needed to be shared with the world. Thus, the @OliverBarkBark project was born."

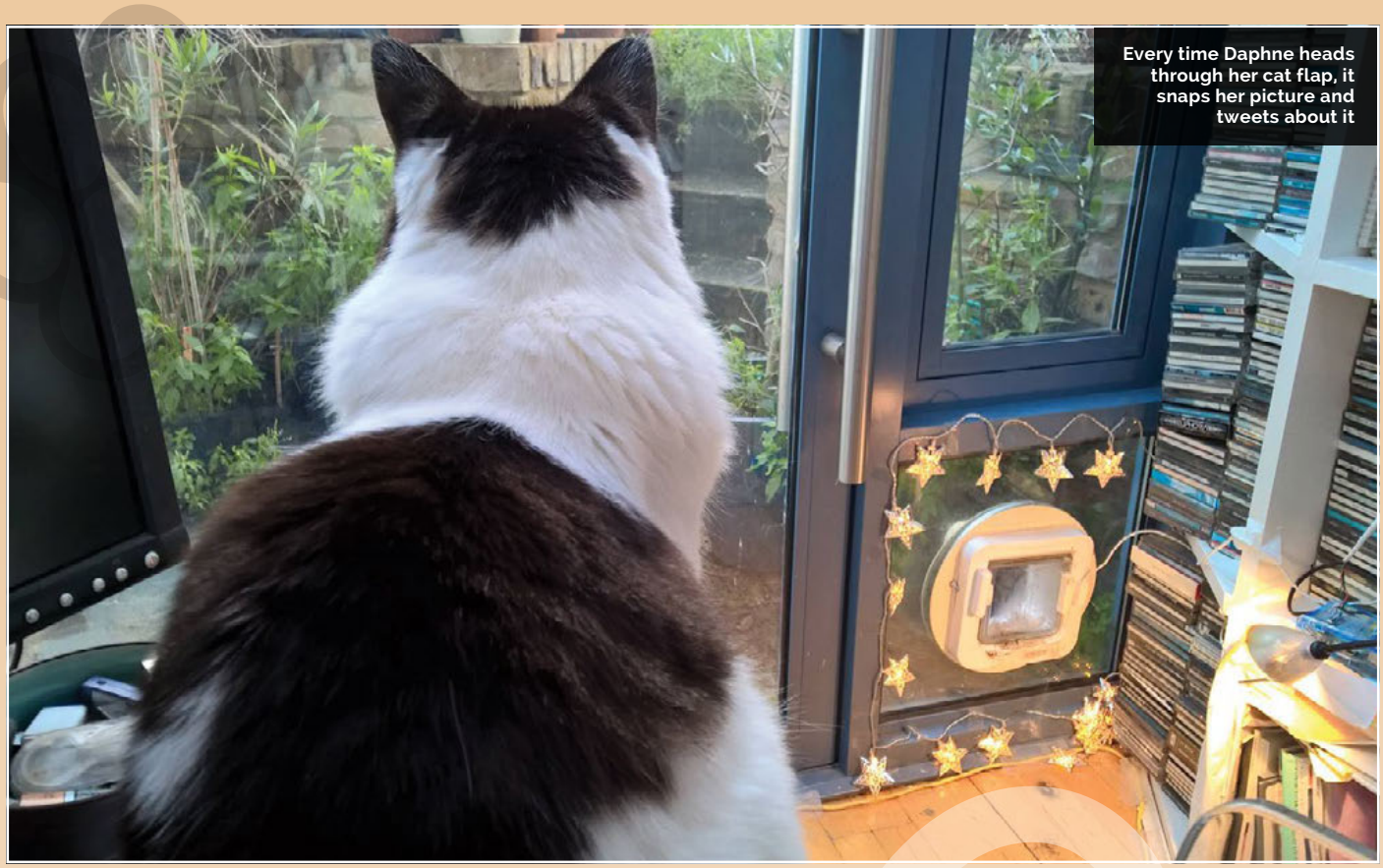
By connecting a Raspberry Pi, WiFi dongle, and microphone, Henry was able to make a system that "automatically detected, filtered, and published each and every one of Oliver's deafening vocalisations."

The Bark Detect is much more advanced than it appears. "I took a machine learning approach to filter out the barks," reveals Henry. "I built a model using pyAudioAnalysis (**magpi.cc/1VKsITG**) and a day's worth of barks. I then set up a Bash script to run every ten minutes and classify each recorded sound."

The barks are forwarded to the Twitter API using python-twitter (**magpi.cc/1VKsGet**) and posted under the handle **@OliverBarkBark**: be sure to follow! Currently, the tweets are random strings composed of 'bark', 'ruff', and 'woof'. "I plan to replace that with a bark-to-text translator that will likely produce similar results, but be much closer to Oliver's actual voice."



Below Machine learning is used to detect which noises belong to Oliver



Every time Daphne heads through her cat flap, it snaps her picture and tweets about it

FLAPPY MCFLAPFACE

Daphne announces her arrival to the whole world every time she saunters through her cat flap

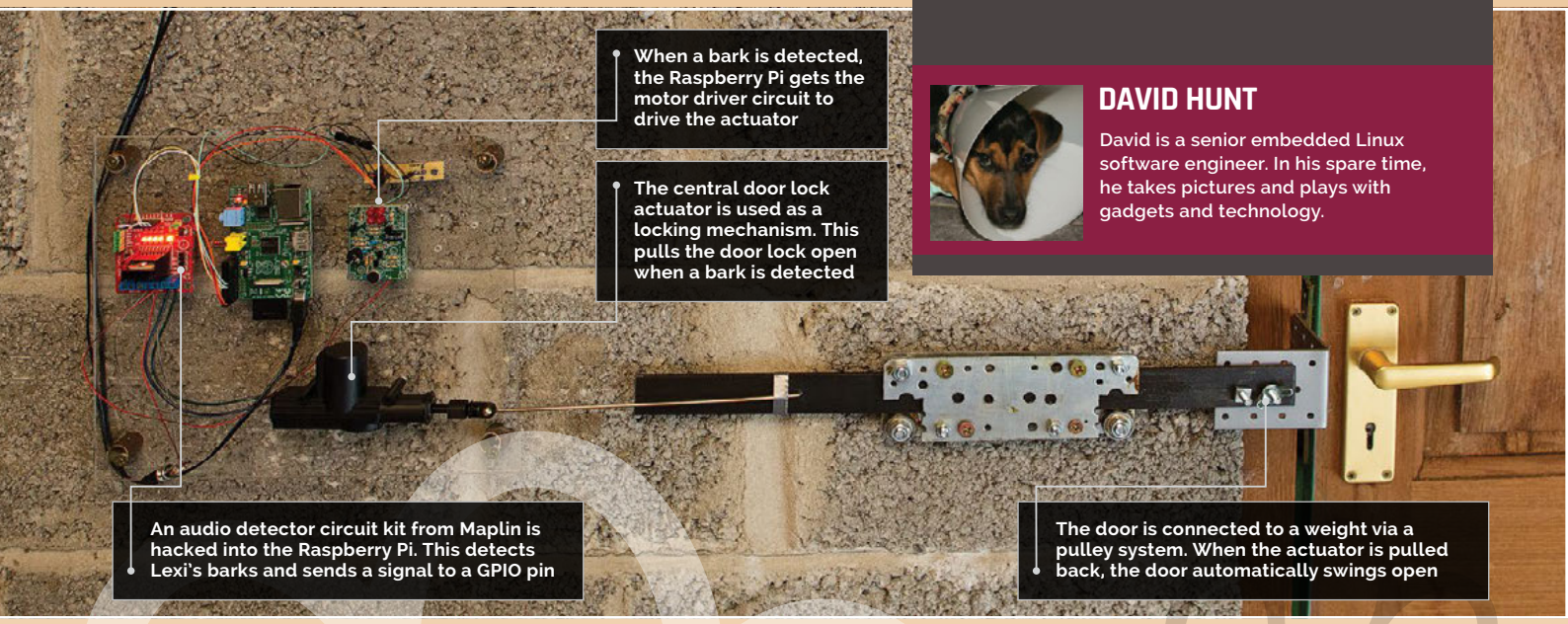
All cats think they're special, but only Daphne has her very own Twitter-enabled cat flap that heralds her arrival online. The Flappy McFlapface project snaps a photo and tweets it, along with a cute randomised phrase.

Built by Bernie Sumption, this cat flap is a thing of beauty. "Daphne often takes to social media to rant about the inadequate service provided by her staff (tech journalist Kate Bevan)," explains Bernie on his blog (**magpi.cc/1VKui85**). "This activity is cathartic and highly recommended for any household pet. Unfortunately, Daphne's cat flap was until recently mute, and couldn't tell the world about its thoughts and feelings."

A £3 reed switch from Maplin provides the input. "This is wired to the GPIO sensors, using a couple of resistors that prevent the Pi from being damaged by drawing too much current. The reed switch is duct-taped to the cat flap so that when Daphne walks through, the voltage on a GPIO pin changes and the software can spring into action."

Left A generative grammar Python program creates tweets for the Flappy McFlapface

The Raspberry Pi uses a generative grammar program to create a tweet for Daphne. "It works by taking a standard sentence structure and replacing nodes in the sentence with one of several options," says Bernie. Tweets include 'Most exquisite Daphne, how soft is your magnificent fwuff. I can expire without regrets,' and 'Oh my! It's Daphne, how noble is your imperial demeanour. No wall-mounted fixture could be luckier than me.' You can follow Flappy McFlapface on **@DaphneFlap**.



DAVID HUNT

David is a senior embedded Linux software engineer. In his spare time, he takes pictures and plays with gadgets and technology.

YOU'LL NEED

- Central door lock actuator
- Angle bracket
- Motor driver PCB

PI-REX

One dog's bark is enough to open doors on command

David Hunt's dog Lexi has her very own door. The Raspberry Pi inside Pi-Rex listens for Lexi's bark and opens the door for her.

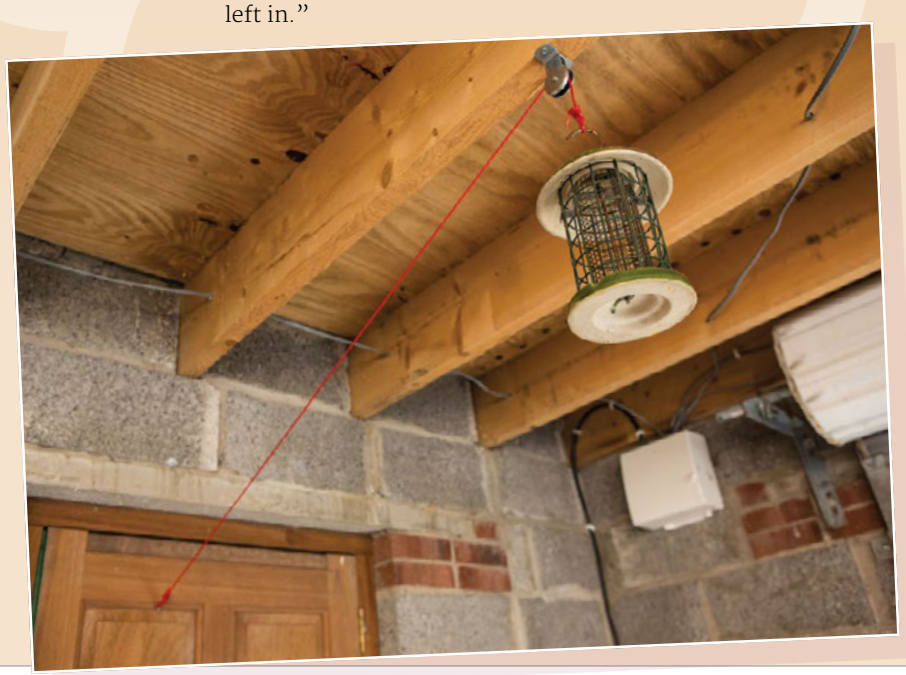
"Sleep deprivation has been driving me mad," says David. "It's all down to a new member of the family, our new dog. She [Lexi] barks at night when she's left out. She barks early in the morning when she's left in."

David decided to build a bark-activated automatic door opener. A noise detector circuit is wired to the input of the Raspberry Pi to detect barks. A motor driver circuit drives the actuator that unlocks the door, and a weight and pulley system swings the door open when it's unlocked.

"I picked up the audio detection circuit in Maplin as a DIY kit," David tells us. "I probed the audio sensor with a voltmeter; when the volume into the microphone was at a decent level, I saw about 3-3.5 volts at one point, so I hooked that directly up to the GPIO on the Raspberry Pi, and it worked beautifully.

"I had a few pieces of metal lying around, and an angle bracket which got around the concrete blocks nicely to meet the door lock. I added a couple of bearings to reduce the friction on the mechanism; I didn't want to burn out the actuator. When it was built it was still a little stiff, so I greased it up and it was then moving nice and smoothly."

Now whenever Lexi barks, the door opens automatically to let her into the house (**magpi.cc/28KAM3z**).



Left The door is connected to a weight via a pulley system. When the actuator moves back, the door automatically swings open

K9

One of our readers has built Doctor Who's robot dog

RICHARD HOPKINS

Richard Hopkins is an IBM Distinguished Engineer in the UK. He is the co-author of a book called *Eating the IT Elephant*.

What if you want to build a pet tech project, but don't have a pet? Well, you could use your Raspberry Pi to build your very own dog.

And if you're going to build a dog, why not go with the best robot dog that ever existed: Doctor Who's K9? Richard Hopkins is an IBM Distinguished Engineer, and he's spent a lot of time bringing K9 to life. "Most people would probably start with something simple, but that's not my style," says Richard. "Sink or swim. My first robot will be as near a facsimile of Doctor Who's dog K9 as I am capable."

The K9 build is based on the K9 schematics from the *Doctor Who* props department, and made from MDF. It's a work in progress, and detailed instructions can be found on Richard's blog (**magpi.cc/28J6O21**).

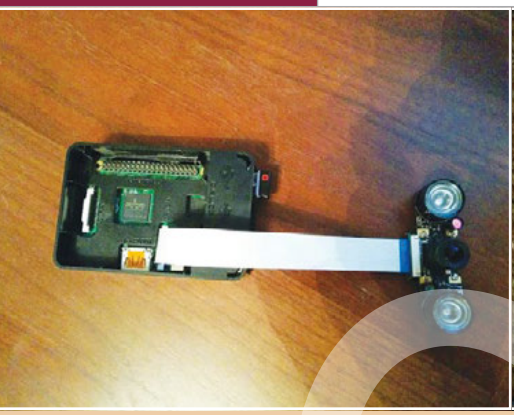
Richard has also purchased several remote-control K9 toys from eBay, but ended up building his version of K9 with motors from an electric scooter. "The servos used for the tail and ears are simply not powerful enough to move a fairly substantial robot around," explains Richard. So he uses e-scooter motors connected to an H-bridge to move the robot around.

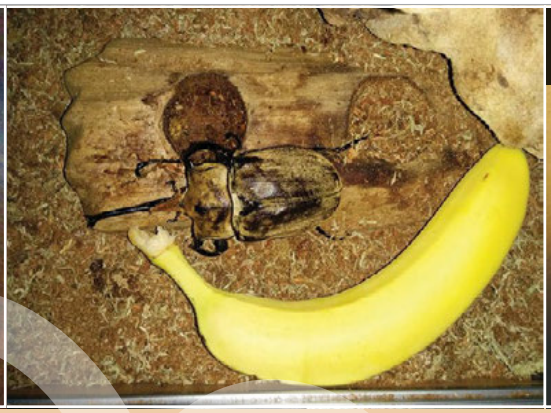
K9 is a lot smarter than most robots. "One of the original goals of building K9 was that he should have a Siri-like function for when he's not a remote presence robot," Richard says. "Apple has put quite a bit of money into developing Siri, so getting an equivalent function working on a Pi is a big ask." Fortunately, Steven Hickson has created a very clever voice command capability, specifically tuned for the Pi, as part of the PiAUISuite (**magpi.cc/22MLXNr**). Behind the scenes, it uses Google voice recognition and Google Translate text-to-speech software to enable the Pi to understand simple spoken commands and respond using voice.

"I've managed to get the basics working," reveals Richard. "You can now hold a conversation with K9, even if there are some embarrassing pauses while he thinks."

YOU'LL NEED

- E-scooter motors
- IR sensors
- Node-RED
- PiAUISuite





The Beetlecam is mounted on the roof of the elephant beetles' tank using Velcro tabs

BEETLECAM

Monitoring nocturnal insects with a Raspberry Pi Camera

Elephant beetles may not be the most strokeable pets, but Will McGugan is a freelance software developer in Edinburgh who loves them all the same. "These insects are mostly nocturnal," he tells us. "During the day, they tend to burrow under their bedding material or hang out on a branch. But during the night, they can be quite active. I know this because in the morning they have rearranged the branches in their tank."

Will wanted to make use of the Pi's Camera Module to create a webcam. "I also wanted to be able to create time-lapse movies, because I didn't want to watch 12 hours of video to see what they get up to at night." The result was Beetlecam (**magpi.cc/22MLURG**).

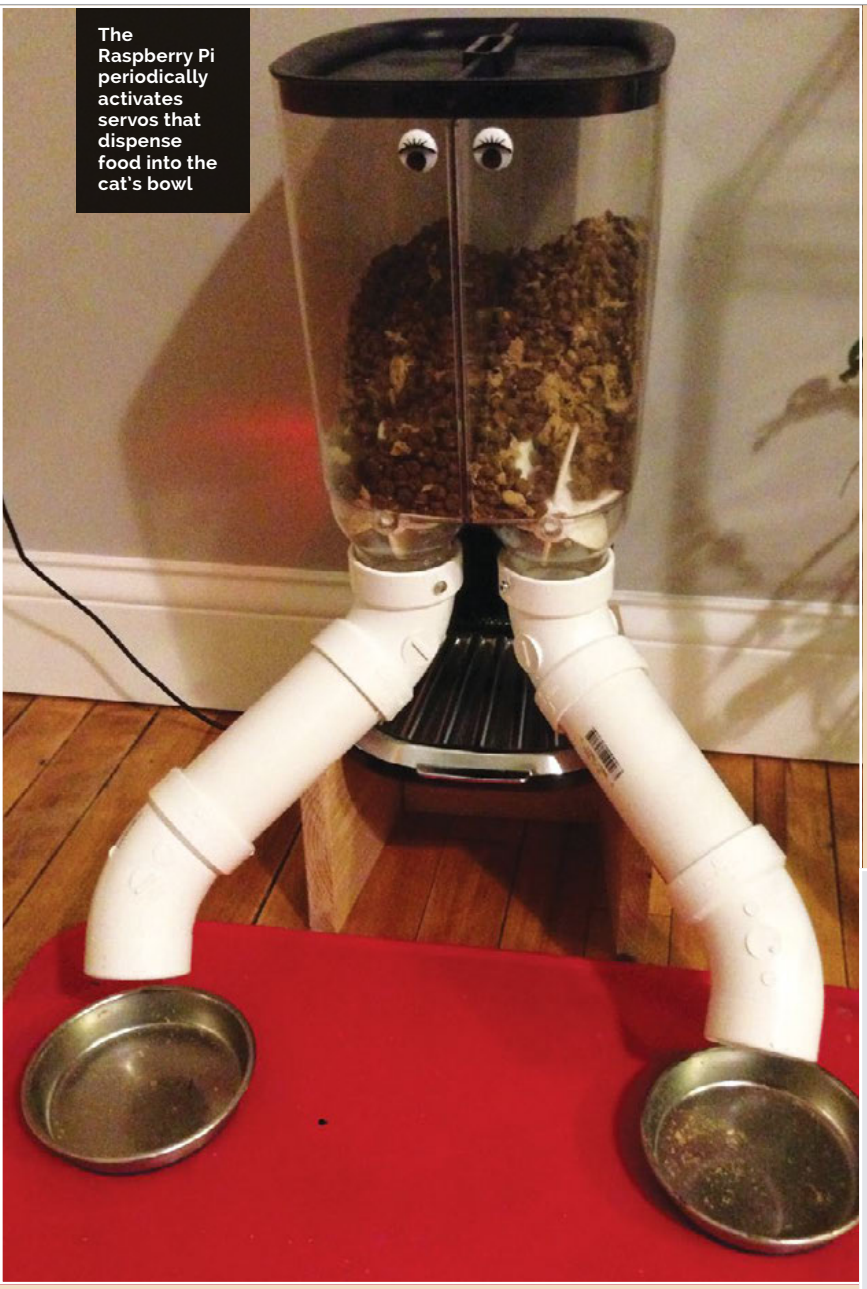
SNIFFUR

Keep track of your pet, using beacons to triangulate their position

We looked at Matt Reed's Sniffur project in *The MagPi* 42, and it's still an impressive piece of kit. Sniffur uses a Bluetooth beacon and Raspberry Pis to triangulate the position of Bean, the dog at Matt's office. But all dogs like to get out and play. "When [greyhounds] do, they're very hard to catch because they're so fast," says Matt. "The need to know where she is at any moment and see if she's close to the front doors is the reason Sniffur was built."

Three Raspberry Pis are used to monitor the beacon device attached to Bean's collar (**magpi.cc/28L5NIe**).

Left A beacon is attached to Bean's collar, and three Raspberry Pi devices are used to triangulate the dog's position



The Raspberry Pi periodically activates servos that dispense food into the cat's bowl

PI-POWERED CAT FEEDER

Keep your cat fed with this automated cat feeder

Anybody with a cat knows how useful a cat feeder is. David Bryan took things one step further by building this Pi-powered cat feeder (**magpi.cc/28KAD01**).

"It's simple," says David, "and can be easily assembled in about 4-6 hours once you have all the parts."

CATWHEEL

Build a giant hamster wheel for your moggy

We looked at Jasper's Cat Exercise Wheel in *The MagPi* 45, and it's still one of the coolest pet tech projects around. This giant mechanised wheel uses a laser pointer to attract cats, and then a motor turns the wheel (**youtu.be/dbPTwewy1SA**). The Pi gathers data on motion and reports back via a web interface.

Keep your cats trim with this smart exercise wheel

PETBOT

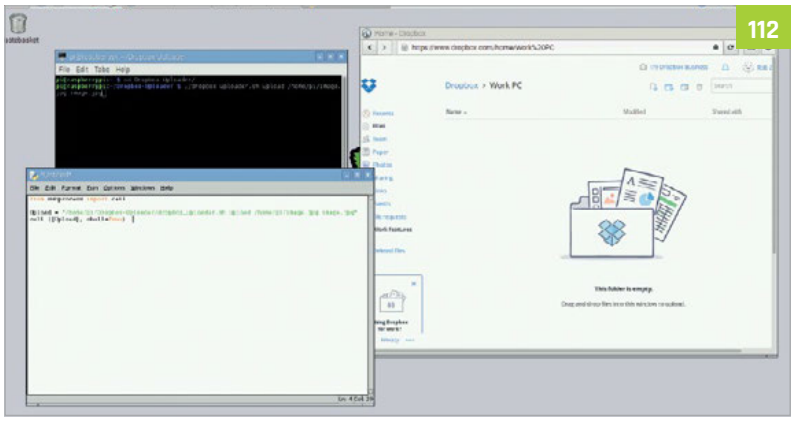
Keep an eye on your pet with this open-source robot

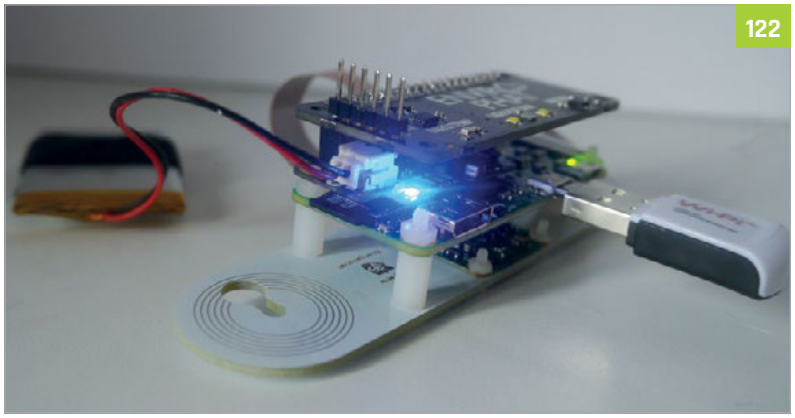
PetBot (**petbot.com**) is the only commercially available project here, but don't worry: it's all open-source, and you're free to use the source code to build your version (or buy a pre-built kit). PetBot trundles around the floor of your house and enables you to interact with your pets from afar. It comes with a remote-controlled webcam, image recognition software, and treat dispenser, and is powered by the Raspberry Pi.

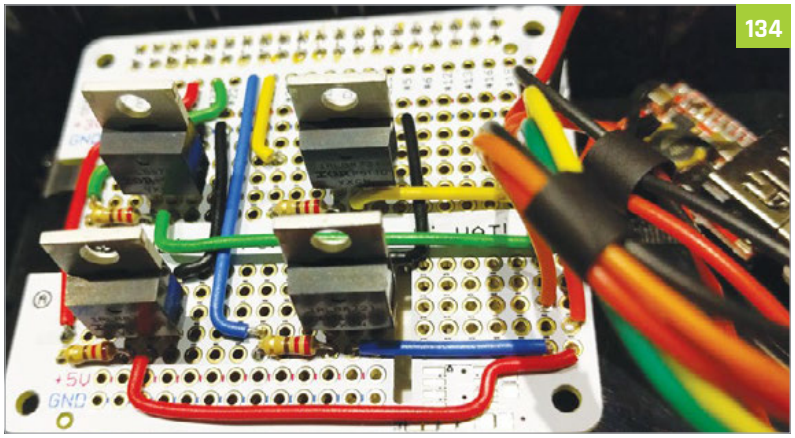
Play with your pet remotely with PetBot, a robot designed for pet interaction

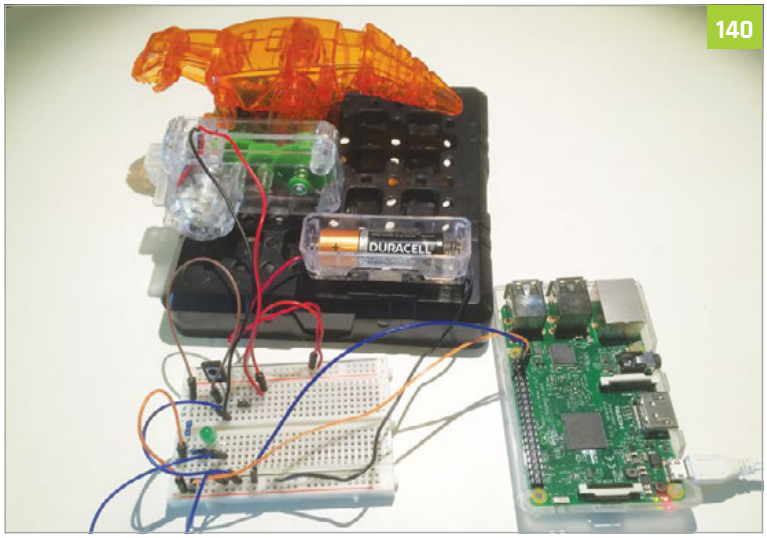
TUTORIALS

Get making with our expert guides and learn how to make the project of your dreams









Tutorials

INCREDIBLE RASPBERRY PI PROJECTS

Get started with digital making with these incredible projects

Digital making is really important to the Raspberry Pi community. It's not enough for us to just buy a Raspberry Pi and just set it up. We have to make stuff too.

Here at *The MagPi*, we want to help people make things. Making is fun. It's about getting your creative juices flowing.

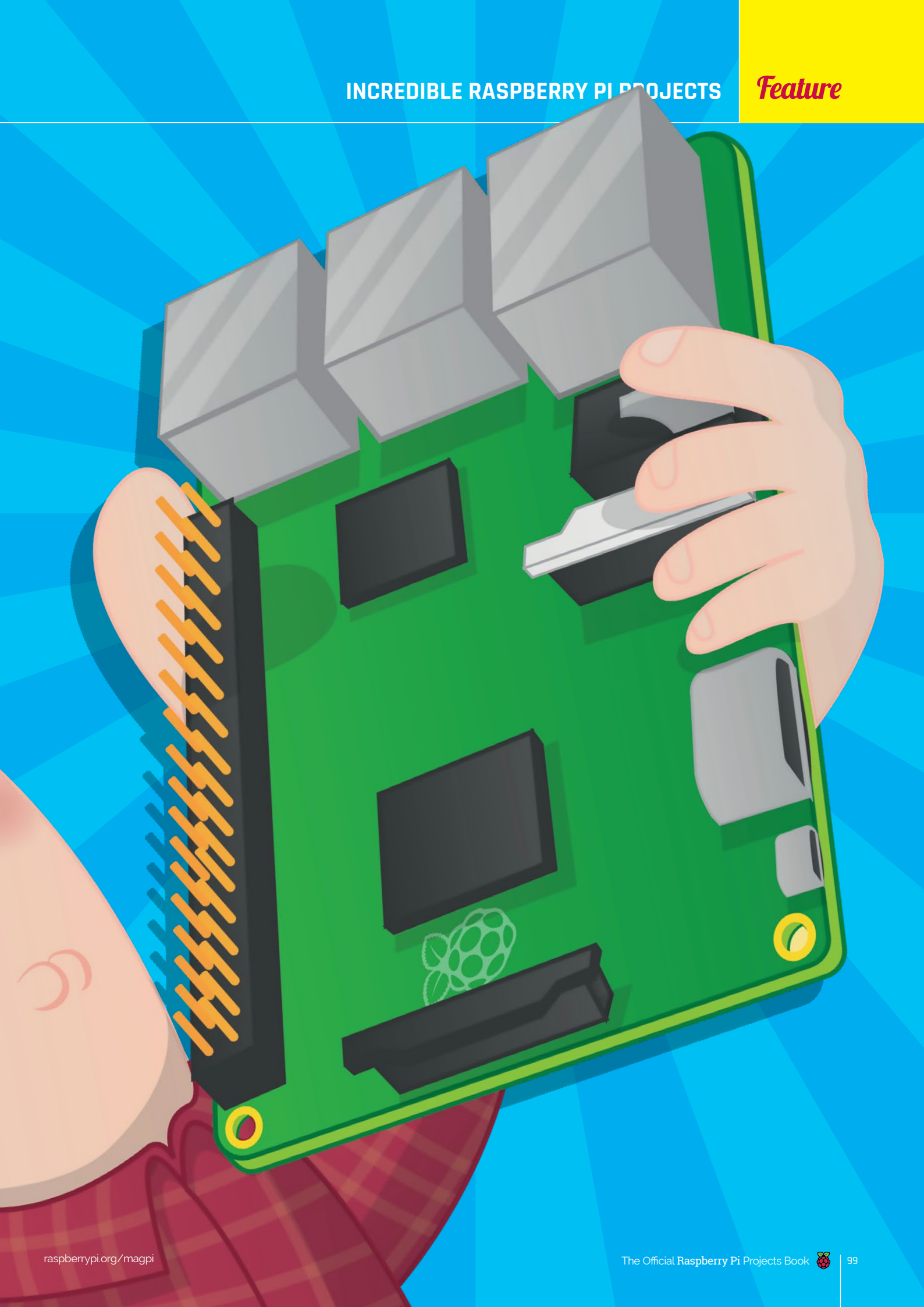
In the following pages, we present a stack of ideas for things to research and make with your Raspberry Pi.

These are practical projects that anybody can make. Some are simple kits like the LEGO NES case; others are *Blue Peter*-style hacks like the Pi-powered binoculars.

They're designed to be challenging, but definitely doable.

More advanced makers will want to look at Sixlab's YouTube Drone. This combines drone technology with a Pi Camera and 4G connectivity. It's the cutting edge of maker-tech.

There's little more important than being a digital maker in the modern world. The world is full of the 'new shiny': toys and trinkets that people pay for but never truly understand. It's way more important to make your own things. They may not always be as shiny, but they're worth so much more.



Beginner

BUILD AN EASY ROBOT

Get a CamJam robotics kit and make a wheeled rover in an afternoon

MIKE AND TIM

Michael Horne and Tim Richardson run the Cambridge Raspberry Jam and are active members of the Raspberry Pi community.
camjam.me

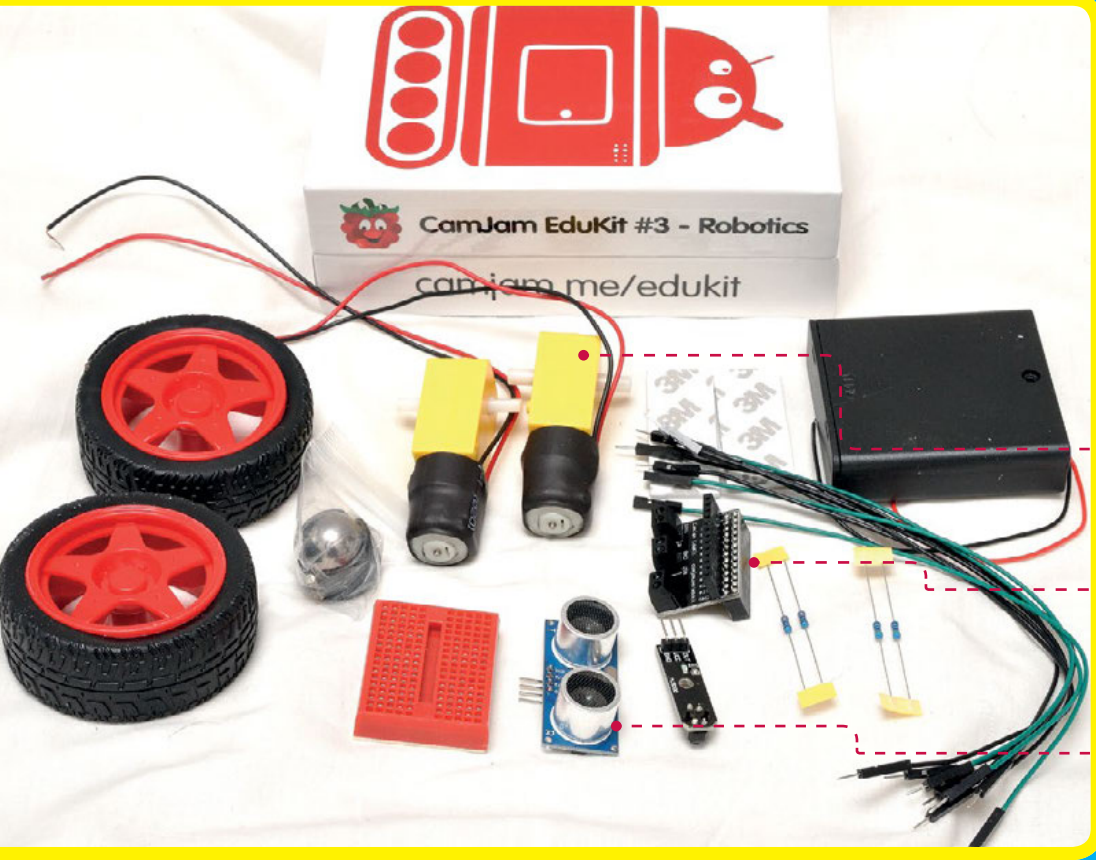
Building a robot is a dream for many new Raspberry Pi owners, and it's way easier to get started than you think.

One of our favourites is the EduKit 3 Robotics by Cambridge Raspberry Jam. The kit is designed by Michael Horne and Tim Richardson, the guys who run CamJam and Pi Wars.

Inside the box, you get a custom motor controller made by 4Tronix (**4tronix.co.uk**). This sits between the two DC motors and a battery compartment, which takes four AA batteries – make sure you buy high-quality batteries, as cheap ones often don't have enough juice to move the wheels.

The kit comes with two sensors: an SR-04 distance sensor and a line-following sensor. The only thing you don't get is a chassis. This is part of its charm, though, as you can build a robot out of any box capable of containing the Raspberry Pi. It's even possible to use the box that all the components come in.

What makes the CamJam EduKit 3 better than other robotics projects is the quality of the instructional material. There are ten different worksheets, covering everything from building the robot to driving the motors and setting up the sensors.



These two yellow motors power the red wheels and are used to move the robot around

A custom motor control unit is included. This enables you to control the robot using Python code

Two sensors are included with the kit. A distance sensor lets the robot stop before hitting a wall, and a line sensor enables the robot to follow a line drawn underneath it

The NesPi kit contains 111 pieces of LEGO and a 30-page manual. Like all LEGO kits, it's a lot of fun to put together as you build the case

Putting the parts together results in this Raspberry Pi case. It looks like a classic NES console, but with access to the Raspberry Pi ports. The lid (used for cartridges in the original NES) provides access to the GPIO pins

BUILD A NES FROM LEGO

Make a retro gaming console from toy bricks

The Danish company's plastic bricks are the toys that many makers love the most. LEGO has provided a sturdy start in life for many a maker.

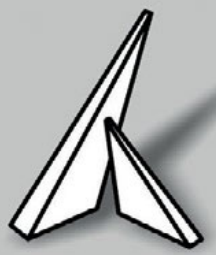
The idea of building a case for the Raspberry Pi from LEGO is by no means unique. There are many, many designs on the internet for Raspberry Pi LEGO cases.

> "A tender homage to the 1985 Nintendo Entertainment System"

But we like this NesPi design the best. Made from 111 pieces of regular LEGO, it is a tender homage to the 1985 Nintendo Entertainment System.

The result is a design that looks close to a NES, but has openings on the side enabling access to all the ports on your Raspberry Pi. This makes it possible to connect USB peripherals and the HDMI cable.

You can pick up a NesPi for just under €35 (including VAT) from the Spanish retailer, RaspiPC (**magpi.cc/2mj6QAJ**).The website is in Spanish, but it's easy enough to navigate (leave the telephone field blank, as it may struggle with international phone numbers).



BERNARDO MARTINEZ

Bernardo is a digital maker who runs RasPi PC, a Spanish retailer specialising in Raspberry Pi equipment. He designed the NesPi kit.
magpi.cc/2mj9fLJ

Alternatively, if you already have the required LEGO pieces, it's possible to create a similar-looking NES case from scratch. One maker, André Rinas, used the LEGO Digital Designer (**ldd.lego.com**) to create a similar case. You can find the instructions on his website (**magpi.cc/2mjhM1b**).

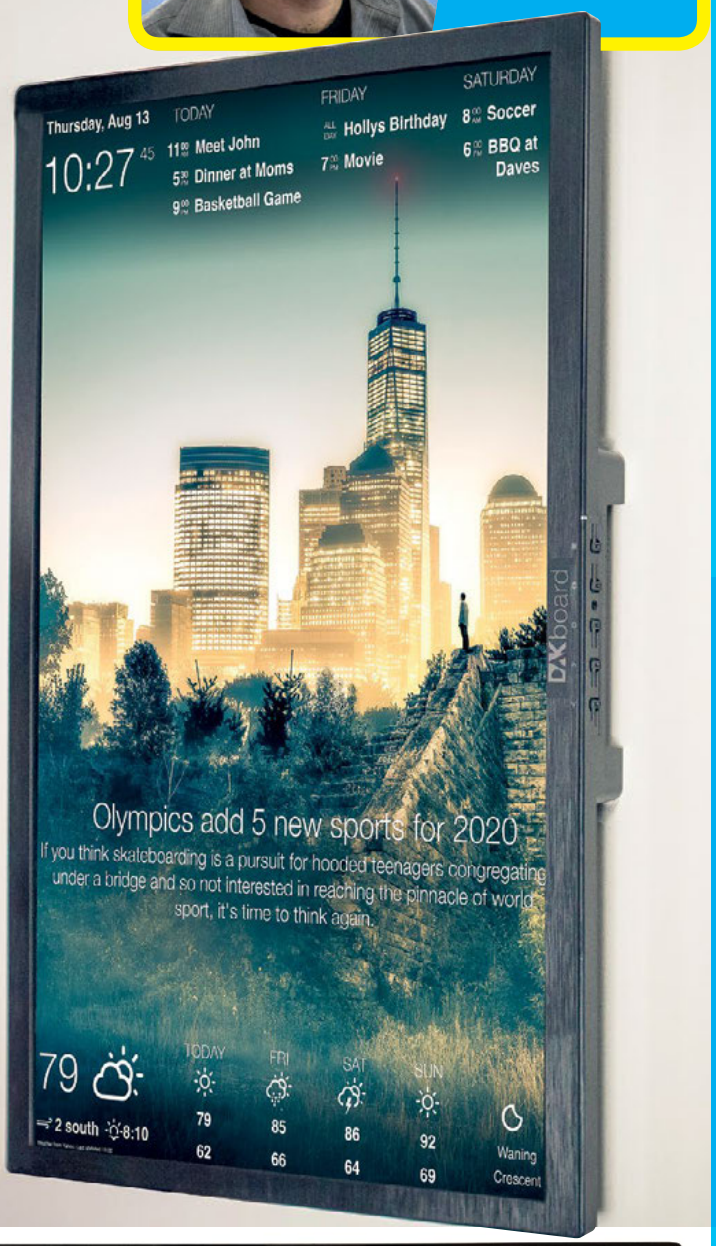
Intermediate

CREATE A WALL DISPLAY

Turn an old monitor or picture frame into a connected wall display with DAKboard

DAN KING

Dan is a software developer and digital designer from Syracuse, New York.
magpi.cc/2mJz9e4



YOU'LL NEED

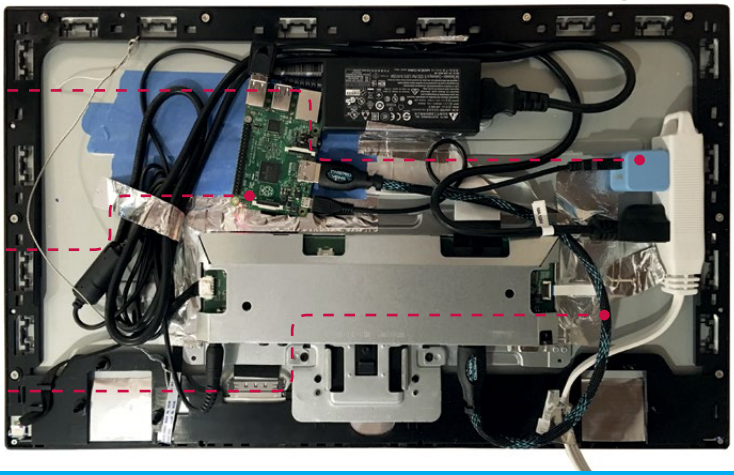
- Raspberry Pi computer (plus WiFi dongle, if using the Pi version 1 or 2)
- 8GB+ SD/micro SD card
- Micro USB charger (for Raspberry Pi)
- Power extension cord
- Photo frame wire

Hooking up a Raspberry Pi to a monitor is one of the easiest things to do. In fact, that's pretty much what you do when you first set up a Raspberry Pi.

That's one of the things that makes DAKboard such an enticing project. DAKboard is a gorgeous web interface that displays photographs, weather and other information (such as events from your calendar or Wunderlist to-do list).

Start with a fresh installation of the Raspbian OS on your Raspberry Pi.

Begin by connecting to a wireless network. Click on the networking



A power adapter is attached to the rear of the display, allowing both the monitor and USB adapter to be hooked up to a single cable

The Raspberry Pi is fixed to the rear of the display. The DAKboard software is set up to launch in full-screen mode when the power is turned on

A short HDMI cable is used to connect the Raspberry Pi directly to the HDMI input on the display

icon in the Menu bar and choose your wireless network.

Now open the Chromium web browser and set up an account at **dakboard.com**. You'll be taken to the DAKboard app interface. Before going any further, click the Options tab in DAKboard. Click on your account name (in the menu to the left) and choose Account. See the bit where it says 'Private URL' – you'll need this URL later, so minimise Chrome for now.

Configure Raspbian

Back in Raspbian, choose Menu > Preferences > Raspberry Pi Configuration. Ensure that Boot is set to Desktop and Auto Login is ticked.

We're going to turn SSH on, so it's a good idea to click Change Password first. Enter a new password for your wall display so you can access it remotely. Choose Interfaces and set SSH to Enabled.

Finally, click Localisation and Set Locale. Ensure that the Country is set to your location; it's 'GB: Great Britain' by default. Click on Set Timezone and choose the correct time zone for your locale; it's UTC (which is the same as GMT) by default.

Now we'll hide the mouse pointer after a few moments of inactivity. Open Terminal and enter:

```
sudo apt-get install unclutter
```

Next, make a couple of system configuration changes.

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

...and add:

```
# Display orientation.
Landscape = 0, Portrait = 1
display_rotate=1

# Use 24 bit colors
framebuffer_depth=24
```

Use **CTRL+O**, **ENTER**, and **CTRL+X** to save and exit Nano.

Now we want to force the screen to stay on, and load the Chromium browser with the DAKboard website on boot. Enter this in Terminal:

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

Delete the contents of the **autostart** file and replace it with this:

```
@xset s off
@xset -dpms
@xset s noblank
@chromium-browser
--noerrdialogs --incognito
--kiosk http://dakboard.com/
app/?p=YOUR_PRIVATE_URL
```

After saving the file and exiting nano, enter **sudo reboot**. The Raspberry Pi will restart and when it boots, you will see DAKboard running in full-screen mode (without a mouse pointer).

The next step is to customise the DAKboard interface. Click on Options and use the Date/Time, Calendars, Photos, Weather, and News options to customise the display.

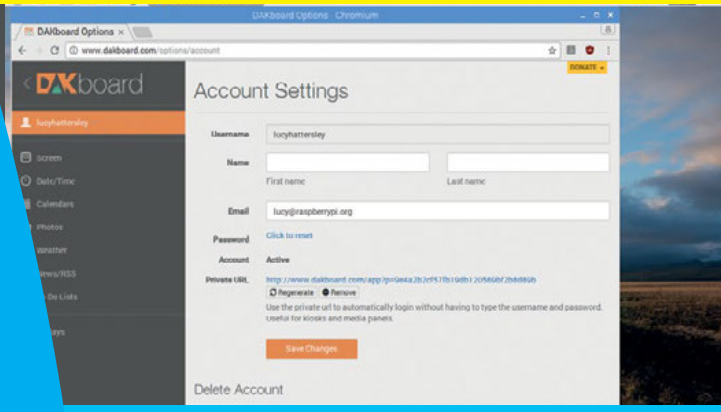
Now you need to fix your Raspberry Pi to the rear of your monitor and hang it on the wall. Dan from DAKboard suggests an IPS display: "You'll still be able to see it when viewing at an angle greater than 90 degrees," he explains.

If you plan to wall-mount the monitor, make sure the USB ports face down and not straight out (or the cables will stick into the wall).

Dan recommends a Dell S2340L 23-inch screen. "The plastic case is perfect for this setup," he says. "There's almost no bezel on the front, and the back cover pops off easily, leaving the frame which I then attached the photo frame wire to use for hanging."

You can find more details on the build on Dan's website (**magpi.cc/2mKaDcO**). DAKboard also sells a pre-built model, which is a monitor containing a Raspberry Pi, for $299.

SET UP DAKBOARD



>STEP-01

Set up DAKboard

Using a regular monitor and keyboard, set up the DAKboard software on your Raspberry Pi. Make sure it's connected to your wireless network and set up SSH so you can access it remotely.

>STEP-02

Fix the Raspberry Pi

Attach the Raspberry Pi to the rear of the monitor. Using a short HDMI lead will help to contain the cables. A professional edition uses the DSI cable on the Raspberry Pi.

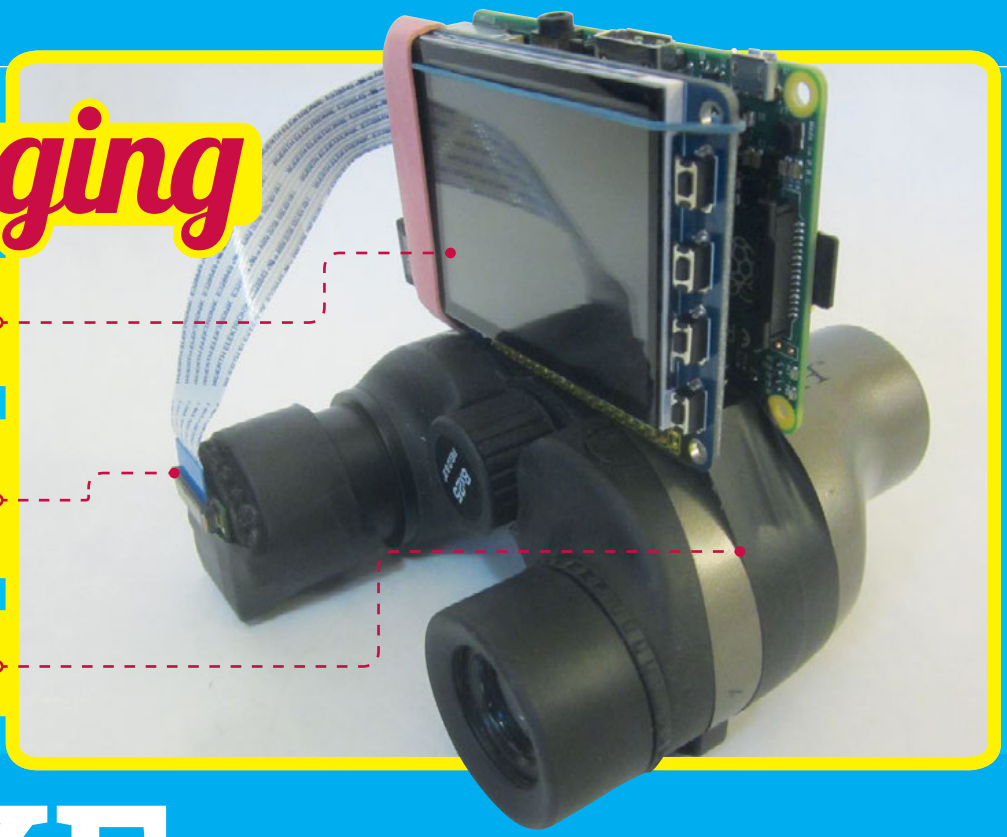
>STEP-03

Wall-mount the display

Now mount the monitor on your wall. Most monitors have brackets available for this purpose. You can also remove the bezel and use photo frame wire to hang the screen.

Feature

Challenging



MAKE PINOCULARS

The PiNoculars project is an excellent way to recycle an old pair of binoculars into a high-tech zoom recording device

YOU'LL NEED

- Binoculars
- Raspberry Pi
- Camera Module
- Adafruit touchscreen display
- Foam

The Raspberry Pi Camera Module is a great tool for digital making. With it you can quickly add an 8-megapixel camera to the Raspberry Pi board.

It connects directly to the Raspberry Pi using the CSI camera interface, a thin cable socket on the Raspberry Pi board.

The Camera Module is fantastic for video and still photography projects, such as time-lapse cameras. OpenCV (**opencv.org/**) software makes it easy to add computer vision to your projects. With this, you can train a Raspberry Pi to identify objects and react accordingly.

The PiNoculars project is one of our favourite Camera Module hacks.

Created by digital maker Josh Williams, PiNoculars are a regular set of binoculars, with a Raspberry Pi Camera module fixed over one of the eyepieces.

The Raspberry Pi is connected to the top of the binoculars along with a touchscreen display. The whole assembly is powered by a Goal Zero AA battery pack (**magpi.cc/2m9w6xr**).

"I was on 18-hour road trip back from Colorado to Michigan with my wife, and I was restless," Josh tells us. "I had brought the Raspberry Pi and Pi Camera Module along to play around with time-lapse photos in the mountains."

When Josh looked at his pair of binoculars, he had a brainwave. "Raspberry Pi and Camera Module and duct tape made for a crude prototype," says Josh.

When he got home, Josh set about refining the build. He now has

JOSH WILLIAMS

Josh helps run local maker spaces (All Hands Active and Maker Works). He builds amazing places for people to work on their projects.
magpi.cc/2m9A4pz

The finished project enables you to record long-distance video footage using an intelligent device

detailed instructions for two different types of PiNoculars. One follows the duct tape and foam route, while the second is a more complex build using laser cutting to create a mount for the Raspberry Pi and touchscreen.

"I used Adafruit's [PiTFT] capacitive touchscreen," says Josh. "Their tutorials made it incredibly easy to attach to the Pi (**magpi.cc/2mCpVy3**). Josh suggests that makers read Adafruit's DIY WiFi Raspberry Pi touchscreen camera tutorial by Phillip Burgess and the Ruiz Brothers (**magpi.cc/2m9Bxfv**).

The most time-consuming part of the build is creating a mount that wraps around the pair of binoculars, but you can skip all this by using foam and duct tape.

After setting up the Raspberry Pi with the Camera Module and Adafruit touchscreen, the whole kit is mounted on top of the binoculars.

The first step is to mark up and cut out a circle of foam. This serves as a mount for the Pi camera, to hold it in front of the eyepiece.

"The camera mount should barely cover the eyepiece," says Josh. It doesn't have to be perfect, but try to position the camera mount as close to the centre of the eyepiece as possible.

Josh then uses rubber bands to hold the Raspberry Pi and screen unit in place on top of the binoculars, and duct tape to fix the binoculars and Raspberry Pi together. "Be careful not to crush your LCD," he says. More duct tape is used to attach the foam mount over the PiNoculars eyepiece.

> "There are a number of people who've combined the Raspberry Pi with microscopes and telescopes"

Then it's just a case of moving everything around until it is firmly fixed, and the camera can record a well-defined circle.

If you're a perfectionist, there's a much more detailed method, involving precision design with Inkscape (**inkscape.org**) and a laser cutter. But we like a quick hack, and this is a great project for making something quick and impressive.

"There are a number of people who've combined the Raspberry Pi with microscopes and telescopes," says Josh, who's fascinated by them.

"Remember to measure twice," he warns us, "and callipers are beautiful tools."

ATTACH A CAMERA MODULE

>STEP-01

Set up the camera and touchscreen

First, attach a Camera Module to your Raspberry Pi board and set up a capacitive touchscreen display. This will enable you to control the project on the move.

>STEP-02

Attach the camera

Cut out a foam circle the same size as the eyepiece on your binoculars (it needs to cover the eyepiece completely). Mark an X in the centre and draw an 8×8mm square in the centre of the circle. Push the Camera Module into the foam and position it in front of the eyepiece.

>STEP-03

Fit it together

Use rubber bands to hold the Raspberry Pi and touchscreen display vertically on top of the binoculars, with the screen facing towards the eyepieces. Now use duct tape to secure the Raspberry Pi firmly, and a smaller piece to secure the foam-encased Camera Module. A portable power back enables you to take the PiNoculars out in the field for long-range video recording.

Feature

Challenging BUILD A YOUTUBE DRONE

Add a Camera Module to a drone to broadcast footage straight to YouTube

MAHMUT

Mahmut is a computer engineer and maker. He is working with his friend, Metin, on Sixfab Raspberry Pi 4G/LTE projects. **magpi.cc/2nSlpvp**

YOU'LL NEED

- Drone Kit (and RC controller)
- Sixfab 4G/LTE shield
- Camera Module

This smart drone broadcasts footage over a cellular data connection direct to YouTube.

The YouTube Drone is a great project because it incorporates so many different Raspberry Pi features: video recording, flight control (either RC or automated), YouTube integration, and data network access.

It's also fundamentally inspiring. Hooking up a drone to a Raspberry Pi with network coverage lets you test a lot of future technologies.

There's potential to integrate it with computer vision technology, as well as automatic flight.

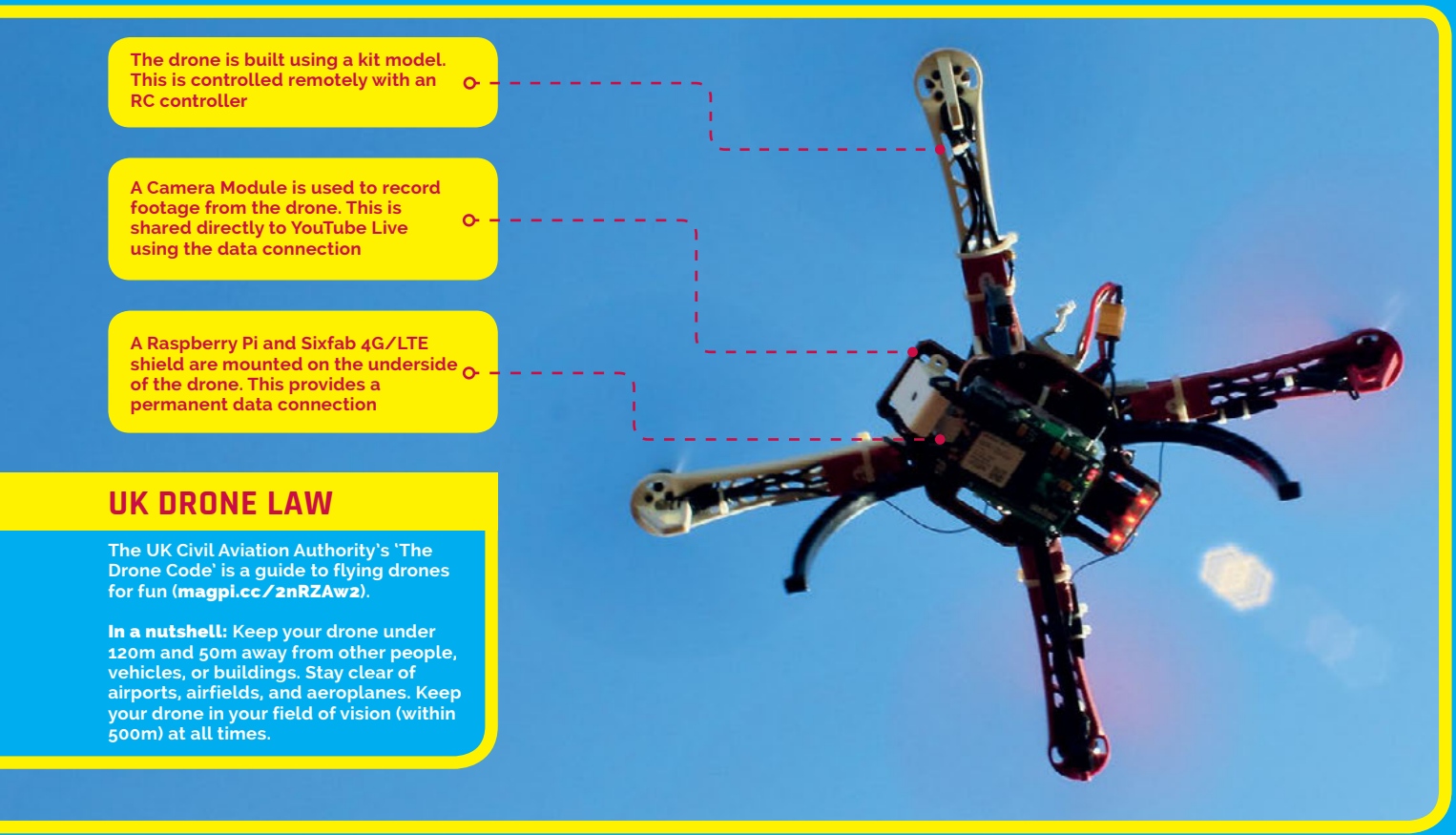
This project was built by Mahmut and Metin, a computer engineer from Sixfab and his maker friend.

"The Sixfab 4G/LTE shield offers high-speed internet connection to Raspberry Pi," says Mahmut.

Mahmut and Metin started with a drone kit: a Super Multicopter Set – Make Yourself Drone Kit from Robotistan (£180/$220, **magpi.cc/2mgXMjV**).

The drone set needs an RC controller with four channels. Robotistan recommends a RadioLink AT9 model RC controller (**magpi.cc/2mh8gQ8**).

The drone comes with a KK2 Multi-rotor LCD flight controller board (**magpi.cc/2mh0CWa**). The flight controller has GPIO pins, so it is possible to create

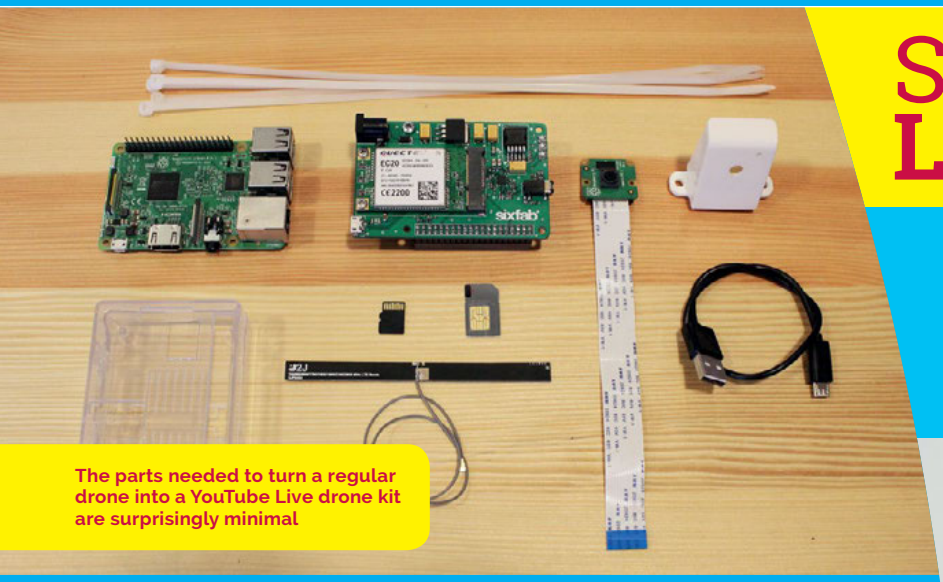


UK DRONE LAW

The UK Civil Aviation Authority's 'The Drone Code' is a guide to flying drones for fun (**magpi.cc/2nRZAw2**).

In a nutshell: Keep your drone under 120m and 50m away from other people, vehicles, or buildings. Stay clear of airports, airfields, and aeroplanes. Keep your drone in your field of vision (within 500m) at all times.

STREAMING LIVE VIDEO



The parts needed to turn a regular drone into a YouTube Live drone kit are surprisingly minimal

an automated drone, although Mahmut and Metin's project currently flies via remote control.

The Raspberry Pi community discusses using a Pi with the KK2 flight controller in this forum thread: **magpi.cc/2nScpGB**.

You're not limited to the Super Multicopter. "You can use any drone capable of carrying an extra 200-250 grams," explains Mahmut.

After building the drone kit, the Raspberry Pi is connected to the Sixfab 4G/LTE shield. The Camera Module is attached to the Raspberry Pi and the whole assembly is mounted to the underside of the drone. Mahmut has produced a video tutorial demonstrating the build **(youtu.be/F44R5PaV25M)**.

And an SD card image has been produced especially for the project (**magpi.cc/2nRY7FQ**).

Setting up the Sixfab 4G/LTE shield is relatively easy. A driver is available direct from the Sixfab website (**magpi.cc/2nRW9W3**). Copy the files to **/lib/modules/4.4.21-v7+/kernel/drivers**, reboot the Raspberry Pi, and run **dmesg** in a Terminal window.

Prospective makers will find this Sixfab tutorial helpful: YouTube/Facebook Live Video Streamer with Sixfab 3G/LTE shields (**magpi.cc/2nRZJzp**).

Please be careful when building a drone. It is important to not to make the connections between propellers and motors before it is complete and fully secured. Wear safety glasses when testing the drone, as spinning blades are dangerous.

"It is very easy for people with all the materials to do the project," says Mahmut. "Because it's very easy to use the Sixfab products and everyone can take it and do whatever they want with ease."

> "Those who want to do the project should not hesitate. They can easily do it with the videos"

"The first time we did it we used a smaller drone and it would not go up," he tells us. "Other than that, we have not had a problem. But during the flight, the wind made our work very difficult.

"Those who want to do the project should not hesitate. They can easily complete it with the videos and the image file."

The current project uses the Pi and Sixfab shield 4G/LTE connection for YouTube Live streaming. "But our goal is to also control the drone via the network connection," says Muhmut.

>STEP-01
Set up the shield

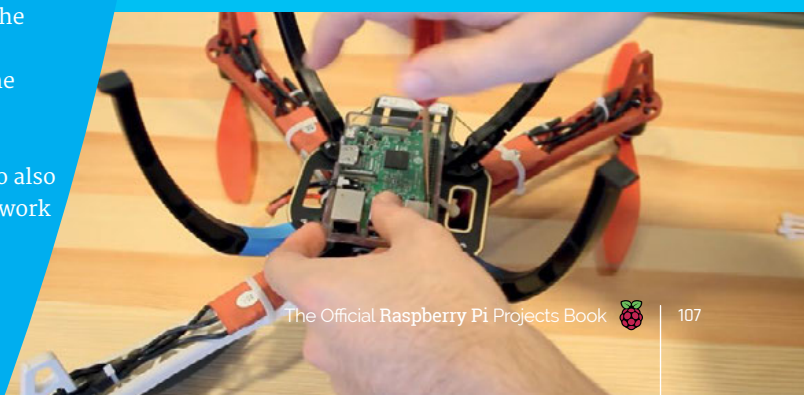
A Sixfab 4G/LTE shield is connected to the Raspberry Pi. This provides a persistent data connection, enabling the Raspberry Pi to stream data to YouTube and other websites.

>STEP-02
Attach the camera

A Camera Module is attached to the Raspberry Pi board. Meanwhile, a USB cable is used to connect the Sixfab shield to the Raspberry Pi. The LTE band provides a more powerful antenna for the shield, so it maintains a steady data connection.

>STEP-03
Mount the device

The Sixfab shield and Camera Module are mounted on the underside of a quadcopter. A battery unit is mounted on the top of the drone to provide steady power.



THE TOP TEN RASPBERRY PI KITS

Want to build a project, but don't want to make everything from scratch? Here are some of the best kits around. Everything from retro gaming systems to wildlife cameras can be bought in kit form

PICADE

£180 $238

magpi.cc/2nSOfvv

Creating a retro arcade system is a dream for many makers. Building a cabinet from scratch can be a costly and difficult enterprise. Picade makes it all a lot easier, though. Its powder-coated cabinet feels like a real arcade cabinet. It comes with a PCB, joysticks, and 12 arcade buttons.

RASPBERRY PI 3 RETRO GAMING BUNDLE

£65 $86

magpi.cc/2nSXHPs

If you're looking for a retro gaming console, then look at this Retro Gaming Bundle from The Pi Hut. It contains a Raspberry Pi 3 and two SNES-style USB gamepads. You also get a long HDMI cable, an official Raspberry Pi power supply, and a 16GB SD card. All you need to do is install RetroPie (**retropie.org.uk**).

PIRATE RADIO

£40 $50

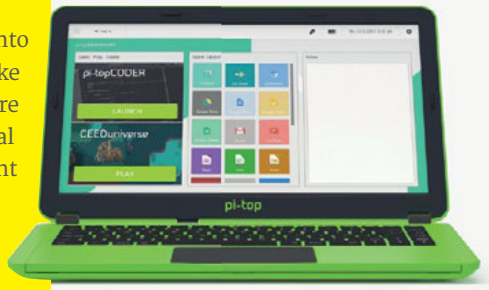
magpi.cc/2nSEOfE

This kit has everything you need to create a radio with your Raspberry Pi. It contains the new Pi Zero W and a pHAT BEAT (DAC and stereo amplifier). You also get a 5W speaker. The plastic case has a VU meter so you can view the sound levels. You will need to solder the GPIO header on to the Pi Zero W, and a female header to the pHAT BEAT.

PI-TOP

pi-top: £215.98 / $284.99
pi-topCEED: £109.99 / $164.99

pi-top.com



The pi-top and pi-topCEED are projects to turn your Raspberry Pi into a laptop or desktop computer. Unlike a regular laptop or desktop, these are module DIY computers that are ideal for mobile making. You can 3D-print the case and buy the components, or get the whole lot in kit form. It's not cheap, but it's designed to teach the basic architecture of a computer, and there's plenty of documentation and tutorials.

PI ZERO CCTV KIT (LITTLE BRO)

£24 $32

magpi.cc/2yFOcMv

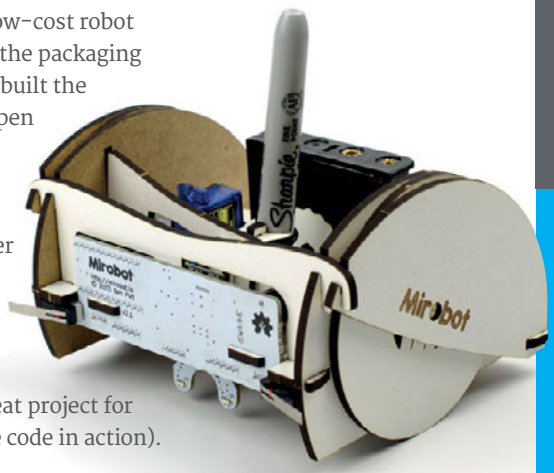
Start your mini surveillance state with this sign that houses a Pi Zero and a Camera Module. The camera logo on the sign has a hole for the Camera Module. You need to buy the latter separately, but can combine it with OpenCV computer vision to create a smart CCTV Camera that recognises people.

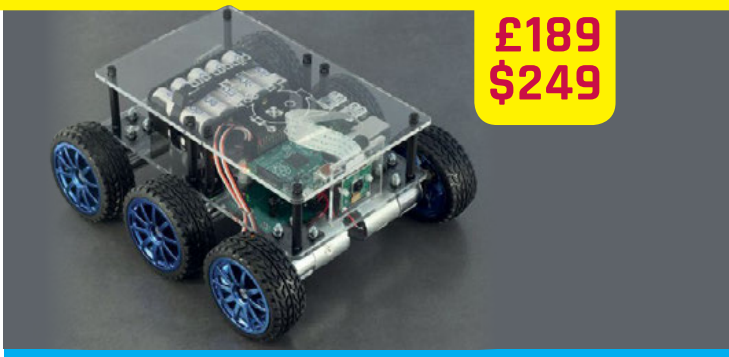
MIROBOT V2

£60
$79

magpi.cc/2nSJMt0

The Mirobot is a small, low-cost robot that you assemble using the packaging it comes in. Once you've built the robot, you add a felt-tip pen and use it to move and draw. The Mirobot acts like your average turtle robot. It can raise or lower the pen to draw shapes on paper. Turtle robots have a long history in computing and maker culture, and they're a great project for learning logic (and to see code in action).





£189
$249

DIDDYBORG

magpi.cc/2mHbmbp

There are many robotics kits for the Raspberry Pi, but PiBorg's DiddyBorg is perhaps the most comprehensive. It has a robust laser-cut chassis with six 60rpm motors. The DiddyBorg comes with a PicoBorg Reverse motor controller and a BattBorg power management unit. You need to add a Raspberry Pi and Camera Module, and you have the option to control the robot using a PlayStation 3 gamepad.

WILDLIFE CAM KIT

£130
$172

naturebytes.org

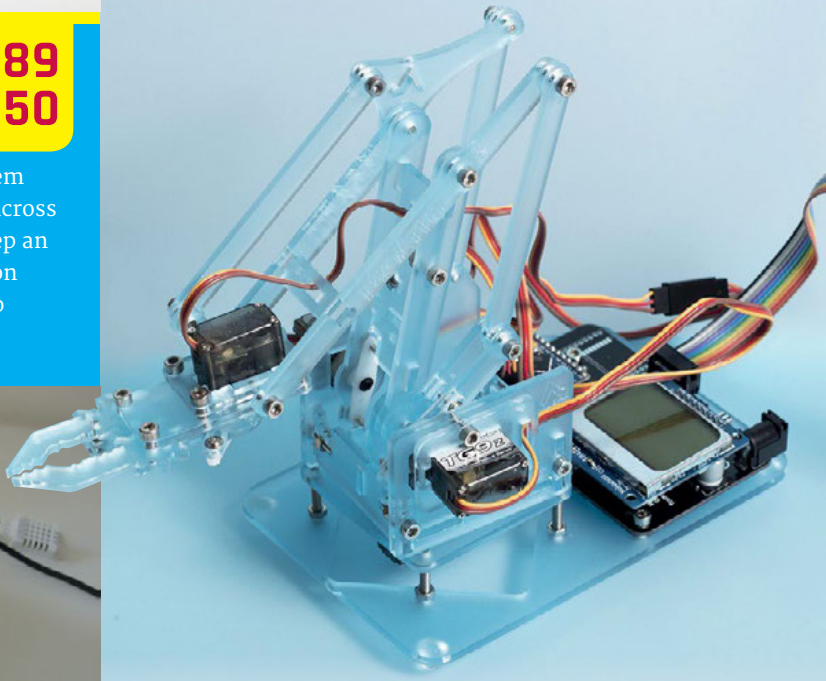
If you've ever wanted to record the critters in your garden, then the Wildlife Cam Kit is the way to go. Its PIR sensor detects movement and triggers the Pi Camera Module to take a stealthy snap. It's ideal for educational use. The Cam Kit is also very versatile and can be used for time-lapse photography, night-time shots (with a Pi NoIR camera and IR LED lighting), or even a live video feed.

MEARM

£32.99
$43.54

magpi.cc/2nSOlDo

Not all robots have wheels. The MeArm is a flat-pack robot arm kit that you build. It can then be controlled via Python or directly with a joystick. It's very easy to assemble, using just a screwdriver, and we think this is a great kit for anyone wanting to step into the world of digital making.



LIV PI

£189
$250

livpi.com

LiV Pi comes from Hong Kong, where pollution has been a problem for many years. With air quality increasingly a concern in cities across the world, it's a great way to learn more about pollution (and keep an eye on levels in your area). Inside the kit are three sensors: carbon dioxide, temperature/humidity, and air pressure. It's not a cheap project, but it is a professional air-monitoring system.

RICHARD SMEDLEY
Having often found words better than pointing at things, Richard stuck with the command line when all around had fled.
twitter.com/RichardSmedley

COMMAND LINE PI

You'll Need

- Raspbian **raspberrypi.org/downloads** – though most of the tutorial will work with the command line running the Linux default Bash shell on any GNU/Linux PC

As close to perfect as Raspbian is, things can go wrong. In this tutorial, we learn that there's no need to turn the Raspberry Pi off and on again: just kill the process!

Ever lost the 'off switch' for a program? Sometimes software you're running seems to have no inclination to stop: either you can't find out how to quit, or the app has a problem and won't respond to your **q**, **CTRL+C**, or whatever command should close it down.

There's no need to panic, and certainly no need to reboot: just identify the process and quietly kill it. We'll show you how, and look at what else can be done with knowledge of processes.

Processes

Find the many processes running on your Pi with the **ps** command. On Raspbian, it's usually called with the **a** and **x** switches which give all processes, rather than just those belonging to a user, the **u** switch shows processes by user, attaching it to a **tty**. **w** adds wider output, and **ww** will wrap over the line end to display information without truncating.

Type **ps auxww** to see, then try with just **a** or other combinations. You'll notice that these options work without the leading dash seen for other commands. Both the lack of dashes, and the letters **a** and **x**, date back to the original Unix ps of the early 1970s; this was maintained through various revisions by one of Unix's two family branches, **BSD**, and baked into the first GNU/Linux ps. Unix's other branch, System V, extended and changed ps with new options and new abbreviations for command switches, so for **ps ax** you may also see **ps -e** (or **-ef** or **-ely** to show in long format).

KEEP ON RUNNING

nohup is useful for a program that will be running for some time in the background – perhaps a sensor project you're working on – until you feel happy enough to add it to Raspbian's startup processes.

The **ps aux** listing has various headers, including the **USER** which owns the process, and the **PID** (process identification number). This starts with 1 for init, the parent process of everything that happens in userspace after the Linux kernel starts up when you switch the Pi on. Knowing the **PID** makes it easy to kill a process, if it's the easiest way of shutting it down. For example, to kill a program with a **PID** of 3012, simply enter **kill 3012**. To quickly find the process in the first place, use **grep** on the ps list. For example, locating vi processes:

```
ps aux | grep -i  vi
```

The **-i** (ignore case) isn't usually necessary, but occasionally a program may break convention and contain upper-case letters in its file name. You can also use **killall** to kill by program name; for example, with **killall firefox**.

Piping commands

Naturally, you can pipe ps's output to select the PID and feed directly to the kill command:

```
kill $(ps aux | grep '[f]irefox' | awk
'{print $2}')
```

We don't have space for an in-depth look at **awk** (we're using it here to print the second field of grep's output, the **PID**), but the **[f]** trick at the beginning

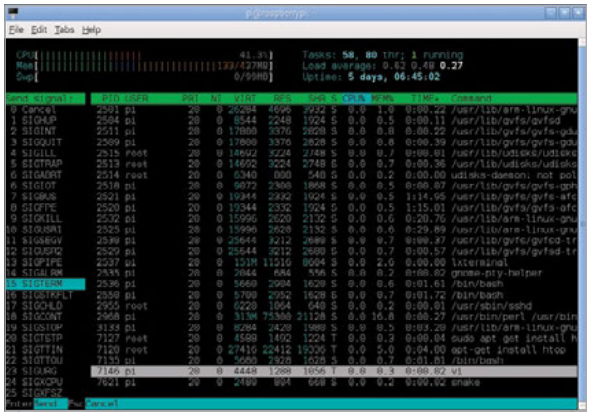


Fig 1 `htop` tells you what's running, what resources it's using, and lets you interact with the process, even killing htop from within htop

of Firefox (or whatever named process you want to kill) singles out the Firefox process. In the vi example above, grep found the grep process itself as well as vi (and anything with the letter sequence vi in its name).

The output of ps also shows you useful information like percentage of memory and CPU time used, but it's more useful to see these changing in real time. For this, use **top**, which also shows total CPU and memory use in the header lines, the latter in the format that you can also find by issuing the command **free**. For an improved top:

```
apt-get install htop
```

htop is scrollable, both horizontally and vertically, and allows you to issue commands (such as **k** for kill) to highlighted processes. When you've finished, exit both top and htop with **q**, although in htop you may care to practise by highlighting the htop process and killing it from there (see **Fig 1**). htop also shows load over the separate cores of the processor if you have a Pi 2 or 3.

Background job

Placing an ampersand after a command in the shell places the program in the background – try with: **man top &** and you'll get an output like **[1] 12768**.

The first number is a job number, assigned by the shell, and the second the PID we've been working with above. man top is now running in the background, and you can use the job control number to work with the process in the shell. Start some other processes in the background if you wish (by appending &), then bring the first – man top – to the foreground with **fg 1**. Now you should see man running again.

You can place a running shell program in the background by 'suspending' it with **CTRL+Z**. fg will always bring back the most recently suspended or backgrounded job, unless a job number is specified. Note that these job numbers apply only within the shell where the process started. Type **jobs** to see background processes; **jobs -l** adds in process IDs (PIDs) to the listing.

Signals

When we sent a kill signal from htop, we were given a choice of signal to send. The most important are **SIGTERM**, **SIGINT**, and **SIGKILL**. The first was the default when we killed from htop, and is the signal that kill sends if not called with a modifier. It tells a process to stop, and most programs will respond by catching the signal, saving any data they need to save, and releasing system resources before quitting.

kill -2 sends **SIGINT**, the equivalent to stopping a program from the terminal with **CTRL+C**: you could lose data. Most drastic is **kill -9** to send **SIGKILL**, telling the kernel to let the process go with no warning. Save this one for when nothing else works.

Mildest of all is the Hang Up (**HUP**) signal, called with **kill -1**, which many daemons are programmed to treat as a call to simply reread their configuration files and carry on running. It's certainly the safest signal to send on a critical machine.

Staying on

nohup will run a program which will continue after the terminal from which it's started has closed, ignoring the consequent **SIGHUP** (**hangup**) signal. As the process is detached from the terminal, error messages and output are sent to the file **nohup.out** in whichever directory you were in when you started the process. You can redirect it, as we did in part 4 (issue 34), with **1>** for **stdout** and **2>** for **stderr**; **&>** is a special case for redirecting both **stdout** and **stderr**:

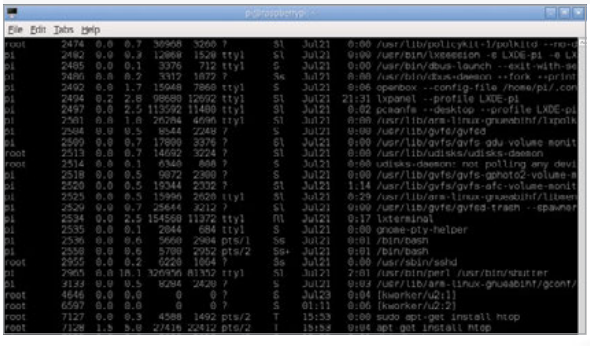
```
nohup myprog &>backgroundoutput.txt &
```

One use of **NOHUP** for Pi users is to be able to set something in motion from an SSH session, that will continue after an interruption to that session. For example, restarting the network connection to which you are connected:

```
sudo nohup sh -c "ifdown eth0 && ifup eth0"
```

Note that the **nohup.out** log file created here will need sudo privileges to read – or reassign with:

```
sudo chown pi:pi nohup.out
```



Above Everything running has a process ID (PID) that can be used to control that program; find them all with `ps aux`

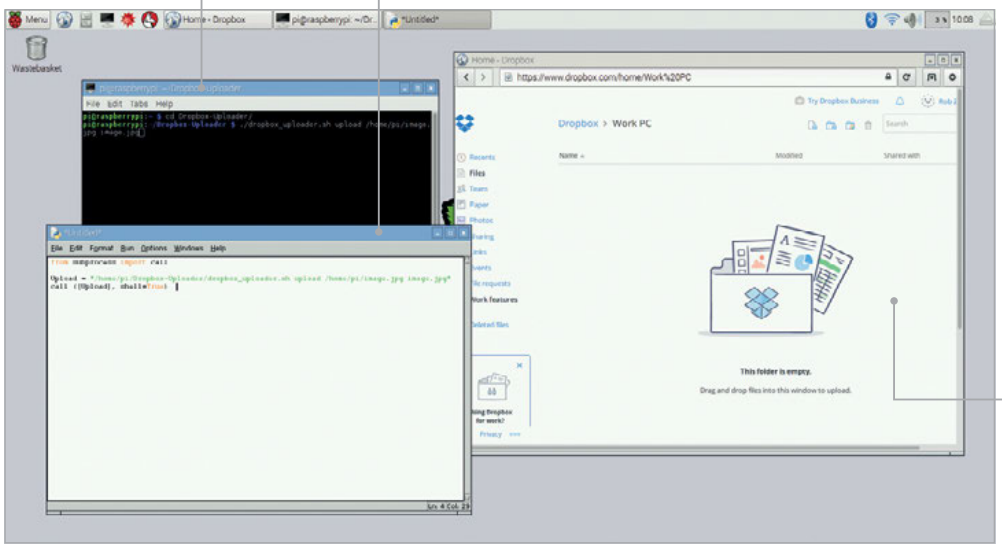
QUICKER BOOT

The startup process of Raspbian is controlled by SysVinit, but like other GNU/Linux distributions will eventually change to the new, faster SystemD. This will change startup processes, but instructions here will still be relevant.

KEEP ON TOP

Using a virtual console, it can be worth keeping htop running so that if there are any problems, you can CTRL+ALT+Fn there for a quick look – even if the GUI freezes.

ROB ZWETSLOOT
Tinkerer, sometime maker, other-times cosplayer, and all-the-time features editor of *The MagPi*.
magpi.cc



GET DROPBOX ON RASPBERRY PI

You'll Need

- Dropbox account **dropbox.com**
- Dropbox Uploader **magpi.cc/2aaHoJN**
- Your Dropbox API key

Connect to the most ubiquitous cloud service on your Raspberry Pi, perfect for uploading pictures and video in a project!

Dropbox's relationship with Linux has always been slightly weird, and as Raspbian is a version of Linux, that too means it's not so straightforward to get the file-syncing behaviour of Dropbox to work. There are definitely ways around this, though, and with a little bit of hacking and tweaking, we can get automatic uploads (and downloads!) of items to Dropbox. This method was created by Alex Eames of **RasPi.TV** and is perfect for many types of Raspberry Pi project, especially those where you're taking pictures and want to view them remotely or free up some space on the Raspberry Pi after they've been taken.

>STEP-01

Get a Dropbox account

If you don't already have one, sign up for a Dropbox account at **dropbox.com**. It offers a couple of GB for free, but you can pay a small amount a month for a whopping 1TB of space. There are some other cloud services around, such as Google Drive, but they have even less Linux support than Dropbox. As with most cloud storage services, you can view, download, and upload files from the browser. So if you want to download anything to the Raspberry Pi, it can be quick and easy to go through there.

>STEP-02

Get Dropbox uploader

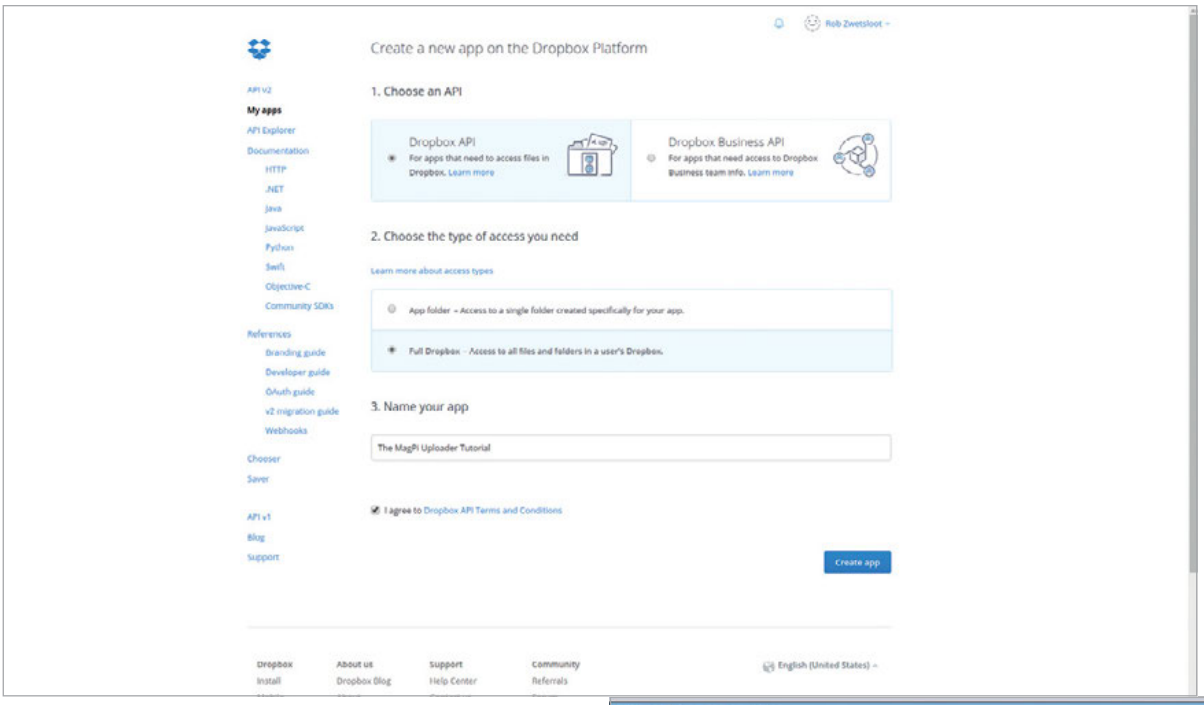
Boot into Raspbian if you're not already using it, and either open a Terminal or SSH into the Raspberry Pi if you prefer. From there, you'll need to download the install files with:

```
git clone github.com/andreafabrizi/Dropbox-Uploader.git
```

Once that's downloaded, you'll need to move to the folder (`cd Dropbox-Uploader`) to begin installing. You can start this off with:

```
./dropbox_uploader.sh
```

It will ask for your API key, which is our cue to move onto to the next step.



Left Creating an app on Dropbox is easy; just make sure it has a unique name so you can get it working

FREEING UP SPACE

In a Bash or Python script using the function, you can always set it to delete the upload once it's sent, to free up space on the Pi.

>STEP-03

Find your API key

You need to head to the developers' section of Dropbox (**magpi.cc/2aaQnKQ**) so you can create a new app and get a unique API key to use on the Raspberry Pi. Click on **Create App** to start.

As we're working towards a personal use application, the first option we'll chose is Dropbox API rather than business. The next two options don't really matter: if you want to access full Dropbox, you can, but it may be better for privacy and security reasons if you're just able to use a specific folder on your Dropbox. Finally, name it whatever you want and click **Create App**.

>STEP-04

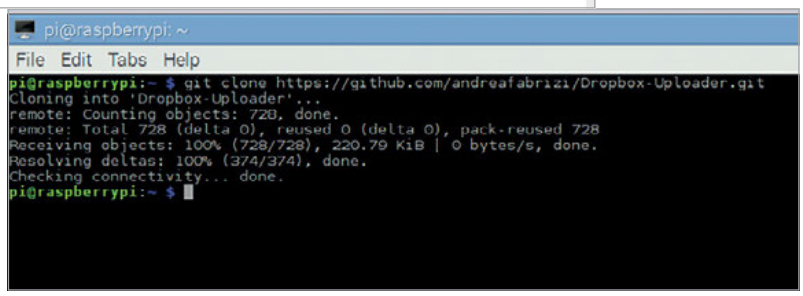
Enter your API key

On the settings page for the app you created, there will be an 'App key' field. Note it down or simply copy and paste it in into the Terminal if you're still on your Raspberry Pi. It will then ask for the 'App secret', which is right below the key in your settings page. Click on 'show' and then enter that. It will then ask you to confirm what type of permission you gave it (full or just a folder) and then it will drop a link to put in the browser to confirm everything. Press **ENTER** to finish the setup and if everything has gone correctly, it will flash up a message to let you know!

>STEP-05

Add a progress bar

Without a progress bar, you won't always know if everything is working. Luckily, you can add one to this project: open up the installed file we just used (**`nano dropbox_uploader.sh`**) and look for the line that says **`SHOW_PROGRESSBAR`** under Default values.



Above From the Terminal, all you need to do is download the project to begin with: it's just a simple git request

If this line ends with **`=0`**, then change the **`0`** to a **`1`** and save the file. There are also some other options under Default values (such as the ability to skip existing files), so have a quick look and see if there's anything else you feel confident to change.

>STEP-06

Start uploading!

Now everything should be working and you can start uploading. Everything revolves around the **dropbox_uploader** file, so stay in the folder or make sure to have your code point towards the folder in the future. The code to upload is something like:

```
./dropbox_uploader.sh upload path/to/file
dropbox_filename
```

You can use this code in Python 3 by creating an OS call, using something like:

```
from subprocess import call
Upload = "home/pi/Dropbox_Uploader/
dropbox_uploader.sh upload path/to/file
dropbox_filename"
call ([Upload], shell=True)
```

Time to get uploading and experimenting!

OTHER COMMANDS

As well as uploading, you can use it to download files. A full list of commands is at: **magpi.cc/2aaHoJN**

VOICE CONTROL ON YOUR PI

Use Amazon's Alexa service on a Raspberry Pi and open up a world of voice-control projects

ROB ZWETSLOOT
Tinkerer, sometime maker, other-times cosplayer, and all-the-time features editor of *The MagPi*.
magpi.cc

You'll Need

- Alexa AVS **magpi.cc/2boDnjB**
- A constant internet connection
- External speaker
- A USB microphone

In the UK, all the *Star Trek* shows are currently on Netflix, reminding us of the desire to ask the computer for Earl Grey tea or Klingon coffee (we can't start the day without a raktajino, you know). So it's exciting to see Amazon's Alexa is quite readily available on the Pi now. Let's get it working, then, and make some projects.

We need to start by installing VLC. Not just normal VLC, though – we need to install this one slightly differently. Open up the terminal and enter:

```
sudo apt-get install vlc-nox vlc-data
```

This might take a while; once its done, we then need to set the environment variables so we can access VLC from Alexa properly later. Do this with:

```
export LD_LIBRARY_PATH=/usr/lib/vlc
export VLC_PLUGIN_PATH=/usr/lib/vlc/plugins
```

Next, it's time to download the Alexa files we need:

```
git clone https://github.com/amzn/alexa-avs-raspberry-pi
```

Now we need to install our dependencies: Node, JDK, and Maven. In the terminal, enter:

```
curl -sL https://deb.nodesource.com/setup | sudo bash -
```

And let it work. It will end by prompting you to install Node.js. Do that with:

```
sudo apt-get install nodejs
```

Next, use **cd** to move to **/alexa-avs-raspberry-pi/samples/companionService** and install npm with:

```
npm install
```

Once that's finished, we need to then install a specific version of the Java Development Kit (JDK). Use **cd** to move to the **alexa-avs-raspberry-pi/samples/javaclient** folder and run:

```
./install-java8.sh
```

You will get a message from Oracle Java installer that you must accept the terms of service for the Java SE platform, which you need to now do.

Once that's complete, download Apache Maven from **magpi.cc/2bDPluf**. Move to the **Downloads** folder and extract the contents with:

```
sudo tar xvf apache-maven-3.3.9-bin.tar.gz -C /opt
```

You then need to create a file with some system settings for Maven. Start by creating the file like so:

```
sudo touch /etc/profile.d/maven.sh
sudo nano /etc/profile.d/maven.sh
```

Add the following to the file you just opened:

```
export M2_HOME=/opt/apache-maven-3.3.9
export PATH=$PATH:$M2_HOME/bin
```

Save and exit the file. Reboot your Raspberry Pi before continuing.

Certification

We now need to generate self-signed certificates:

```
sudo apt-get install openssl
```

Once installed, move back to **/alexa-avs-raspberry-pi/samples/javaclient** and run the script:

```
./generate.sh
```

It will ask you to enter some information. Enter the following details exactly as shown:

```
Product ID: my_device
Serial Number: 123456
```

Just press **ENTER** when it prompts you for a password and then let it run and generate a key.

Now we can get our details for the Alexa Voice Service; this does require an Amazon account, though. Go to **developer.amazon.com** and log in – we then had to 'complete' our registration before continuing, so be prepared to do so as well.

Once you're at the dashboard, click on the Apps & Services tab, then Alexa. On Alexa Voice Service, hit Get Started. From the drop-down menu 'Register a Product Type', select Device.

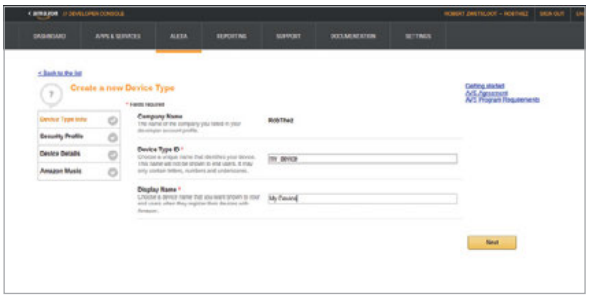
On the first page, fill in Device Type ID as **my_device** and Display Name with **My Device**. Click Next to go to the security profile. Click on the Security Profile drop-down and choose 'Create a new profile'.

Enter the following:

```
Security Profile Name: Alexa Voice Service
Sample App Security Profile
Security Profile Description: Alexa Voice
Service Sample App Security Profile Description
```

Click Next and your Client ID and Client Secret will be generated for you. Go to the Web Settings tab and make sure the security profile you just created is selected in the drop-down menu, then click the Edit button.

On Allowed Origins, click Add Another and then enter **https://localhost:3000** in the text field that appears. For Allowed Return URLs, enter **https://localhost:3000/authresponse** and then click on Next. We're now on Device Details; first set the Category as Other. Use a description of 'Alexa Voice Service sample app test' and then choose 'Longer than 4 months / TBD' for the expected timeline question. Finally, enter 0 for for the number of devices you plan to commercialise and hit Next once more. On the next tab, click for Amazon Music and hit Submit. You're done!



Above You'll need to set up the app on your Amazon developers page, which is linked to your Amazon account

Final configurations

In a browser, go to **magpi.cc/2bvWrNu**. At the top of the page, select the security profile we created and click Confirm. You'll need to enter a dummy web address for the consent privacy notice URL.

Click Save. Click on Show Client ID and Client Secret and make a note of them.

Back on the Pi, move the **alexa code** folder to the Desktop and rename it **alexa-avs-raspberry-pi-master**. In the terminal, open the following file:

```
nano /home/pi/Desktop/alexa-avs-raspberry-
pi-master/samples/companionService/config.js
```

Post your Client ID and Client Secret in the fields clientID and clientSecret respectively, and then save and close the file.

Now we can get the Alexa service running. Start in the terminal with:

```
cd /home/pi/Desktop/alexa-avs-raspberry-pi-
master/samples/companionService
npm start
```

Open a new terminal window. Move to **alexa-avs-raspberry-pi-master/samples/javaclient** and use:

```
mvn install
```

It will say 'build success' when done. Run it with:

```
mvn exec:exec
```

Talk to me

A window will pop up asking you to register the device. Copy the URL into a browser and log into Amazon, and click Okay on the next page to confirm everything. You'll be redirected to a page saying 'device tokens ready'.

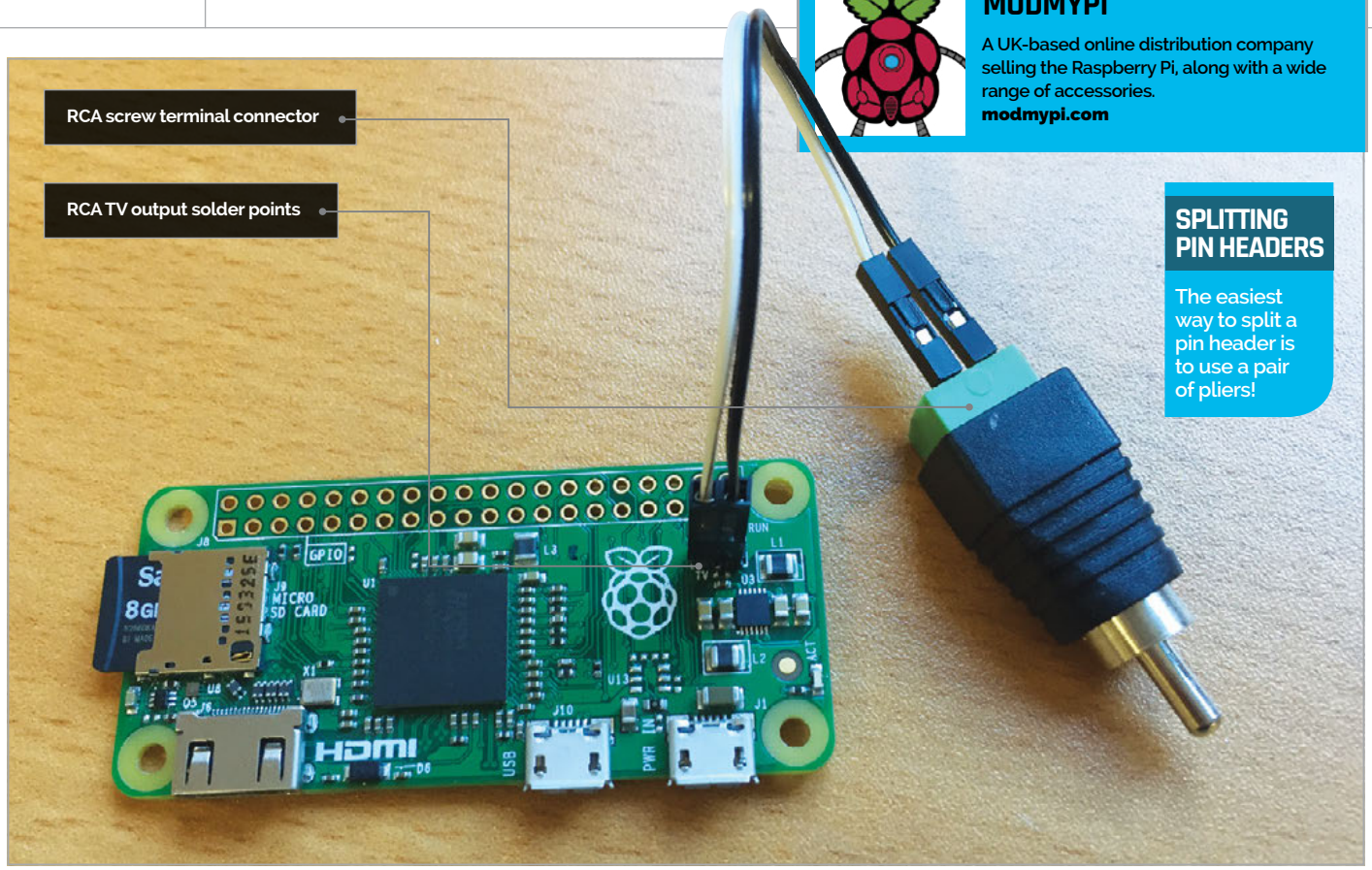
Click OK on the original pop-up and you can start asking Alexa stuff. Hit the Start Listening button, wait for the audio cue, then ask it something like 'what is two plus two'. Hit Stop Listening and it should reply 'four'. And that's it, you're ready to play with Alexa!

ADAPTED FROM

This tutorial was adapted from this excellent tutorial by Akash Chandran: **magpi.cc/2bljvC2**

ALEXA CONTEST!

Want to see how people have been using Alexa on Pi? Check out the entrants and winners to a contest held by Hackster.io for inspiration: **magpi.cc/2baErdf**



MODMYPI

A UK-based online distribution company selling the Raspberry Pi, along with a wide range of accessories.
modmypi.com

SPLITTING PIN HEADERS

The easiest way to split a pin header is to use a pair of pliers!

RCA OUTPUT FOR YOUR PI ZERO

Did you know that the Pi Zero has a composite video out port which is very easy to access? If you'd like to connect your Zero to an old TV, read on and we'll show you how...

You'll Need

- Pair of header pins **magpi.cc/1U76ZW3**
- Male-to-female jumper wire **magpi.cc/1U774sY**
- Screw terminal RCA **magpi.cc/1U776AV**

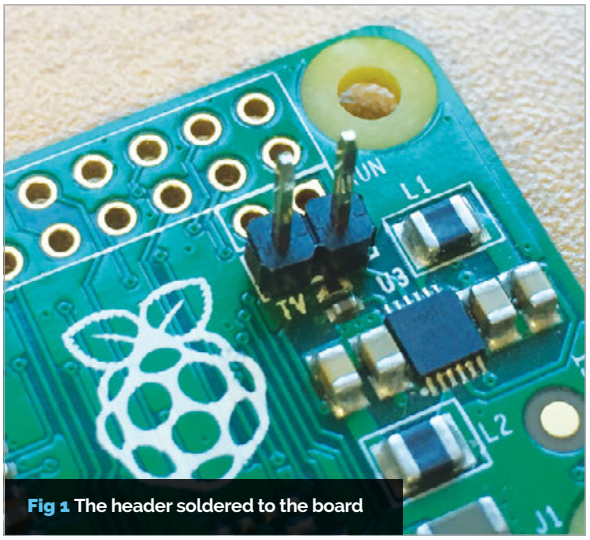


Fig 1 The header soldered to the board

In this simple, easy-to-follow tutorial, we'll be showing you how to hook up your Pi Zero to a TV via an RCA cable. That's right – believe it or not, the Pi Zero isn't limited to just HDMI video. By soldering a header pin, hooking up a couple of jumper wires, and adding a screw terminal RCA connector, you can easily access the RCA video output so you can use an old CRT TV in your next Raspberry Pi project.

We're going to be soldering two pin headers onto the Pi Zero. Start by soldering a pin header to the square pad labelled 'TV' on your Pi, then solder another header onto the circle pad next to the square pad. Both pads are contained within a white outline. See **Fig 1** for reference.

You could solder wires directly to these pins, but by using pin headers you get a nice neat solution that

allows your connection to be removed when required. Once the headers have been soldered to your Pi, you can then move on to attaching the jumper wires to your RCA screw terminal. Using the male side of the jumper wire, attach them to each of the terminals, making sure to screw them up nice and tight. Make a note of which wire is plugged into the positive pin and negative pin.

Now plug each of the wires into the pin headers you previously soldered onto your Pi. Make sure the wire connected to the positive terminal is connected to the pin labelled 'TV'.

That's it for setting up the hardware. Now your Pi should automatically detect which video method you're using, either HDMI or RCA. If it doesn't, however, read on and follow our software configuration to get it working.

Setting up the software

First things first: either SSH into your Raspberry Pi, or open up a terminal window.

We need to make some changes to the **config.txt** file, but before we do that it's probably a good idea to make a backup of the original, just in case:

```
sudo cp /boot/config.txt /boot/config.txt_backup
```

Now we have our backup, we can edit the original and make some changes. Start by opening **config.txt** in your editor of choice. We'll be using nano:

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

There are two lines in the file that you need to edit. Firstly, you need to remove the comment '#' from the following line:

```
#sdtv_mode=2
```

So it should now look like this:

```
sdtv_mode=2
```

Then we need to add a comment '#' to the following line:

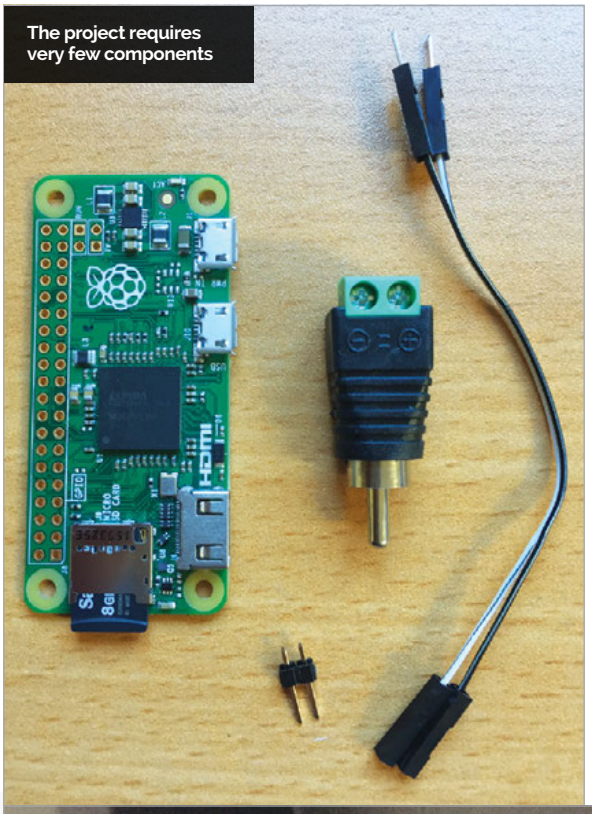
```
hdmi_force_hotplug=1
```

So it should now look like this:

```
# hdmi_force_hotplug=1
```

That's it. Remember to save your file – if you used nano, press **CTRL+X** to exit; then, when asked if you want to save changes, enter **Y** then press **RETURN**.

Now you can plug your RCA cable into your TV/monitor, and you should hopefully see the video output. Happy days!



The project requires very few components

SOLDERING PIN HEADERS

Try putting the pin headers in a breadboard to hold them upright!

TYPING FILE PATHS

When typing file paths in the terminal, try pressing **TAB** to auto-complete!

Hook it up to a video input

ALAN MCCULLAGH

Irish and French. Biggest Raspberry Pi supporter in Orange. Ambassador and national coordinator for Europe Code Week. Helping develop Code Club in France.
twitter.com/tyrower

MAKE A RASPBERRY BERET

We'll add some wearable electronics to our hat: music, lights, camera, and some 3D prints in a digital homage to Prince

You'll Need

- A hat – ideally a beret
- Piezo buzzer **magpi.cc/28IjuBH**
- WiFi dongle **magpi.cc/28IjAsZ**
- Battery packs (5V for Pi; 5-9V for NeoPixels)
- Camera Module **magpi.cc/28IjIsz**
- Pisoundo package, originally from onebeartoe **magpi.cc/28IjRMy**
- Rpi_ws281x library from jgarff **magpi.cc/28IjT6X**

For some people hearing about the Raspberry Pi for the first time, a certain song entered their head. This earworm turned out to be the eighties classic *Raspberry Beret*. It's now become more poignant with the untimely departure in April 2016 of legendary musician Prince.

As far back as August 2011 in the forum, there was even a merchandising suggestion of a Raspberry (Pi) Beret. Since nothing came of it, we're going to hack our hat for some fun – adding colour, sound, and vision.

>STEP-01

Prepping the Pi

We start out with a fresh installation of Raspbian. Use the WiFi dongle to connect to the network. Make sure everything is up to date:

```
sudo apt-get update && sudo apt-get upgrade
```

Take note of the Pi's IP address:

```
hostname -I
```

We'll then be able to work remotely via SSH, and later point our smartphone or web browser at the Pi. You could also find the IP via the graphical interface, a smartphone app (such as Fing), Bonjour via Mac, via a router admin page, or a tool such as Nmap.

>STEP-02

Making the sign (of the times)

If you have access to a 3D printer, you can creatively spruce up the hat with some colourful 3D prints. Prince famously became associated with a shape, later known as the 'Love symbol', even replacing his name at one time. We easily found a vector graphic (SVG) online (**magpi.cc/28Ii66C**), as created by onebeartoe. Bringing the file into Tinkercad (**tinkercad.com**), via **Import URL**, giving it a height in the process, allows us to generate a 3D printing file. We rescale it so our NeoPixels shine and our camera can shoot through it. We must then generate our final 'G-Code' file to be printed.

>STEP-03

What it sounds like when piezos cry

The piezo is no simple buzzer. Much like phones of old, we can program it to play very basic tunes, although don't expect quality – it's more like a novelty musical Christmas tie than high-fidelity.

First, check the version of Java (it should be 1.8):

```
java -version
```

If needed, install/upgrade via:

```
sudo apt-get install oracle-java8-jdk
```

The latest JAR package of our Raspberry Beret version of Pisoundo can be downloaded and copied to the Pi from **magpi.cc/28IqaUQ**.

If you've already hooked up the piezo as in the schematic, you can launch the application via :

```
sudo java -jar pisoundo-0.0.1-Raspberry-Beret.jar
```

From the network, we then point a browser at **http://[IP_address_of_Pi]:2111/ui/index.html** and choose a tune from the list. Alternatively, it's possible to inject your own musical code by clicking on the On-the-Fly link on the page.

>STEP-04

Add some lights

Having hooked up our individual NeoPixels, as per the diagram, we need to carry out the following commands to get our lights on and flashing.

```
sudo apt-get install build-essential
python-dev git scons swig
https://github.com/jgarff/rpi_ws281x.git
cd python
sudo python setup.py install
```

In the example given, we see three NeoPixels, but you can choose to add as few or as many as you want (individual, rings, or strands). To configure this, type:

```
cd examples/
sudo nano strandtest.py
```

Then adjust the following line to suit your setup:

```
LED_COUNT = 3 # Number of LED pixels
```

Finally, to launch our program, use:

```
sudo python ./strandtest.py
```

Our LEDs should start to flicker, dance, and change colour.

>STEP-05

Cameras, action...

A final step is to hook up our Camera Module to take photos or even broadcast a live video stream. One of the easiest ways is with the RPi Cam Web Interface (**magpi.cc/28InKFJ**). Don't forget to activate the camera via the **Configuration** menu on the **Interfaces** tab. Here's a simple installation method:

```
git clone https://github.com/
silvanmelchior/RPi_Cam_Web_Interface.git
cd RPi_Cam_Web_Interface
chmod u+x *.sh
./install.sh
```

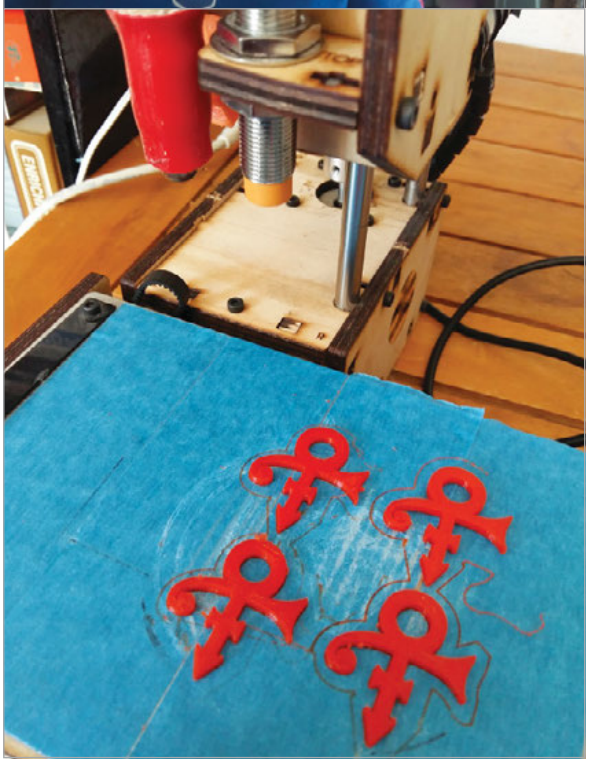
Once we've set this up, all we need to do is point a web browser at the IP address of our Pi to access the RPi Cam Control.

>STEP-06

Explore the possibilities

If you tested out this project on a breadboard to get it working, it's now time to install all the electronics inside our hat. Use your imagination to modify and make it better. Hack the hat. What else could we add to our chapeau?

So now we have a Raspberry Beret – the kind you wouldn't find in a second-hand store. Enjoy!



Left Placing the piezo on the top, NeoPixels around the edge, and camera on the front, we add our 3D 'symbols' for a final Princely touch

Left Printing off 3D decorative symbols adds colour to our headwear and hides our electronic connections. Hot glue is our friend when attaching them

Language
>PYTHON

DOWNLOAD:
magpi.cc/28IqenE

LE BERET BASQUE

The beret is often thought of as French, but for us it's usually more associated specifically with Basque regional tradition.

THE MELON HAT

In France, they think of the bowler hat as being British. A fruit connection – it's called a 'melon hat' in French!

WESLEY ARCHER (AKA RASPBERRY COULIS)

Self-taught Raspberry Pi enthusiast, guide writer for Pi Supply and creator of raspberrycoulis.co.uk. Enjoys trying out new Pi projects and sharing them!
raspberrycoulis.co.uk

You'll Need

- Lisiparoi LED light ring (infrared version) **magpi.cc/1SQVFrW**
- Cyntech Raspberry Pi case **magpi.cc/2ipch47**
- Raspberry Pi NoIR Camera Module
- USB WiFi adapter
- MotionEyeOS **magpi.cc/1UCw1Jk**
- Drill and small drill bit (1mm or 2mm ideal sizes)
- Small file (we used a metal nail file)
- Pencil and a sharp knife (craft or Stanley knife)

RASPBERRY PI NIGHT VISION CAMERA HACK

Have a spare Raspberry Pi Model B lying about? Turn it into a night-vision CCTV camera with this simple case hack...

Many people have a number of Raspberry Pis at home. Some might say those folks are obsessed, but Pis are so versatile that trying out new projects where possible is great fun! Putting them to good use is always exciting, so why not turn a spare Model B into a night-vision CCTV camera, using an inexpensive case, the Camera Module, and a nifty little accessory called the Lisiparoi to provide infrared lighting? If you saw the previous guide on adding push notifications to MotionEyeOS (*The MagPi* 43), then combining this with night vision would be a great addition!

>STEP-01

Mark where you will cut

Before cutting into the case, it makes things easier to mark it out first. Grab your pencil, place your Camera Module and Lisiparoi on the case, and then draw around them. You should be able to see the pencil line, and it's easy to remove if you need to. Make sure you allow enough room around the outside so that it's not too close to the edge of the case, though! You'll also need to cut out space for the pins on the Lisiparoi, holes to mount the Camera Module board, and a hole for the camera lens itself.

>STEP-02

Drill the mounting holes

Using a small drill bit (1mm / 2mm is ideal), drill two small holes in the case. If you are very careful, you can use the Lisiparoi as a template. You should now have two small holes in the case and a pencilled outline of the Lisiparoi. It's now time to drill holes for the header pins, which are a few millimetres away from the bottom edge of the Lisiparoi. It's best to start small and increase the size of the holes as you go; it doesn't have to be exact, as the Lisiparoi will hide everything once in place.

>STEP-03

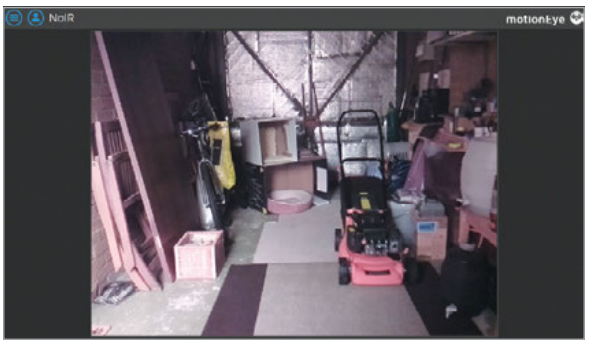
Drill, file, test, and repeat

Whilst doing this, remember – you can always cut more off! To make sure our header pins on the Lisiparoi fit nicely, we drilled the holes to the approximate width of the header and then used a small file to square off the holes. We then placed the Lisiparoi in place to see if it fit; if it didn't, a little more was filed off until it did. It doesn't have to be surgically accurate, since the Lisiparoi sits on top of the holes; even so, take your time, as you can't undo a hole if it's too big!

>STEP-04

Raspberry Pi NoIR Camera Module

It's now time to mark and cut a hole for the Camera Module. Again, we drew around the module with a pencil and then drilled small holes along the inside edge of the outline (otherwise the hole will be too big). Next, very carefully use a sharp knife to cut out the hole; the drilled holes along the outline should help with this step. Make sure this is done on a tough, steady surface (we used an old chopping board). Fine-tune the fit by filing the hole and testing that the Camera Module fits. Repeat until it does.



Left Once it's complete, you should hopefully see something like this from your Raspberry Pi night-vision CCTV camera!

>STEP-05

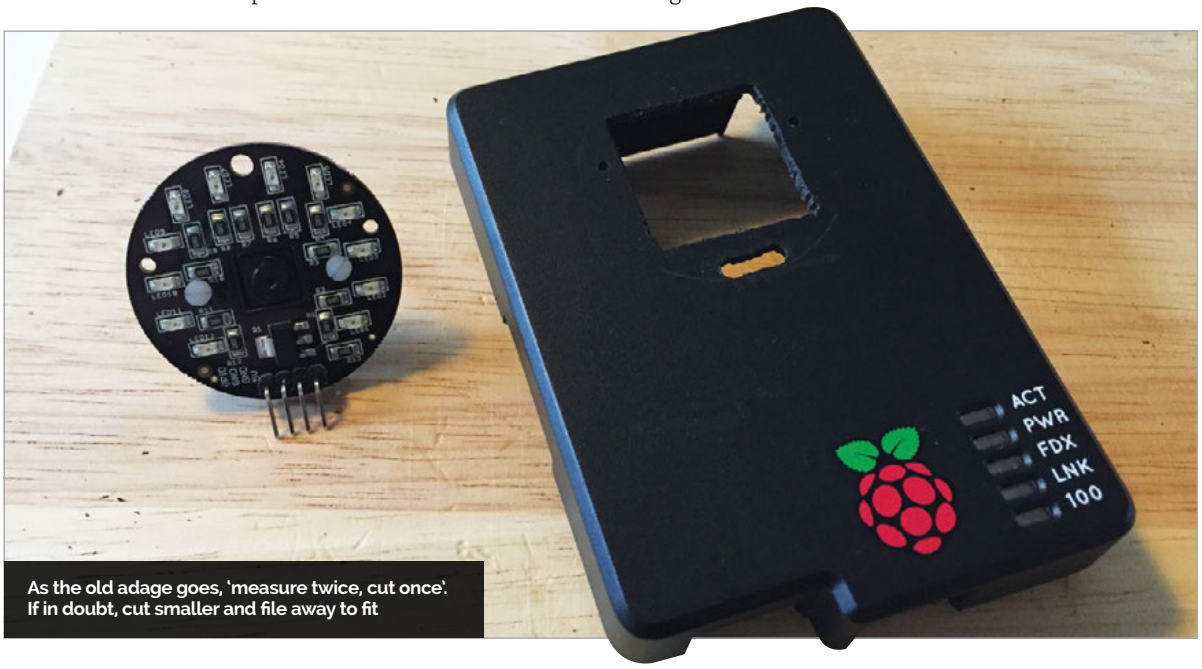
Put everything together

Now that the holes are all cut and fit nicely, it's time to put everything together. Make sure you've soldered the header onto your Lisiparoi first, though! The header needs to be soldered so the pins are on the inside of the case and pointing downwards, so that the jumper cables can be fitted to the Pi's GPIO once fully assembled. Screw your nuts and bolts together and connect up your cables, including the ribbon cable for the Camera Module. For more information on using the Lisiparoi, including wiring diagrams, check out the official site here: **magpi.cc/1SQWvoo**.

>STEP-06

Install MotionEyeOS and test everything

If you have connected the Lisiparoi so that it turns on whenever the Pi does, then you just need a CCTV operating system. MotionEyeOS is perfect for this, because it's simple to use and works very well! Download the relevant image from **magpi.cc/1UCw1Jk**, then write it to your SD card. You won't be able to see if the infrared LEDs are on, but test it out in the dark and you should see more than you would normally without them! As the LEDs are small, they're not very powerful, but should provide enough illumination to see better in the dark!



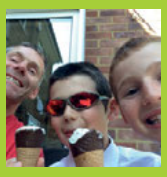
As the old adage goes, 'measure twice, cut once'. If in doubt, cut smaller and file away to fit

MEASURE, THEN MEASURE AGAIN!

They always say 'measure twice, cut once', and so do we! It is better to be safe than sorry.

LESS IS MORE

You can always cut more, but you cannot replace what is gone. Think about this when drilling and filing!



THE HAYLER-GOODALLS

Ozzy, Jasper, and Richard are mentors at CoderDojo Ham and gave a talk at the Raspberry Pi birthday party about their AstroPi adventures.
richardhayler.blogspot.co.uk
coderdojoham.org

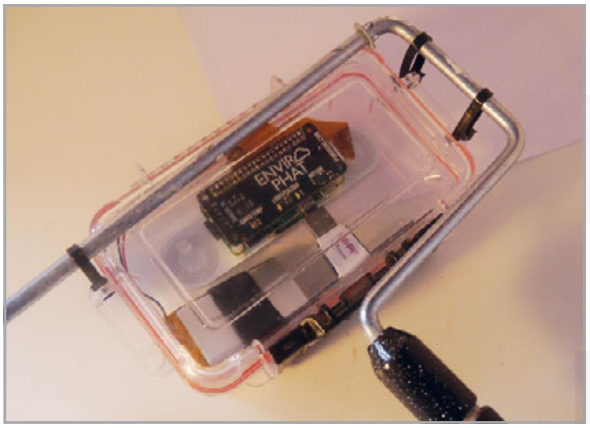
AQUAPI-CAM

Explore the underwater world with a Raspberry Pi camera

You'll Need

- A transparent, waterproof box **magpi.cc/2e8beBX**
- Pi Camera Module
- Portable power source
- hostapd and dnsmasq packages
- Python Flask library
- WiFi dongle (if not using a Pi 3)
- Enviro pHAT (optional) **magpi.cc/29NHB3T**
- ZeroView (optional) **magpi.cc/2e8ghWt**

There are plenty of underwater sports cameras available, but they can be quite expensive, especially if you want to control them remotely. In this tutorial we're going to use readily available Pi add-ons to make a cheaper, customisable camera unit. There are lots of options and alternative sources of components for a project like this. For example, the Pimoroni Enviro pHAT is a really useful option that can report back information about the environment in which the camera is operating, especially how much light is available. There's a fair bit of software configuration involved, but example config files are in the GitHub repo for this article (**magpi.cc/2e8dtFk**).



>STEP-01

Find a suitable container

This needs to be watertight and have at least a see-through lid. You can find Tupperware boxes with a very tight seal, but these tend to be translucent rather than transparent. The size of box will probably determine your choice of Pi and power source. Zeros are great as they're so small, but then you'll need a WiFi dongle and shim. You can also save space by using a LiPo battery instead of a power bank, although you'll need a boost regulator too, such as the Pimoroni Zero LiPo.

>STEP-02

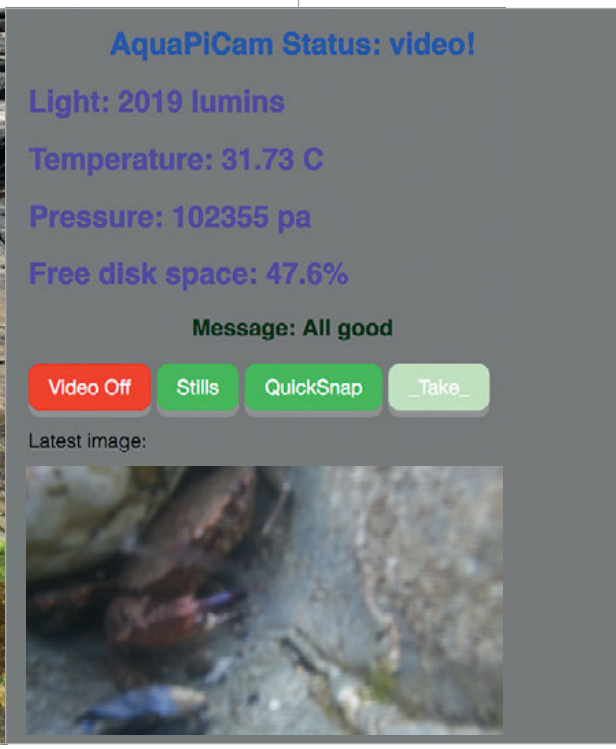
Configure your Pi to be a WiFi access point

Start from a fresh Raspbian Jessie Lite SD card and install the following:

```
sudo apt-get update
sudo apt-get install -y dnsmasq hostapd python3
python3-dev python3-flask python3-picamera
```

First, configure your wireless interface to have a static IP address by editing **/etc/network/interfaces**. Then set it to not use DHCP by adding this line:

```
denyinterfaces wlan0
```



Above left You'll still have to get pretty close to the water yourself
Above right The web interface shows environmental information and lets you control the camera

…to the end of your **/etc/dhcpcd.conf** file. Next, create the **/etc/hostapd/hostapd.conf** file, using the example in this tutorial's GitHub repository as a template. Change the **interface**, **ssid**, and **passphrase** parameters as needed. Finally, edit **/etc/dnsmasq.conf**, ensuring that the IP addresses are consistent with your settings in **/etc/network/interfaces**. Then reboot!

>STEP-03

Add the Enviro pHAT

You have the option of soldering this board directly onto the Pi's GPIO pins, or you can use the supplied female header if you want to reuse it in other projects. After that, install the Python library and dependencies using the following command:

```
curl -sS https://get.pimoroni.com/envirophat | bash
```

The library comes with some example programs and you should run these to test that everything is working correctly.

>STEP-04

Fit everything into your container

To cut down on reflections and get the best possible images, the camera should be as close to the transparent side of your container as possible. The ZeroView from the Pi Hut is a clever mounting plate that uses suction cups and will also hold your Pi securely. Alternatively, you could make a mount out of cardboard and glue this to the inside of the container. Velcro tape can be a good solution for power sources (which normally need to be removable for recharging).

>STEP-05

Add some code, HTML and CSS

Clone the entire **Flask** folder from the project repository onto your Pi. Flask is a small web framework written in Python which allows you to create simple web services; in this case, it's a webpage that allows us to see data from the Enviro pHAT and the latest captured images. We can also switch between recording modes (movie or continuous still frames) or take photos on demand. This control of the camera is achieved via the excellent Python **picamera** library. You could enhance the project by adding additional exposure and shutter speed controls to your interface if you want.

>STEP-06

Set the code to run at boot

To set the AquaPiCam program to run when the Pi boots up, add this line to your **/etc/rc.local** file, immediately above the **exit 0** line:

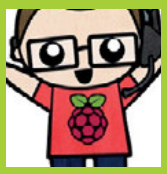
```
python3 /home/pi/Flask/apc.py &
```

It's also a good idea to configure the Pi to only boot to the command line, using:

```
sudo raspi-config
```

…and selecting 'console' under option 3.

Now go and find somewhere wet! You might want to run a few tests in the bath before venturing further afield!



SPENCER ORGAN

Chemistry and physics teacher, Raspberry Pi enthusiast and Certified RPi Educator from the West Midlands with a passion for running workshops and building fun, educational, and practical things with the Pi!
magpi.cc/2bkQ53q

PROGRAMMABLE MOTION TIME-LAPSE CAMERA RIG

Take stunning motion-controlled time-lapse frames with your Raspberry Pi and Arduino wherever you go

You'll Need

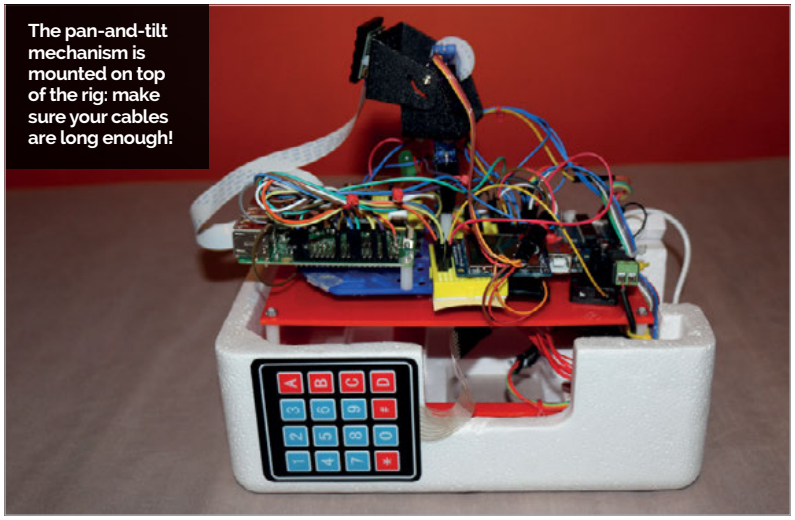
- Camera Module
- Arduino Uno **magpi.cc/2gTVB0Z**
- 16×2 character I²C LCD display **magpi.cc/2gVCROB**
- Nokia 5110 screen (optional) **magpi.cc/2gTPWrA**
- Adafruit mini pan-tilt kit - assembled with micro servos **magpi.cc/2eQBU6a**
- Membrane 4×4 button pad **magpi.cc/2gU14F1**
- 5V mini DC relay, LED, 270Ω resistor, male-to-male and male-to-female jumper cables, various breadboards

The small form factor, lower power use and the high-quality camera on the Raspberry Pi makes it an ideal platform for capturing time-lapse frames. In this project, we'll use an Arduino Uno to control the motion of the Raspberry Pi Camera Module and to trigger the photos being taken.

>STEP-01

Connect pan-tilt kit

We start the project by connecting the Adafruit pan-tilt kit to the Arduino. Use a breadboard to connect a common 5V and ground line from the Arduino. Connect the red power cables on each of the servos to the common 5V line, and the brown wire to the common ground. Use jumper cables to connect the orange signal wire of the pan servo to digital pin 9 on the Arduino, and the orange signal wire of the tilt servo to digital pin 8.



The pan-and-tilt mechanism is mounted on top of the rig: make sure your cables are long enough!

>STEP-02

Connect the I²C display

Once we have connected the servos, we can add the I²C LCD display to the Arduino. For this, we'll need four jumper cables. Connect the GND pin to the lower ground connector, then connect the VCC pin to the common 5V line we were using just now for the servo motors. Connect the SDA pin to the analogue A4 connector, and the SCL pin to the analogue A5 connector. We'll need to download the libraries for the I²C LED display for the Arduino. The latest libraries can be downloaded from here: **magpi.cc/2bkO5br**.

>STEP-03

Connect the keypad

Connecting the keypad can be one of the trickiest parts of the build. There's very good documentation for setting up and using a keypad here: **magpi.cc/2baan3b**. We'll also need to download and install the keypad libraries from this page. We have connected the rows to digital pins 5, 4, 3, and 2, and the column pins to 13, 12, 11, and 10. If you find incorrect characters being displayed when you press the keys, you'll need to try reversing the order of the row and column pins; with a bit of trial and error, this should be easy to fix.

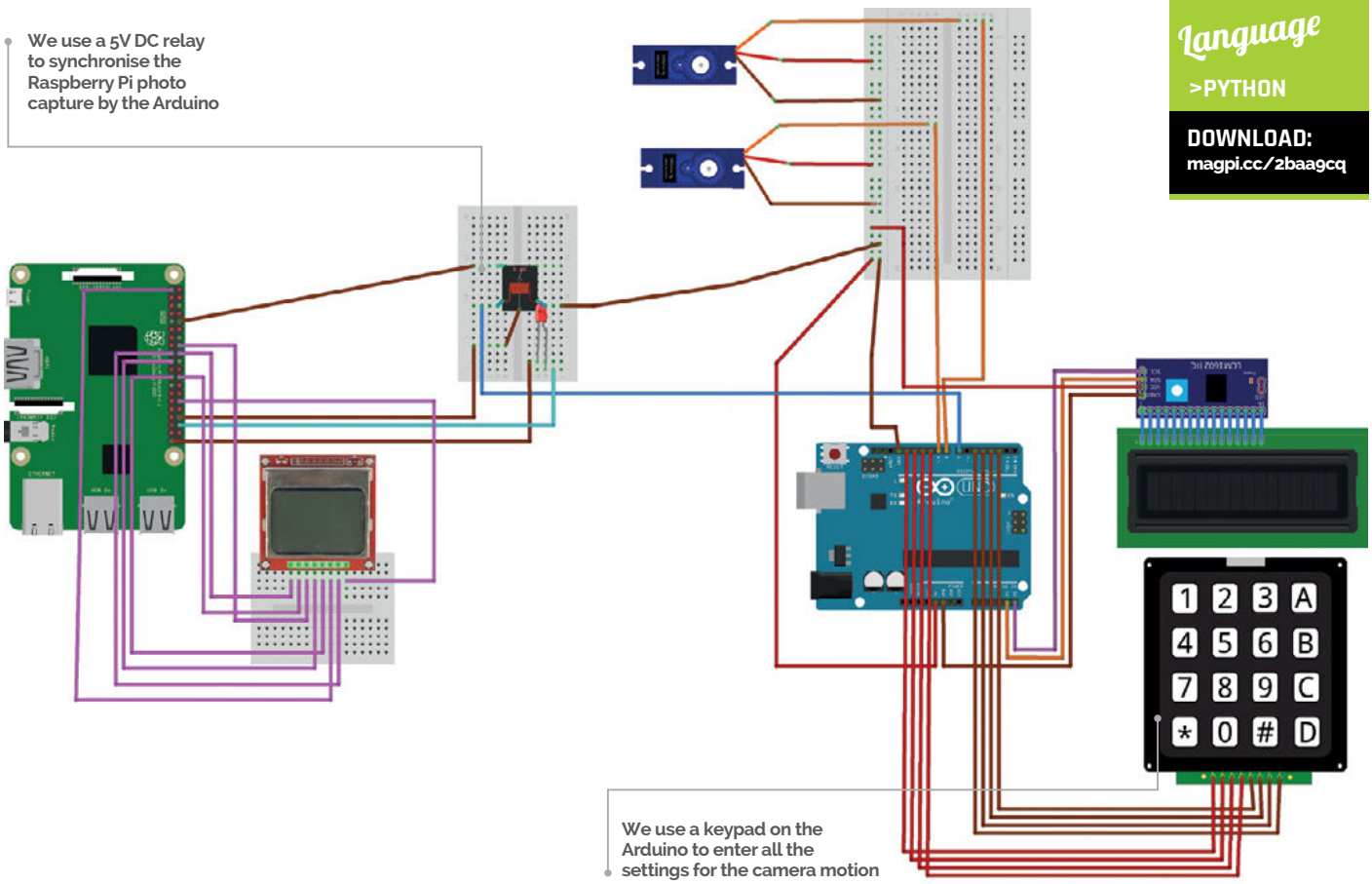
>STEP-04

Mount the Pi camera

Mount the Camera Module onto the pan-and-tilt mechanism. We have found that a longer camera connector cable works better and prevents the camera getting stuck. Connect the other end of the camera connector to the Raspberry Pi, with the printed side of the ribbon cable pointing towards the USB ports.

Language
>PYTHON

DOWNLOAD:
magpi.cc/2baagcq



>STEP-05

Connect the Nokia 5510 screen (optional)

This optional LCD display allows you to see how many pictures have been taken. There are two versions of this display, so we advise you to check the pin layout and adjust as necessary. Excellent documentation and links to the Python libraries can be found in this PDF guide: **magpi.cc/2bkPU8g**.

LCC VCC is connected to 3.3V
LCD GND is connected to a ground pin
LCD D/C is connected to GPIO 23
LCD RST is connected to GPIO 24
LCD CS is connected to SPI CE0
LCD CLK is connected to SPI SCLK
LCD DIN is connected to SPI MOSI

There is an optional Backlight pin, which can be powered off the 5V from the Raspberry Pi or from the Arduino. If powering it from the Arduino, you'll also need to connect the LCD GND to a ground pin on the Arduino.

>STEP-06

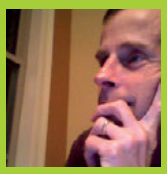
Adding a relay and LED

We now come to the stage were we connect the Raspberry Pi and the Arduino together. The Arduino controls the movement of the servo motors, moving the camera as well as telling the Raspberry Pi when to take a photo. To ensure that the camera isn't moving, we have included a three-second countdown and delay in the Arduino code before each photo is taken. The Raspberry Pi simply waits for a switch to be closed between GPIO 15 and ground. This switch is provided by a small 5V relay powered by digital pin 7 on the Arduino.

Depending on the relay you have purchased, you'll first need to connect the coil to digital pin 7 and ground on the Arduino. The switch part of the relay can then be connected to ground and GPIO 15 on the Raspberry Pi. You should hear a satisfying click every time a photo is taken, as the relay closes and then opens again.

One final step is to add an LED to give a confirmation that a photo has been taken successfully. Connect the longer LED leg (positive) to a 270Ω resistor and then to GPIO 16, and the short leg to ground.

Above Create beautiful time-lapse videos: see an example at: **magpi.cc/2bagMOQ**



DAVE PROCHNOW

A writer whose numerous projects, contraptions, and inventions have appeared in *Popular Science*, *Nuts & Volts*, and *SERVO Magazine*.
magpi.cc/2cc7uvj

GO, PIONEER

DIY PORTABLE ACTION CAM

Build your own battery-powered, slow-motion, 90fps action camera to prove you've 'been there and done that'!

You'll Need

- Raspberry Pi Camera Module
- 7× 5mm (T-1 3/4) LEDs
- 7× 430Ω resistors
- Big SPST switch **magpi.cc/2bY3Pmn**
- 3.7V 1000mAh LiPo battery **magpi.cc/2bY3Kiq**
- 5V step-up **magpi.cc/2gTMM77**
- JST connector **magpi.cc/2bLzvc0**
- 4× 2-inch #6-32 screws
- 4× ⅜- or ½-inch rubber grommets
- Jumper wires

Most adventurers who are thinking about recording an action event would choose to explore the GoPro camera system. While being a very capable recording device that's housed inside a diminutive package, the GoPro cameras have an exorbitant price tag that stops weekend warriors in their tracks. What if an action camera cost less than $50? What if you could build this camera yourself? What if this camera had a fully functional Linux computer inside? Although it might sound like pie-in-the-sky dreaming, these are the hallmarks for our piOneer slo-mo action camera project.

>STEP-01

Print it

Borrowing a page from the Astro Pi design book, piOneer consists of a three-part 3D-printed case that holds a Raspberry Pi, Camera Module, user interface, LiPo battery, and large capacity micro SD card. The case consists of three separate STL files that can be printed on any 3D printer with a 150×150mm build surface. The case's top piece should be printed with a removable support structure. You can download the STL files from here: **magpi.cc/2c3XdUE**.

>STEP-02

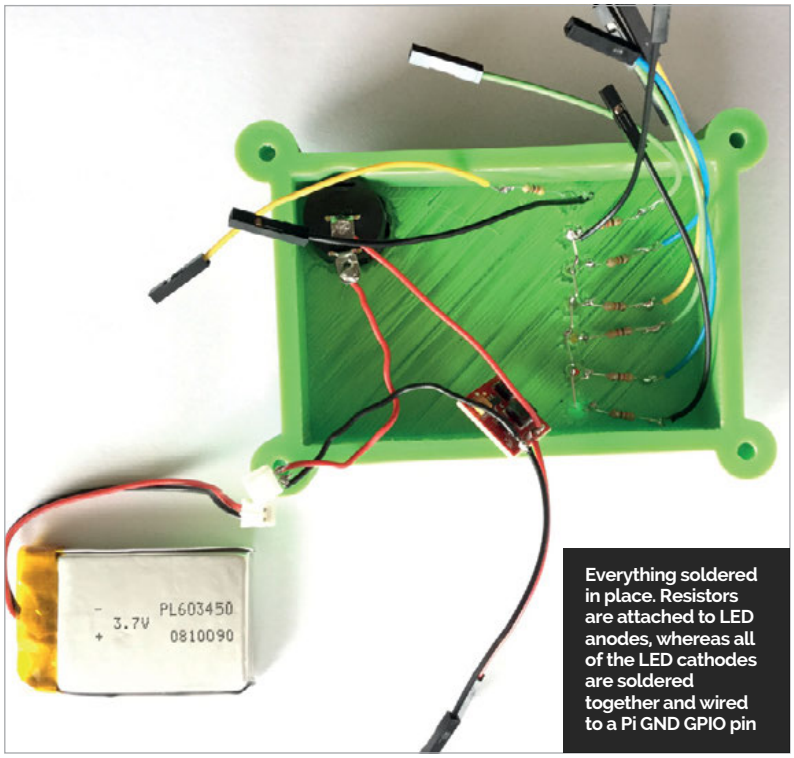
Build your UI

The user interface (UI) for piOneer is dead simple. Consisting of seven LEDs and one switch, the entire UI is housed in the case top piece. Begin your UI construction by gluing the LEDs and the switch to the upper surface of the case top. While you can use the same colour LEDs for piOneer, we opted to mix it up a little: a red LED for power (PWR), blue for ready, white for countdown 3 and 2, a yellow LED for countdown 1, red for recording (Rec), and a green LED for finished. Just one caveat about gluing the LEDs: ensure that all of the cathodes are lined up with each other; this makes the next step very easy to complete.

>STEP-03

Solder your UI

We're going to use a little bit of wiring trickery here! Rather than routing each LED cathode to a separate Pi GPIO GND pin, we're going to connect each cathode together in a daisy chain fashion, thereby using only one GND pin for six LEDs! Begin this wizardry by gently bending the 'finished' LED cathode so that it's touching the 'recording' LED cathode. Now solder these two cathodes together. Continue up the daisy chain, carefully bending the 'recording' cathode until it's touching the 'countdown 1' cathode, and soldering that connection. Likewise, for the remaining LEDs, bend and solder each LED cathode to the next until you reach the 'Ready' LED. Stop at this LED. Do *not* solder the power (PWR) cathode to this chain: it must be connected to its own GND pin.



Everything soldered in place. Resistors are attached to LED anodes, whereas all of the LED cathodes are soldered together and wired to a Pi GND GPIO pin

Mount the camera module to the 3D-printed base with M2 fasteners

>STEP-04

Wiring your UI

The resistors are each individually soldered to the anodes of *all* LEDs. Begin this process by carefully wrapping one end of a resistor lead around an LED anode lead. The resistor should be parallel to the inside of the top case. Solder this connection, and trim and discard the leftover resistor lead. Add the six remaining resistors to the final LEDs. Solder a jumper wire to the free end of the resistor's other lead. Complete the LED's cathode wiring by soldering a female jumper wire to the power (PWR) LED cathode lead, and one final female jumper wire should be soldered to the daisy-chained cathodes that we created in step 03. The piOneer UI is now complete.

>STEP-05

Adding battery power

The power system for piOneer consists of three sub-assemblies: the power (PWR) LED, the SPST switch, and the battery. While the LED and switch are pretty straightforward fixtures in a do-it-yourself (DIY) project, the battery has one additional component that enables the Pi to run without being tethered to a power outlet. A 5V step-up converter takes the output from the 3.7V LiPo battery and increases it to 5 volts.

The completed project, ready to capture your life's next great adventure

>STEP-06

Wiring your battery

Two wires are individually soldered to the switch's terminals. One of the switch's wires is soldered to the positive terminal of the JST connector. The JST connector is keyed for proper power signal orientation. Use the battery for identifying which terminal is which – remember, the battery's red wire is positive, while the black wire is GND. The other wire from the switch is connected to the IN pad on the 5V step-up converter. A female jumper wire is connected to the OUT pad of the converter. The final pad on the converter is a 'common' GND connection. In other words, both the battery's GND and the Pi's GND must be connected together via this pad. Therefore, solder a wire from the GND terminal of the JST connector to the 5V step-up converter GND pad *and* solder a female jumper wire to this pad. All of the soldering is now finished for piOneer.

>STEP-07

Begin final assembly

The Raspberry Pi and Camera Module will be fastened to the base portion of the case. Note the orientation of the two openings in the base. These openings will hold the Camera Module's lens and status LED. Line the module up with these openings and use four M2 screws for attaching the Camera Module to the base. Lay a rubber grommet on each of the four tall mounting posts, set the Pi on top of the grommets, route the Camera Module's ribbon cable out and over the GPIO pins, and use four more M2 screws for securing the Pi to the base. These rubber grommets will provide a small amount of cushion for the Pi during your upcoming rough-and-tumble adventures. Insert the Camera Module's ribbon cable into the Camera interface connector on the Pi.

>STEP-08

Connecting to the GPIO

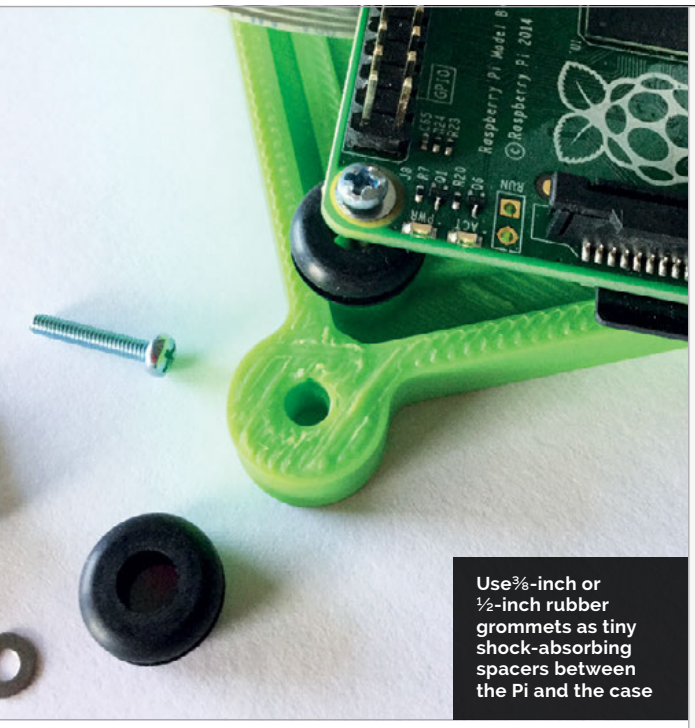
Place the wired-up top case next to the base that holds the Pi. Slip the mid case onto the top case. Connect each of the female jumper wires from the top case to the Pi GPIO pins by following this listing:

GPIO	JUMPER
Pin	Wire
1	PWR LED anode
4	5V step-up converter OUT
6	5V step-up converter GND
9	PWR LED cathode
22	Ready LED anode
16	Countdown 3 LED anode
15	Countdown 2 LED anode
32	Countdown 1 LED anode
33	Rec LED anode
37	Finis LED anode
39	Daisy-chain LED cathodes (see step 03)

>STEP-09

Program the Pi

Carefully and thoroughly examine each and every solder joint and wiring connection, looking for touching wires, solder blobs, connection mistakes, and so on. Fix any problems before connecting your Pi to a power source! When your piOneer passes muster, connect the Pi to a monitor, keyboard, mouse, and USB power source; now set up your Pi so that it boots, with automatic login, to a command-line interface (CLI). These settings will enable piOneer to run automatically without user input every time you



Use⅜-inch or ½-inch rubber grommets as tiny shock-absorbing spacers between the Pi and the case

flick the power switch. Download or enter the piOneer code and save your code to the home directory on the Pi. You can alter the code to suit your recording tastes. As it stands, piOneer will record 1 minute of 640×480-pixel video at 90 fps. The final H264 file will be approximately 65MB in size. In order to make the Python code run automatically, open your user profile for editing:

```
sudo nano /etc/profile
```

...and add this line to the end of that file:

```
sudo python /home/pi/pioneer.py &
```

Save, exit, and disconnect piOneer.

>STEP-10

Get out there!

Ensure that the power switch is in the OFF position and connect the battery to the JST connector. Slowly bring the case edges together, ensuring that *no* metal leads, wires, or solder joints are touching the Pi! Insert a long #6-32 screw through each of the cases's connection lobes and fasten with a nut. We used wing nuts for our fasteners, which makes the case easier to open and close. In order to record a video, just flick the power switch and within 30–45 seconds, piOneer will start recording your slo-mo video. Videos are recorded and saved with a date/time stamp file name. In order to view the videos on your Pi, use:

```
omxplayer date-time-stamp-filename.h264
```

Have fun doing your daredevil deeds, because now you can prove it.

pioneer.py

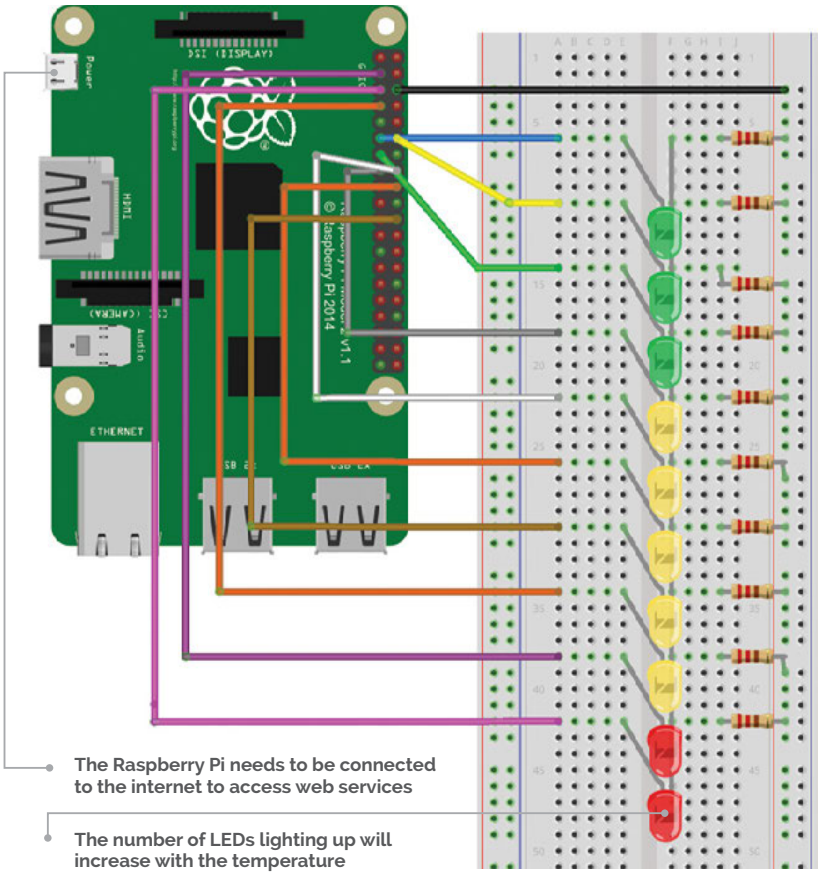
Language
>PYTHON

DOWNLOAD:
magpi.cc/2coG1um

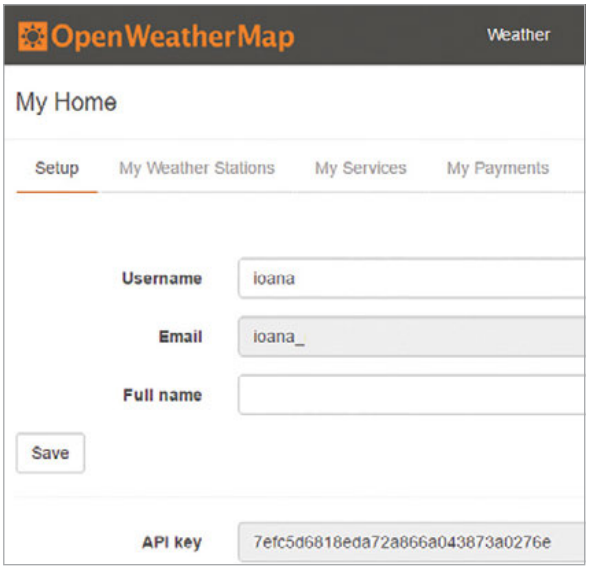
```
import RPi.GPIO as GPIO
from time import sleep
import datetime as dt

import picamera

GPIO.setmode(GPIO.BOARD)
# Set up LED pins
# Ready LED
GPIO.setup(22, GPIO.OUT)
# Countdown 3 LED
GPIO.setup(16, GPIO.OUT)
# Countdown 2 LED
GPIO.setup(15, GPIO.OUT)
# Countdown 1 LED
GPIO.setup(32, GPIO.OUT)
# RECord LED
GPIO.setup(33, GPIO.OUT)
# finisH LED
GPIO.setup(37, GPIO.OUT)

# Camera is ready, begin countdown
GPIO.output(37, False)
GPIO.output(22, True)
sleep(2)
GPIO.output(16, True)
sleep(2)
GPIO.output(16, False)
GPIO.output(15, True)
sleep(2)
GPIO.output(15, False)
GPIO.output(32, True)
sleep(2)
GPIO.output(32, False)
GPIO.output(22, False)

# Begin recording video
GPIO.output(33, True)

# Records 60 seconds of video at 90 fps; change wait_recording
# for length, in seconds, for video
with picamera.PiCamera() as camera:
    camera.resolution = (640, 480)
    camera.framerate = 90
    camera.exposure_mode = 'antishake'
    filename = dt.datetime.now().strftime('%d-%m-%Y-%H:%M:%S.h264')
    camera.annotate_text = dt.datetime.now().strftime('%d-%m-%Y %H:%M:%S')
    camera.start_recording(filename,format='h264')
    start = dt.datetime.now()
    while (dt.datetime.now() - start).seconds < 60:
        camera.annotate_text = dt.datetime.now().strftime('%d-%m-%Y %H:%M:%S')
        camera.wait_recording(0.2)
    camera.stop_recording()

# Finish
GPIO.output(33, False)
GPIO.output(37, True)
sleep(10)
GPIO.output(37, False)

GPIO.cleanup()
```

IOANA CULIC

Ioana is an IoT specialist, and is co-author of *Getting Started Guide for Raspberry Pi and Arduino Using Wyliodrin*. **wyliodrin.com**



Below You need to copy the API key in order to use the OpenWeatherMap service



PI-THERMOMETER

You'll Need

- Wyliodrin STUDIO **goo.gl/Sgj9HB**
- 3× Green LEDs
- 5× Yellow LEDs
- 2× Red LEDs
- 10× 220-ohm resistors
- Breadboard and jumper wires

Connect your Pi to existing web platforms and easily build your own internet-based thermometer out of LEDs

How often do you check your browser or smartphone to see how cold it is outside? Nowadays, we read the temperature using applications that take data from the internet and display them in a nice interface. This reduces the interactivity and simplicity that old mercury thermometers had, while offering the great flexibility of reading the temperature anywhere on the globe. But what if you could have the best of both worlds? You can put your Raspberry Pi to use and replace the computer with a thermometer-like device, which has the capability of displaying the temperature at whichever location you wish.

>STEP-01

Set up the Raspberry Pi

Wyliodrin STUDIO is an IDE (integrated development environment) that makes it very easy to create complex applications for your Pi. First of all, you need to download it (**goo.gl/Sgj9HB**) and follow the first two steps of the Getting Started tutorial for the Raspberry Pi. Once you have an SD card with the extra service installed, you need to connect the Pi to the internet, in the same network as your computer. As soon as the board has booted, you will see it available under Port. Press Connect and you're ready to go.

>STEP-02

Create a new visual application

In order to create the thermometer, press the 'projects' button and create a new application. Name it 'pi-thermometer' and select 'visual' for the programming language. Open the newly created app and you'll see the frame in which you need to connect the visual elements. The blocks gallery is on the left. We have chosen this way of programming because it allows you to rapidly create the code, and it doesn't require you to know any specific programming language. In addition, if you're familiar with Python, you can press the 'Show code' button and see how the code gets generated as you drag and drop the visual elements.

>STEP-03

Build the thermometer

From the hardware point of view, you simply need to connect ten LEDs to the GPIO pins of the Pi. You can visualise the number assigned to each pin by selecting the 'Pin layout' tab in Wyliodrin STUDIO. We have chosen to connect the LEDs on pins 0 to 9. We will scale the read temperature and light up a variable number of LEDs accordingly.

>STEP-04

Weather API

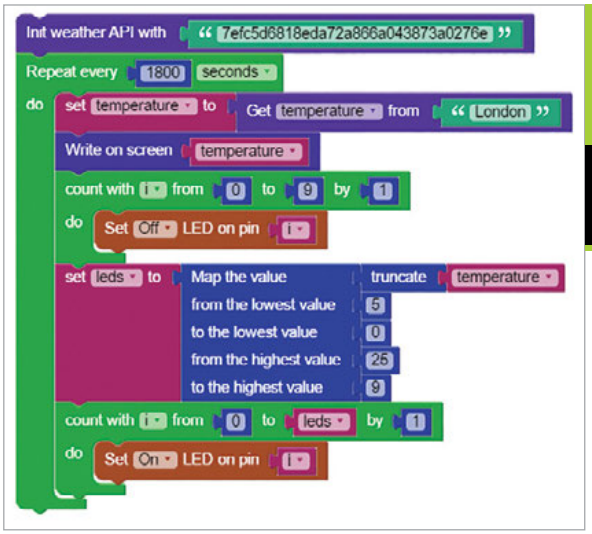
The website **openweathermap.org** offers a complex API that allows you to get weather details from cities around the world. To access the API, you need to sign up for a key at **openweathermap.org/appid**. You only need to provide your email and choose a password. Once you have created an account, you can use the API key available at **home.openweathermap.org**.

>STEP-05

Writing the application

The application reads the weather from the city of your choice and maps the value to the number of LEDs. The string used within the 'Init weather API' block is the key you got in the previous step.

In order to do the mapping, we specified the average minimum and maximum temperature values according to the current time of the year. It's recommended that you use a reduced range, to make the differences in temperature visible on the thermometer.



Language
>VISUAL PROGRAMMING

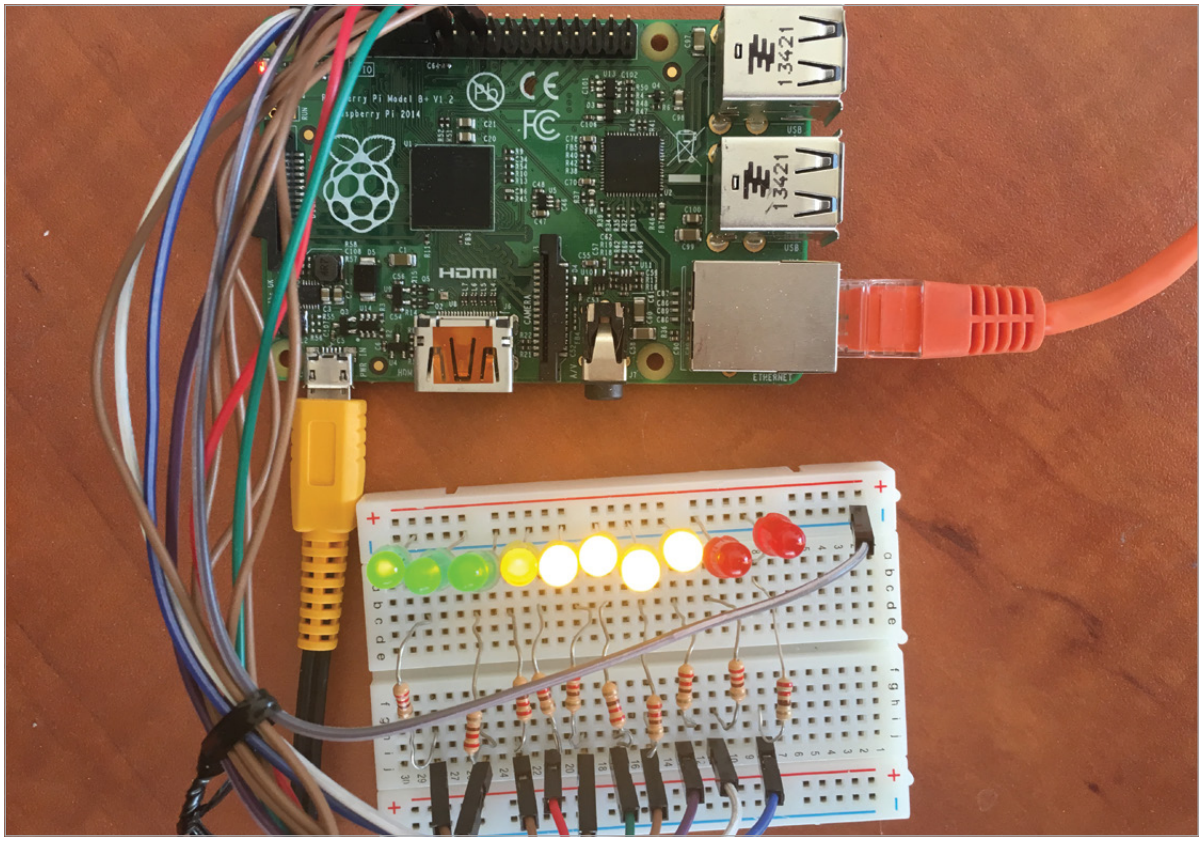
DOWNLOAD:
magpi.cc/1S8XD6o

If you would like to see the Python code that will get executed once you run the project, you can select 'Show code'.

>STEP-06

Check the thermometer

All that's left to do is to run the application. As long as you don't stop it, the device will read the temperature from the internet every 30 minutes and update accordingly. You could add more LEDs to make it suitable for a larger range of temperature values. We suggest you also put the device in a box so that it looks just like a thermometer. Now you'll know how warm it is outside with just a turn of your head.



Left The thermometer displaying a temperature of 15° Celsius

THE HAYLER-GOODALLS

Ozzy, Jasper, and Richard are mentors at CoderDojo Ham and gave a talk at the Raspberry Pi birthday party about their AstroPi adventures.
richardhayler.blogspot.co.uk
coderdojoham.org

RGB LED TWEET-O-METER

You'll Need

- RGB LED
- Breadboard
- Jumper wires
- 3× 100 ohm resistors
- Twitter developer account
- TextBlob Python library
- Twython Python library

Use the GPIO Zero Python library to control an RGB LED and see how well your tweets are doing

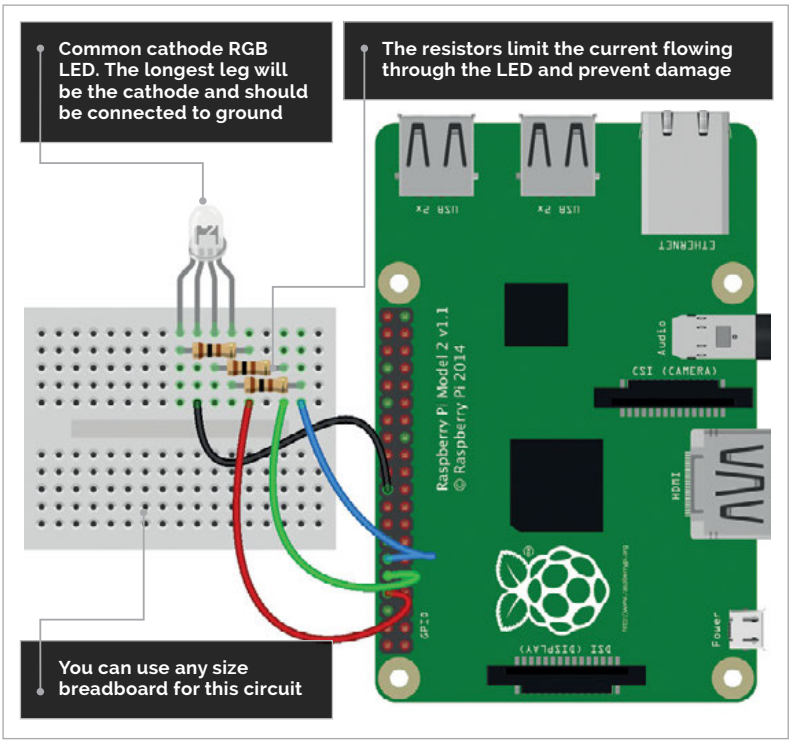
Keeping up to date with Twitter can be very time-consuming, especially if there are lots of tweets. What if you could see at a glance what the Twittersphere thinks about a certain topic? In this tutorial we're going to build a simple RGB LED circuit, and program it to change colour to indicate whether the tweets that include a given hashtag or keyword are using positive, negative or generally neutral language.

>STEP-01

Install Python libraries

Update your Pi to the latest version of Raspbian and download and install the additional software you'll need.

```
sudo pip3 install twython textblob
```



There are two libraries that make our project really easy. Twython allows you to contact Twitter using Python and collect tweets (you'll need to register for a Python developer account – see step 5). Then, to read the tweets in the code, we're going to use TextBlob; there are other libraries available, but this is one of the simplest.

>STEP-02

Do you like sausages?

Let's take a look at a simple example. Open a Python 3 interpreter (either use the command line or IDLE) and type:

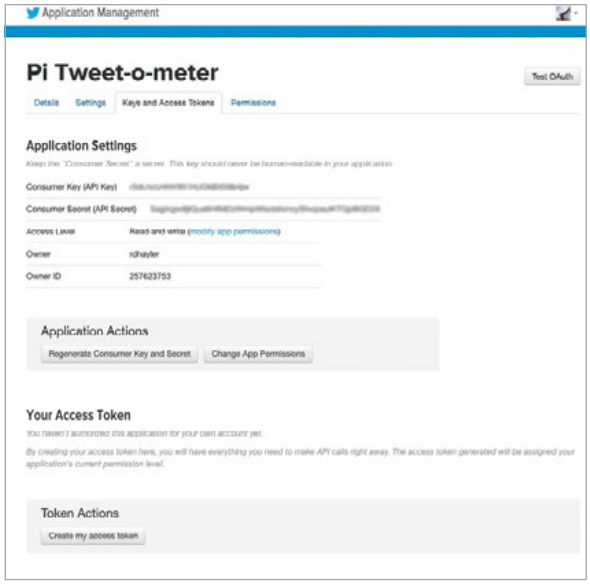
```
>>> from textblob import TextBlob
>>> sentence = TextBlob('I really like
sausages, they are great')
>>> sentence.sentiment.polarity
0.5
```

Any value for polarity greater than 1 indicates a positive sentiment (like); a value less than 1 suggests negative sentiment (dislike). Try changing the sentence and see how a different phrase will give a different result. Results will be more accurate if you have more text, although a 140-character tweet is normally good enough.

>STEP-03

Select your RGB LED

Light-emitting diodes (LEDs) are cool. Literally. Unlike a normal incandescent bulb which has a hot filament, LEDs produce light solely by the movement of electrons in a semiconductor material. An RGB LED has three single-colour LEDs combined in one package. By varying the brightness of each component, you can produce a range of colours, just like mixing paint. There are two main types of RGB LEDs: common anode and common cathode. We're going to use common cathode.



Above Once you've created your Twitter app, generate the required access tokens; copy these into your code

>STEP-04

Connect up the RGB LED

LEDs need to be connected the correct way round. For a common cathode RGB LED, you have a single ground wire and three anodes, one for each colour. To drive these from a Raspberry Pi, connect each anode to a GPIO pin via a current-limiting resistor. When one or more of these pins is set to HIGH (3.3V), the LED will light up the corresponding colour. Connect everything as shown in the diagram.

>STEP-05

Register as a Twitter API developer

Anyone with a Twitter account can register as a developer, although you might be asked to provide a mobile phone number or other identification details. Once you've registered, you need to create a new application at **apps.twitter.com**. Click the button to create a new app and then fill in the required fields. Once that's done, select the 'Keys and Access Tokens' tab and click on the 'Create my access token' button.

>STEP-06

Process some tweets

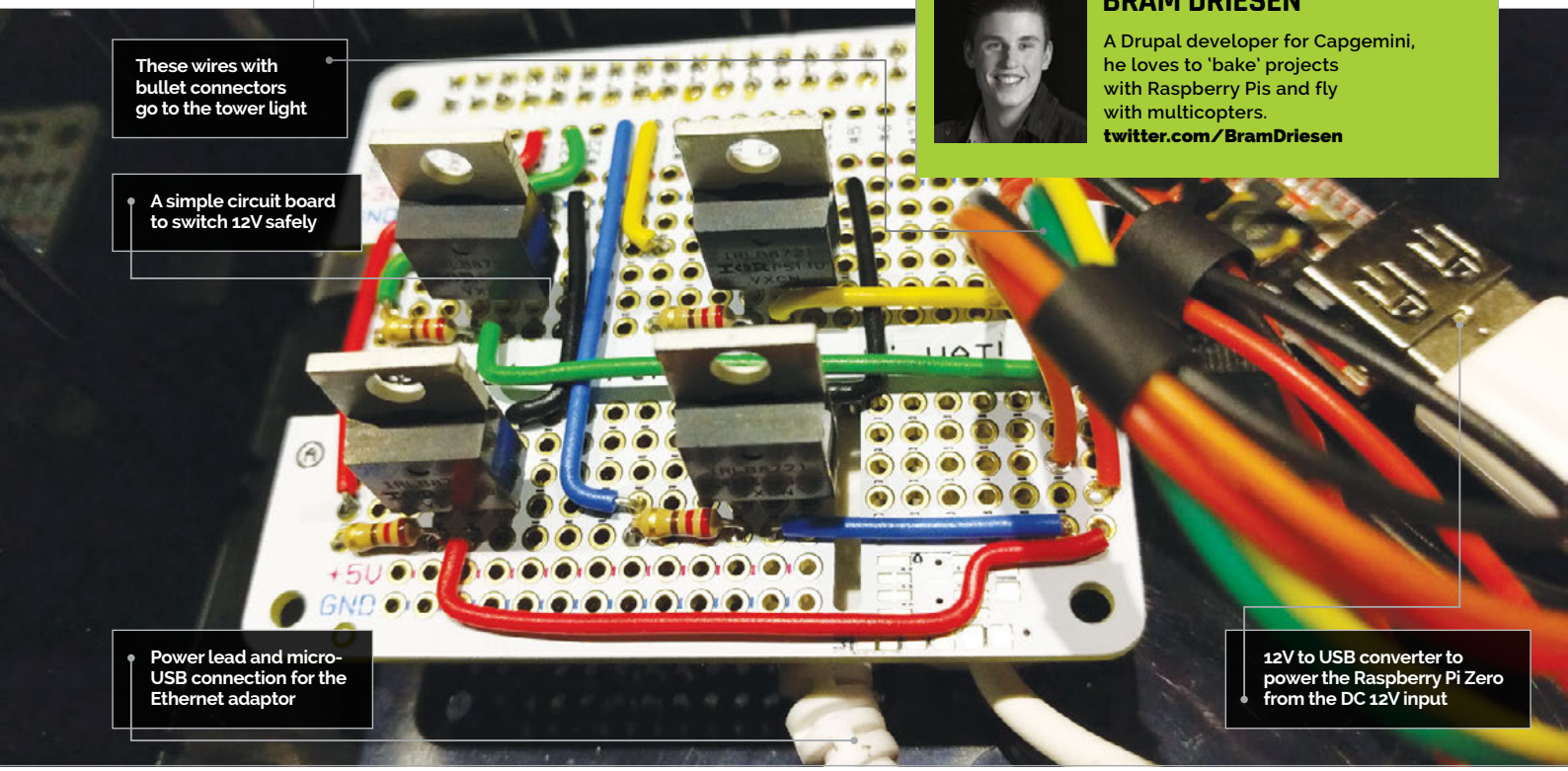
Download or type up the code from the **tweetometer.py** listing. Add the Twitter API keys and tokens generated in step 5 at the appropriate places. Now pick a hashtag or keyword for testing. The internet loves cats and dogs, and likes to argue about which is better, so we found that using 'cat' or 'dog' generated more than enough data! Run the code: you should see a running count of the analysed tweets on the console, and the LED should flash with each new matching tweet. Between new tweets, the LED will remain the colour of the sentiment with the biggest count.

tweetometer.py

Language
>PYTHON

DOWNLOAD:
magpi.cc/1WBerda

```
#Tweet-o-meter: Add your own Twitter API developer
keys (lines 9-12)
# and choose your own keyword/hashtag (line 56)
import time, sys
from textblob import TextBlob
from gpiozero import RGBLED
from twython import TwythonStreamer

# Add Python Developer App tokens and secret keys
APP_KEY ='ENTER APP KEY HERE' # <- CHANGE
APP_SECRET = 'ENTER APP SECRET HERE' # <- CHANGE
OAUTH_TOKEN = 'ENTER OAUTH_TOKEN HERE' # <- CHANGE
OAUTH_TOKEN_SECRET = 'ENTER OAUTH_TOKEN_SECRET HERE' # <- CHANGE

# Set our RGB LED pins
status_led = RGBLED(14,15,18, active_high=True)
# Set active_high to False for common anode RGB LED
status_led.off()
totals = {'pos':0,'neg':0,'neu':0}
colours = {'pos':(0,1,0),'neg':(1,0,0),'neu':(0,0,1)}

class MyStreamer(TwythonStreamer):
    def on_success(self,data): # When we get valid data
        if 'text' in data: # If the tweet has a text field
            tweet = data['text'].encode('utf-8')
            #print(tweet) # Uncomment to display  each tweet
            tweet_pro = TextBlob(data['text']) # Calculate sentiment
            # Adjust value below to tune sentiment sensitivity
            if tweet_pro.sentiment.polarity > 0.1: # Positive
                print('Positive')
                status_led.blink(on_time=0.4, off_time=0.2, on_color=(0, 1,
0), n=1, background=False)
                totals['pos']+=1
            # Adjust value below to tune sentiment sensitivity
            elif tweet_pro.sentiment.polarity < -0.1: # Negative
                print('Negative')
                status_led.blink(on_time=0.4, off_time=0.2, on_color=(1, 0,
0), n=1, background=False)
                totals['neg']+=1
            else:
                print('Neutral') # Neutral
                status_led.blink(on_time=0.4, off_time=0.2, on_color=(0, 0,
1), n=1, background=False)
                totals['neu']+=1
        overall_sentiment = max(totals.keys(),key=(lambda k: totals[k]))
        status_led.color = colours[overall_sentiment]
        print(totals)
        print('winning: ' + overall_sentiment)
        time.sleep(0.5) # Throttling

    def on_error(self, status_code, data): # Catch and display Twython errors
        print( "Error: " )
        print( status_code)
        status_led.blink(on_time=0.5,off_time=0.5, on_color=(1,1,0),n=3)

# Start processing the stream
stream2 = MyStreamer(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
while True:  # Endless loop: personalise to suit your own purposes
    try:
        stream2.statuses.filter(track='magpi') # <- CHANGE THIS KEYWORD!
    except KeyboardInterrupt: # Exit on Ctrl-C
        sys.exit()
    except: # Ignore other errors and keep going
        continue
```



BRAM DRIESEN

A Drupal developer for Capgemini, he loves to 'bake' projects with Raspberry Pis and fly with multicopters.
twitter.com/BramDriesen

You'll Need

- IRLB8721 MOSFETs
- 220-ohm resistors
- 12V DC tower light **magpi.cc/1rGNbwB**
- 12V power supply and jack
- 12V to USB converter
- Perma-Proto HAT **magpi.cc/1TQQcn8**
- Micro-USB to Ethernet adaptor
- Miscellaneous other parts

CREATE A JENKINS BUILD STATUS LIGHT

Jenkins is a great tool to monitor code quality. For greater visibility of the build status, you can build an awesome tower light that uses it

Right Using some neodymium magnets, you can mount the tower to any metal object, such as a closet or fridge door

Imagine being able to show the status of your Jenkins projects to anyone in the same room without the hassle of logging in to the dashboard. With some LEDs, a few other cheap components, and some soldering skills, you can complete this project. You can take it to the next level, however, by using an industrial tower light instead of regular LEDs. Employing an existing API, you can retrieve almost any data from Jenkins and get a nice visual indicator of the status of your project.

>STEP-01

Make a HAT

First, we need to create the HAT. We used a Perma-Proto board from Adafruit, but you can also use a breadboard if you don't want a permanent solution. In this version, the following GPIO pins were used, since they best fit the layout:

PIN 18: Red
PIN 23: Buzzer
PIN 24: Yellow
PIN 27: Green

You can use the GPIO output voltage in combination with a MOSFET and a resistor to switch 12V safely to the LEDs. There's also a 12V to USB converter to power the Raspberry Pi from the barrel jack, and a small power switch.

>STEP-02

The Python Jenkins-API

You need to start off with a clean Raspbian Jessie image (not the Lite version) and a Raspberry Pi Zero (make sure to solder the GPIO header pins!). After you have burned the image and completed the basic setup, make sure to set up an internet connection. You can either use a standard WiFi dongle or a micro-USB to Ethernet adaptor. Once you are ready, proceed with the installation of the Jenkins API. In the terminal window, enter the following command:

```
sudo apt-get install python-jenkinsapi
```

>STEP-03

Create the Python script

Most of the coding work has already been done for you and is available on GitHub. To get everything you need for this project, clone the project from GitHub on your Raspberry Pi by using the following command in your terminal:

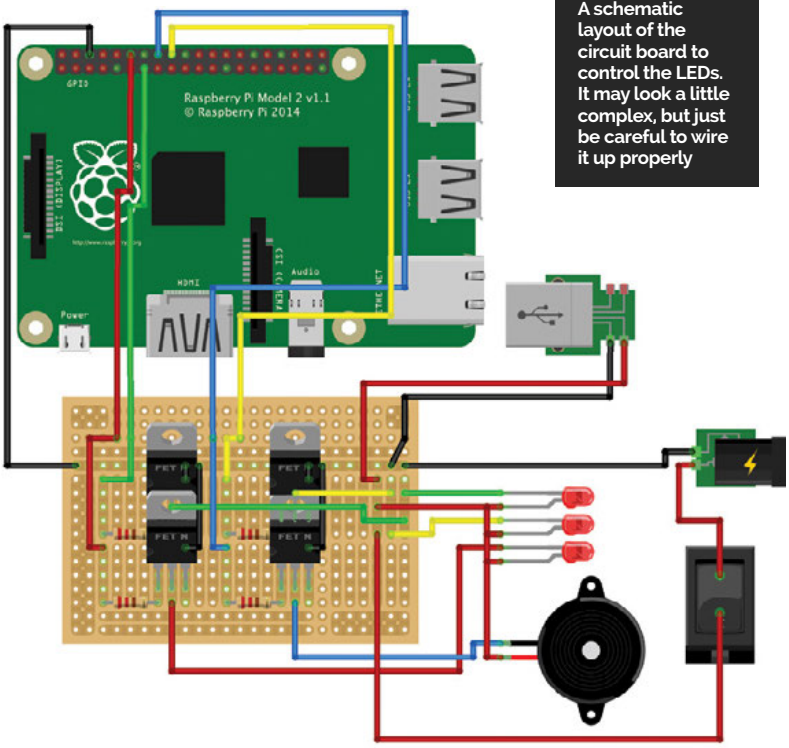
```
git clone https://github.com/BramDriesen/rpi-jenkins-tower-light.git
```

Make sure to clone the project in a directory where you can easily find it afterwards, such as the home directory. You can use **`cd <folder>/<name>`** and **`cd ../`** commands to navigate through folders in your terminal. You can also download a ZIP file of the source code from the versions page: **magpi.cc/1rGOjAt**.

>STEP-04

Configure the code

Now we have cloned the project, we'll browse into the directory to configure the Python script. Copy the default configuration file **default-config.py** and rename it to **config.py**. You can use the command **`cp default-config.py config.py`** to do this all in one go. After you have copied the file, edit it with your favorite text editor; we like to use nano, so in this case we'd issue the command **`nano settings.py`**. Now change the default configuration lines so it contains your credentials, job name(s), and correct GPIO pins according to your situation. If you have only one job, make sure the jobs parameter remains an array with one item inside.



A schematic layout of the circuit board to control the LEDs. It may look a little complex, but just be careful to wire it up properly

Default-config.py

Language
>PYTHON

DOWNLOAD:
magpi.cc/1rGOsnx

```
# Default configuration file for Jenkins
# Copy this file and name it config.py
jenkinsurl = "http://example-url.com:8080"
username = "your-username"
password = "your-password"
jobs = ['job-name-1', 'job-name-2']
gpios = {
    'red': 18,
    'buzzer': 23,
    'yellow': 24,
    'green': 27,
}
```

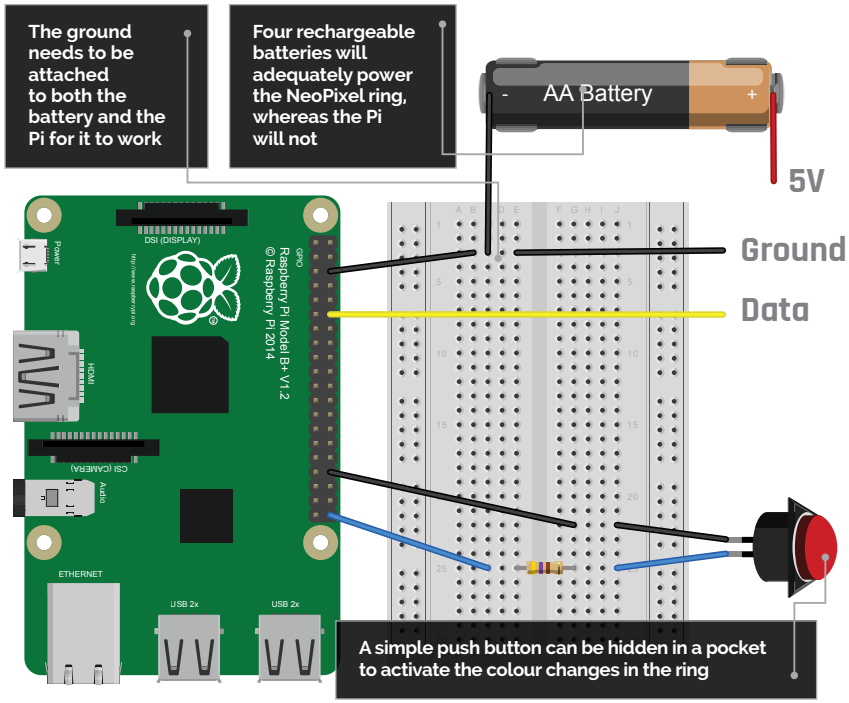
>STEP-05

Auto-run on boot

Since we're working on a permanent solution, we want to ensure that our script launches immediately on boot. We can edit our **rc.local** file to launch the script when the Raspberry Pi is booted up. Edit your rc.local file with **`sudo nano/etc/rc.local`** and add the following line to direct the script to auto-start upon boot:

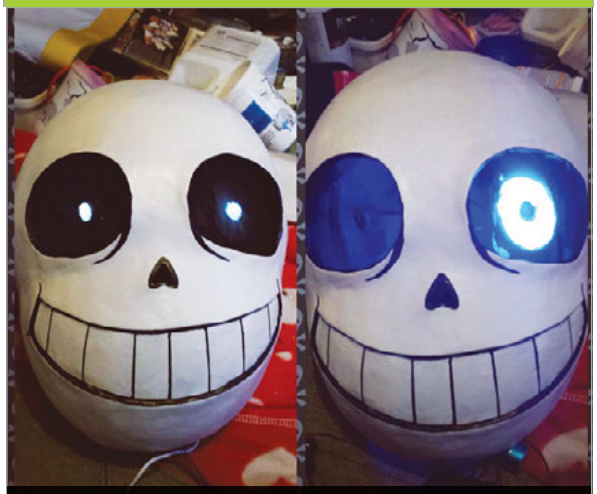
```
python /path/to/the/script/rpi-jenkins-tower-light/jenkinslight.py &
```

Reboot your Pi and you should be good to go! For maintenance, you can always SSH into the Raspberry Pi in the future.



ROB ZWETSLOOT

Tinkerer, sometimes maker, other-times cosplayer, and all the time features editor of *The MagPi*.
magpi.cc



The lights are attached behind black chiffon to slightly obscure them, and allow the cosplayer's face to remain hidden while still being able to see

You'll Need

- First-generation Raspberry Pi (A+, B+, Zero, etc.)
- NeoPixel ring
- 4× AA battery pack, preferably with a switch
- 4× rechargeable AA batteries
- Portable mobile phone power bank
- Push button
- Various wires and resistors

MAKE NEOPIXEL COSPLAY EYES WITH PI

Create your own powerful eyes for Sans from Undertale, by getting some LEDs and a NeoPixel ring. It'll be a skele-ton of fun!

NeoPixels are amazing, addressable RGB LEDs from Adafruit. You can get them in strips, rings or individually, and create incredible effects with them. However, it's not always clear how to control them on the Raspberry Pi.

Recently, friend of the mag Mel Nurdin (of Freyarule Cosplay: **magpi.cc/2iqoofE**) expressed an interest in using NeoPixels in a cosplay of theirs. We decided to cut through the information and create a Pi-powered pair of eyes for their costume: skeleton pun-maker and hotdog salesman Sans from the excellent *Undertale* game. Follow along to make your own Megalovania eyes, or learn how to use NeoPixels in general for your own projects.

>STEP-01

Prepare your Pi

A Raspberry Pi Zero was used in the final build, so it would fit better within the confines of the costume. Update a version of Raspbian Jessie by going into the terminal and using **`sudo apt-get update`** then **`sudo apt-get upgrade`**, before installing the software needed for the NeoPixels:

```
sudo apt-get install build-essential
python-dev git scons swig
```

Now download the library for the NeoPixels:

```
git clone https://github.com/jgarff/
rpi_ws281x.git
cd rpi_ws281x
scons
```

Finally, we can install the module to the Pi:

```
cd python
sudo python setup.py install
```

>STEP-02

Wire up the NeoPixel

It's worth testing this out on a breadboard first, so you can understand the concept of the circuit. Basically, the NeoPixel needs to be powered separately by the four AA batteries, with a data cable coming from a PWM-enabled pin on the Pi to control the LEDs.

Language
>PYTHON

DOWNLOAD:
magpi.cc/SansEyes

eyes.py

```
#!/usr/bin/env python

import time

from gpiozero import Button

from neopixel import *

button = Button(21)

# LED strip configuration:
LED_COUNT   = 16      # Number of LED pixels
LED_PIN     = 18      # GPIO pin
LED_FREQ_HZ = 800000  # LED signal frequency in hertz
LED_DMA     = 5       # DMA channel to use for
# generating signal
#LED_BRIGHTNESS = 255  # LED brightness
LED_INVERT  = False   # True to invert the signal

# Function to control colour change

def Megalovania(strip, color):
    iterations = 0
    for i in range(strip.numPixels()):
        strip.setPixelColor(i, color)
        strip.show()

# Create NeoPixel object with
# appropriate configuration
strip = Adafruit_NeoPixel(LED_
COUNT, LED_PIN, LED_FREQ_HZ,
LED_DMA, LED_INVERT)
# Initalise the library (must be called once before
# other functions)
strip.begin()

Megalovania(strip, Color(255,255,255))
time.sleep(2)

while True:
    Megalovania(strip, Color(0,0,0)) # Eyes off
    time.sleep(0.5)
    button.wait_for_press()
    Megalovania(strip, Color(0,0,255)) # Blue eye
    time.sleep(0.5)
    button.wait_for_press()
    time.sleep(0.5)
    while button.is_pressed == False:
        Megalovania(strip, Color(255,255,0)) # Yellow eye
        time.sleep(0.1)
        Megalovania(strip, Color(0,0,255)) # Blue eye
        time.sleep(0.1)
```

You don't need to step up the 3V3 signal from the Pi to do this, so to keep it simple, we're not. The main thing to remember is that the ground of the NeoPixel needs to be connected to the negative on the battery pack and a ground on the Raspberry Pi. We've connected the data pin for the NeoPixel to pin 6 (GPIO 18).

>STEP-03

Wire up the button

The button is the easy one to wire up. It doesn't matter which way around it goes, but we have it connected to pin 40 right at the end and to ground on pin 34, to keep it away from the NeoPixel wires. You'll also need a resistor suitable for your button – ours uses a 470 ohm resistor – and it can be connected to either side of the button. This is controlled by GPIO Zero, so it's addressed at GPIO 21 in the code.

>STEP-04

Add the code

Type up or download the code to the Raspberry Pi. You can either put it in the home folder or in its own folder; either way, you can test it by running it in the terminal with:

```
sudo python eyes.py
```

It should light up the LEDs white for a couple of seconds before turning them off. A button press will make them turn blue, and another button press will make the LEDs flash blue and yellow quite quickly, emulating a specific scenario in the game *Undertale*. Add this command to the end of the file **/etc/profile** so it will run on boot.

>STEP-05

How the code works

Each NeoPixel in a series can be addressed individually. As the code only needs to have all the lights the same colour at one time, we've created a **for** loop in the main function that goes through and sets each LED to the same colour one by one. We've also told the code how many LEDs there are, where they're connected, and what frequency to use for refreshing them. The rest of the code is fairly straightforward, waiting for button presses and using delays to manage them.

>STEP-06

Finishing up

That's the pure circuit, but how is it used in the costume? Mel installed a ring of frosted acrylic into the skull she made and stuck the NeoPixel ring behind it: this served to diffuse the individual lights into a more coherent ring. Normal white LEDs were mounted in both eye sockets and controlled by a separate switch for the 'normal' eyes. The button from the Pi was on a long enough cable that it fit sneakily into her pocket – useful, as the character keeps his hands in them!

GARAGE TECH

Advocate of physical computing, digital fabrication, and tech recycling.
garagetech.tips
twitter.com/GarageTechTweet

You'll Need

- USB 5V voltage booster step-up module **magpi.cc/1U5Fz0p**
- LiPo battery charger **magpi.cc/2ip57g8**
- Lithium polymer battery – 1000mAh **magpi.cc/2ipBZ8B**
- Nylon stand-offs **magpi.cc/1WD2eVf**
- Red LED bar graph display **magpi.cc/1WD1tLN**
- 220Ω SIL network - 8 commoned resistors **magpi.cc/1WD1SxG**
- 36-way header strip **magpi.cc/1WD1YFA**
- Case fan
- Stripboard

WRITE TEXT IN THIN AIR WITH A PI ZERO

Create a persistence of vision illusion, an interesting project that's fairly easy to achieve and from which there is plenty to learn

In this article, we're going to show you how to build a Raspberry Pi Zero setup to display text from a file using an LED bar. The fun part is that the Pi and all the rest of the hardware are spinning, glued onto a PC fan, to deliver a fantastic persistence of vision (POV) effect whilst remaining wirelessly connected to your network!

What is POV?

You can experience persistence of vision in many ways. From something as simple and obvious as a flip book or a film, to seeing car wheels spinning in the other direction, the principle is very similar. These are an illusion: what we perceive as motion is given by a series of discrete images shown to us at a specific rate. If the number of images per second is too low, we perceive them as just pictures; get the right frame rate and you see motion.

Persistence of vision with LEDs has been a thing for a while and has been perfected over time to extremely sophisticated levels.

Our example works with eight LEDs arranged in an LED bar and installed on a rotating platform. When the platform, our CD, starts spinning, an LED which is turned on will give the illusion of a bright disc. This is very similar to what happens when you start writing your name in the air with a sparkler on bonfire night.

Imagine that you're able to control the sparkler very precisely, and it can also be switched on and off at very precise moments during its motion. In such conditions, you could be able to see something that would remind you of Morse code.

In the same way, as the CD rotates, we'll be controlling our LED bar with very precise timings, and these will give you the impression of very complicated light patterns which we eventually use to display characters.

Building the POV

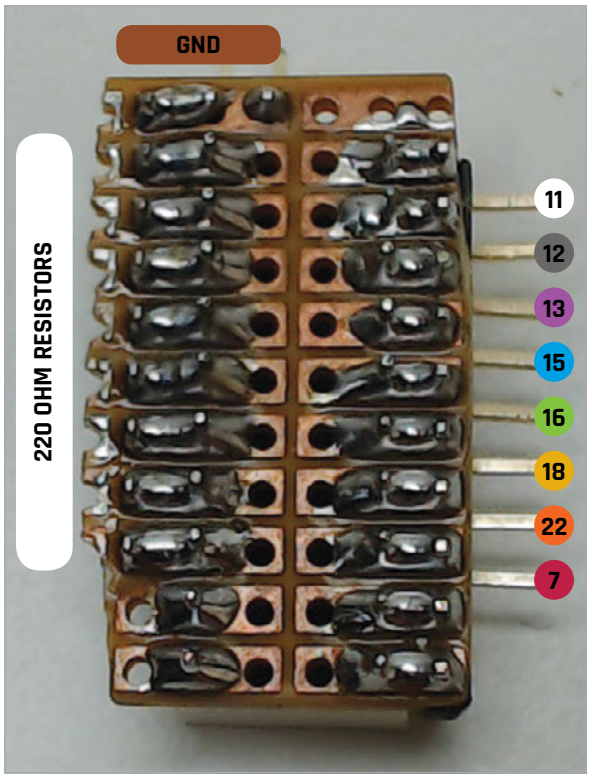
The most important part of getting this POV to run smoothly is the way you fix the system onto the CD. Placing the Pi Zero in the correct place will make it so that, when turning, the whole setup will have as few vibrations as possible. The rule of thumb we followed was to drill two 3mm holes on opposite sides of the central hole of the CD, so that they would sit on one of the disc's diameters. The two holes correspond to the ones on the Pi Zero's PCB, which are diagonally opposed to each other.

Fix the Pi with the spacers and install the rest of the components as shown in the picture. The LiPo battery gets tucked in under the Pi.

Be creative about how you fix things, and use the correct amount of Blu-Tack and cable ties.

We modified the LiPo battery terminals so that we could easily plug it into the charger without needing to take anything apart.

Use a spacer between the fan and the CD to keep things away from the fan frame. Lastly, put together the LED bar as shown in the picture below, following the corresponding colour and GPIO pin numbers.



CONTAIN THE VIBRATIONS

Even the cable connected to the LED bar will introduce substantial imbalance if not properly anchored. Pull all the slack of the cable to the centre of the CD and fix it tightly so that it won't move once the disc is spinning. You might want to experiment, drilling a few holes to find the sweet spot for where to place the LED bar, but after a bit of trial and error you should have a stable enough setup.

Above Have it write anything you want in the air – it looks better in real life than in photos!

Tuning the rotation speed

Getting the timing right works better by trial and error than by making very complicated calculations, and that's how we approached the problem. We needed to make it so that the frequency at which the PC fan is spinning, and the frequency with which we're switching the LEDs on and off, somehow converge on a point at which the illusion of floating text is convincing. We started by choosing about 5-6V with a bench power supply to power our 12V PC fan. That meant the fan was turning slower or faster, depending on the voltage selection of the power supply. In the meantime, we started by blinking one of the LEDs with some regular timing and went off to see if a dotted line was appearing instead of the bright disc.

Changing the timing between flashes helped us tune the dotted line so that it looked like a succession of bright and dark lines which appeared about the same length. Once we got there, it was just a question of putting together the right code to show the characters we wanted.

In your own setup, you will only need to adjust the voltage provided to the fan once the script is running, until you see something floating in the air.

Going for a spin

We provide the demo code **PiZeroPOV.py** on GitHub (**magpi.cc/1WD4ADE**), which will display the contents of **pov.txt**. The script will check the contents of the text file every 10 seconds and display them on the POV. As the Pi Zero is connected to your network via its wireless adapter, you can change what's being displayed at any time without needing to shut down or alter the spinning system in any way. The only limitations of this demo script are that it only accepts lower-case characters and spaces – anything else will break it miserably, but, hey, anybody can easily improve it with a bit of time and dedication.

So, as you log into your rotating Pi Zero, just type:

```
sudo python PiZeroPOV/PiZeroPOV.py &
```

...and start editing **pov.txt** by using nano:

```
nano PiZeroPOV/pox.txt
```

This will get your messages in the air!

Language
>PYTHON

DOWNLOAD:
magpi.cc/1WD4ADE

LUCY ROGERS

Tamer of robot dinosaurs, hacker of the IoT, author. Translates science into plain English. Positively affecting the lives of a billion people.
lucyrogers.com

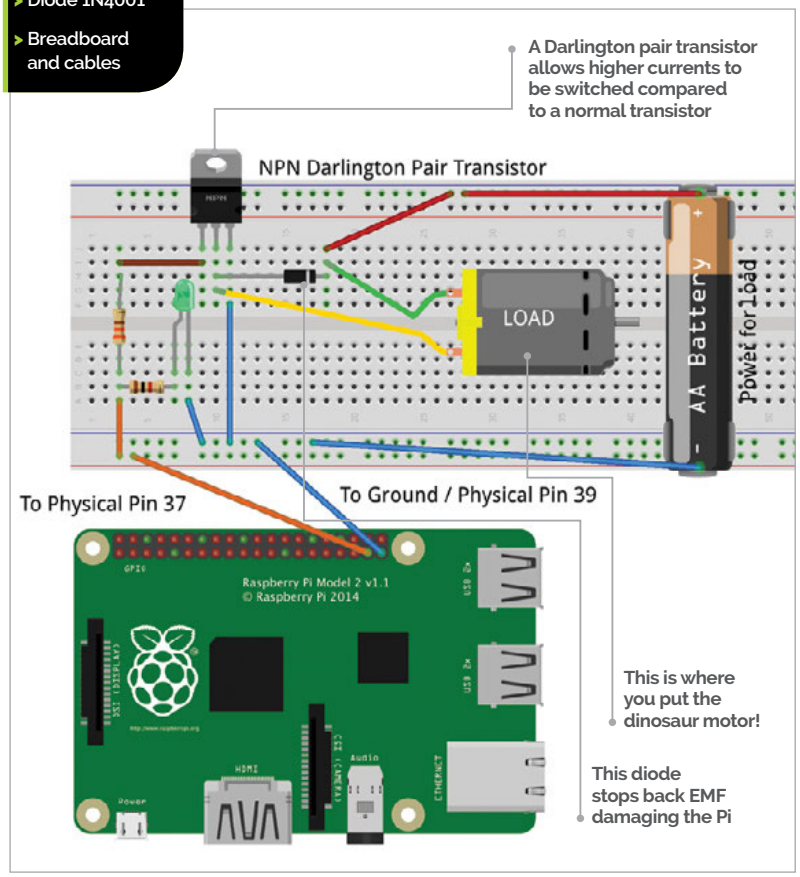
MAKE A DINOSAUR REACT TO YOUR EMAIL

Hack a toy dinosaur with Node-RED to provide email notifications

You'll Need

- Connex Action Dinosaur Electronics Kit: **magpi.cc/1ScJbZX**
- LED & 1kΩ and 330Ω resistors
- NPN Darlington pair transistor
- Diode 1N4001
- Breadboard and cables

Want to let the kids know it's time for dinner by waking a toy in their bedroom? Or have a fun alert on your desk when you receive an email? This project takes you step by step through the process of hacking a cheap toy and using a Raspberry Pi, Node-RED drag-and-drop visual programming, and simple electronics to make a dinosaur move on demand. The electronic circuit used in this project can also be used to control larger motors, relays, or high power LEDs. First step, toy dinosaurs – next step, email control of all your home automation devices?



>STEP-01

Install Node-RED on your Pi

Before connecting anything to the GPIO pins of your Pi, we'll draw the code. If you have the latest version of Raspbian Jessie, Node-RED comes installed in your Programming folder. If not, you can install it via a terminal (see **magpi.cc/28PJz2O**). You also need the Firefox browser. Connect your Pi to the internet, open Node-RED (**Menu>Programming>Node-RED**), and paste the address at the top of the Node-RED window into Firefox. This address should be something like **http://192.168.X.XXX :1880**. You should now see the Node-RED flow page. If you find everything is sluggish, you may like to access that address from another computer's browser.

>STEP-02

Intro to Node-RED

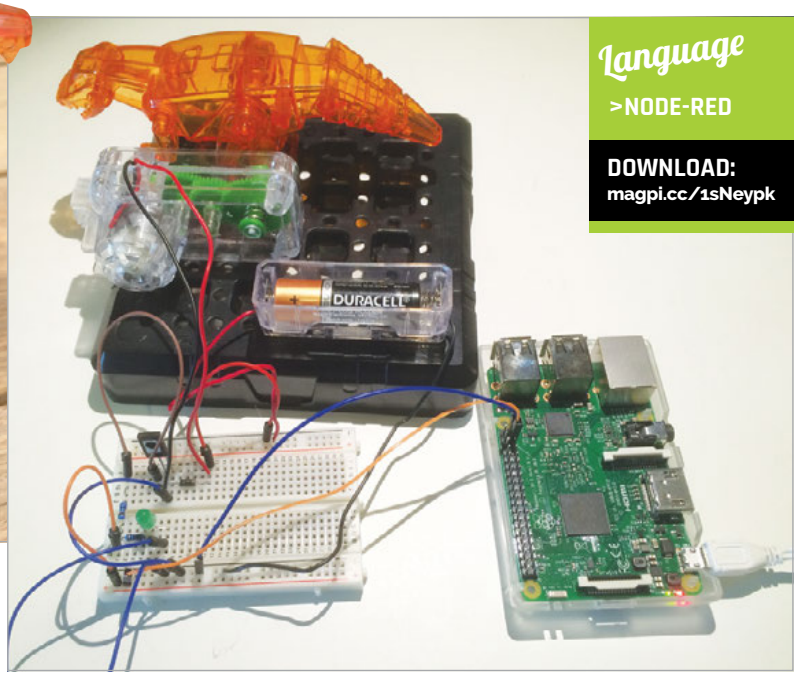
If you haven't used Node-RED before, start by making an 'inject-debug' flow. Drag an **inject** node from the Input section of the left menu into the middle pane. Its name should change to **timestamp**. Repeat with a green **debug** node from the Output menu. Join the two by clicking one of the grey pimples and drag it to the pimple on the other node. Click the big red 'deploy' button at the top-left. Activate the **inject** node by clicking the square to the left of the word **timestamp**. The resulting message (payload) appears in the debug tab.

>STEP-03

Draw the flow

Drag the **email** node with the connector on the right into the work area. Double-click it and enter your email credentials. Connect a **debug** to the **email** node, to check what messages are arriving. The message contains the email text and who it's from. Drag in a **switch** node and double-click to make some changes. For Rule1, change the '==' to 'Contains' and type

Set your dinosaur free! There are already a few dinos in the wild – tamed by @andysc, @neilcford, @fortoffee and @andypiper



Language
>NODE-RED

DOWNLOAD:
magpi.cc/1sNeypk

'Dinner' (without quotes) into the text string box. Any email that contains the word 'Dinner' will output to the first (top) output node. Click '+ rule' near the bottom, and change '==' to 'otherwise'. All other messages will go to the second (bottom) output node.

>STEP-04

Turn the motor on and off

The message must just be '1' or '0' to turn the motor on or off. To change it from the text, add a **change** node and change the message to a 1. Connect an **inject** node to the input of the **change** node. This will allow you to text the flow without sending an email. Add a **delay** node and another **change** node, and change the message to zero, otherwise the dino won't stop. Connect both the first **change** node and the **delay** node to a **GPIO out** node (connector on left). Set the pin to physical pin 37. Deploy the flow.

>STEP-05

Make the circuit and dinosaur

Connect the circuit as shown in the diagram. Turn the Pi off before connecting anything to the GPIO pins. As this motor is low current, we could use a transistor rather than a Darlington pair. However, the latter allows higher currents to be switched, and so this circuit can be used for other projects – but never mains or AC. Assemble the dinosaur kit. Make sure the dino can nod freely. The kit's battery holder and motor connect straight to the breadboard. You don't need the switch. On the Raspberry Pi, use the ground on physical pin 39 and GPIO pin 37.

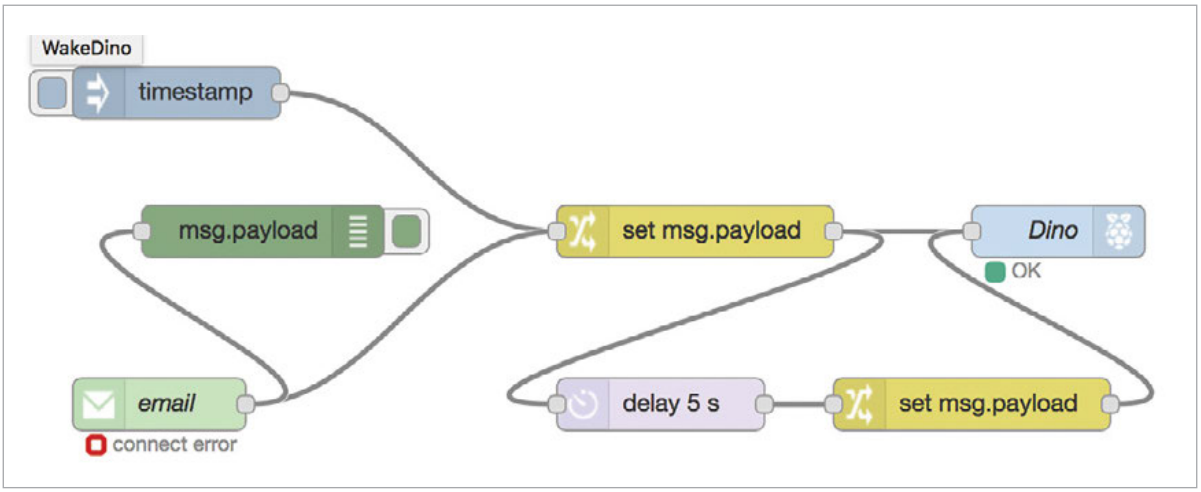
Above The complete project. Now you just need to email 'Dinner' to get everyone's attention

>STEP-06

Make dino nod and roar

Turn the Pi back on and open Node-RED again. Click the **inject** node and the dinosaur should nod. Now send an email with 'Dinner' in the message (not in the subject). The dino should nod again!

To get your dino to speak, attach a speaker to the Raspberry Pi, then add an **exec** node into the flow. You may need to install MPlayer onto the Pi (use **sudo apt-get install mplayer** from a terminal). Make sure your 'roar' MP3 file is on the Pi. Double-click the **exec** node and type in the command box: **mplayer /home/pi/.node-red/roar.mp3**



Left This Node-RED flow will trigger if any email is received. The delay node stops the dino after five seconds

PYTHON

If you want to do this using Python, see @ForToffee's blog (**magpi.cc/1sNeTZj**) or Chris Robbins' (**magpi.cc/1sNeMNi**).

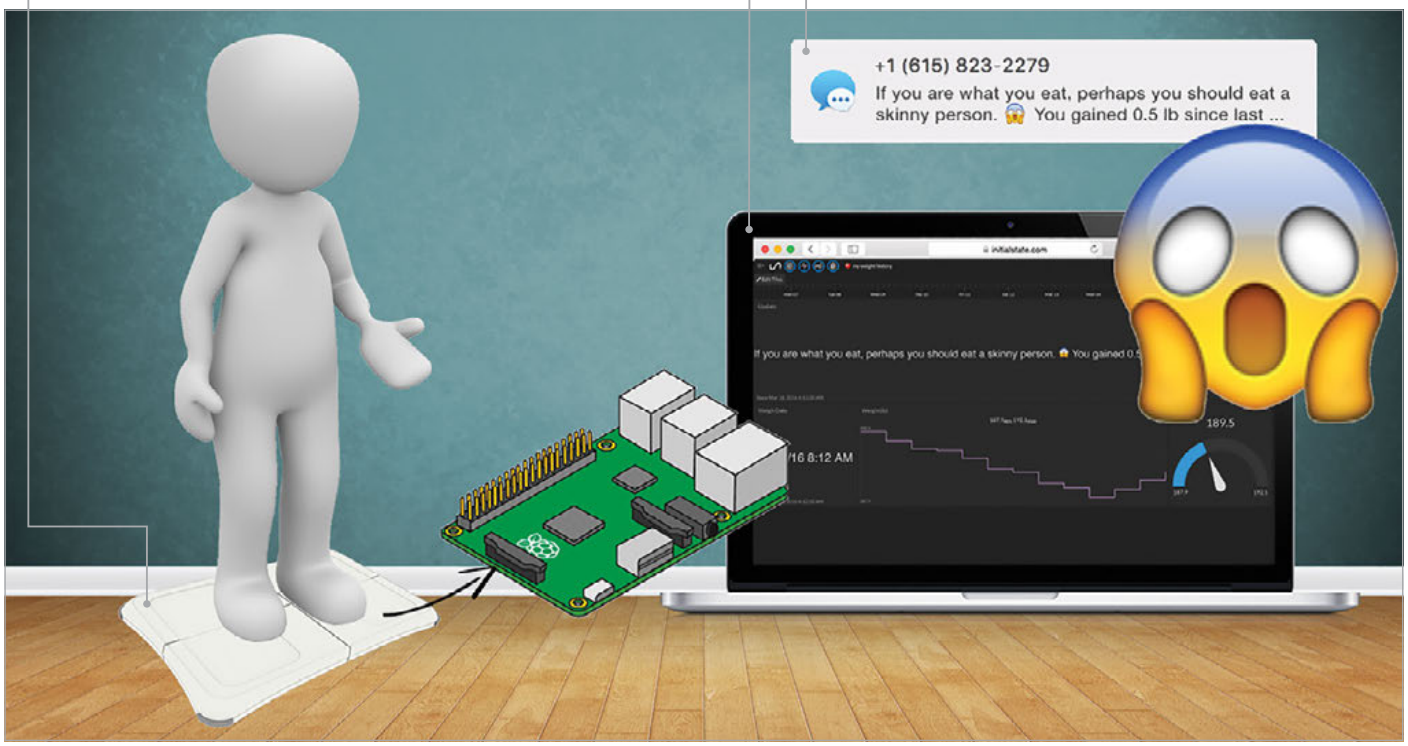
JAMIE BAILEY

Jamie is an electrical/system engineer, integrated circuit designer, and CEO/founder of Initial State, a data analytics service for Internet of Things devices.
initialstate.com

The Wii Balance Board is a Bluetooth scale that can be controlled from a Python script

Weight measurements are sent to your dashboard on Initial State's website

A snarky yet informative SMS is sent every time you weigh yourself



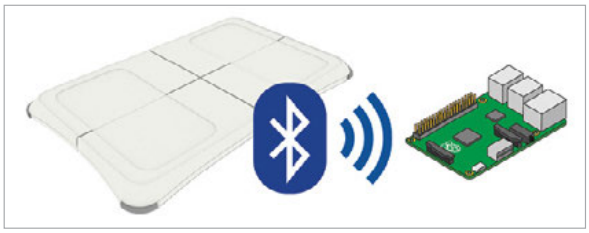
BUILD A WEIGHT-TRACKING WISECRACKING SCALE

You'll Need

- Wii Balance Board
- Wii Fit rechargeable battery pack
- ⅜″ Felt pads
- Pencil

Connect a set of scales to the web that tracks your weight and sends you text message updates with an attitude

Oh, that boring, soulless bathroom scale. We love to hate you when you don't show us that number we want. We swear at you, as if you'd care. Why hasn't anyone made a scale that's actually fun to use? It's time to create a scale that's not only smart, but has a bit more personality to brighten your day. We're going to build our very own hackable, weight-tracking, text-messaging bathroom scale that comes with a built-in sense of humour.



Right The Wii Balance Board pairs with your Pi through Bluetooth, making it easy to read your weight with a script

Setting up the Wii Balance Board

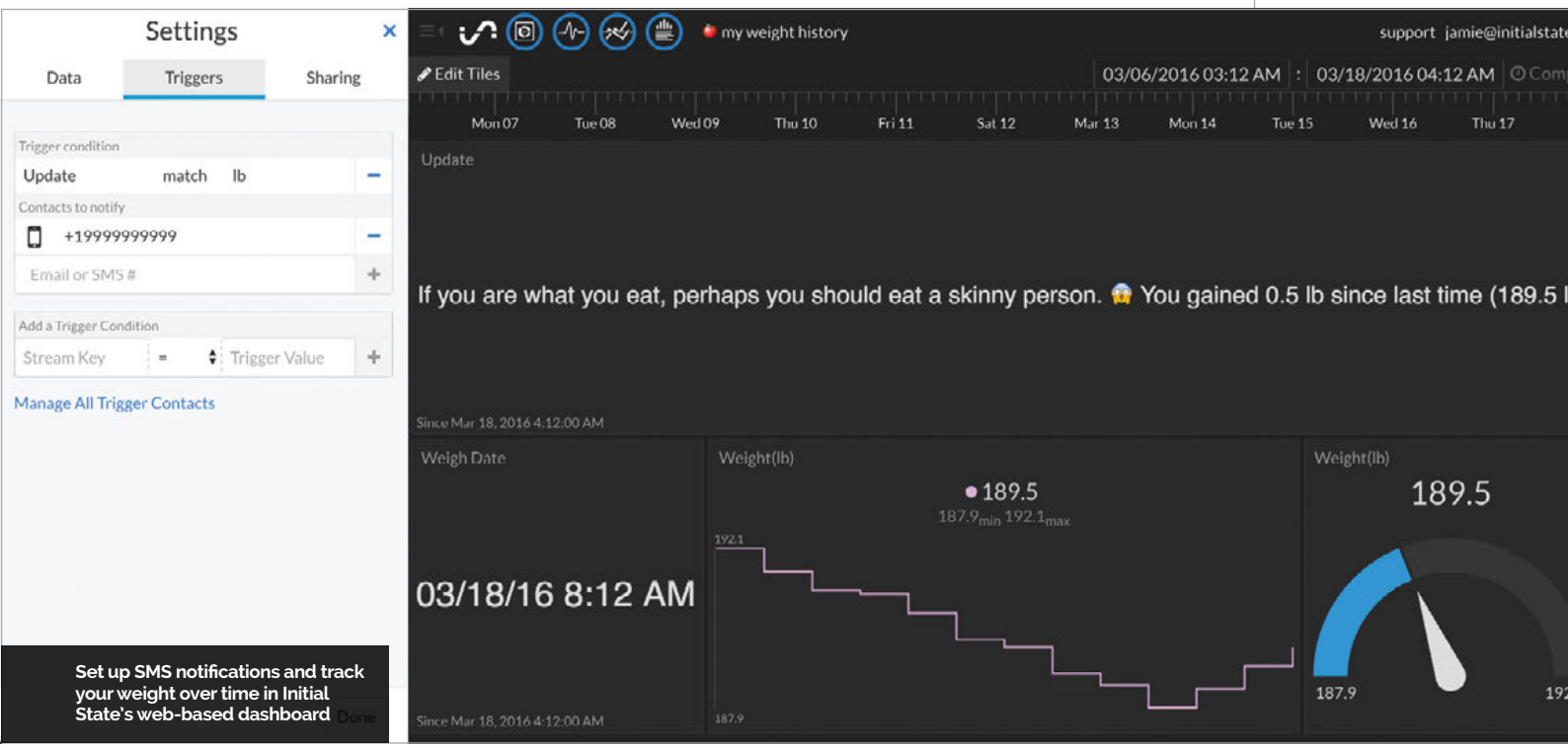
As the Raspberry Pi 3 comes with Bluetooth built in, it makes it very easy to communicate with the Wii Balance Board. If you have a Raspberry Pi 1 or 2, you'll need to use a USB adapter such as an inexpensive USB Bluetooth 4.0 Low Energy adapter (**magpi.cc/2ioms8V**).

Power on your Pi and open up a terminal window. You can see the address of your Bluetooth dongle by entering the following command:

```
hcitool dev
```

Install the Bluetooth modules that we will be using in our Python scripts:

```
sudo apt-get install python-bluetooth
```



Set up SMS notifications and track your weight over time in Initial State's web-based dashboard

After installation completes, we're ready to connect and communicate with the Wii Balance Board. We won't be permanently pairing our Board with our Pi, like we do with most of our Bluetooth devices. The Wii Balance Board was never intended to be paired with anything other than a Wii. Pairing will happen every time we run our Python script.

It's time to connect our Wii Balance Board to our Raspberry Pi. We'll do this by running a Python script. To get the scripts we'll use for this project, clone the GitHub repo:

```
cd ~
git clone https://github.com/InitialState/
smart-scale.git
cd smart-scale
ls
```

You should see two Python files in the new **smart-scale** directory: **smartscale.py** and **wiiboard_test.py**. Run the wiiboard_test.py script to test communication and take weight readings from the Wii Balance Board:

```
sudo python wiiboard_test.py
```

You'll see the following response:

```
Discovering board...
Press the red sync button on the board now
```

Remove the battery cover underneath the Wii Balance Board to locate the red sync button. Make sure that you press the button within a few seconds of running the script or a timeout will occur. Once successful, you'll see something similar to the following on rhe screen:

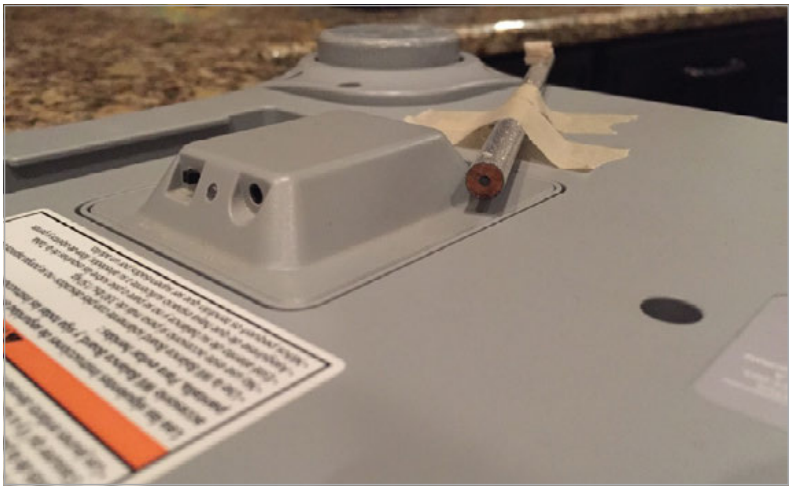
```
  Found Wiiboard at address
00:23:CC:2E:E1:44
  Trying to connect...
  Connected to Wiiboard at address
00:23:CC:2E:E1:44
  Wiiboard connected
  ACK to data write received
  84.9185297 lbs
  84.8826412 lbs
  84.9275927 lbs
```

You have now successfully converted your Wii Balance Board into a Raspberry Pi-connected scale. Now, let's make it a cool scale.

INITIAL STATE

Initial State is an easy-to-use platform for collecting data from connected devices and turning that data into dashboards, waveforms, notifications, and more. **initialstate.com**

Create a hacky lever to access the sync button on the bottom of the Board, using a pencil and a few ⅜″ felt pads



Above The Wii Fit rechargeable battery pack allows you to constantly power your scale from a wall outlet, to avoid having to do a Bluetooth sync each time you weigh

Hardware tweaks

Nintendo assumed you would always power your Wii Balance Board with four AA batteries and so included no AC power adapter. Having only battery power would be inconvenient because we cannot permanently pair our Wii Board to our Pi through Bluetooth. We need to sync it, then allow it to remain synced without draining the batteries, so that we can simply step on the scale and weigh. Luckily, there are several third-party adapters made for the Wii Balance Board that we can use to provide constant power from a wall outlet. A Wii Fit rechargeable battery pack (**magpi.cc/2inVSg2**) is a perfect solution to our power problem. Replace the batteries with this battery pack, and plug the AC adapter into a wall outlet.

Having to pair the Wii Balance Board and Raspberry Pi every time we run our Python script presents another inconvenience due to the location of the sync button. The sync button is at the bottom of the Wii Board, which means we would have to flip it over every time we sync. We can fix this by making a hacky little lever using a pencil and three ⅜″ felt pads. The rechargeable battery pack exposes the sync button to the underneath surface of the Board. Tape a pencil (or something similar) that runs from the sync button to the outside front of the Board. Stack three ⅜″ felt pads (or something similar) on the centre(ish) of the pencil to create a stationary pivot. Flip the Board over and you can press the sync button by simply pressing down on the lever. Hacky but effective.

ADD YOUR OWN HACKS

You can modify the messages that get sent, the units used for measurement, and even add your progress toward your own weight loss goals by modifying the existing Python script.

Initial State

We want to stream our weight/data to a cloud service and have that service turn our data into a nice dashboard that we can access from our laptop or mobile device. Our data needs a destination; we'll use Initial State as that destination.

Go to **magpi.cc/1TFlhaz** and create a new account, then install the Initial State Python module onto your Pi:

```
cd ~
\curl -sSL https://get.initialstate.com/
python -o - | sudo bash
```

When prompted to automatically get an example script, type **Y**. This will create a test script that we can run to ensure that we can stream data to Initial State from our Pi. Run the test script to make sure we can create a data stream to your Initial State account:

```
python is_example.py
```

Go back to your Initial State account in your web browser. A new data bucket called 'Python Stream Example' should have shown up on the left in your bucket shelf. Click on this bucket and view it in the visualisation apps Tiles, Waves, and Lines.

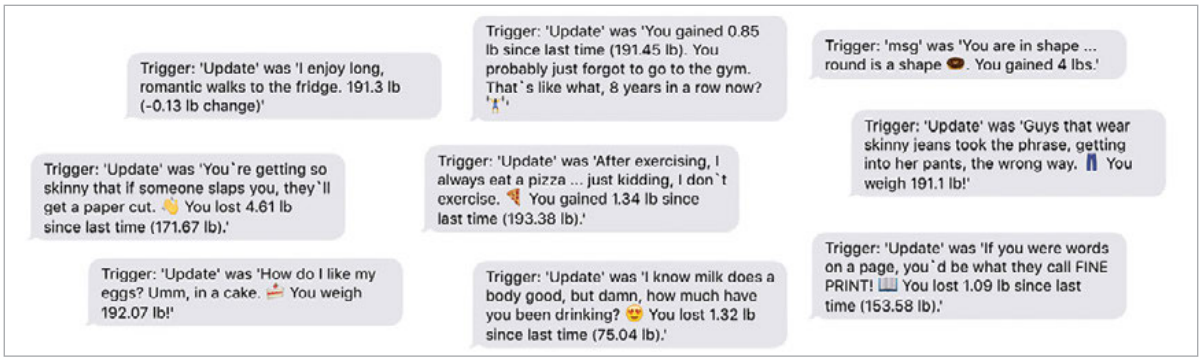
You are now ready to start streaming real data from your scale.

The final script

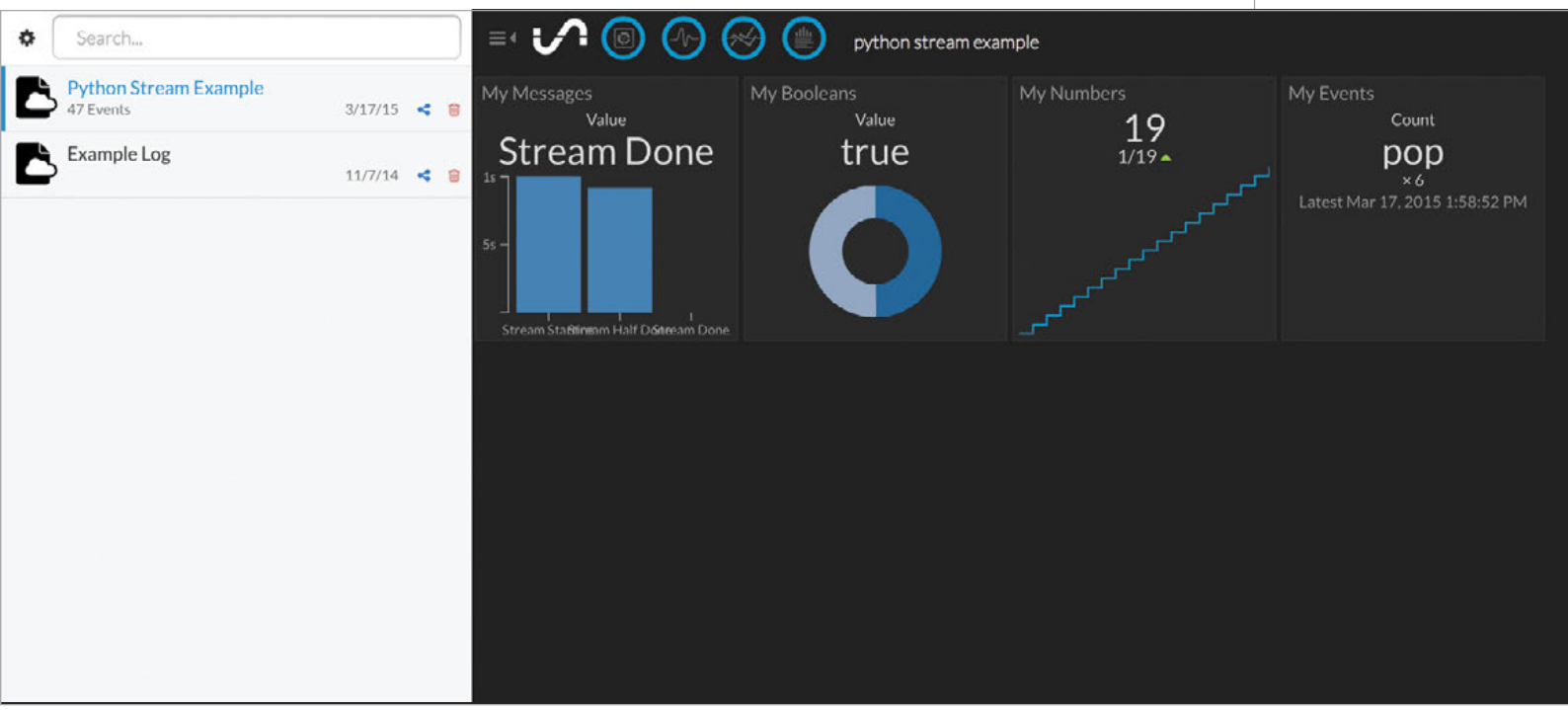
The final script that puts everything together is called **smartscale.py**, in your **~/smart-scale** directory. A few settings need to be set in the script before you can run it. Open up **smartscale.py** in your favorite text editor, such as nano:

```
cd ~
cd smart-scale
nano smartscale.py
```

Near the top of this file, there is a User Settings section. These are all of the settings that you can tweak to your taste. The only setting you have to set is your ACCESS_KEY, which is your Initial State account key. If you don't put your ACCESS_KEY in this field, your data won't show up in your account. Go to your



Right Receive a variety of funny, inspiring, and insulting SMS messages from your smart scale



Above Run a test script from your Pi to make sure you can create a data stream to your Initial State account

> Luckily, there are several third-party adapters made for the Wii Balance Board that we can use to provide constant power from a wall outlet

Initial State account in your web browser, click on your user name in the top-right, then go to 'My Account'. You'll find your access key at the bottom of the page, under 'Streaming Access Keys'.

Once you have specified each parameter in this section and saved your changes, you're ready to run the final script:

```
sudo python smartscale.py
```

Step on the scale to take a measurement. Go to your Initial State account and click on the new data bucket with the name corresponding to the BUCKET_NAME parameter (i.e. My Weight History). Click on Tiles to view your weight history dashboard. You can customise your dashboard by resizing and moving tiles, as well as changing view types and even adding tiles.

SMS setup

Let's create a SMS alert whenever the scale takes a weight measurement. First, make sure your weight history data bucket is loaded, then click on the bucket's settings in the data shelf (under its name). Click on the Triggers tab; if you don't see a Triggers tab, make sure you're subscribed to an Initial State account plan that allows Triggers.

Select the data stream to trigger on; you can use the drop-down list to select from existing streams once a data bucket has loaded, or you can type in the stream name/key manually. Select the 'Update' data stream and then select the conditional operator, in this case 'match'. Select the Trigger value that will trigger an action (manually type in the desired value). Type in **lb** if you are using imperial units, or type in **kg** if you are using metric units. Whenever the Update stream contains 'lb' (or 'kg'), you will get a text message notification.

Next, click the '+' button to add the Trigger condition. Select the action ('notify by SMS') and then click the '+' button to add the action. Input any verification code, if adding a new phone number, to complete setup.

Your trigger is now live and will fire when the condition is met. Click Done to return to the main screen. Every time you weigh yourself, you will get an SMS that contains your weight, how much it has changed since the last measurement, and a random joke, insult or compliment.

Customisation

Not a fan of the humour built into this project? You can change the jokes in the `messageWeighFirst`, `messageWeighLess`, `messageWeighMore`, and `messageWeighSame` functions to whatever wacky brand of humour you have. You can also stream data from other sources into the same weight history dashboard to create your own personal health dashboard. Hack away.

ADD MORE DATA

You can stream data from other sources into the same weight-tracking dashboard to create your own personal health dashboard. Learn more at **magpi.cc/1NqK8mp**

THE HAYLER-GOODALLS

Ozzy, Jasper, and Richard are mentors at CoderDojo Ham and gave a talk at the Raspberry Pi birthday party about their Astro Pi adventures.
richardhayler.blogspot.co.uk
coderdojoham.org

You'll Need

- Initial State account **initialstate.com**
- ISStreamer and Python 3 library
- psutil Python 3 library

TERRAFORMING MINECRAFT

Everyone has their favourite Minecraft block. What if you could have an entire world made out of them?

Imagine fields of gold, fit for King Midas or the dragon Smaug. Or how about a frozen landscape where everything has been turned to ice? Just think what you could do in a world where everything is primed TNT.

Using Python, we can start a terraforming process to remake a *Minecraft* world to your specifications. Even on a Pi 3, this will not be a quick process: depending on how complex your landscape is, and how much you want to transform, it may take several days. So we'll monitor our progress by uploading data to an Initial State dashboard so that we can keep track of things remotely. If you just want to do the terraforming, there's another version of the code without the Initial State functionality in the same GitHub repository (**terraforming_no_is.py**).

>STEP-01

Generate your world

Before you start coding, you need to create your *Minecraft Pi Edition* world and select the block type with which you want to fill your world. This can be any block of your choice, but it has to be a solid block (not ladders or torches). Manipulating the *Minecraft* ecosystem can be tricky. For example, if you try to turn water directly to lava, you'll probably end up with lakes of obsidian, so you might need an intermediate step: turn all the water to something inert (like wool), then transform it to lava. There may also be some blocks you want to keep – snow or water, for example.

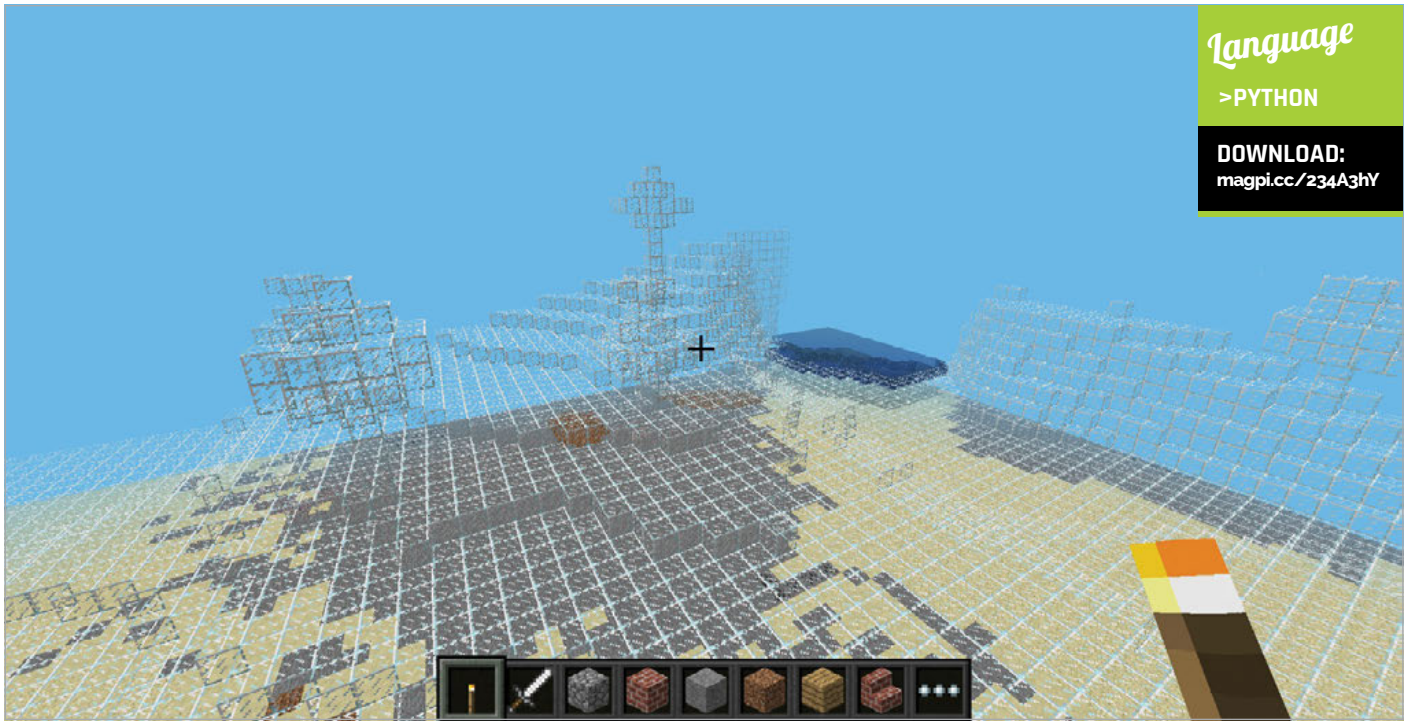
>STEP-02

Get the code

Make sure your Pi is up to date and, if you want to create a remote monitoring dashboard on Initial State, download and install its data streaming library:

```
sudo pip3 install ISStreamer psutil
```

Then download the **is_terraforming.py** code (**magpi.cc/234A3hY**). Note that you'll need to change some of the values to suit your *Minecraft* environment and to include your Initial State account details.



Language
>PYTHON

DOWNLOAD:
magpi.cc/234A3hY

>STEP-03

Tune the code

Terraforming can take a long time – we're talking days rather than minutes. However, we can tune our code to speed things up. Explore your world and find the tallest mountain range and deepest valley. Make a note of the height (the third value displayed in the top-left corner of the screen). You can then plug these values into the code.

We have set the default terraforming height range from -3 to 35 on the y axis, but you can make this bigger (this will take longer) or shorter (this will take less time), depending on the size of the geological features in your world.

>STEP-04

Set the speed for the power of your Pi

This code should work on any type of Pi, but older, less powerful models may struggle if you terraform at full speed. If *Minecraft* can't keep up with all the changes it's being asked to make to the landscape, it may hang. Therefore, it's a good idea to pause after a certain number of blocks, to allow *Minecraft* to catch up. On a Pi 3, you can comfortably transform 500+ blocks before the need for a pause, but for a Model B you may need to deal with 50 blocks at a time. You'll probably want to run a few experiments to find the optimum configuration for your setup.

>STEP-05

Register for an Initial State account

Initial State allows you to upload live data and plot interesting charts and graphs. A free account lets you stream 25,000 events a month and examine the last 24 hours' worth of data in any bucket. Once you have registered for an account, click on the 'create HTTPS bucket' button (the plus symbol) and give it a suitable name. Then check 'Configure Endpoint Keys' and copy the Bucket Key and Access Key into your version of the code.

Above You can create some very strange-looking worlds, like this one where everything on the surface is made of glass

>STEP-06

Start terraforming!

If you're using a free account, edit the code and set the **Free_account** variable to True. This will throttle the amount of data sent to Initial State and allow you to record the whole process without exceeding the data cap.

Start your code running and check the console for any errors. You can fly to the corner of your world and should soon be able to see the changes taking place. Once the first data reaches Initial State, you can create a cool dashboard: use the Tiles interface and play around with the different types available.

Below Initial State has a range of graph types to make your terraforming data look informative and cool at the same time

PAUL TANNER, ROSS ATKIN & TINA ASPIALA

Paul, Ross, and Tina are IoT developers, engineers, and designers. They enjoy testing the limits of the latest tech with Bare Conductive.
magpi.cc/2dDDdXP

Four of the Pi Cap's 12 capacitive electrodes are being used as proximity sensors

The game runs on a Pi Zero with a Pi Cap add-on and Ethernet cable

Players move their hands between the Electric Paint sensors to control the paddles on screen

You'll Need

- Pi Cap and Electric Paint **magpi.cc/2e8kmGK**
- micro-USB cable
- Pi power supply
- Crocodile clips
- Acrylic
- Glue
- Cardboard
- Aluminium foil

CAPONG
A PONG GAME

Make a physical version of Pong! Use capacitive sensing and Electric Paint to make a fun and addictive two-player game to play with your friends

Capong breaks Pong out of the screen and into your hands. Map the Pong paddles to the position of your hands, using a Pi Cap and Raspberry Pi, to create a simple and addictive game. Capong is a physical reinterpretation of the original video game. Instead of mouse or arrow keys, it uses sensors arranged on a laser-cut stand so that each player moves her hand between a pair of sensors. The game is based on SimplePong, available on **openprocessing.org** and released under Creative Commons. It was modified to use input from the Pi Cap sensors and converted to two-player operation.

CROCODILE CLIPS

Make sure you leave enough length for each crocodile clip to reach its designated electrode.

First steps

First, we need to set up the Pi Cap. Run through the 'Setting up your Pi Cap on the Raspberry Pi Zero' tutorial found at **magpi.cc/2emLB1K**, and don't miss any steps. (You need to know the IP of the Pi to log into it.) Run through the Pi Cap intro to see the code examples, particularly the one that streams the sensor data via OSC to your laptop terminal window. Notice the DIFF data; that's what we'll be using.

Once you've done this, download and install Processing if that's not already on your laptop. Unzip and install the code **mpr121_pong** in Processing's sketch folder, usually **/Documents/Processing**. Open the sketch in Processing and start it running. To run the OSC demo standalone, go to your **PiCapExamples** folder on the Pi and **cd** to **cpp/picap-datastream-osc-cpp**. Use **`./run`** to see the Pi Cap datastream. Find out the IP of your laptop then use **`./run -host [IP address of laptop]`** to stream it to Processing. Pong should now be running. Click the laptop mouse to start a game; it finishes when a player misses the ball. Click the laptop mouse to start another game.

If you want to build the acrylic stand, as seen in our version, you can download the Illustrator files online (here: **magpi.cc/2enaB7V** and here: **magpi.cc/2enc6Tn**) and follow the tutorial instructions, courtesy of **@rossatkin**. You will need a laser cutter to cut these out, or you can make it out of foam board.

To assemble your stand, glue one of the I-shaped pieces of acrylic to the white rectangular piece with no holes in it.

To build the acrylic stand seen here, you can download the Illustrator files online

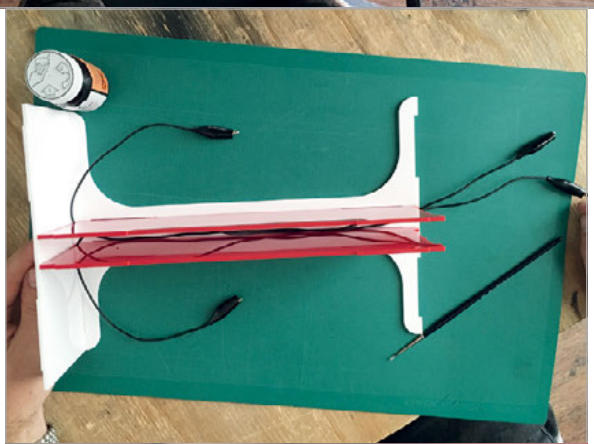
Before you glue in the two red rectangular pieces, make sure to insert two of the crocodile clips inside the structure; there should be a slot for the cables to exit when you attach the sides. This will ensure the wires are concealed within your stand, but still accessible.

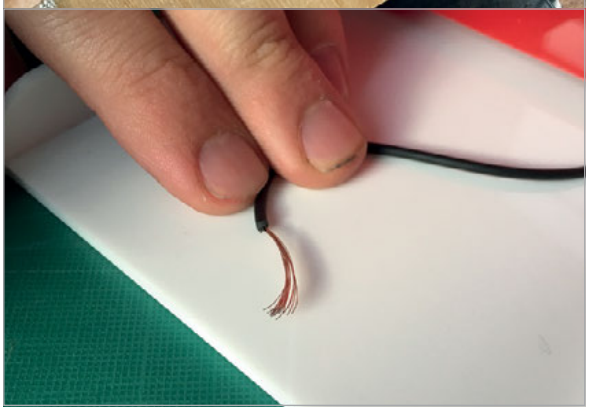
Glue the two red rectangular pieces to the white structure. Using a small paintbrush, dab all the joints of the stand with the acrylic glue; this adhesive will melt the plastic pieces together. Careful with the red acrylic: it may melt and release some colour. You should still have one acrylic piece remaining: the white rectangle with two holes. Don't glue this piece on yet.

Stand your Capong upright, so the white piece with no holes is touching the tabletop. Make sure you have enough wire so that your crocodile clips protrude at the top; you need at least 7.5cm of croc clip visible, as shown in the picture.

Now, leave some length of the wire out the bottom of the stand, and cut and strip the wire. You should have about 1.5cm of copper wire protruding. You're going to use this to attach the copper wire to the sensors.

Cut out two cardboard squares and two rectangles. These will go on your stand so you can measure the





Left Be patient with the build and you'll get there

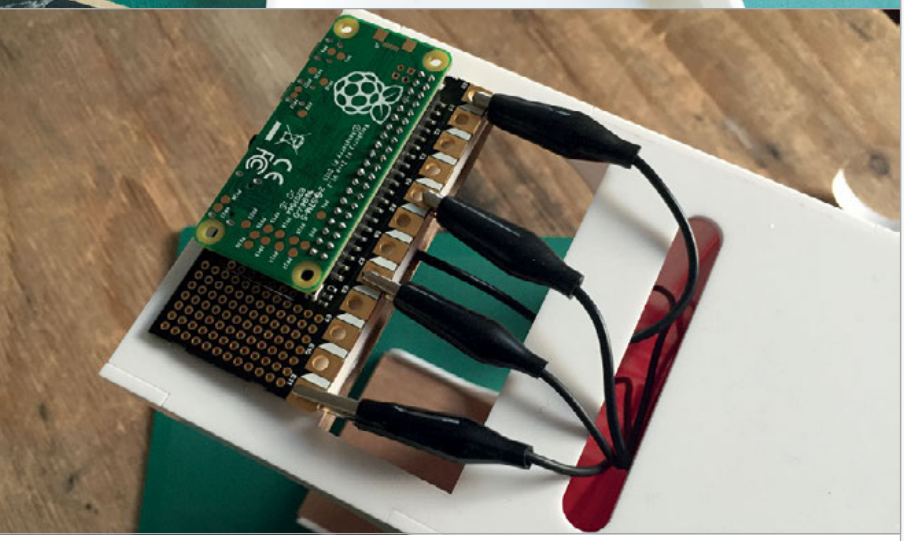
Bottom left You may need to shorten your crocodile clips during construction

RED ACRYLIC

Careful when gluing the red acrylic – it may melt and release some colour.

STRAIN RELIEF

Secure each cable on its sensor so they don't move (you can use double-sided tape).



Above top Paint Electric Paint to make your sensor, and to cold-solder to your crocodile clip

Above Attach each of the four crocodile clips from each sensor to the correct electrode

size of the interior surfaces. If you're using Electric Paint, you can paint directly onto these squares.

Once dry, apply some double-sided tape; you're going to glue the sensors face down against the acrylic. But first, you must cold-solder the paint!

Using your Electric Paint tube, squeeze out a generous amount of paint onto the exposed copper. You should make sure the wire is held in place so that it doesn't move around; you can use double-sided tape. When you're finished, you should have four sensors – two square, two rectangular – connected to each of the sides of the Capong stand.

If you don't have Electric Paint, you can make your sensors using aluminium foil. Just follow the same steps as above, but sandwich the exposed wire between the aluminium foil and the cardboard.

You can now firmly attach your sensors to the stand and get out your Pi Cap and Pi Zero. Take the crocodile clips that are protruding from the top of the Capong stand and attach them to your Pi Cap's electrodes. Make sure you're connecting to the correct electrodes, the ones you've programmed for functionality.

Now you can connect your Pi Zero, upload the code, and get playing!

TROUBLE-SHOOTING

Make sure you've mapped the correct crocodile clip to each sensor and electrode combination.

mpr121_pong.pde

```
01. import oscP5.*;
02. import netP5.*;
03.
04. final int numElectrodes  = 12;
05.
06. boolean serialSelected = false;
07. boolean oscSelected    = false;
08. boolean firstRead      = true;
09. boolean soloMode       = false;
10.
11. boolean gameStart = false; //true;
12.
13. float x = 150;
14. float y = 150;
15. float speedX = random(3, 5);
16. float speedY = random(3, 5);
17. int leftColor = 128;
18. int rightColor = 128;
19. int diam;
20. int rectSize = 150;
21. float diamHit;
22. int vpos1 = 0;
23. int vpos2 = 0;
24.
25.
26. OscP5 oscP5;
27.
28. int[] diffs;
29.
30. int globalGraphPtr  = 0;
31. int electrodeNumber = 0;
32. int serialNumber    = 4;
33. int lastMillis      = 0;
34.
35. void setup() {
36.   size(500, 500);
37.   noStroke();
38.   smooth();
39.
40.   // setup OSC receiver on port 3000
41.   oscP5 = new OscP5(this, 3000);
42.
43.   // other setup
44.   diffs = new int[numElectrodes];
45. }
46.
47. void oscEvent(OscMessage oscMessage) {
48.   println("oscevent");
49.
50.   if (firstRead && oscMessage.
51.         checkAddrPattern("/diff")) {
        firstRead = false;
52.   }
53.   else {
```

Language
>PROCESSING

DOWNLOAD:
magpi.cc/2dDE4bo

```
    if (oscMessage.checkAddrPattern("/diff")) {
      // simulate mouse in original game
      updateArrayOSC(diffs, oscMessage.
                arguments());
      vpos1=diffs[10]-diffs[1]+100; // guesswork
      vpos1*=2.5;
      if (vpos1 > 450) vpos1=450;    // limits
      if (vpos1 < 80) vpos1=80;
      vpos2=diffs[0]-diffs[11]+100; // guesswork
      vpos2*=2.0;
      if (vpos2 > 450) vpos2=450;    // limits
      if (vpos2 < 80) vpos2=80;
      print(vpos1, vpos2);
      println();
    }
  }
}

void draw() {
  background(255);

  fill(200,0,0);
  diam = 20;
  ellipse(x, y, diam, diam);

  fill(200,0,0);
  rect(width-30, vpos1-rectSize/2, 10, rectSize);
  rect(30, vpos2-rectSize/2, 10, rectSize);

    if (gameStart) {

    x = x + speedX;
    y = y + speedY;

    // if ball hits movable bar, invert X
       direction and apply effects
    if ( x > width-30 && x < width-20 && y >
              vpos1-rectSize/2 && y <
              vpos1+rectSize/2 ) {
      speedX = speedX * -1;
      x = x + speedX;
      rightColor = 0;
      fill(200,0,0);
      diamHit = random(75,150);
      ellipse(x,y,diamHit,diamHit);
      rectSize = rectSize-10;
      rectSize = constrain(rectSize, 10,150);
    }

    // similar if ball hits the other movable bar
       (2 players)
    else if ( x > 20 && x < 30 && y > vpos2-
       rectSize/2 && y < vpos2+rectSize/2 ) {
      speedX = speedX * -1;
      x = x + speedX;
      rightColor = 0;
      fill(200,0,0);
      diamHit = random(75,150);
      ellipse(x,y,diamHit
                ,diamHit);
      rectSize = rectSize-10;
      rectSize =
constrain(rectSize, 10,150);
    }

    // if ball hits wall, change direction of X
              (single-player only)
    else if (false && x < 25) {
      speedX = speedX * -1.1;
      x = x + speedX;
      leftColor = 0;
    }

    else {
      leftColor = 128;
      rightColor = 128;
    }

    // resets things if ball hits either wall - you lose
    if (x > width || x < 0) {
      gameStart = false;
      //delay(5000);  // auto-restart
      //gameStart = true;
      x = 150;
      y = 150;
      speedX = random(3, 5);
      speedY = random(3, 5);
      rectSize = 150;
    }

    // if ball hits up or down, change direction of Y
    if ( y > height || y < 0 ) {
      speedY = speedY * -1;
      y = y + speedY;
    }
  }
}

void mousePressed() {
  gameStart = !gameStart;
}

void updateArrayOSC(int[] array, Object[] data) {
  if (array == null || data == null) {
    return;
  }

  for (int i = 0; i < min(array.length,
              data.length); i++) {
    array[i] = (int)data[i];
  }
}
```

CLIVE WEBSTER

A professional tinkerer since 2004, Clive just keeps seeing more uses for Pis around the house – how many Pis are too many?
twitter.com/clivewriting

You'll Need

- USB TV tuner **linuxtv.org/wiki**
- USB remote control **magpi.cc/2dDLriE**
- USB hard disk **magpi.cc/2gTSLsM**
- Powered USB hub **magpi.cc/2gUfp4n**
- OSMC **osmc.tv**
- MPEG2 codec **magpi.cc/2dDLT0c**

PiVR

A PI-POWERED PVR

Make your own PVR to record and watch live TV, as well as stream video and audio

Good Telly Season always seems to be just around the corner, so what better time to make a PVR that does everything? From recording TV shows to streaming favourite films from your NAS to playing tunes from your smartphone, a PiVR can do everything… apart, that is, from virtual reality: the name's a little misleading like that.

Better still, your PiVR only uses USB devices and requires a few fairly basic terminal commands to get running. There's nothing here to scare a Pi novice, and plenty to please the experts.

The PiVR is based around a Pi 3, both for its processing power and the current it can feed to USB devices. The TV tuner used in this project can be found here: **magpi.cc/2dDMTS7**. However, with this TV tuner taking up to 500mA, an old laptop hard disk sucking 1A and the Pi itself consuming up to 800mA, we worried about brownouts if we powered everything from the Pi. We therefore based our project around a 3A powered USB hub: plenty of headroom.

OSMC and the software

We chose OSMC (**osmc.tv**) as the basis for the PiTV, as it incorporates the popular Kodi front-end (albeit in skinned form) with a full Debian back-end. Essentially, it's easy to use and easy to modify and add to. Better yet, it's a cinch to install: download the installer and it'll set up your SD card automatically.

You can even enter your WiFi details during the install so the Pi is ready to go once it's booted.

Update OSMC and its apps in the usual way, over SSH. The login credentials are **osmc/osmc**: change these as soon as possible via the **passwd** command.

```
sudo apt-get update
sudo apt-get dist-upgrade
```

If your hard disk has also lived a former life, wipe and format it with **fdisk** and then reboot.

```
fdisk /dev/sda
```

Next, mount the hard disk and link it to a folder called **recordings**:

```
sudo mkfs.ext4 /dev/sda -L storage
sudo mkdir /mnt/recordings
sudo mount /dev/sda /mnt/recordings
sudo chown osmc:osmc /mnt/recordings
sudo chmod 777 /mnt/recordings
```

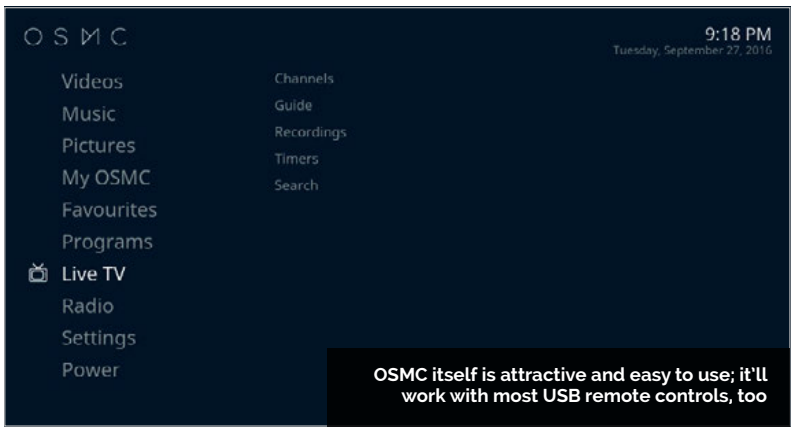
Now you can find your Pi's serial number, which is needed to buy an MPEG-2 codec licence for your Pi (this licence will be tethered to your Pi). Without an MPEG-2 codec licence, the Pi will have to decode the TV signal in software rather than its bespoke hardware, which can lead to overheating and general performance issues. Follow the instructions at **magpi.cc/2dDLToc** and pay the £2.40. Your licence code should arrive within 72 hours; add the entire line of the received MPEG-2 licence to your config file:

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

Getting your TV tuner up and running might require some detective work, but your first step is to find your tuner on **magpi.cc/2dDNsLv**. If you're lucky, your tuner won't require specific firmware, otherwise you'll have to download it, typically from GitHub: see **magpi.cc/2dDMI9h**. However, our PCTV TripleStick (292e) required even newer firmware, which we found at **magpi.cc/2dDMiQm**:

```
wget http://palosaari.fi/linux/v4l-dvb/
firmware/Si2168/Si2168-B40/4.0.25/dvb-demod-
        si2168-b40-01.fw
sudo mv dvb-demod-si2168-b40-01.fw /lib/firmware
sudo reboot
dmesg
```

The return from **dmesg** shouldn't list any errors regarding firmware not downloading. If so, you can proceed to setting up Tvheadend, the server and client combination that handles all the live TV duties for the PiVR.



OSMC itself is attractive and easy to use; it'll work with most USB remote controls, too

Tvheadend on OSMC

The easiest way to install the latest, correct version of Tvheadend is via OSMC's front-end. First, you must navigate its awkward setup procedure; use only a keyboard, as it's too easy to get confused as to which level of menu or option you're selecting with a mouse. Once negotiated, head to **My OSMC** and track across to **Remotes** to set up your remote. Then track back to the **OSMC Store** and install Tvheadend (it's free, don't worry), not forgetting to select **Apply** to actually start the installation.

Once installed, you'll need to switch to another computer to set up Tvheadend; point a browser to **http://<pi-ip-address>:9981** and log in with **osmc/osmc**. Now follow this setup order to avoid getting into awful tangles with Tvheadend. First, head to **Configuration > DVB Inputs > Networks**. Click **Add** and then choose DVB-T as the Type; on the next screen give your 'network' a relevant name and select the correct Predefined Mux for your TV area (see **digitaluk.co.uk** if you're not sure). If you're on the edge of two masts' coverage, add a network for both.

Now go to the **TV adapters** tab and select your TV tuner; on the right-hand pane, tick the Enabled box and add any and all networks via the Networks field. Head to the **Muxes** tab and you should see many entries with a scan status of PEND; after a while, these will switch to Active, and hopefully then to OK. The last job in the Tvheadend webpage is to head to the **Recording** tab and change the recording location to your hard disk, in our case **mnt/recordings**. Click **Save** which is toward the top-left for this section.

Now you can switch back to OSMC on your Pi. Head to **Settings > Add-ons > My add ons > PVR clients > Tvheadend HTSP Client**. Press **Enter** on your remote, then select Configure. Enter the Tvheadend login details (**osmc/osmc**) and then select Enable. Finally, head to **Settings > TV > General** and tick Enabled; you should see OSMC update a few things. Head back to the main OSMC menu and you'll now see an option for Live TV. Open that, and you'll see an EPG and other such options. To watch a show, select it from the EPG and then press **Back** on your remote until you go 'beyond' the main menu into full-screen live TV.

NEATER BUILD

Using a 314GB WD PiDrive (which only consumes 0.55A) might allow you to drop the USB hub from the build.

AUDIO ADD-ON

OSMC supports AirPlay (from iTunes servers) while the Pi 3 has Bluetooth – upgrade the audio output and you've got a streaming jukebox.

SCREEN OUT

Add a VFD display for extra slickness, perhaps using a £10 ZeroSeg with the two buttons left off (**magpi.cc/2dOtGBg**).

K.G. ORPHANIDES

K.G. has been tinkering with computers since the mid-'80s and writing about them since the late '90s. They've mostly survived the experience.
twitter.com/KGOrphanides

You'll Need

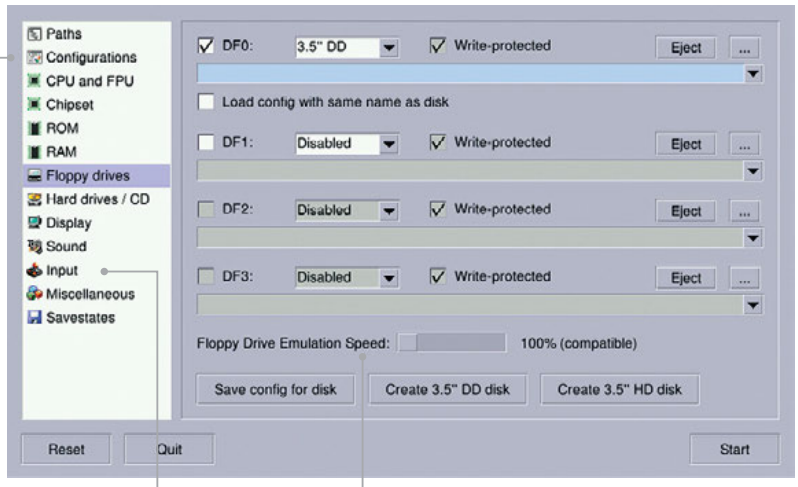
- microSD card
- USB stick
- Wired Xbox 360 controller
- Amiga Kickstart ROMs amigaforever.com
- Amibian bit.ly/Amibian

TURN YOUR PI INTO AN AMIGA

Recapture the glory days of 16-bit computing by turning your Pi into a faithful Amiga emulator

The Commodore Amiga's top-notch sound and graphics made it one of the most desirable home computers of the '80s and early '90s, at a time when your average IBM PC was still plodding along with EGA graphics and an internal beeper. Amiga games from the era have aged incredibly well, and look and play brilliantly on everything from a portable display to a widescreen TV. We'll take you through turning your Raspberry Pi 3 into a perfect modern-day Amiga emulator. You'll need a Windows, Mac OS X, or Linux desktop operating system to copy the Amibian Linux distribution to your SD card and unpack the Kickstart ROMs required to make it work smoothly.

Start by downloading the Amibian distro. Format a microSD card, decompress the Amibian RAR file, and use Win32DiskImager or Linux's **dd** command to copy the IMG file to the card. A 4GB card should be plenty, as Amibian only occupies around 300MB.



You can load and create emulated hardware configurations for specific Amiga computers

Game controllers, mouse, and keyboard configurations can be selected and tweaked in Input

Once you've set up your emulated hardware and firmware config, just mount a floppy disk image and click Start

Slot the microSD card into your Pi and power up. It'll boot directly into the UAE4ARM emulator, but there's some extra configuration to do before we start playing. Quit UAE4ARM to get to the command line and run:

```
raspi-config
```

Select Expand Filesystem, which will give you access to the entirety of the SD card's capacity for storage, then Exit and select Yes to reboot.

If your Pi won't output sound via HDMI properly, enter this at the command line:

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

Make sure the following lines are present and aren't commented out with a preceding hash (#):

```
hdmi_drive=1
hdmi_force_hotplug=1
hdmi_force_edid_audio=1
```

Save and return to raspi-config:

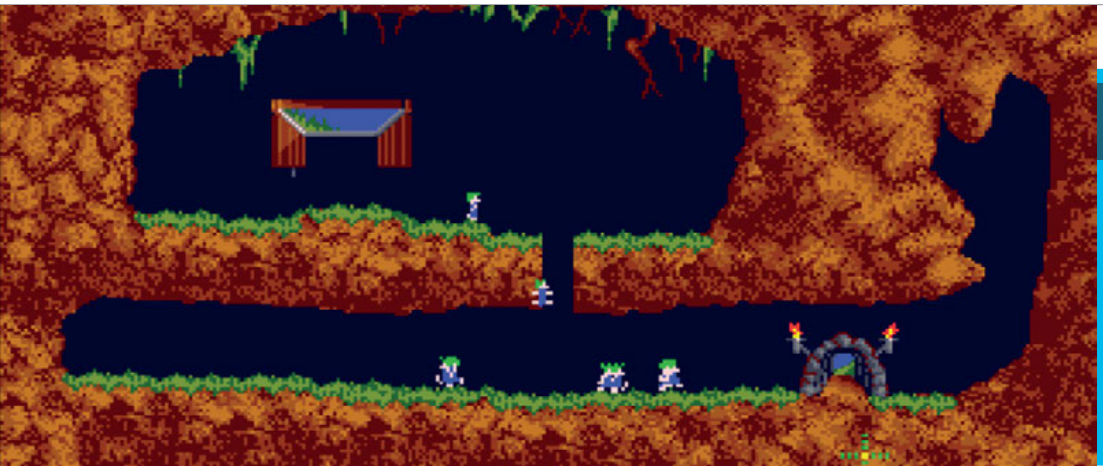
```
raspi-config
```

Select **Advanced Options > Audio > Force HDMI** and then reboot.

Kickstart me up

To run Amiga programs, you'll need a Kickstart ROM – firmware from the original computers. UAE4ARM comes with the open-source AROS ROM, which can run only some Amiga programs, so we recommend using genuine Amiga Kickstarts for reliability.

The Amiga's Kickstart ROMs and Workbench GUI are still being maintained, thanks to Italian firm Cloanto. Amiga Forever Plus Edition, priced at €29.95, gets you a complete, legal set of Kickstarts for every Amiga computer and console. Cloanto is still working



PUBLISHER-APPROVED GAME DOWNLOADS

Ami Sector One
magpi.cc/2dDLEIL

Dream17
dream17.info

Games Coffer
gamescoffer.co.uk

Gremlin Graphics World
magpi.cc/2dDKZ3S

on a Raspberry Pi edition, so you'll currently have to install Amiga Forever on a Windows PC or Wine and copy the files onto a USB stick.

There are other ways of obtaining Kickstart ROMs, but most are legal grey areas. You can extract them from an Amiga using a tool such as TransRom or find them on abandonware sites, but we strongly recommend supporting Cloanto's continued development of Amiga Forever.

Classic Amiga software is even easier to find. You'll get 50 games with Amiga Forever Plus, while some major publishers have made the Amiga versions of their games available for free (see above).

Many more games are only available online as legally dubious abandonware. They're easily found using any search engine, but inform yourself of the legal status of such software in your region before you download.

One true path

As Amibian doesn't include a window manager, it's easiest to download and copy everything to a USB stick using your operating system of choice. Helpfully, UAE4ARM can read Amiga ADF floppy images even if they're in a ZIP file.

We recommend copying everything to your microSD card. Fire up your Pi, exit UAE4ARM, and run:

```
mc
```

Copy your game files from **/media/usb** to **/root/amiga/floppys**, and your Kickstart ROMs, including a Cloanto rom.key file if you have one, to **/root/amiga/kickstarts**. Quit Midnight Commander and reboot:

```
reboot
```

In the latest version 1.313 of Amibian, two different versions of UAE4ARM are supplied. If you plan on using two Xbox 360 controllers, button mapping on controller two works best using the 'old' version, although the 'new' edition generally provides more options. To switch between the two, at the command line type either **rpiold** or **rpinew**. The following configuration instructions work with both versions.

Configure UAE4ARM

First, go to the Paths tab and click Rescan ROMs so UAE4ARM knows where to find everything.

The Configurations tab lets you select from several preset hardware emulations, with the default being an A1200 – just select and Load your chosen computer. You can tweak your virtual hardware in the CPU and FPU, Chipset, and RAM tabs.

Your configuration selection doesn't always set the relevant Kickstart ROM for you, so check the ROM tab, where you can choose Kickstarts from a pull-down menu. Note that many games require a specific ROM or hardware configuration to work properly, depending on which system they were originally released for.

To run most software, you'll need the Floppy drives tab. Just press the … icon next to drive DF0's Eject button, select the desired disk image, and click Start.

By default only drive DF0 is active, and most titles expect this configuration. To swap disks when prompted, press **F12**, eject the disk image in DF0, select the disk image you're asked for, and click Resume.

F12 will always pause and return you to UAE4ARM's main interface, so you can create a save state – a stored image of your progress in a game – or give up and load something new. The Reset, Quit, and Start/Resume buttons are always visible in UAE4ARM's GUI. Reset completely reboots your emulation and Resume returns you to your current game.

UAE4ARM automatically detects Xbox controllers. You can use two controllers for multiplayer gaming – if the second is unresponsive, you may need to press **F11** to disable your mouse and switch control to the pad. If you're running the 'new' version of the emulator, first select your controllers from the pull-down Port0 and Port1 menus in the Input settings.

Now you've got your Amiga emulator up and running, there's plenty of scope to build on the project, from setting up virtual hard disks to install Workbench and other software onto, to creating floppy images from your own original Amiga disks and using the Pi's GPIO to connect a classic '80s joystick.

The Saturn V rocket is one of the most powerful vehicles of all time, and necessary for us to get to the moon

APOLLO PI

Emulate the Apollo mission computers on the Raspberry Pi and make your own small step to the moon

The Apollo space programme is the stuff of legend. Six manned moon landings resulting in twelve people walking around on a completely different celestial body to the Earth. The only twelve people in history to have done so – and they did it nearly 50 years ago.

The legacy of these historic missions is felt and revered to this day, and it's safe to say the world would be a different place if they hadn't happened. Will we ever return? Hopefully one day, and it might be our stepping stone to the rest of the solar system and eventually distant stars.

For now, though, let's honour the Apollo legacy by investigating the computers on board these incredible spacecraft, and how we can make our own Apollo computer on a Raspberry Pi.

Buzz Aldrin stands on the moon in one of the most famous photos of all time from a legendary mission

The Apollo code, printed out and stacked, next to Margaret Hamilton who was the director of software programming for the Apollo missions

GET THE SOURCE CODE ON GITHUB!
magpi.cc/2abpPcb

MOON CODE

The computers on the Apollo spacecraft needed programming as well

You've probably heard someone say before how modern pocket calculators are more powerful than the Apollo spacecraft; they're mostly correct, although it's tricky to properly compare. The Apollo Guidance Computer (AGC) was created for the Apollo program, which featured a 1.024MHz clock speed, 16-bit word length, and 2,048 words of RAM. Not bits or bytes, words.

As 'primitive' as it may seem 50 years later, it was powerful enough for the task. Of course, the computer needed more than power and that's where the code comes in. Programmed during the 1960s, the project was fundamental in creating what we know of today as software engineering.

The code is written in assembly, which is a much 'lower-level' programming language to something like Python, but was much more common in the Sixties, when programming computers was a fairly new concept.

It was a marvel for its time. Now, as with all of NASA's work, it's open to the public. While you may have been able to access it in some way for a few years now, the Apollo 11 version of the code is now up on GitHub. Modern-day code collaboration software being used to house and distribute the code that got humans to the moon – an incredible time for computer science.

All images courtesy of NASA

MOONPI

From space to your Raspberry Pi

The code for the moon landings is an amazing piece of history, but what does that have to do with the Raspberry Pi? A couple of years ago, the AGC code was ported to various versions of the operating system Linux to create a virtual AGC that people could use and learn from. It's not a simulator in any sense of the word, but it can give you an idea of how working the computers in space might have gone.

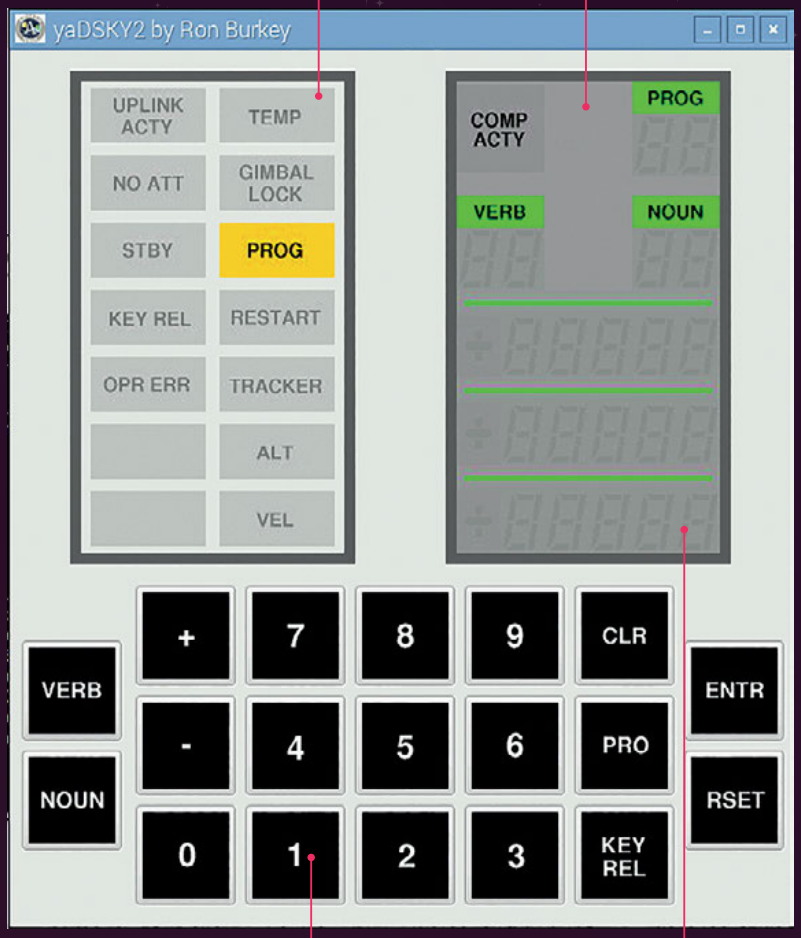
Raspbian, the main Raspberry Pi operating system, is a version of the OS called Debian that has been tweaked to work on the Pi. Debian itself is a popular distribution of Linux and the virtual AGC worked on normal Debian, so getting it working on Raspberry Pi is quite simple! Over the next few pages we'll teach you how to get it working on the computer powering the Astro Pis currently up in space, in homage to the Pi's moon-landing ancestor.

The AGC computer and its control pad. Computers were very different in the Sixties, relying mostly on magnetic ribbon for storage

THIS VIRTUAL AGC WAS CREATED BY RON BURKEY:
magpi.cc/2b2oasx

SET UP YOUR APOLLO PI

Make your Raspberry Pi ready to pilot a spacecraft straight to the moon



These fixed information indicators light up as different operations where performed by the computer

Programs were run on this display, with the VERB and NOUN and PROG boxes showing what was being run by the astronaut

These fixed information indicators light up as different operations where performed by the computer

The results from the programs were shown on the last three lines – luckily all the results were numbers so they could be easily read

It's actually quite simple to get the virtual AGC running on Raspberry Pi – all we need is a few libraries and the specific code. Luckily, Dave Honess of the Raspberry Pi Foundation has already built the code so we can download it and run it on the Pi without having to build it from scratch ourselves.

>STEP-01

Train up your Raspberry Pi

We'll need the latest version of Raspbian. If you've not reinstalled Raspbian in a while it may be best just to do a fresh install of Raspbian to your SD card. You can find the latest image of Raspbian here: **magpi.cc/1MYYTMo**

If you're installing fresh or not you'll have to make sure your Raspberry Pi is up-to-date. You can do this by opening the terminal and using the following:

```
sudo apt-get update
sudo apt-get upgrade
```

>STEP-02

Launch prep

For the code to work, we need some extra software on Raspbian. You can install this with the following command in the terminal:

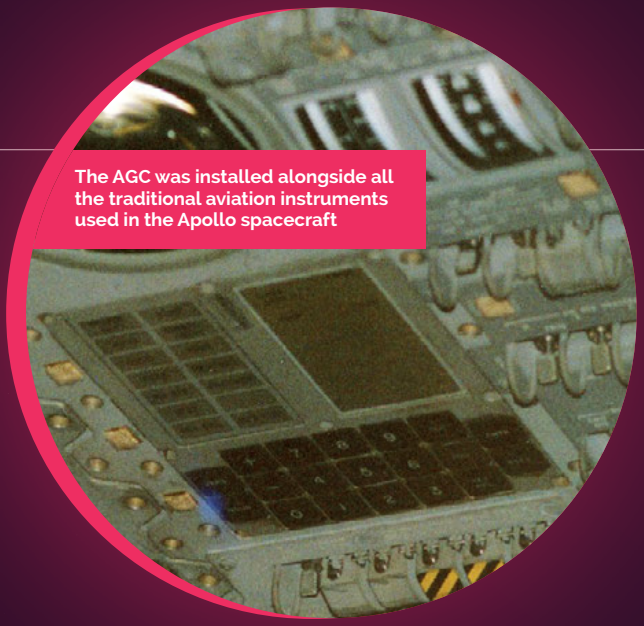
```
sudo apt-get install wx2.8-headers
libwxgtk2.8-0 libsdl1.2debian libncurses5
```

Thw wx2.8-headers and libwxgtk packages allows us to use the graphical interface that's been created for the virtual AGC, which we'll discuss over the page. The libsdl and libncurses packages lets the AGC have better access to Raspbian so it can work properly.

>STEP-03

Ignition sequence

Once everything is installed, it's time to download the code. You can either open a browser on your Raspberry



The AGC was installed alongside all the traditional aviation instruments used in the Apollo spacecraft

and go to **magpi.cc/2gUodHb** to get the zip file, or you can download it in the terminal with:

```
wget https://github.com/themagpimag/
projects-book-3/blob/master/agc.zip
```

You'll need to unzip the file once it's downloaded (unzip agc.zip if you're using the terminal). Move it to its own folder in the home directory to make sure it's all nicely contained before unzipping if you wish.

>STEP-04

Blast-off!

This part you need to do in the terminal within the desktop environment. If you're in the command line, use startx and then open a terminal window.

From there use cd to move to the IVirtualAGC folder that you unzipped (e.g. cd IVirtualAGC). After thatm cd into the bin folder within and run the Virtual AGC with:

```
./VirtualAGC
```

The option interface will start up. Select Apollo 11 Command Module, click on Full on the DSKY option in the right hand column, and finally hit Run to use the AGC.

LUNA PROGRAMMING

Operating the AGC is quite different to how we use computers today. Calculations and queries were made using a verb and a noun code – two-digit numbers that told the computer what to do. The verb was the action that the astronaut wanted the computer to do, while the noun was the data that the action needed to be done on. For example, pressing VERB and then 05 followed by NOUN and 09 and then hitting Enter will display (the action) the alarm codes (the data) if there's any problems with the AGC. In short hand this **is referred to as V05N09E.**

IMPORTANT CODES

verbs:

- 05 Display Octal Components 1, 2, 3 in R1, R2, R3.
- 06 Display Decimal (Rl or R1, R2 or R1, R2, R3)
- 25 Load Component 1, 2, 3 into R1, R2, R3.
- 27 Display Fixed Memory
- 37 Change Programme (Major Mode)
- 47 Initialise AGS (R47)
- 48 Request DAP Data Load Routine (RO3)
- 49 Request Crew Defined Maneuvre Routine (R62)
- 50 Please Perform
- 54 Mark X or Y reticle
- 55 Increment AGC Time (Decimal)
- 57 Permit Landing Radar Updates
- 59 Command LR to Position 2
- 60 Display Vehicle Attitude Rates (FDAI)
- 63 Sample Radar Once per Second (R04)
- 69 Cause Restart
- 71 Universal Update, Block Address (P27)
- 75 Enable U, V Jets Firing During DPS Burns
- 76 Minimum Impulse Command Mode (DAP)
- 77 Rate Command and Attitude Hold Mode (DAP)
- 82 Request Orbit Parameter Display (R30)
- 83 Request Rendezvous Parameter Display (R31)
- 97 Perform Engine Fail Procedure (R40)
- 99 Please Enable Engine Ignition

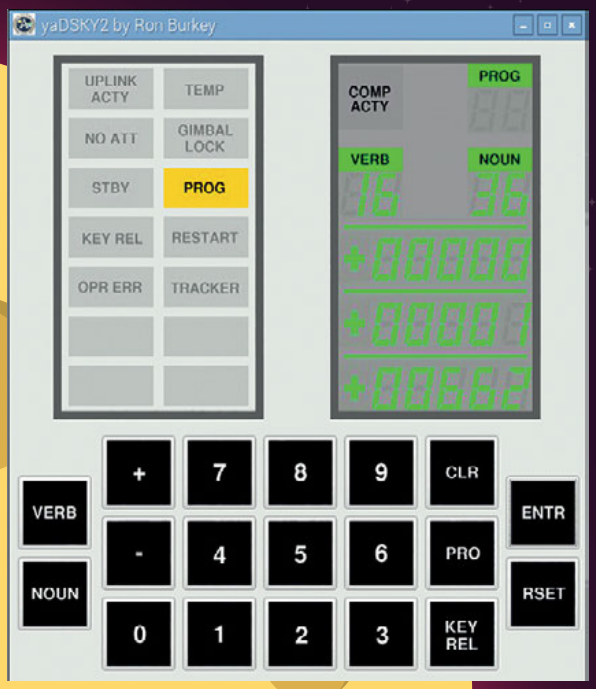
nouns:

- 11 TIG of CSI
- 13 TIG of CDH
- 16 Time of Event
- 18 Auto Maneuvre to FDAI Ball Angles
- 24 Delta Time for AGC Clock
- 32 Time from Perigee
- 33 Time of Ignition
- 34 Time of Event
- 35 Time from Event
- 36 Time of AGC Clock
- 37 Time of Ignition of TPI
- 40 (a) Time from Ignition/Cutof
  (b) VG
  (c) Delta V (Accumulated)
- 41 Target Azimuth and Target Elevation
- 42 (a) Apogee Altitude
  (b) Perigee Altitude
  (c) Delta V (Required)
- 43 (a) Latitude (+North)
  (b) Longitude (+East)
  (c) Altitude
- 44 (a) Apogee Altitude
  (b) Perigee Altitude
  (c) TFF
- 45 (a) Marks
  (b) TFI of Next/Last Burn
  (c) MGA
- 54 (a) Rang
  (b) Range Rate
  (c) Theta
- 61 (a) TGO in Braking Phase
  (b) TFI
  (c) Cross Range Distance
- 65 Sampled AGC Time
- 66 LR Slant Range and LR Position
- 68 (a) Slant Range to Landing Site
  (b) TGO in Braking Phase
  (c) LR Altitude-computed altitude
- 69 Landing Site Correction, Z, Y and X
- 76 (a) Desired Horizontal Velocity
  (b) Desired Radial Velocity
  (c) Cross-Range Distance
- 89 (a) Landmark Latitude (+N)
  (b)Longitude/2 (+E)
  (c)Altitude
- 92 (a) Desired Thrust Percentage of DPS
  (b) Altitude Rate
  (c) Computed Altitude

APOLLO 11 MADE ITS HISTORIC MOON LANDING AT 20:18:04 ON 20 JULY 1969!

MOON TIME

Check the time since launch and set yourself up an Apollo clock on your Raspberry Pi

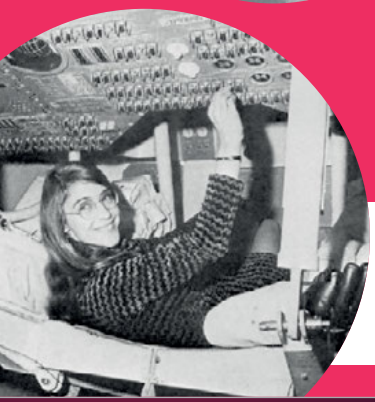


Right: The hours, minutes, and 100ths of seconds are listed on the display

One of the most basic functions of the AGC was for the computer to keep track of the time. It was also an important function, aiding with mission planning and also figuring out if it's too early in San Francisco to give someone a call.

The virtual AGC keeps track of time since launch, or in this case time since the AGC was turned on. We can check this time by keying in **V16N36E**. From the list of codes, this means we're asking for the time (V16) of the AGC clock (N36). You might see this split into LGC, which is the lunar guidance computer that would have been the computer in the Lunar Module, or CGC which is the Command Module's computer. Both use the same AGC hardware and code.

After typing in the code, you'll get three lines of numerical readouts. The top display will be hours, the second display is the minutes, and the third display is in 100ths of a second. The display is updated by the second, so you don't need to keep repeating the code to keep an eye on the time since launch.



CODING IN THE SIXTIES

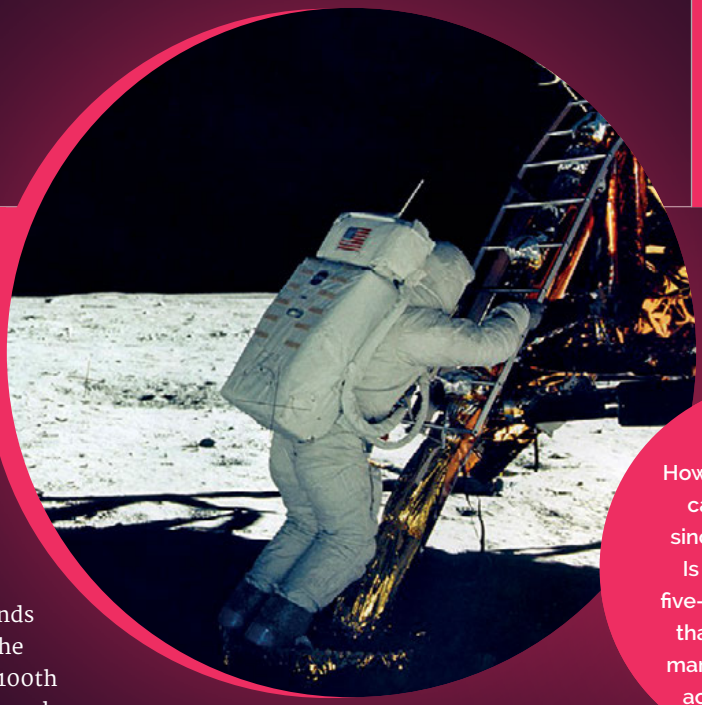
Pioneering software engineering at the dawn of computing

Top: James Lovell (of Apollo 13) can be seen here taking a star reading during Apollo 8 – next to him is the AGC's control pad
Left: Margaret Hamilton, then lead software designer on Apollo, tried to give legitimacy to software engineering in a time when it was looked down upon

The Apollo missions were a huge undertaking and the brightest minds in the United States were called upon to help on any relevant area. This means when the computer was to be designed and programmed, NASA went to MIT. In 1962, the project began and paved the way not only for modern computers but also modern software.

In the Sixties, the term software was not as widespread as it was today – it was only really known to those who made it or were very close to the projects that required it. Coming off the back of older computers, the concept of software to the hardware engineers was foreign and distrusted as it wasn't a physical thing they could see, even if it was a fundamental necessity.

The whole thing was written in assembly language, as discussed earlier, but many new programming techniques were invented to make sure the whole thing would work. Software could



A giant leap that has a legacy which can still be felt nearly 50 years later
Image courtesy of NASA

SPACE CLOCK

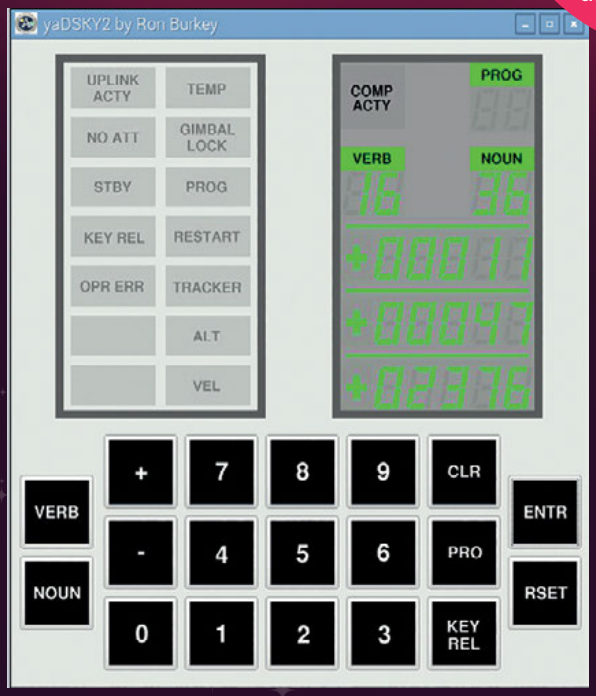
The AGC is a programmable computer, so it stands to reason we can reprogram the clock to show the current time. N36 can be modified down to the 100th of a second and we can modify it using V25; this verb allows us to load a component (change the number) in the readout of the noun, in this case the clock.

On the AGC, use **V25N36E** and the top line (R1) will clear and you can change it to be the current hour by pressing **+** on the virtual keyboard and then using the numpad to key in the time. If you make a mistake, you can press **CLR** to start again, but once you're happy you can press **ENTER** and move onto the middle line (R2) and set the minutes the same way. Remember, for the seconds it's in 100th of a second increments so 5 seconds would 500, 10 would be 1,000, etc.

Use **V16N36E** to display the current time from this edited state. This will update every second like it did before and allow you to use the AGC as a clock. With a smaller screen and some inventive setting customisation, you can make it your main clock somewhere in your house. If you want to find out the time since bootup, you can always use a different key combination of **V25N65E**, and then return to your clock with **V16N36E**. When you restart the AGC, you'll need to reset the clock, though.

Challenge!

How would you go about calculating the time since Apollo 11 landed? Is it possible with the five-digit display to count that many hours? How many years can the AGC actually count up to?

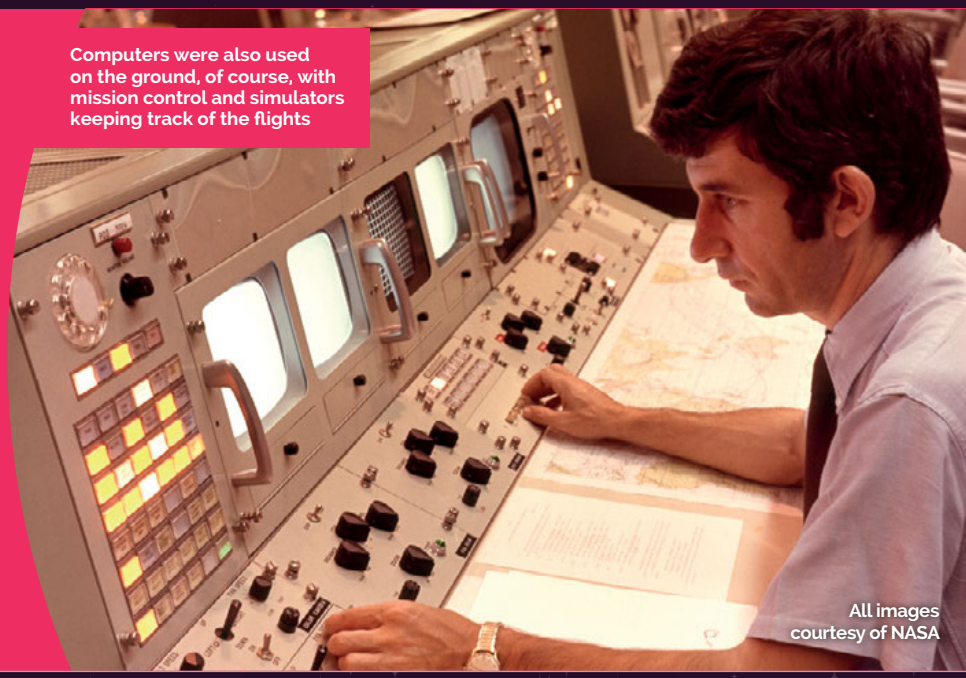


Left: The clock is set to your specific time. It's not a 24-hour clock, though, so you may need to reset it sometimes

be run asynchronously, and a priority scheduler allowed tasks for the computer to be executed when they were needed.

These innovations were key to the successful landing of Apollo 11 on the moon: due to faulty power supplied to the Lunar Module's rendezvous radar (for the return journey), the AGC was overloaded with interrupts and an abort was nearly made. Due to the scheduling system and asynchronous program running, the computer was able to cope with the extra load, resulting in Apollo 11 landing safely on the surface of the moon.

The software was continually updated and worked on throughout the rest of the Apollo missions. To work around the limitations, many little tricks were employed and in some cases the readability of the code suffered – a great reminder to always document your code!



Computers were also used on the ground, of course, with mission control and simulators keeping track of the flights

All images courtesy of NASA

The crew of Apollo 13 required a special startup process for their Command Module computer during the final stages of their fateful return home

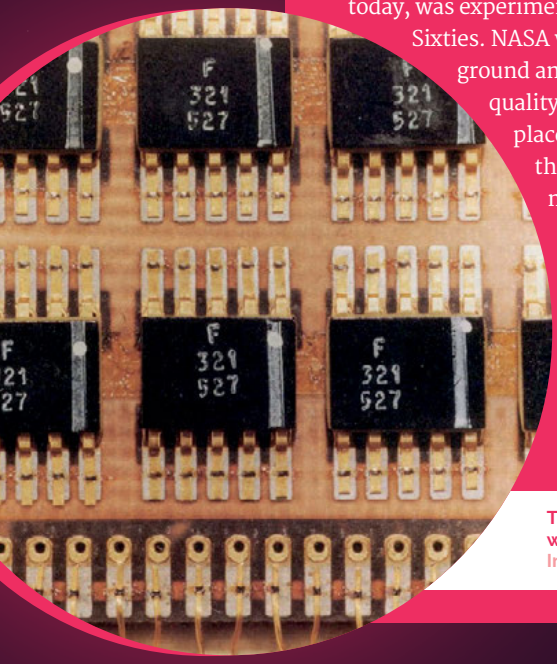
COMPUTERS IN THE SIXTIES

Revolutionising hardware to send men to the moon

One of the biggest problems with sending men to the moon was that all the equipment needed to do it didn't exist yet. There were no rockets powerful enough to get that far and computers were the size of a room. In order to fit a computer into the confines of a very tightly designed spacecraft, a new technology needed to be invented: the microchip.

Fairchild Semiconductor, which still operates today, was experimenting with the idea in the Sixties. NASA was keen to get them off the ground and make sure they were high quality and well researched, so they placed an order for a million of them, knowing they would only need a few hundred.

It worked and the integrated circuits were able to reduce the size of the computer down by a sizeable amount, allowing it to be small and light enough to fit in the craft and not hamper the flight to the moon.



These microchips used in the AGC were some of the first
Image courtesy of NASA

SPACE TESTS

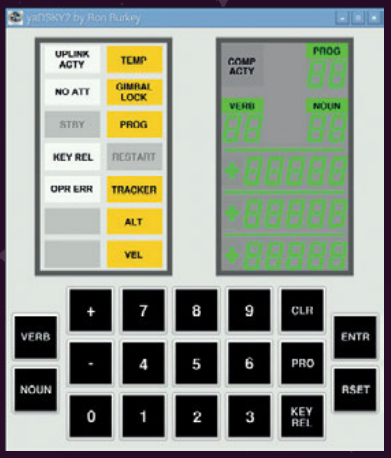
Perform the vital tests needed to start up your AGC and get to the moon

Consider the situation – you've just launched into space on a Saturn V rocket on your way to the moon. Your spacecraft has performed its docking operation between the Lunar Module and the Command Module and you're well on your way. This is when you need to check to make sure your computer is working properly – you don't want any problems when you're 100,000 miles from the nearest layby. Check the status of your on-board computer by using Apollo 13 Lunar Module and then follow these steps:

>STEP-01

Lamp check

CODE: V35E

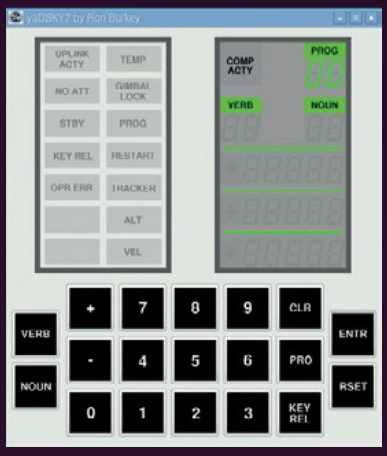


There aren't any LEDs as this is 1969, so to start the test we need to turn all the indicator lamps on to make sure they're all working. If one is burnt out and it will tell you something important, you need to know.

>STEP-02

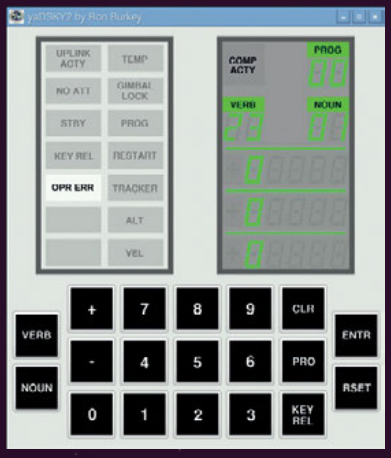
Start the main program

CODE: V37E, 00E



Program Poo, affectionately named Pooh after the bear, is one of the main programs for the AGC. PROG in the top right should show 00; this means your software is initialising and ready to work.

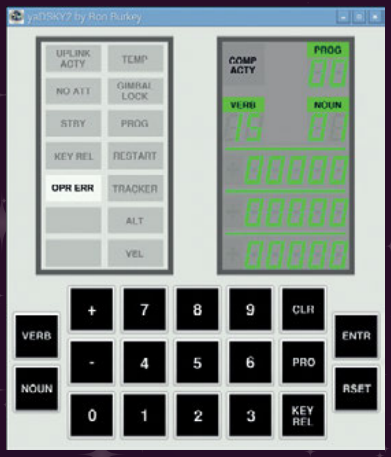
>STEP-03
Error counting



CODE: `V25E, N01E, 01365E, 0E, 0E, 0E`

Before we begin the tests, we need to set the count of total failed self-tests, total started self-tests, and successfully completed division tests to 0. We want to make sure we know exactly how many errors we get in this test alone.

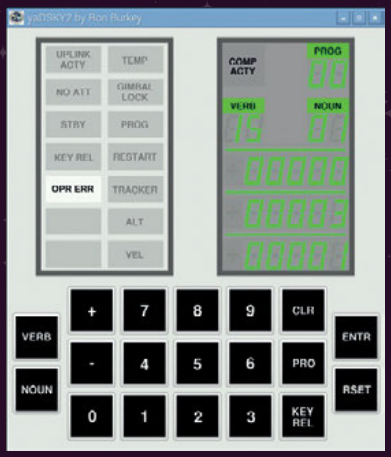
>STEP-04
Monitor the test



CODE: `V15, N01E, 01365E`

We've reset the counts; now we get ready to monitor the tests. We have to set up the three lines of output to do this first. The first row (R1) shows the number of failed tests, R2 displays how many test have actually been made, and R3 shows the number of completed division tests.

>STEP-05
Begin the tests



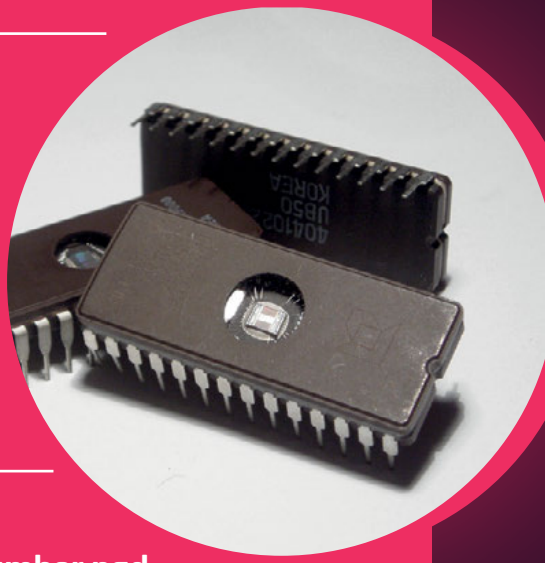
CODE: `V21N27E, 10E`

The tests will start and go through the computer. These will continue on as long as you want them to and you can stop them with **V21N27E** followed by **0E**. Hopefully your computer will be fine and you'll be on your way to the moon!

APOLLO COMPONENTS USED TODAY

Integrated circuits

These were a revolution at the time, heralding a new future for computers. These are still widely used in almost all electronics in varying ways. You can also use a few with the Raspberry Pi on a breadboard, such as an analogue-to-digital converter chip.



Number pad

The calculator-style interface on the AGC was the first of its kind to use a number pad. As well as the calculators it inspired, you can see a very similar evolution of it on the number pad found on the side of a full computer keyboard.

Image courtesy of NASA

Seven-segment display

While it didn't change the world like the integrated circuit did, the seven-segment display for showing numbers is still used today – in fact, patents for it go back to 1908. So when you plug one into your breadboard, remember this is a technology that is a century old!

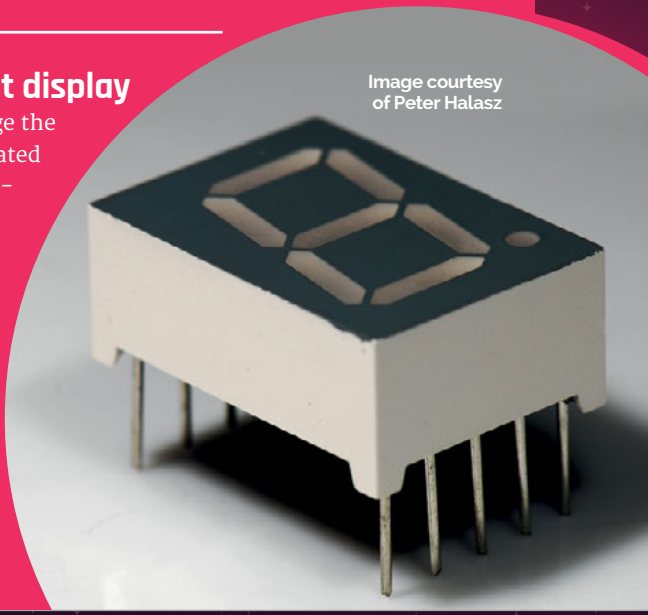
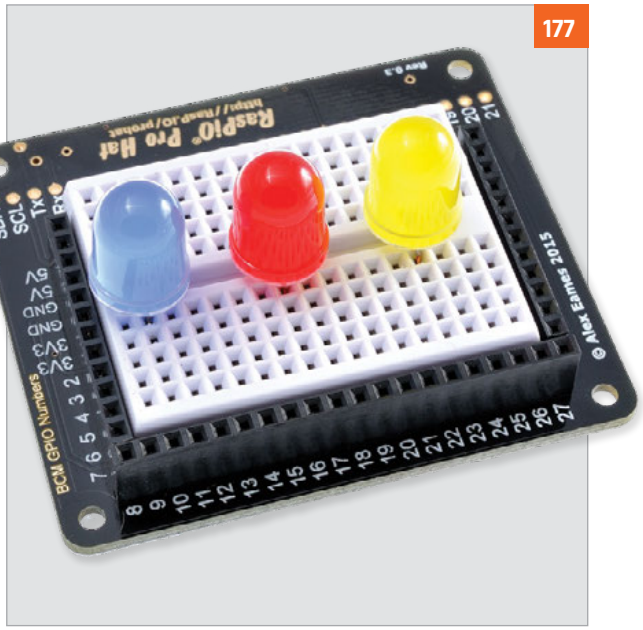


Image courtesy of Peter Halasz

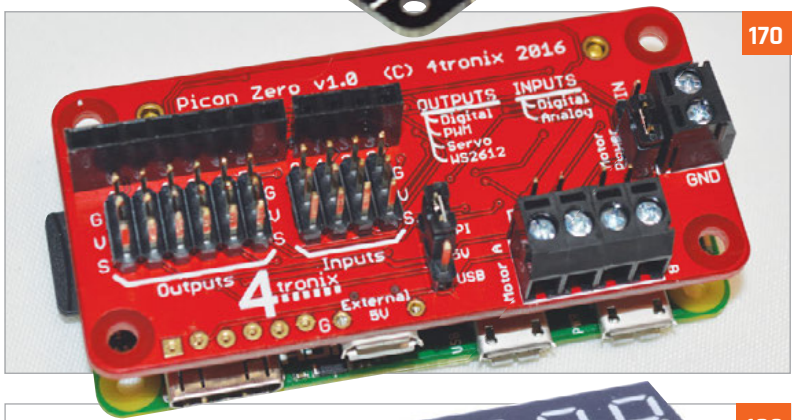
REVIEWS

Improve your Raspberry Pi projects by discovering the best accessories and add-ons with our definitive reviews

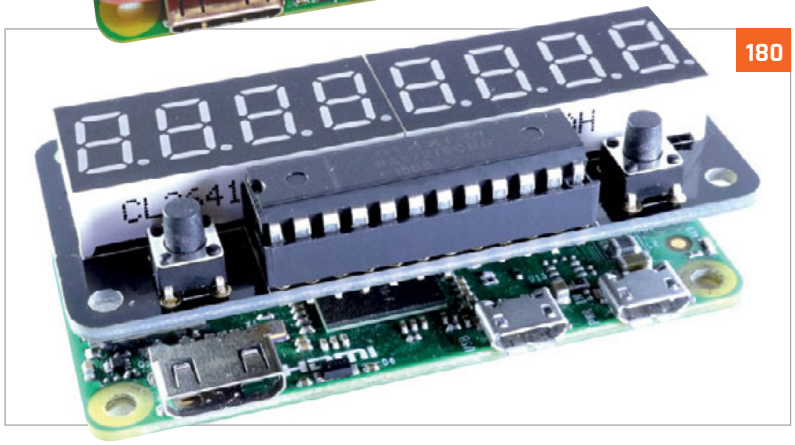
168



177

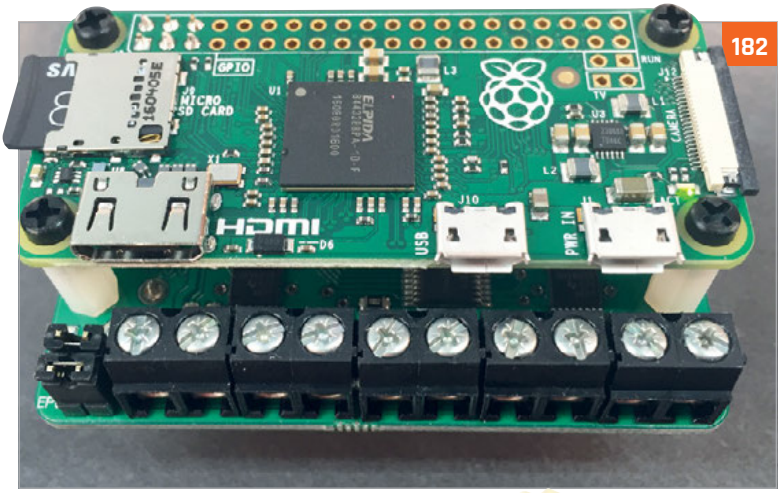


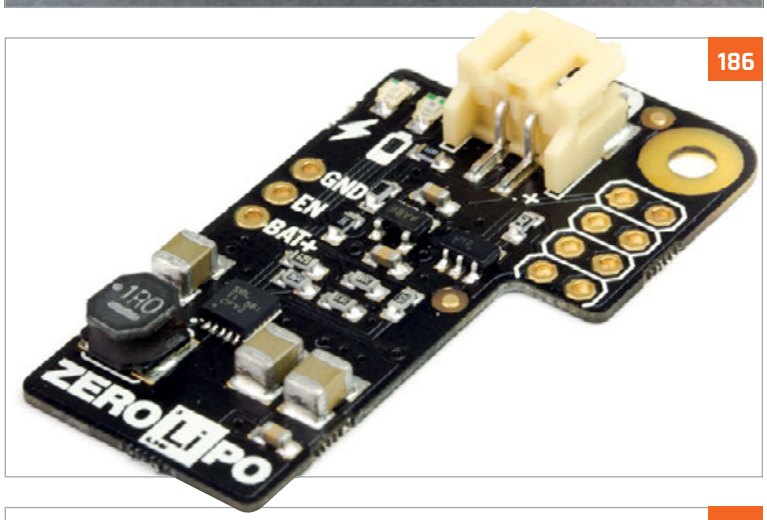
170

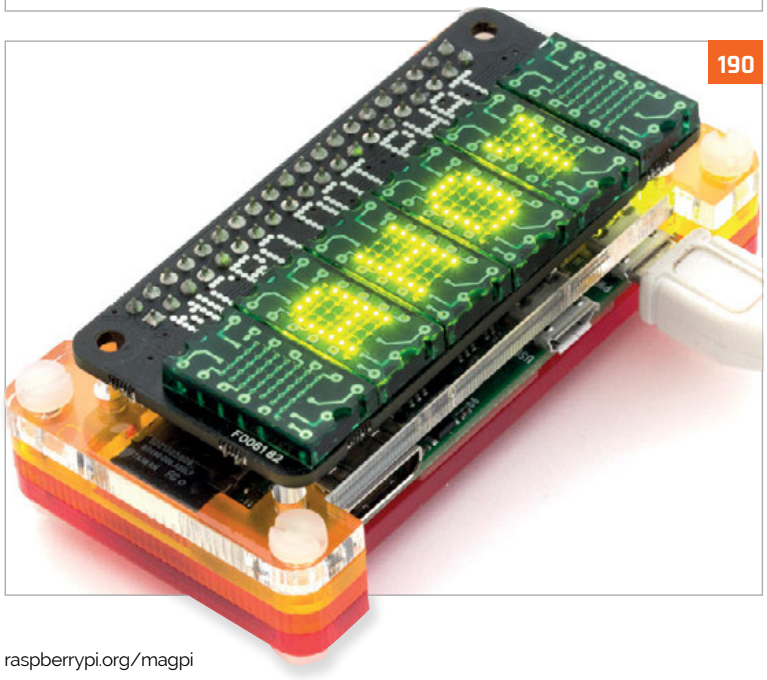


180

181







Reviews

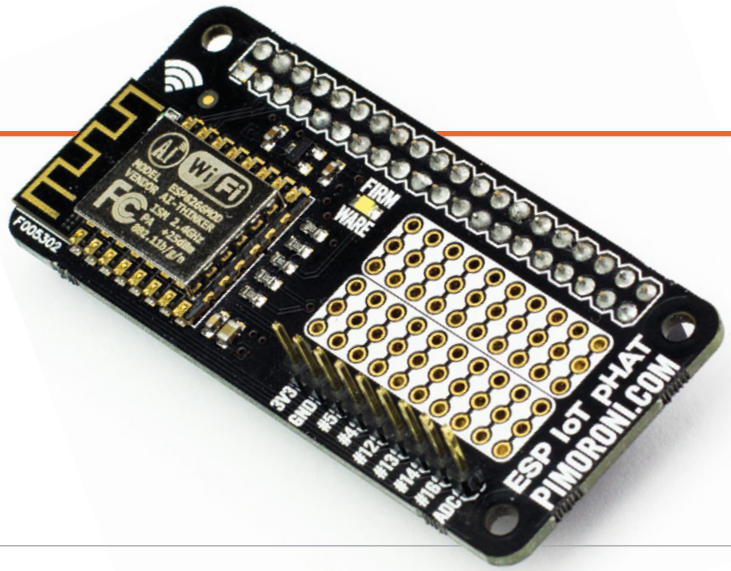
magpi.cc/21ZVQL1

£10 / $13

Maker Says

"There is a thriving community building up around the ESP8266 platform"

Pimoroni



PIMORONI ESP8266 IOT PHAT

Add a low-cost WiFi HAT to your Raspberry Pi Zero and take control of wireless IoT devices

The addition of wireless and Bluetooth to the Raspberry Pi 3 has piqued our interest in IoT devices, but what about the Pi Zero? The smallest board in the Raspberry Pi range is ideal for low-cost IoT builds, but it doesn't feature built-in wireless.

While it's tempting to think of the ESP8266 IOT pHAT as WiFi for your Raspberry Pi Zero, this is a mistake, or at the very least an understatement. Sure, it can add WiFi to your Pi Zero, but if all you want to do is head online, you're much better off using a USB WiFi dongle.

The ESP8266 IOT pHAT is a way to get started with ESP8266, an extremely low-cost WiFi chip with a full TCP/IP stack and SoC (system on chip). ESP8266 was created by Chinese-based Espressif Systems, and hackers quickly realised it would be incredibly useful (and cheap) for building IoT devices. Hackaday's Richard Baguley explains: "[It] can connect to 802.11b/g/n networks on the 2.4GHz band. It can be addressed with SPI or a serial connection, and has an AT command set that makes it behave rather like an old-style modem. It has everything you would need to connect a device to a WiFi network."

A wireless Zero

The chip itself came out in late 2014 and, during 2015, Western IoT enthusiasts translated its datasheets into English. Pimoroni has taken the ESP8266 and turned it into a HAT-like board, enabling you to combine the processing power of the Raspberry Pi with the wireless capabilities of the ESP8266.

Setup is moderately complex, and you'll need to solder the pins to the board. If you haven't done so already, you'll also need to solder the GPIO header to the Raspberry Pi Zero. Both are quite fiddly tasks, although we found soldering the pHAT easier than the Pi Zero. But it's still a job for confident solderers.

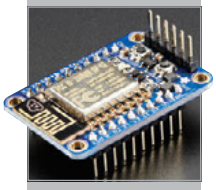
Once you've soldered the board, you'll need to install Minicom (**magpi.cc/1RcdWOp**), a text-based communications program similar to MS-DOS Telix. Raspberry Pi enthusiast Richard Hayler has created a superb guide (**magpi.cc/2ipx87g**).

The community has done a fabulous job of translating the ESP8266 dataset from Chinese, but it's still a complex and niche area. IoT enthusiasts should certainly take a closer look.

Related

ADAFRUIT HUZZAH ESP8266 BREAKOUT

Adafruit's Huzzah adds the ESP8266 chip to a breadboard-friendly breakout board. While it's not an integrated HAT-like unit, it's easier than working with just the microcontroller.



£7 / $10

magpi.cc/2ioAtn6

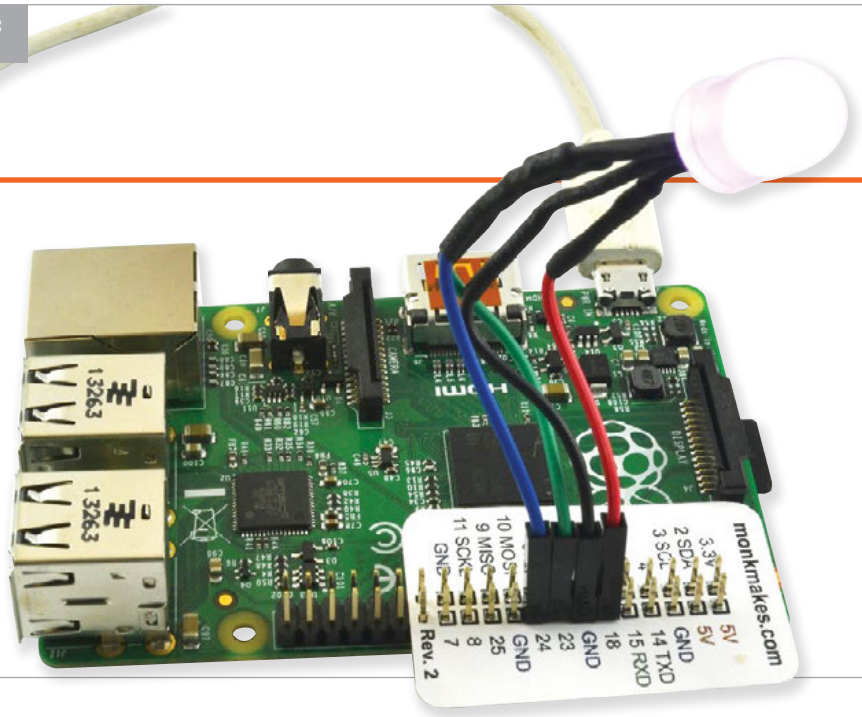
Last word

The ESP8266 is an attractive chip, and the combination of robust TCP/IP communication and Raspberry Pi power is a compelling one. One for serious IoT project makers.

★★★★☆

magpi.cc/1XuzXNf

£10 / $15



Maker Says

"A bright RGB LED with built-in resistors"

Monk Makes

RASPBERRY SQUID COMBO PACK

An RGB LED that makes learning code quick and fun, while also being useful for other projects

One of the first things people usually do with physical computing is work out how to light an LED. A pinout, a resistor, and a ground pin are all you need to get it to work on a Raspberry Pi – and the coding can be done in many ways, depending on what language you're using. The Raspberry Squid Combo Pack takes this concept and allows you to go even further with it by adding buttons and an RGB element to the LED.

The Squid makes everything very easy to get started with; it comes pre-made so that the relevant resistors are in place and it has female pins so you can attach it straight to the Raspberry Pi. There's even a handy overlay for the GPIO so you know where everything is supposed to go.

There are many coding examples available for the LED which rely on a central squid.py library – this acts a lot like GPIO in that you just need to tell it what you want the LED to do, without too much setting up of the GPIO pins in Python. There are many examples to go through and learn from; with a little bit of poking around in squid.py (and even in GPIO Zero), you can figure out how to use it a little more manually.

There are also a couple of chunky red push buttons that come with the full combo pack, so that you can add interaction to scripts. There are some example programs and a library for them as well, which just use the button on their own; however, it should be pretty easy to figure out how to get everything working together to create your own cool little displays. Even when you're done with the original learning side of the components, they'll be good for other projects and can easily be repurposed.

The Squid Combo Pack is a great little kit that, while maybe a touch steep price-wise, features some great components and tools that will be useful for years to come.

Last word

A great beginner's set for learning physical computing that can easily be used for any other projects you might have in the future.

★★★★☆

Related

NEOPIXEL RING

Available in many different sizes and prices, albeit with no buttons, but still great LEDs for projects.

£11 / $10

magpi.cc/1XuAvmk

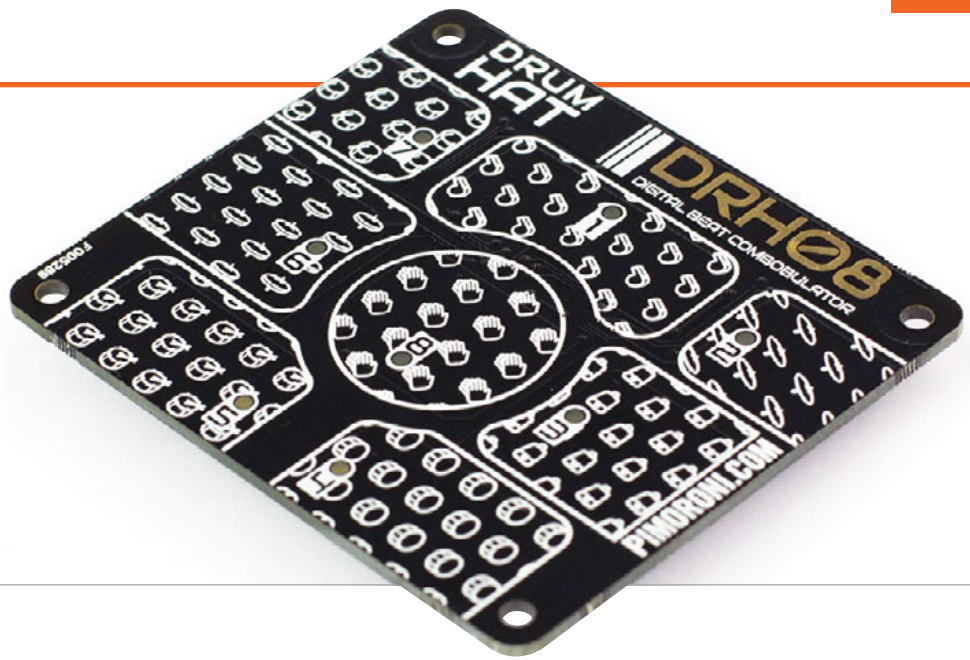
magpi.cc/1VwzNGn

£12 / $15

Maker Says

"Raspberry Pi drum-kit that lets your fingers think they're Stubblefield"

Pimoroni



DRUM HAT

Hot on the heels of Pimoroni's successful Piano HAT comes this hot drum machine. Discover the joy of code-based finger drumming

A while ago we came across the Piano HAT, a snazzy piece of hardware based on Zachary Igielman's PiPiano and turned into a HAT (hardware on top) device by Pimoroni.

The Drum HAT is its funky brother, capable of quickly transforming a Raspberry Pi into a drum machine. On top of the board sit eight capacitive sensor pads; you tap the beat out with your fingers. Each pad also sports an LED that lights up when you tap (or can be programmed separately).

Installation is easy thanks to a command listed on the Drum HAT's GitHub page (**magpi.cc/1VwzZWb**). Just enter `curl -sS get.pimoroni.com/drumhat | bash` to get started. This script installs the Python modules and downloads a bunch of sample WAV files and programs Enter `python drums.py` and you'll quickly have a drum machine ready to play. A file called **direct.py** links the samples in the **drums2** folder, and you can edit the Python code to link to any folder you want. Then it's just a case of creating some drum samples, or downloading sample files from a site like **looperman.com**.

Amen to that

We started by recreating 'Boots and Cats' (**youtu.be/NniorTLg5B8**) using our sampled voices, then grabbed a bunch of samples of the 'Amen Break' (**youtu.be/5SaFTm2bcac**) and set to turning a Raspberry Pi into a kick-ass drum-and-bass machine.

Taking apart the sample code (and reading the GitHub page) enabled us to figure out the Drum HAT code. You can set the pads to react when hit, or released, and you can get the pads to call a function that can do anything.

Building a drum machine is where it's at, though, and we had an awesome amount of fun with the Drum HAT (much more than with the seemingly more complex Piano HAT). A good project to try out.

Related

PIANO HAT

The Piano HAT is more detailed, with 16 touch-sensitive buttons (forming 13 piano keys). You can use the Piano HAT and Drum HAT together using a Black HAT Hack3r device.

£15 / $20

magpi.cc/10ALwNT

Last word

Easy to set up and fun to bash around on, the Drum HAT turns a Raspberry Pi into a home-made 808 drum machine.

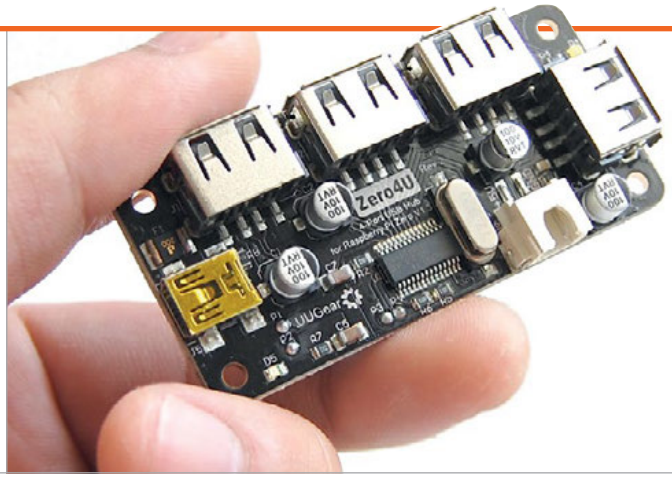
magpi.cc/2aFCXtY

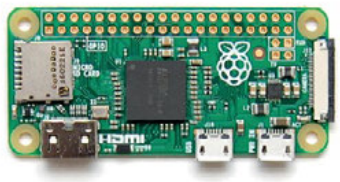
£7 / $9

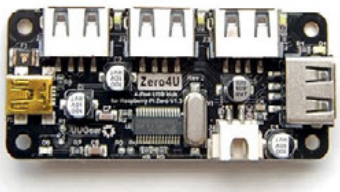
Maker Says

"Can be mounted to Raspberry Pi Zero back-to-back"

UUGear







ZERO4U

Adding four USB ports to the Pi Zero, can it replace a USB hub?

While the Raspberry Pi Zero's compact nature makes it ideal for many projects, the downside is that it only offers a single micro USB port for connecting peripherals. So, to use it with a keyboard and mouse, for instance, you'll need a USB adapter and a standard USB hub. Well, not any more...

Designed by UUGear in the Czech Republic, the Zero4U is a four-port USB hub that's mounted on the rear of the Pi Zero. Its four pogo pins connect to the tiny PP1 (+5V), PP6 (GND), PP22 (USB D+), and PP23 (USB D-) testing pads on the Pi Zero. This enables it to take its power from the latter, in which case it can output up to 2A current to all four USB ports.

Since the pogo pins are only in surface contact with the pads, they need to be kept firmly in place by securing the Zero4U to the Pi Zero using the plastic standoff screws and spacers supplied. We were slightly concerned about the pins maintaining a reliable contact, but didn't experience any problems. One detail to note is that since the testing pad positions are slightly different on the two Pi Zero models – the original v1.2 and new v1.3 with camera connector – there are two versions of the Zero4U to suit, so you need to ensure you order the correct one. Either way, the Zero4U can also be used with any other Raspberry Pi model via its mini USB input, although the power output is reduced in this case unless you power it independently via its JST XH2.54 port.

Once the Zero4U is piggybacking the Pi Zero and powered on, a blue LED lights up to show that it's operating. In addition, each port has a white status LED that's lit whenever a device is connected to it, which is a nice touch. All four ports operate at standard USB 2.0 speed (480Mbps). The only caveat is that if you insert a USB 1.1 device, they'll all be slowed down to 12Mbps, since the hub has a single transaction translator, but it's not a major problem.

Last word

The Zero4U is an ingenious solution to the lack of standard USB ports on the Pi Zero. There's no soldering required and it's relatively easy to attach to the rear of the Zero, which means the GPIO header is kept free and unobstructed. As a bonus, the device can also be used as a standard USB hub for other Raspberry Pi models.

★★★★☆

Related

THREE-PORT USB HUB WITH ETHERNET

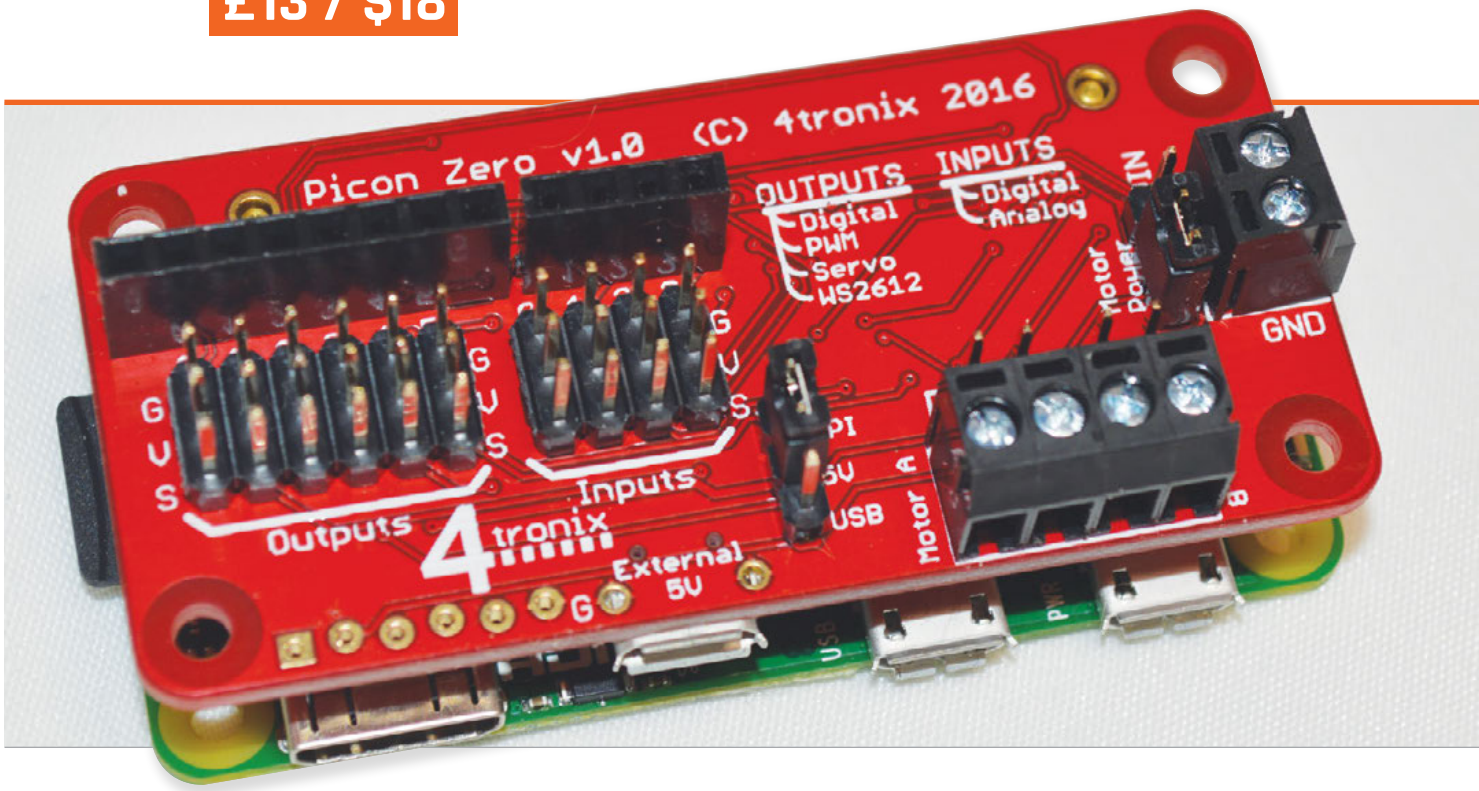
You'll need a micro USB adapter to plug it into the Pi Zero, but it has the bonus of an Ethernet port for wired connectivity.

£10 / $13

magpi.cc/2aFJkx9

magpi.cc/1p9wGaA

£13 / $18



PICON ZERO

A new robotics board from 4tronix that fits snugly on your Pi Zero for some miniature robotics

Related

RASPI ROBOT BOARD V3

Create slightly different robots with this board that allows access to all the GPIO pins for more manual construction.

£15 / $30

magpi.cc/1VBwoYG

Pi Zero robots are quickly becoming popular. This was inevitable, given the nature of the Pi community and just how useful the Zero can be in such a situation. We've already got a new kit from the folks over at PiBorg coming soon after a successful Kickstarter, and plenty of little projects and builds that use it for diminutive automatons. Usually, these small robots use pre-existing small motor-controllers. However, we now also have the Picon Zero from Pi robot veterans 4tronix.

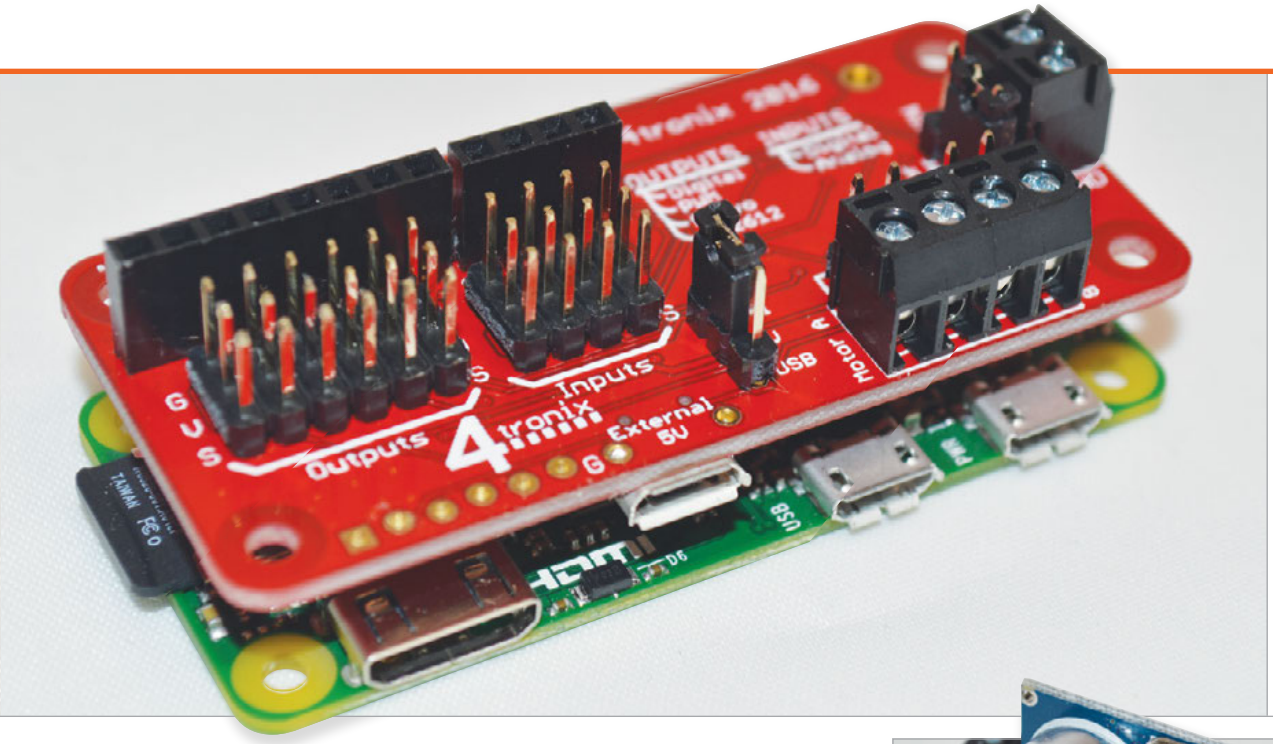
Its claim to fame is that it's a robot control board with the same size/form-factor as the Pi Zero. This means it slots right on top of a Pi Zero without poking over any of the edges. This is particularly handy, as it allows you to easily access the ports and SD card and such. It also means that you'll need to have a set of GPIO pins soldered onto the Pi Zero, which is really not a massive ordeal, although we know the idea of it is still a little scary for some.

It's certainly worth it, as the Picon Zero comes with a huge amount of benefits for robot building with the Pi. As you might expect, it can power two motors; unusually for a motorboard, it can draw power directly from the Raspberry Pi for this. This uses 5V, which the Pi can provide but which it is rarely used for due to the power draw. With two small motors, it's not so high; if it's a little too much for the Pi, however, the Picon can also take USB power. This can be a little easier to hook up and maintain than rechargeable batteries, so it's a neat addition.

The ins and outs

One of the best things about the board is the numbers of inputs and outputs it supports, especially for its size. Dotted around the PCB are inputs that accept digital and analogue signals, as well as having the ability to be configured to use popular temperature and humidity sensors. Output-wise, it's also fairly powerful, with configurations to allow digital output, the ability to use PWM, standard servo controls, and even NeoPixel support.

This is amazing for such a small device. Some full-size HATs or controllers for the Pi can handle motors and a HC-SR04 ultrasonic



Maker Says

"Intelligent robotic controller for Raspberry Pi"

4tronix

Left The board sits well above the Pi Zero, but not so high that it significantly decreases space efficiency

sensor input (which, by the way, the Picon also has a dedicated input for), but then you still might have to play around with some GPIOs to get the rest working. The Picon is incredibly comprehensive; it even opens up a small section of the GPIO port with five GPIO pins and a selection of Ground, 3V3, and 5V pins.

that are very simple to program if you have a cursory understanding of Python. There are also some worksheets for teaching the ins and outs of using the library; at the time of writing, however, they only get so far as to teach you how to read the input of a reed switch, and cover nothing to do with motors.

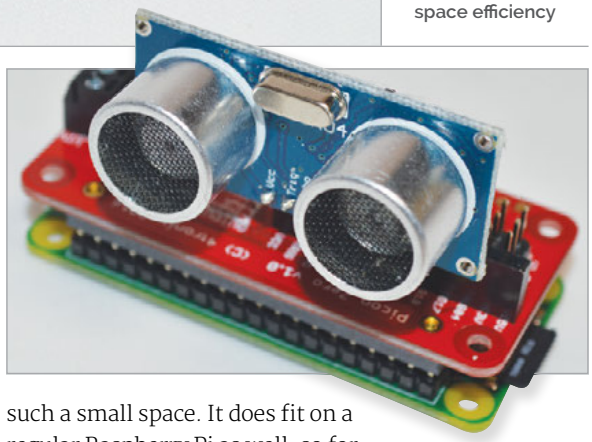
> "One of the best things about the board is the number of inputs and outputs it supports"

Code it

The whole system is powered by a Python library custom-made by 4tronix. It's incredibly simple to install, simply requiring you to download a script that will do all the work. It comes with some example Python scripts and the modules needed to operate the board with your own custom scripts. A full breakdown of the features can be found on the 4tronix blog for the Picon Zero (**magpi.cc/1SMc8N8**), but there's a huge number of functions

They're likely to be on their way very soon and you shouldn't be discouraged by that, as reading up on the functions and looking over the test and example code that comes with the Picon libraries should be enough to get you making full use of the Picon.

The only real issue we have with the Picon is that things can get a little tight on the board if you're using a significant number of inputs and outputs, although that's the price you pay for it all being in



Above An ultrasonic sensor can plug straight into the board if you want to keep things compact

such a small space. It does fit on a regular Raspberry Pi as well, so for the price it may well be something you could consider for any type of Pi robot project.

It's certainly one of the most comprehensive robot HATs we've come across for the Pi, and that low price makes it pretty special as well. Absolutely worth a look if you're wanting to make a more complex machine for your robotic needs.

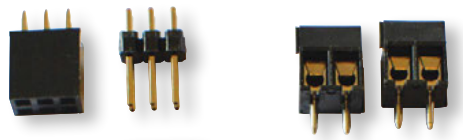
Last word

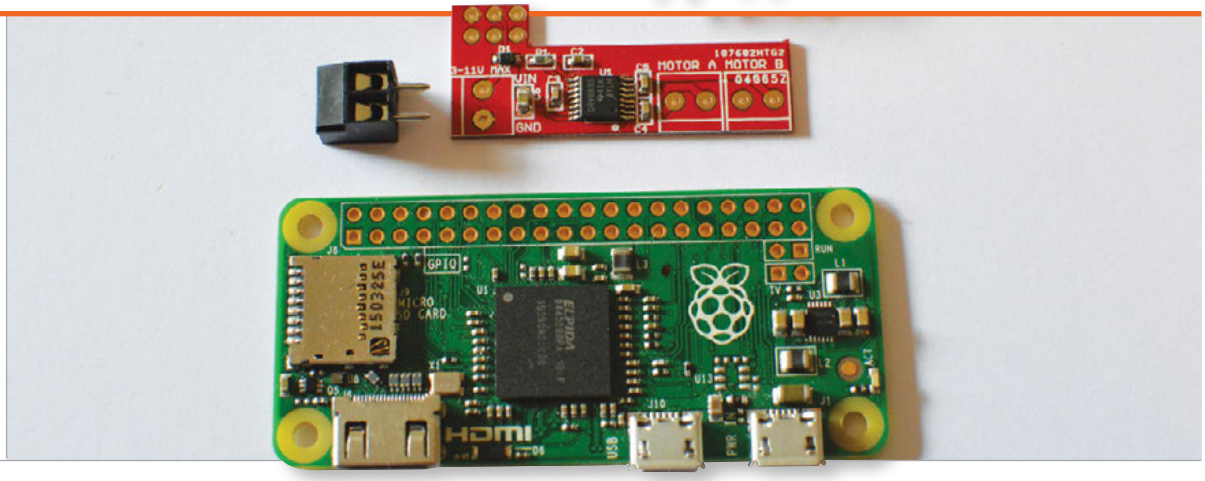
A wonderful robot controller at an amazingly low price. While designed for the Pi Zero, it can be used on any Raspberry Pi.

★★★★★

magpi.cc/1XrL6lv

£5 / $7





Maker Says

"Add motors to (almost) any Pi Zero project"

4tronix

PZM – PI ZERO MOTOR SHIM

Not quite a motor board like the MotoZero, this little shim does let you control some motors with the Pi Zero

We've seen some tiny add-ons for Raspberry Pis and even the Pi Zero in the past, but this motor shim from 4tronix has surprised us the most with its size. About as long as three micro SD cards laid end to end and not much wider (it's 38mm × 16mm), this tiny bit of PCB promises the ability to control two motors while taking up very little space in your project.

The shim comes as a kit you need to construct: the two motor terminals, a power terminal, and a couple of methods of attaching it to the GPIO ports. It does require a little bit of soldering, but even the most novice wielders of a soldering iron should have no problem with this and be done in about 20 minutes.

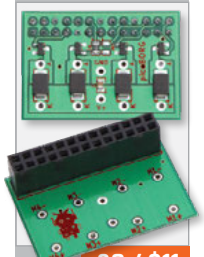
Due to how thin the board is (0.8mm, which is tiny), you can easily mount it directly to the Pi Zero if you wish. It does come with some more straightforward methods, as mentioned above: a female header so you can attach it to a Pi Zero with a set of 40 GPIO pins attached, or a male header so it can just slot through the empty pins. It only needs six, as well, and uses the last six pins so that it doesn't take up the 5V and 3V3 pins at the front of the GPIOs. It's quite clever, although it does mean you need proper external power for it, like most motor boards.

The code for the board is quite simple, much like for the other 4tronix boards. Getting the Python module from their site lets you import it and give such commands as `pzm.forward` and `pzm.spinRight`, and also control the speed. All very easy to use and implement into your project. We feel this particular shim goes beyond robot projects; after all, there are plenty of great dedicated robot-centred motor boards. Motors can do more than turn wheels, after all, although if you're making a particularly tiny robot, the shim should easily do for that as well.

Related

PICOBORG

Another simple motor controller that can be used for robots or other motor tasks. It takes up more GPIO pins, though.



£8 / $11

piborg.org/picoborg

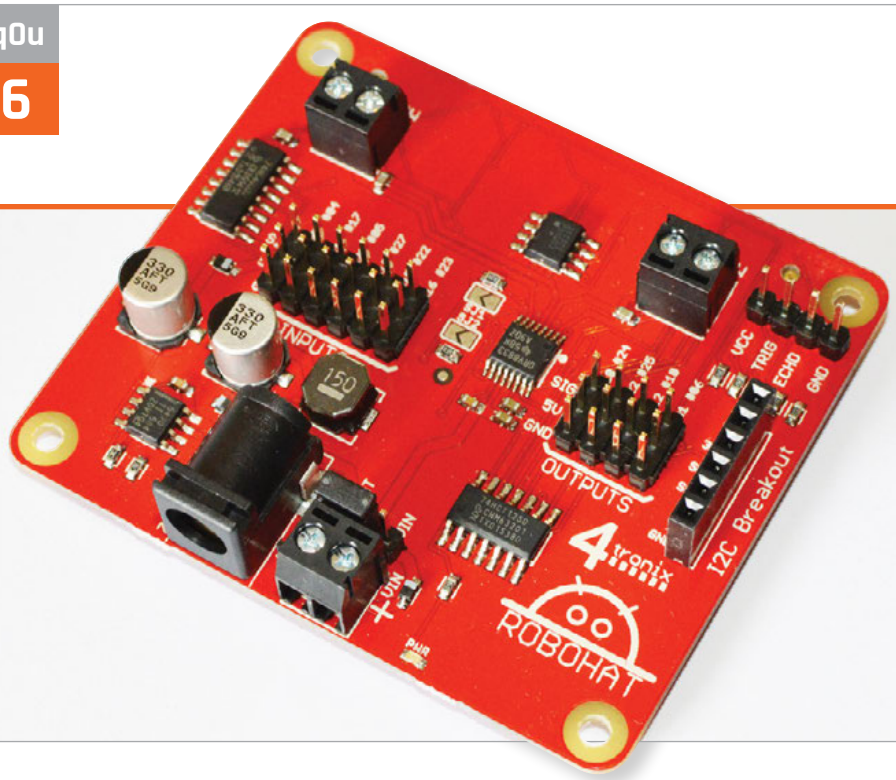
Last word

It barely costs more than the Pi Zero, yet it lets you make tiny robots or add motors to any sort of Raspberry Pi project. The minimal assembly is great as well.

★★★★★

magpi.cc/1TPyq0u

£20 / $26



Maker Says

"Complete robotics controller for Raspberry Pi"

4tronix

ROBOHAT

A full-sized Raspberry Pi robotics controller and the bigger sibling of the Picon Zero. Should this be your controller of choice?

Another month, another robot board for your Raspberry Pi; this time we're looking at the RoboHAT from 4tronix, basically a bigger version of the Picon Zero we looked at last issue. This board is designed to work with all 40-pin Raspberry Pis (so no original Model A or B, sadly) and as it's a HAT, it attaches neatly on top of a compatible Raspberry Pi without hanging off the sides.

Like the Picon Zero, the RoboHAT comes with an amazing number of inputs, outputs, and general robot connections to make use of. It all comes pre-soldered on a very sturdy piece of PCB, so you're ready to start using it right away. On the board are the requisite DC motor screw terminals; the RoboHAT comes with two, enabling it to control two motors. There's a set of pins for connecting the popular HC-SR04 ultrasonic distance sensor, and a mixture of ten input and output pins, along with a series of I²C breakout pins. It's powered by a DC jack that can also handily power the Raspberry Pi while in use, cutting down on power inputs.

It's all more spread out than on the Picon Zero, so the board becomes a bit less of a mess when everything is connected up and in use. There are also custom Python scripts and libraries to get it all working, making programming very easy. For example, importing the robohat module allows for `robohat.forward()` to move the robot forward. There are many examples included to get your head around how the library works, and they're all very easily installed using the instructions on the 4tronix blog: **magpi.cc/1TPBtWJ**.

It's a good board for novices and also some advanced users, allowing you to take the skills you may have learnt from a very simple kit and apply them to a custom project of your own design quite easily. It's easy to find out which GPIO pin controls which aspect, so you can create further custom code if that's your thing, making it fairly open and versatile. It's also a pretty good price at only £20.

Last word

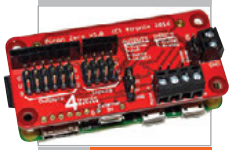
A well-designed robot HAT that works very well with the Raspberry Pi and many common robotics components. Definitely worth a look if you're planning to advance your Raspberry Pi robotics.

★★★★★

Related

PICON ZERO

A much more compact version of the RoboHAT, designed for the Pi Zero, although it can also be used on a standard Raspberry Pi.



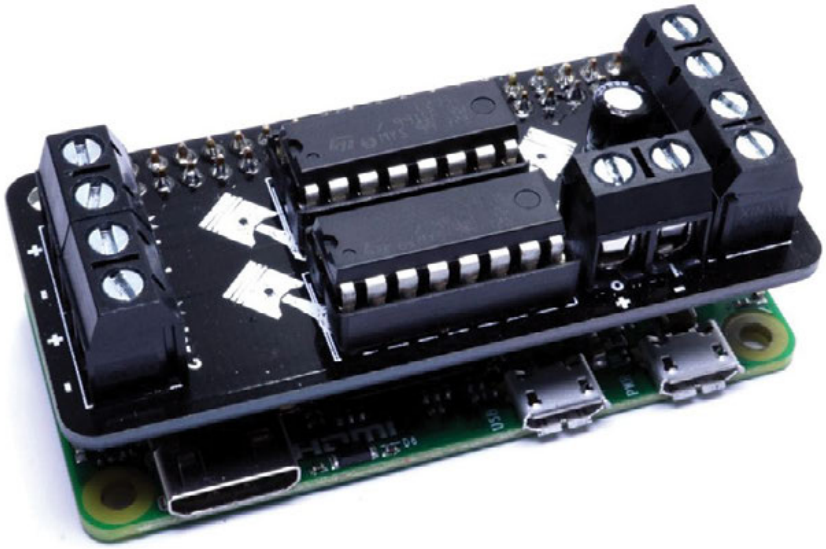
£13 / $18

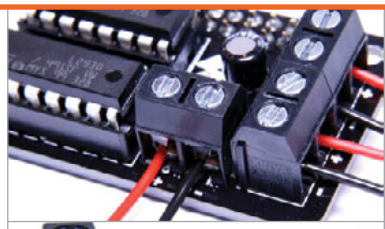
magpi.cc/1p9wGaA

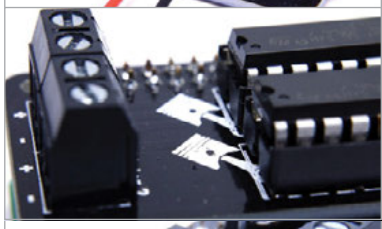
Maker Says

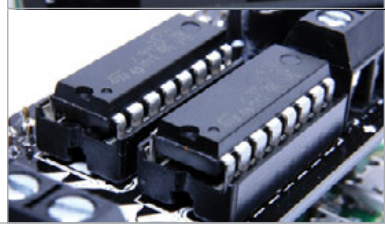
"It's simple, easy to use, and looks great"

Average Man









MOTOZERO

This Zero-size board lets you control up to four motors independently

Related

PICON ZERO

With numerous, configurable outputs and inputs, this fully featured Zero-size motor board is ideal for complex projects.

£13 / $18

magpi.cc/1p9wGaA

Richard Saville, who blogs as the Average Man (**magpi.cc/1Ypo4ez**), came up with the design for the MotoZero while watching his favourite TV show, *Sons of Anarchy*. That biker influence has led to the coolest-looking motor board we've ever seen, resembling an exposed engine with its chunky terminals and twin socket-mounted driver chips. The attention to detail even extends to some piston graphics on the board. That's right, by the way: you get not one but two motor driver chips, enabling the MotoZero to control four motors independently. While the L293D chips used here have been around quite a long time, they do the job well enough, as they are fairly low-powered motors.

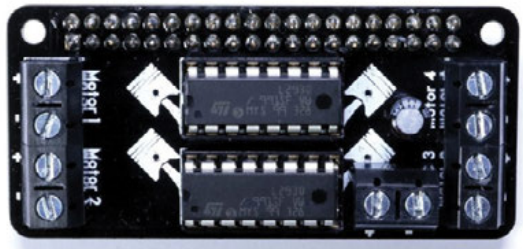
Design considerations

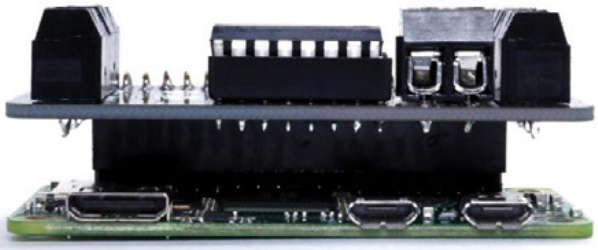
It's obvious that a lot of thought has gone into the MotoZero's design. The terminal blocks are high-quality and chunky, making it easy to connect your motor wires; their screw-heads are a decent size, too. These blocks should certainly stand up to prolonged use. The main terminal blocks are grouped in pairs, after sliding the ridge edges together, to avoid them being easily twisted out of place. Their positioning on either side of the Zero-sized board is ideal for powering the wheels on each corner of a robot. A fifth terminal block is supplied for the wires from your chosen power supply; alternatively, you could solder them directly to the board holes. One caveat to mention here is that, due to its low cost, the MotoZero board lacks any protection from reverse polarity, so you need to make sure that your battery pack is connected the right way round!

Since the MotoZero is supplied in kit form, you'll need to assemble it, which involves a fair amount of soldering. While some people might be put off by this, Richard reckons it's all part of the fun of playing around with electronics. Depending on your soldering skills, it should take 30-60 minutes to put together; helpfully, the comprehensive PDF manual contains a step-by-step assembly guide illustrated with photos, while the board itself is clearly marked with component positions and labels. It's recommended to solder

magpi.cc/1XRfqGQ

£10 / $13





MOTOR CONTROL PINS

The MotoZero allocates three GPIO pins to each of the four motor channels, as follows...

MOTOR 1:	MOTOR 2:	MOTOR 3:	MOTOR 4:
Enable pin – 5	Enable pin – 17	Enable pin – 12	Enable pin – 25
Positive – 24	Positive – 6	Positive – 23	Positive – 13
Negative – 27	Negative – 22	Negative – 16	Negative – 18

on the 40-pin GPIO header first, followed by the two chip sockets, then the four motor connection terminals, capacitor, and optional fifth terminal block. Just slot the two L293D chips into the sockets and you're good to go.

Twin drivers

Once the MotoZero is assembled, it's ready to control connected motors once slotted onto a Raspberry Pi. Incidentally, while the MotoZero's form factor is a perfect match for the Pi Zero, and Zero-sized robots, it can be used with any 40-pin Pi model.

As mentioned, the L293D driver chips are old technology, but still a good choice here since they are low-cost – thereby helping to keep the overall price down – and able to handle a wide range of voltages: from 4.5V to 36V. Handily, they also have built-in overheating protection and feature integrated flyback diodes to prevent damage from sudden voltage spikes from the motors. And even if they do get damaged, the socket mounting on the MotoZero makes them a cinch to replace.

One slight downside to using L293D chips is that they only supply 600mA continuous current per channel (i.e. motor). This means that you will need to use motors with a stall current not much higher than that: the manual states that you can get away with around 700mA, so long as you're careful to avoid stalling the motor and also keep an eye on chip temperature.

When it comes to controlling them, each motor channel is assigned three of the Pi's GPIO pins (which have been chosen carefully to avoid using any with special functions such as I²C, SPI, and UART). While the first two pins are turned HIGH/LOW or LOW/HIGH to make the motor turn forwards or backwards, as per usual, the third pin acts as a master on/off switch. While you might well wonder what the point of this 'enable' pin could be, one benefit is that it can be used with pulse-width modulation (PWM) to control the speed independently of direction.

Although the Python coding process will be similar to that of most other motor boards, one major plus point is that the MotoZero is simple enough to be able make use of the GPIO Zero library, which should make it even easier for robotics novices to control motors with very few lines of code. Do bear in mind, however, that if you do want to attach any robot sensors – such as a line follower or ultrasonic distance sensor – you're going to need to either wire them directly to pins by using a stacking GPIO header (instead of the one supplied), or you'll have to stack a HAT underneath the MotoZero.

Last word

While it lacks some of the advanced functionality of more expensive motor boards, including servo control and GPIO inputs/through-holes for sensors, the MotoZero offers great value for money, looks very cool, and has the unusual ability to control four motors independently.

★★★★☆

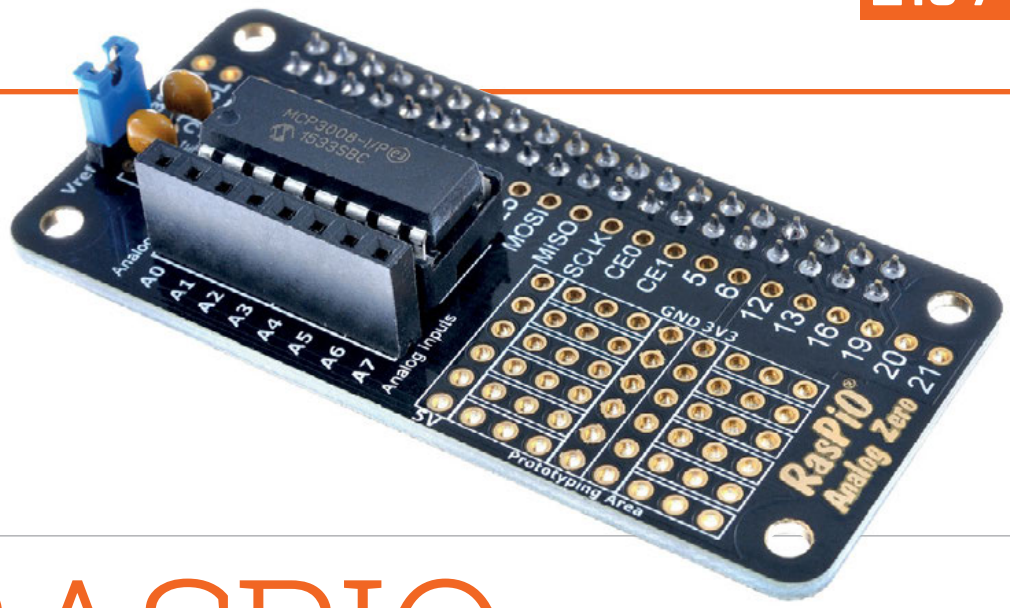
rasp.io/analogzero

£10 / $13

Maker Says

"Read up to eight analogue inputs at once"

RasPiO



RASPIO ANALOG ZERO

It makes the reading of analogue sensors as easy as Pi

While its mini form factor makes the Analog Zero a perfect partner for the Pi Zero, it's a great way to add easy-to-use analogue inputs to any Raspberry Pi model. Supplied as a kit, it's based around the MCP3008 analogue-to-digital converter (ADC) chip, but avoids all the intricate wiring usually required when using an ADC. The great thing about using this particular chip is that it's already supported by the GPIO Zero Python library with its own class, so it's a doddle to start writing programs to read and compare up to eight analogue inputs at once. Just use jumper wires to hook up your analogue sensors – temperature probes, light-dependent resistors, humidity sensors, gas detectors, potentiometers etc. – to any of the eight inputs in a female header, then write a few lines of code to get instant readings. Voltages up to 3.3V can be read directly; if the input is higher, you'll need to use a voltage divider made from resistors. Potential projects include a digital thermometer, voltmeter, and weather station, and kits for all of these were offered as part of the Analog Zero's successful seven-day Kickstarter campaign.

If you require greater accuracy than the MCP3008's 10 bits, you always have the option of swapping it out for your own 12-bit MCP3208 ADC chip for extra precision, since it fits the same socket and is also supported by GPIO Zero. Even so, the MCP3008's 1,024 steps should be enough for most projects.

Although the Analog Zero makes things easier once assembled, note that you do have to do a bit of soldering beforehand, but everything's well marked out on the board. As well as the chip socket, you'll need to solder on the small female header for the analogue inputs, a couple of capacitors, a jumper switch, plus a 40-way female header to connect to the Pi's GPIO port. Handily, the board features through-holes for 25 GPIO pins, along with a mini 54-point prototyping area. There's also the option to create a sleeker version by soldering the chip directly to the board and using surface-mount capacitors on the rear.

Last word

We'd have appreciated a pre-assembled option, but once you've soldered the kit components onto the board, the Analog Zero really does make it much easier to use multiple analogue inputs for projects, particularly when using GPIO Zero.

★★★★☆

Related

AD/DA EXPANSION BOARD

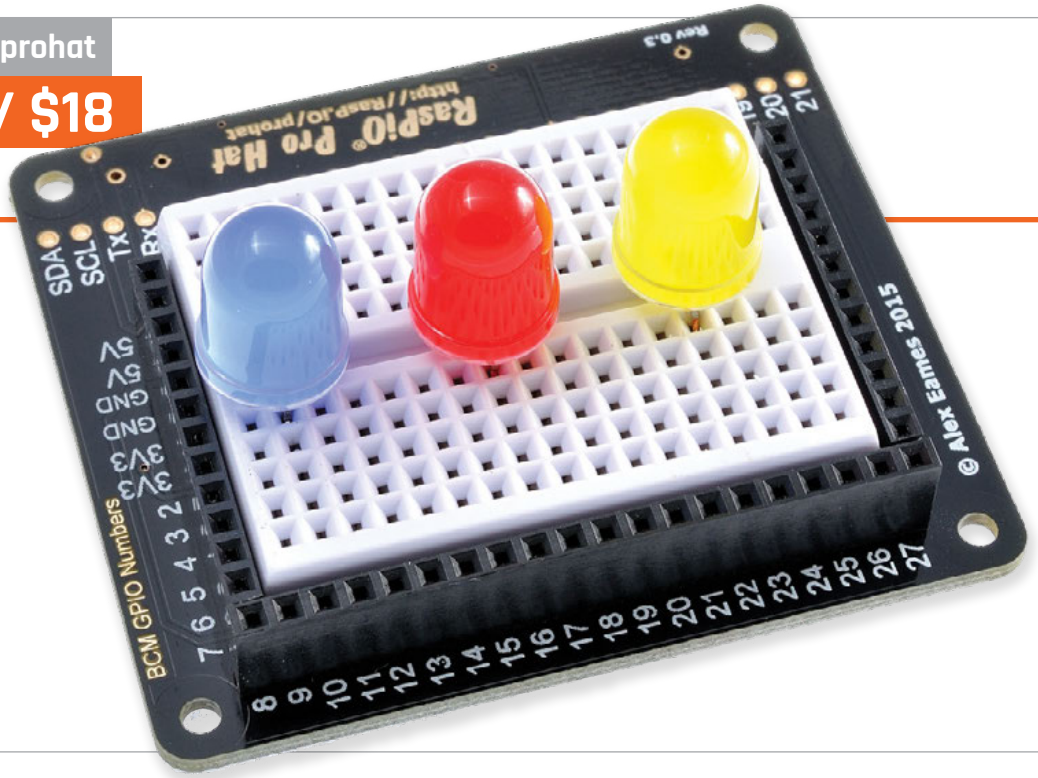
Based on the PCF8591T 8-bit ADC chip, it has four analogue inputs, WiringPi support, and can also do DAC conversion.

£10 / $13

magpi.cc/1syGYDs

rasp.io/prohat

£14 / $18



Maker Says

"Putting your ports in perfect order and protecting them from human error"

RasPiO

RASPIO PRO HAT

Is this electronics prototyping HAT the easiest-to-use ever?

One obstacle that newcomers to Raspberry Pi physical computing come up against is trying to figure out which pin does what on the Pi's GPIO header, particularly since they're not labelled. Previous solutions have included printing out a GPIO pin reference guide on such items as rulers and key-rings. Alternatively, users could just refer to an online diagram, such as at **pinout.xyz**. Even then, you still need to count down the two lines of pins on the Pi to locate the one required, which increases the chance of a potentially damaging wiring mistake.

Enter the RasPiO Pro HAT. Designed by Alex Eames of **raspi.tv**, it makes using the GPIO pins a whole lot easier. Around the edges of its built-in breadboard are female header sockets connected to the Pi's GPIO pins, arranged and labelled in numerical (BCM) order. So, for instance, if you want to hook up a component to GPIO 18, you simply plug its jumper wire into that labelled socket: no more pin-counting! Nor is any extra software required. The HAT works perfectly with GPIO Zero, as well as other libraries. It's such as simple solution, it's a wonder no one's thought of it before.

The third female header comprises six extra connections apiece for 3V3 and GND; this comes in handy, since the mini 72-point breadboard doesn't feature the '+' and '–' power rails of larger versions. In addition, you won't need quite so many jumper wires to power/ground all your components as they are, in effect, wired straight to the relevant GPIO pins. You may need to stock up on a few more male-to-male cables, though.

The other big advantage of the RasPiO is that it makes prototyping a lot safer, since a protection circuit with a 330Ω resistor and 3V3 Zener diode is built into each GPIO port. This means that you can use LEDs without extra resistors, and you won't damage your Pi's GPIO by wiring something up incorrectly, although you still need to take care not to directly short 3V3 or 5V power to GND! One side effect of this protection circuitry is that some components requiring more than the 10mA limit, such as buzzers, may be underpowered; fortunately, a line of unprotected through-hole ports is included for this purpose.

Last word

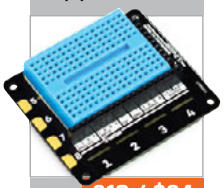
The RasPiO Pro HAT really does make electronics easier, eliminating the need for counting GPIO pins while also protecting them against incorrect wiring. About the only drawback is a lack of room on its half-size breadboard, but if you do need extra space for components, you could easily hook it up to another breadboard.

★★★★★

Related

EXPLORER HAT PRO

While not as easy to use, it includes two H-bridge motor drivers, analogue inputs, built-in LEDs, touch pads, and capacitive clip pads.



£18 / $24

magpi.cc/10AKy46

WILDLIFE CAM KIT

Capture wildlife photos with this weatherproof, Pi-powered camera trap

Related

CAMDEN BOSS ENCLOSURE

This mini, wall-mountable acrylic case will protect just the Camera Module, not the Pi, and isn't weatherproof.

£10 / $13

magpi.cc/29Hsnh0

Ever wondered what kind of critters visit your garden whenever you're not around to scare them off? With the Wildlife Cam Kit, you can find out. Its PIR sensor will sense any movement in the vicinity and trigger its Pi Camera Module to take a stealthy snap of whatever's passing by.

We followed the progress of this Kickstarter-funded project in several issues of *The MagPi*, but now it's finally out in the wild. Designed by Naturebytes, a trio of digital makers and wildlife enthusiasts, its aim is to give users a fascinating insight into the natural world while also enhancing their digital making skills. To this end, it comes in kit form, although no soldering is required. It takes an hour or so to put together, following the detailed online PDF instructions (**magpi.cc/29HamiP**). The latter are well-illustrated with plenty of photos, even if a couple were slightly misleading.

Constructing your camera

A laser-cut plastic insert is provided to suit whichever Raspberry Pi model you're using; the standard kit comes with an A+ due to its lower power usage, but it could even be used with a Pi Zero v1.3 or W. Screws and plastic spacers are supplied to fit most of the components – including PIR sensor, Camera Module, and Pi – to the insert, threading jumper wires through its strategically placed holes. It's clear that much thought has gone into its design; you even get bendy ties to push through pairs of holes to secure the wires tidily. Still, it wasn't so easy to fit the jumper wires to the Pi's GPIO pins through the large cut-out, requiring a small screwdriver to push them into place. A real-time clock module is also fitted to five of the pins (since the Pi doesn't have one built-in), to enable accurate date/time-stamping of photos without an internet connection.

Finally, an Adafruit Powerboost is fitted to the insert and connected to a LiPo battery. This is to boost the latter's 3.7V output to the 5.2V needed to power the Pi and other components. A power switch has been added to the Powerboost to make it easier to switch the camera on and off when outdoors. Before you can use it, however, you'll need to charge up the battery, which takes around 17.5 hours; it might be advisable to start doing this before assembling the rest of the kit. Naturebytes claims it will provide around 30 hours of power; in our tests it lasted up to three days between charges.

magpi.cc/2902evf

From £130/$171

Maker Says

"A camera that anyone can build to take stealthy high-definition images of wildlife"

Naturebytes

Testing it out

Before positioning the camera trap outdoors, it's a good idea to hook it up to a TV to check everything's working properly. The supplied SD card features a customised version of the Raspbian operating system, featuring a Naturebytes logo and desktop shortcuts for various tools, including PIR and camera tests. The main Python script for taking photos can also be tweaked. By default, it adds graphical overlays – a date/time stamp and Naturebytes logo – to each photo taken. However, we found that this was causing a lengthy delay (of up to two minutes) when writing to the supplied USB stick, possibly caused by a lack of RAM in the 256MB Pi A+ to handle the I/O process. Naturally, this is undesirable when capturing wildlife photos, as the critter may have vanished in the meantime. Fortunately, commenting out the code lines for the overlays solved the problem. Writing to an SD card is even quicker, although it's far more convenient to simply remove the USB stick and plug it into another PC in the house to view the photos, without having to bring the whole device back inside.

Venturing outdoors, a Velcro strap is used to fasten the Wildlife Cam to a tree or fence post. The green acrylic case itself seems very tough and durable and snaps tightly shut, secured by two plastic clamps. Naturebytes has tested it in all weather conditions, and we experienced no leakage whatsoever following a downpour. On the front, a circle of clear plastic protects the Camera Module, while another translucent cover guards the PIR sensor.

Start snapping

Once in position, toggling the Powerboost switch turns on the camera and the picture-taking Python script will start running automatically. Our first attempt yielded a shot of a wood pigeon in flight, plus numerous photos of nothing: false positives triggered by the changing amount of sunlight. This effect can be reduced by repositioning the camera to avoid morning or afternoon glare, or by adjusting the sensitivity of the PIR. You can lure wildlife in front of the camera trap by placing it next to a bird table – we improvised by pouring some seeds onto an old garden table and managed to obtain a great shot of a magpie! There's also an optional wooden arm to hang a bird-feeder in front of the camera.

Whatever tactic you use, viewing the photos at the end of the day is an exciting prospect, just to see what has been captured. While not ready at the time of writing, Naturebytes is also working on an online community hub where users can share their wildlife photos, which should add an extra dimension to the project. Ideal for educational use, the Cam Kit is also quite versatile and could be used for time-lapse photography, night-time shots (with a Pi NoIR camera and IR LED lighting), or even a live video feed.

Last word

While more expensive than originally envisioned, the Wildlife Cam Kit is easy enough to assemble and fun to use. Once we'd sorted out the photo-taking delay issue, it worked well outdoors and the anticipation of seeing what it had captured was exciting. That lengthy battery recharge time is a little annoying, but it does power the unit for quite a while. Most importantly, the weatherproof case is very robust and may soon also be available as a standalone unit.

★★★★☆

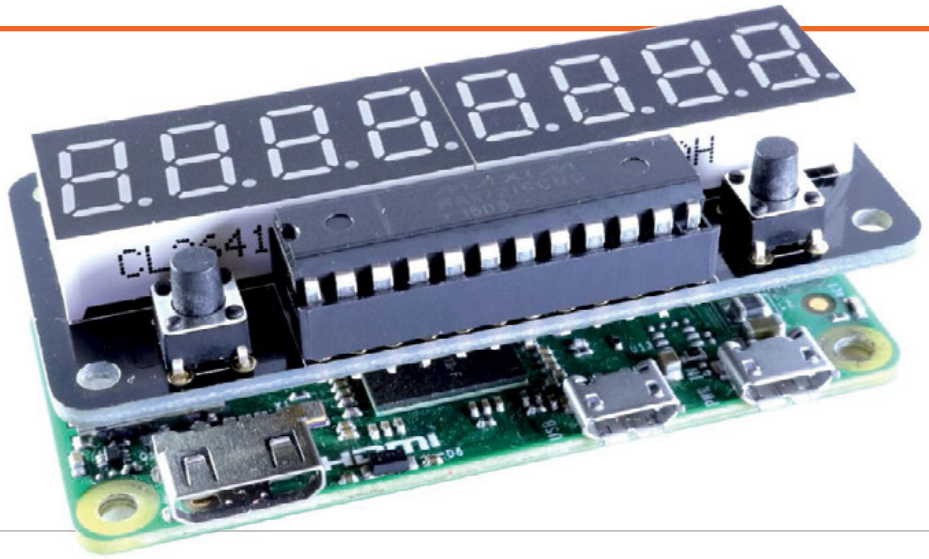
magpi.cc/2d0tGBg

£10 / $13

Maker Says

"An easy-to-use 8-character 7-segment display add-on board"

Average Man



ZEROSEG

Build your own old-school red LED display

While playing around with some generic spare parts, including a standard seven-segment LED unit, Richard Saville – aka Average Man vs Pi (**averagemanvsraspberrypi.com**) – had the idea of creating a more polished, Pi Zero-sized display. Following a lot of reverse-engineering, trial and error, and prototyping, he came up with the ZeroSeg, which features two four-digit display units.

The first thing to note is that it comes in kit form, with numerous components to solder onto the rear and front of the small board, including various resistors and capacitors. Fortunately, there's an excellent online assembly guide to help you, as the parts need to be added in a specific order. Quite a bit of precision is required, too. For instance, the MAX7219CNG chip socket must be flush with the board edge to enable you to cram in the two LED units; when soldering the latter, you also need to take care not to touch previously added components on the rear. Still, it's fun to put together and you get a sense of achievement when it's completed.

To get it working, you need to install the ZeroSeg Python library and spidev, and enable the SPI interface on the Raspberry Pi. In addition to power and ground, it only uses five GPIO pins: 8, 10, 11, 17, and 26; this means there are still plenty to play with if you're breaking them out or stacking the ZeroSeg on top of another board.

The ZeroSeg code library includes a few Python examples to get you started, including a demo that shows off its capabilities, such as the ability to fade the brightness through 15 levels and scroll digits across the display. It's fairly easy to program by adapting examples, although we couldn't figure out a way of showing text; this may well have been added by the time you read this, although some letters (such as M and W) are impossible to reproduce on a seven-segment display. So it's best suited to displaying digits; use cases include a temperature monitor and time/date display. The two programmable mini-buttons are a nice bonus and can be used to switch what's shown.

Last word

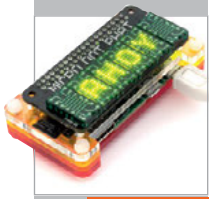
While not as flexible as a matrix display, the ZeroSeg is great for value for money and fun to assemble. More suited to displaying digits than text, it's fine for numerical data, which may be scrolled across the two LED units. The latter are bright enough at full power, although in daylight the white non-lit sectors are very visible.

★★★★☆

Related

MICRO DOT PHAT

Able to display text and numbers, this LED matrix display is very versatile and renders characters in great detail.

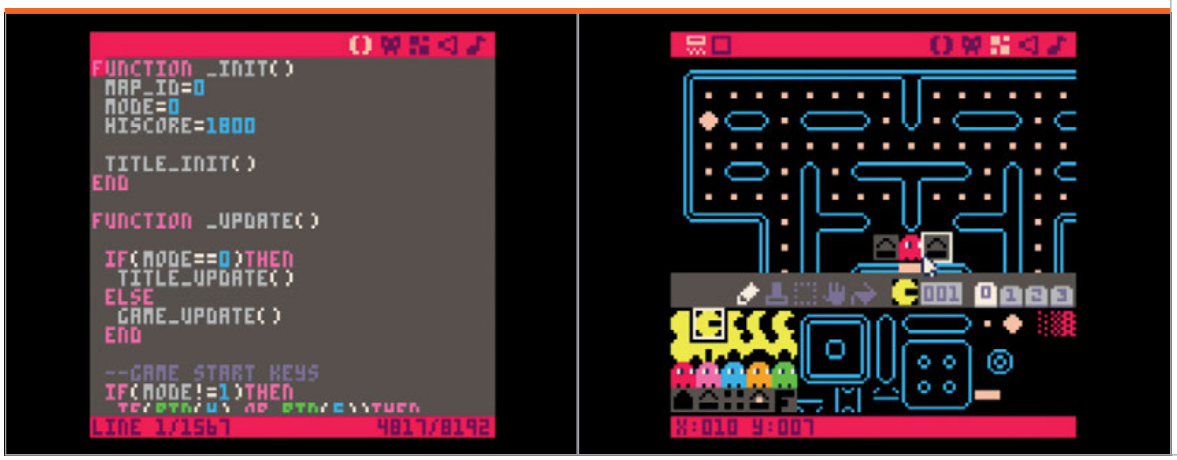


£22 / $30

magpi.cc/2cfq70b

lexaloffle.com

£11 / $15



Maker Says

"A fantasy console for making, sharing, and playing tiny games and other computer programs

Lexaloffle

PICO-8

Build, play, and share 8-bit games with this imaginary console with built-in code-, sprite-, and music-editing tools

Imagine the best console from the 1980s that you never owned. As well as being able to play free games, you would be able to stop them at any time and dive backstage.

You could tweak the code to make it easier, harder, or different. The built-in sprite editor and sound effects tools would let you customise the graphics. And there'd be a level editor for building new stages or changing existing ones.

That's PICO-8. It's a virtual games console, but instead of being an actual 8-bit console, it recreates a virtual console with built-in editing tools.

PICO-8 was developed by Joseph White from his base at Pico Pico Cafe in Kichijouji, Tokyo. It's been around for Windows, Mac OS X, and Linux machines, but feels more naturally at home on the Raspberry Pi, where it can become the console it was meant to be.

PICO-8 runs just fine on the Raspberry Pi hardware. We'd be surprised if it didn't, because a key aspect is a deliberately limited feature set. The display is just 128×128 blocks with 16 colours. Sprites are 8×8 pixels, and you can only have 128 on screen at once.

The virtual cartridges (saved as PNG files) are limited to 32kB, and it has a four-channel sound chip.

Imagine it sitting alongside a NES or PC Engine in terms of technical prowess. These deliberate limitations make PICO-8 more engaging. The games have a retro-cool aesthetic and they're easier to build. The low-spec nature of the console helps newcomers get started.

Code is written in Lua. It's a relatively simple language to learn and mostly used for scripting Adobe programs. While it's no Python, Lua is worth learning. Games can be shared in a web browser, so you can see what sort of thing is possible on the Lexaloffle website.

Community service

A vibrant community has sprung up around PICO-8, with one of the most active fan bases we've seen. The forums are packed with interaction between developers, with everybody chipping in and offering to help.

If there's one downside, it's that PICO-8 is a paid-for program. But we think it's great value given the amount of fun we had, along with the active community.

Last word

We had a huge amount of fun with PICO-8, and it's a natural fit for the Raspberry Pi.

★★★★★

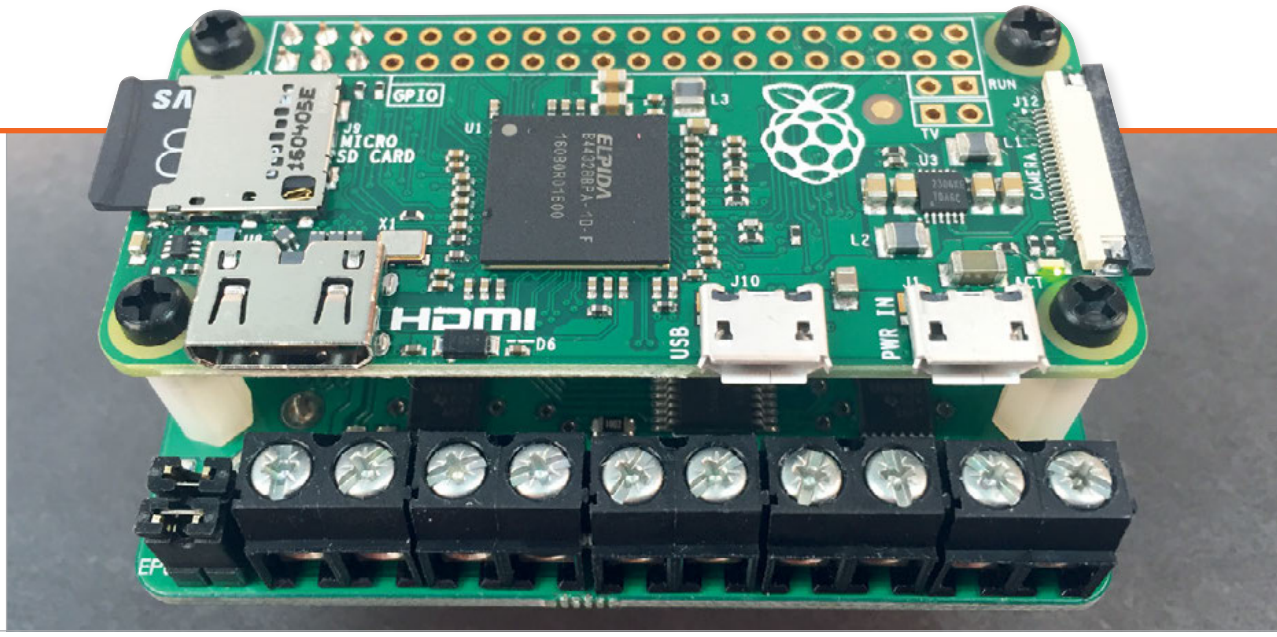
Related

PYGAME

Pygame is a more powerful set of gaming modules that uses Python. It's not got the same holistic environment as PICO-8, though, but you can build some impressive games with it.

Free

pygame.org



Maker Says

"Opens up endless possibilities for tiny robot designs"

PiBorg

ZEROBORG

Control four motors independently with this versatile Zero-size board from the robot experts at PiBorg

Raspberry Pi robotics specialists PiBorg have turned their attention to the Pi Zero and the possibilities of using it to make very small robots. The result is the ZeroBorg, a diminutive motor controller board that's only marginally wider than the Zero itself. When mounted to the rear of the Pi Zero, the whole setup (including optional 9V battery) weighs a mere 65g. It's so lightweight and nifty that PiBorg used it to control the YetiBorg racing robots in the first rounds of the Formula Pi series.

The inclusion of four H-bridges means that the ZeroBorg can control four standard DC motors independently. Add some special Mecanum wheels and you can get your robot to scuttle sideways like a crab! Even when using standard wheels, the ZeroBorg offers extra control since the bidirectional PWM (pulse-width modulation) signal sent to each of the four wheels can be varied precisely. Each H-bridge can deliver 2A peak or 1.5A RMS current, so it should work with most small motors. Alternatively, the board can be used to run two four-, five-, or six-wire stepper motors.

Stacks of fun

One curious aspect of the ZeroBorg is that it's designed to be connected to a Pi Zero that has an unpopulated GPIO header. Instead, it's supplied with a small female header to fit to the rear of the Zero, at the 3V3 end of the GPIO header; into this you slot the ZeroBorg's six pins, two of which connect to SDA and SCL for I²C communication. Now, while it's possible to do this without soldering the small header to the Pi Zero, and instead simply holding the two units together firmly using the supplied standoff screws, we were unable to get this method to provide a reliable enough connection. Once we'd soldered the header to the Pi Zero, however, everything worked absolutely fine, so we'd strongly advise doing this. Alternatively, if your Zero already has a full GPIO male header attached, you could always use two 3-pin female-to-female connectors to connect it; this method would also enable you to use the ZeroBorg with any other Raspberry Pi model.

It's important to note that the ZeroBorg comes in three main versions. While the basic KS1 model comes pre-assembled, the KS2 adds a DC/DC regulator and battery clip (supplied loose or pre-soldered) so that the ZeroBorg, motors, and Pi Zero

Related

MOTOZERO

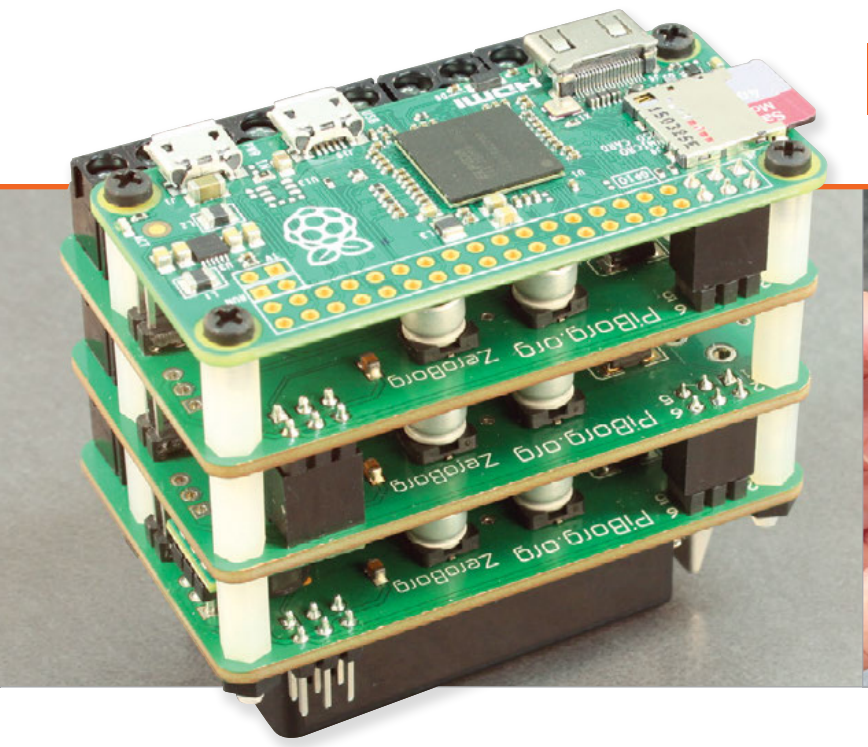
Resembling an exposed engine, it can control four motors independently, though it lacks any sensor inputs.

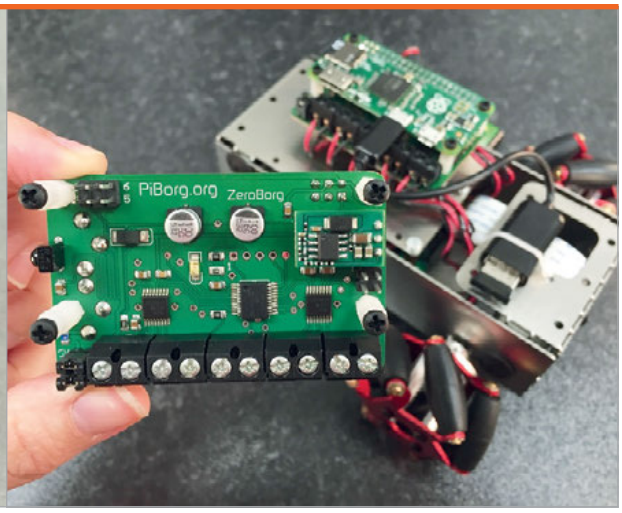
£10 / $13

magpi.cc/1XRfqGQ

magpi.cc/2aFKJ60

From £18 / $24





Above left
The ZeroBorgs are designed so they can be stacked on top of one another

Above
Build full robots with the tiny ZeroBorg

can all be powered by a standard 9V PP3 battery. Alternatively, an external power source such as a battery pack can be attached to two of the ZeroBorg's terminals, enabling you to mount it flat. The KS2 model also includes an infrared sensor (more on that later) and a second six-pin male I²C header for daisy-chaining with other add-on boards,

Motoring on

We tested a pre-soldered KS2 ZeroBorg for this review, so all we needed to do was solder the female header to the Pi Zero, screw in the standoffs, insert the battery, and we were ready to roll. Well, almost. First, you need to ensure I²C is enabled on the Pi, then install the ZeroBorg software using a single terminal command. It's then just

> ZeroBorg software includes a special Python library, along with numerous examples

including the UltraBorg, PicoBorg Reverse, or another ZeroBorg. Indeed, the KS3 option comprises a stack of three ZeroBorgs, the middle of which features two female I²C headers to allow communication between the three boards. While overkill for your average robot, this version could prove particularly useful for animatronics projects or running multiple servos in a CNC machine, for instance. All ZeroBorg models also include two analogue inputs (plus power and ground) for attaching sensors.

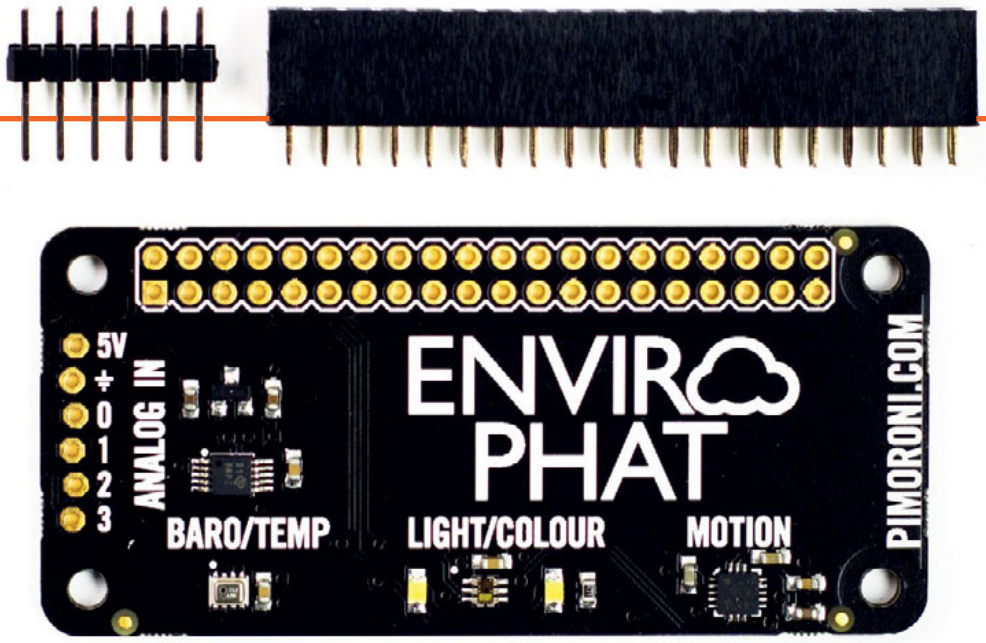
a matter of wiring up your motors as usual; the terminals are all located on one edge of the ZeroBorg, which isn't quite as intuitive as on the rival MotoZero, but they're nice and chunky so they should prove durable. In addition, the ZeroBorg features short circuit prevention to prevent any damage from incorrect connections, along with overheat protection, under-voltage lockout, and a fast-blow 5A fuse.

The ZeroBorg software includes a special Python library, along with numerous examples to get you started. It's all fairly straightforward: for example, the `ZB.SetMotor1(1)` command is used to supply maximum speed to motor 1. Use a lower number for less power, zero to stop it, and a negative value to reverse. The examples include joystick control, stepper motor sequence, analogue inputs, and control using an infrared TV remote; if yours isn't supported by default, it's easy to record and save the raw IR codes and add them to the main script. We were soon using a TV remote to control our swiftly assembled 'Tubbybot', made from a small plastic storage tub to which we strapped four micro metal-gear motors and wheels. While not the fastest off the blocks, Tubbybot was able to do some nifty spin-turns by powering one pair of wheels forwards while reversing the others.

Last word

While its connection method is a little unorthodox, the ZeroBorg is a mini marvel for motor control. The ability to power both the motors and Pi Zero using a single 9V battery should prove particularly useful when designing small robots, while the daisy-chaining options offer extra flexibility for other possible uses.

★★★★★



Maker Says

"It's ideal for monitoring conditions in your house, garage or galleon"
Pimoroni

ENVIRO PHAT

This Zero-size add-on features four built-in sensors plus analogue inputs

While not an official Raspberry Pi standard, Pimoroni's pHAT class of half-size add-on boards are great fun and match the Pi Zero's form factor perfectly, although they'll work with any 40-pin Pi model. The latest addition to the line is the Enviro pHAT, which is all about taking environmental and motion measurements. Along with several built-in sensors, it features four analogue input channels to connect your own external sensors. In effect, the Enviro pHAT is Pimoroni's Flotilla weather, colour, and motion modules rolled into one, with the addition of an analogue-to-digital (ADC) converter.

First things first: the Enviro pHAT comes in kit form, so you'll need to get your soldering iron out to attach the 2×20-pin female header and six male pins for the analogue inputs. Alternatively, you could even solder the pHAT straight onto the GPIO pins of a Pi Zero, if you wanted to use them together as a permanent room-monitoring or motion-measuring device.

Once the pHAT is assembled and mounted on the Pi's GPIO header, installing the software requires just a single command in the terminal. Assuming your Pi already has I²C enabled, you're then able to start coding to obtain readings from the sensors, using the pHAT's own Python library. The latter is partitioned into five separate modules: **`light`**, **`weather`**, **`motion`**, **`analog`** (inputs), and **`leds`**.

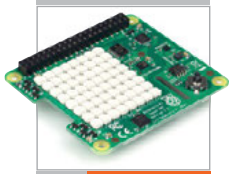
Modular sensors

The **`light`** module offers two main methods for reading the built-in TCS3472 sensor, which monitors four different values: clear, red, green, and blue. As well as an ambient light level reading using **`light.light()`**, you can obtain RGB colour values with **`light.rgb()`**, for a tuple which can easily be split into separate values. As you can see, the function naming structure used by the library couldn't be simpler, so it's all very easy to code. To aid accuracy of colour readings, the board has two small white LEDs located on either side of the light sensor, which can be switched on and off using the **`leds`** Python library module. Even so, the colour values produced are for a duller shade than the real item analysed, so may require some calibration.

The library's **`weather`** module enables you to obtain temperature and barometric pressure (in hPa) readings from the Enviro pHAt's BMP280 sensor, but it doesn't measure humidity. Since the sensor is mounted on the PCB rather than remotely, its temperature reading is greatly affected by the heat of

Related

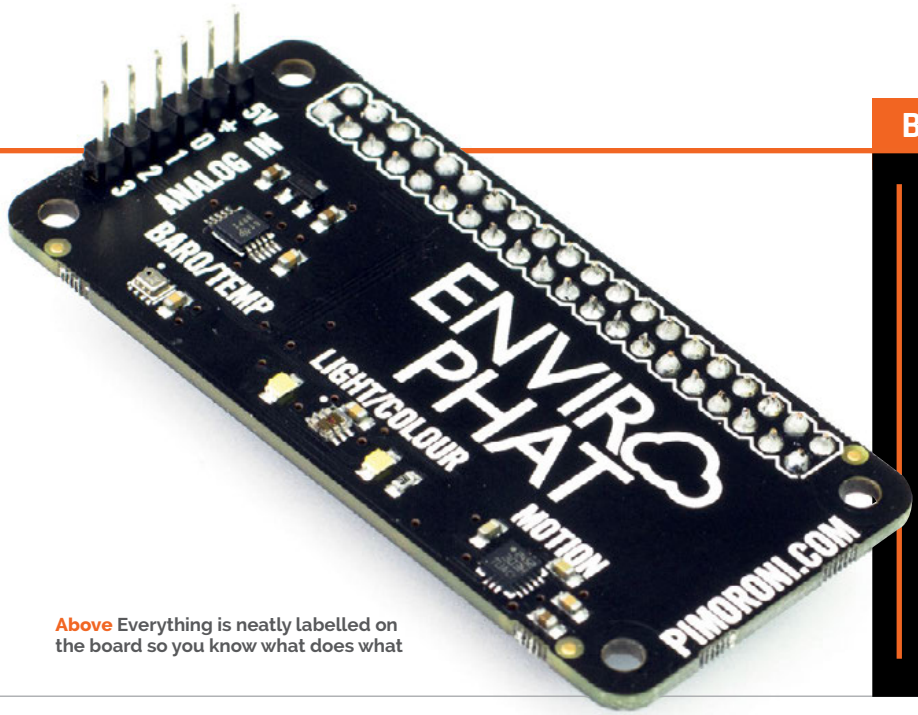
SENSE HAT
As used in the Astro Pi devices aboard the ISS, the Sense HAT features multiple built-in sensors and an 8×8 LED matrix display.



£32 / $40
magpi.cc/1TGGUt5

magpi.cc/29NHB3T

£16 / $20



Above Everything is neatly labelled on the board so you know what does what

BUILT-IN SENSORS

Light: The highly sensitive TCS3472 colour sensor enables you to measure the ambient clear light level and RGB colour values, aided by twin LEDs to illuminate objects.

Weather: A BMP280 sensor measures atmospheric pressure and temperature, although the latter is affected by the heat produced by the pHAT.

Motion: The built-in LSM303D 3D accelerometer/magnetometer can detect the board's orientation, motion, and compass heading.

Analogue Input: An ADS1015 ADC enables it to convert analogue readings from external sensors on four channels.

the Raspberry Pi CPU beneath it. Therefore, you'll need to calibrate it by comparing the real ambient temperature, using a standard thermometer, to discover the difference; for us it was around 7°C, but it may vary depending on the setup. For a more accurate reading, you could always use a remotely placed temperature sensor connected to the pHAT's analogue input section: more on that later.

Detecting motion

The Enviro pHAT includes an LSM303D accelerometer/magnetometer for detecting the board's motion through three axes (pitch, roll, and vertical) and its compass bearing. The latter can easily be calibrated to north, so long as you already know where that is; it's done by setting a variable to its value and subtracting it from the reading (with modulo 360) to get the correct compass heading in degrees. Meanwhile, the `motion.accelerometer()` tuple can be split into three variables, one for each axis. You can also obtain the raw magnetometer data if you prefer. Since the combination of Enviro pHAT and Pi Zero has such a small form factor, it's ideal for measuring the motion of people carrying it or objects attached to it, although it'll require a portable power source such as a phone charger.

Last but not least, the Enviro pHAT features an ADS1015 ADC for reading external analogue sensors. Located on a short edge of the board are six pins: 5V power output and ground, plus four input channels to take readings from sensors. Note that the input pins are designed to measure signals between 0 and 3.3V, so if your sensor's output is 5V you'll need to create a voltage divider, using three identical resistors on a breadboard, to lower it to 3.3V. While Pimoroni says that in its tests, running 5V into the ADC inputs didn't cause any adverse effects, the readings won't be reliable unless you use a voltage divider. It's not much of a hurdle, though, and the inclusion of an analogue input section for connecting extra sensors is a major bonus.

Overall, with its similar functionality, the Enviro pHAT is a cheaper, more portable alternative to a Sense HAT, although without the LED matrix and a few other features, but with the addition of analogue inputs for extra sensors. The Python library is very intuitive and easy to use, aided by an online tutorial (**magpi.cc/29maHZT**) to get you started and a few helpful code examples in the GitHub repository (**magpi.cc/29M8bdD**). With its small form factor, we can see the Enviro pHAT being used with a Pi Zero to create IoT devices for monitoring room temperature, light levels (to possibly trigger electric lighting), and various other remote uses. By using a stacking header, it could also be combined with another Pimoroni pHAT, such as the Scroll pHAT with its LED matrix, to display its readings in situ.

Last word

For portable projects requiring sensor data, the Enviro pHAT could prove particularly useful. You could just mount it on a Pi Zero and leave it on a shelf to monitor room conditions, for instance, logging its readings into a file or database. The inclusion of an ADC and analogue inputs for external sensors is a bonus for what is a fun, easy-to-use add-on with plenty of possibilities.

★★★★★

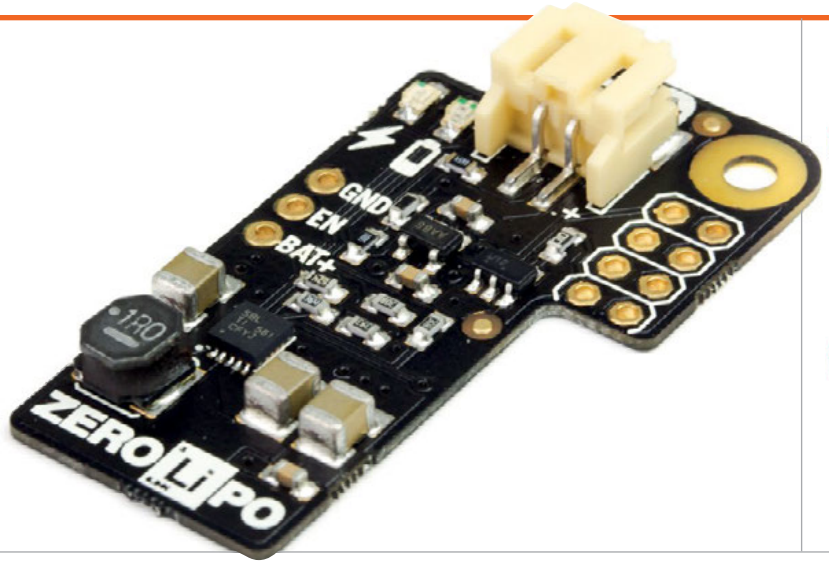
magpi.cc/2coJyZ2

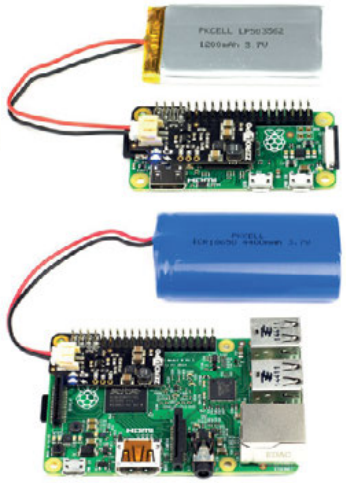
£10 / $10

Maker Says

"Aims to give you the most compact Raspberry Pi power supply possible"

Pimoroni





LIPO SHIM

Reliable and portable power for the Raspberry Pi

The LiPo SHIM is a tiny add-on board for the Raspberry Pi that enables you to power it from lithium polymer and lithium ion batteries.

The board itself is tiny. Measuring just 0.8mm thick and weighing 2.9g, it doesn't interfere with your usage of the Raspberry Pi.

It attaches to the eight GPIO pins at the top of the pin layout (covering the two 5V pins, 3.3V, GPIO 3 and 4, and Ground). From here it provides power to the Raspberry Pi, while leaving the micro-USB power socket free.

There are two ways to attach the LiPo SHIM to the Raspberry Pi. You can either solder the board to the bottom of the standard GPIO header, or you can solder it to a 2×4 female header.

This second option enables you to attach, and remove, the LiPo SHIM to/from the GPIO pins, but it blocks the pins. Both options require deft soldering skills.

In addition to the Pi Zero, you can use the LiPo SHIM on any Raspberry Pi model.

Once attached to the Raspberry Pi, the board gets its power from a battery via a JST connector. Just charge up your battery, then plug it in to power up the Raspberry Pi.

Pimoroni is selling a range of LiPo and Li-ion batteries, ranging from £4.50 to £18.

We tested a 1200mAh LiPo and a 2200mAh Li-ion battery. Both battery types function largely the same (plug and go). The board uses a step-up boost converter (TPS61232) to convert the 3-4.2V from the batteries to the 5V used by the Raspberry Pi.

As the battery runs out, you get an alert at 3.4V: a red low voltage warning LED comes on. The board automatically shuts down at 3.0V. This protects the battery.

The Pimoroni blog (**magpi.cc/2d1YwCm**) has detailed tests on how much usage you can expect to get in a variety of common tasks, across the range of batteries.

In addition to the Pi Zero, you will need a JST charger for your battery, such as the Adafruit Micro Lipo (£7.50) or PowerBoost 1000 (£20.50), both also available from Pimoroni.

Related

ANKER ASTRO E1 5200MAH

You can power a Raspberry Pi using most USB battery packs designed for smartphones. These are bulkier and less integrated than the Zero LiPo.

£13 / $18

magpi.cc/2d21w1E

Last word

It's a slick piece of kit, but make sure you factor in the price of the charger and batteries. It's a great option for providing safe power for a portable project, though.

★★★★☆

thepihut.com

£7 / $9

Maker Says

"The ZeroView is a clever window/glass mount for your Pi Zero and Camera Module

The Pi Hut

ZEROVIEW

Stick your Pi Zero with Camera Module to a glass window with this suction cup mount

One of our favourite uses for the Pi Zero is recording time-lapse video with the camera connector (found on the newer Pi Zero v1.3 and W).

So we were delighted to get hold of the ZeroView. This simple board provides a suction cup mount for the Pi Zero, so you can stick it to glass.

It's ridiculously easy to set up the Pi Zero to record pictures, videos, or capture time-lapse photography. A device that effectively mounts the Pi Zero and holds the Camera Module comes in useful in a range of projects, from home-built in-car dash cams to time-lapse fish tank recordings.

The Pi Zero is mounted using plastic screws. Inside the pack you get a PCB (but just a plain board with no electronic components), two suction cups, and spacers, screws and nuts to mount both the Pi Zero and Camera Module. It took us about five minutes to screw it all together following the PDF instructions at **magpi.cc/2e89hWt**.

The Camera Module is mounted and the cable tucked between the Pi Zero and ZeroView. The end result is a compact, self-contained camera device that can be stuck to any glass surface. Combine it with a battery pack, and set up a script to automatically start recording, and you get a neat camera package.

We've had trouble with suction cups before, where devices have dropped. With this in mind, we stuck the ZeroView to a window to capture a time-lapse video, and started a stopwatch to see how long it lasted. After an hour, we decided that it was going to be there all day and stopped the test.

"We've hunted down the best quality suction cups we could find," says The Pi Hut, "using only the best 'Adams' cups made in the USA. We're so impressed with the performance of these suction cups that we just couldn't use any other brand."

Whether it's the high-quality cups or the general lightness of the package, it's hard to fault the ZeroView. It's easy to set up, looks cool, and sticks around all day.

Last word

A neat product that transforms the Pi Zero and Camera Module into a portable, stickable camera package ideal for time-lapse and slow-motion photography projects.

★★★★★

Related

RASPBERRY PI CAMERA MOUNT

A cheaper option is to buy a mount for just the Camera Module, but this doesn't provide a combined package.

£3 / $4

thepihut.com

Maker Says

"Build your computer through Minecraft"

Piper

PIPER

Build a computer and then keep building it as you play through a Minecraft adventure

Once again, we've come face-to-face with a crowdfunded Raspberry Pi laptop. With the pi-top not even a year old, it's interesting to see something that, on paper, is a competitor for the same space. A 'build-it-yourself' laptop that gamifies learning computing through a custom operating system, the Piper is very different from the pi-top when it comes down to it, however.

First of all, construction of the laptop is very different. While the pi-top feels like you're assembling the components for a real laptop, Piper feels like putting together a Meccano kit or wooden model. Laser-cut, engraved wooden sections slot into place, held together by the odd screw. There's a big sprawling poster with the steps needed to put the box together, with the engravings giving you some visual clues on what goes where. The poster is a little unwieldy and you need lots of space for it, but construction is fairly simple, if not a little lengthy. We sat through at least a couple of episodes of *Star Trek: Deep Space Nine* getting it built, so it took about 90 minutes.

Some assembly required

The final build is chunky and sturdy. The computer parts include a nice 7″ LCD display in the top, a Raspberry Pi, a USB mouse, and a portable power bank to power the whole lot. This makes it quite mobile, although you'll need to remember to charge up the power bank and keep an eye on its levels.

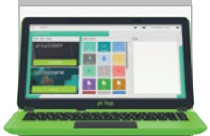
The most ingenious thing about the Piper, though, is that you can carry all the electronics pieces, speaker, and mouse inside the laptop. It's not really so much of a laptop as a digital toy chest, with all your Power Rangers (buttons) and Barbies (jumper wires), and whatever kids actually play with these days (Star Wars figures?) kept inside, latched up and ready to take with you wherever you go. The only thing it's really missing is a carry handle, although we really wouldn't want to be swinging it around with loads of bits inside.

The initial instructions take you as far as getting the case built, and the Raspberry Pi and screen working. Plug it all into the battery pack and you boot up into the Piper's OS. This starts with a fun little video before launching you into the Minecraft adventure that helps you continue to build your laptop, adding the extra buttons

Related

PI-TOP

A similar concept but a very different execution, the pi-top may be more suitable for older kids and young adults.

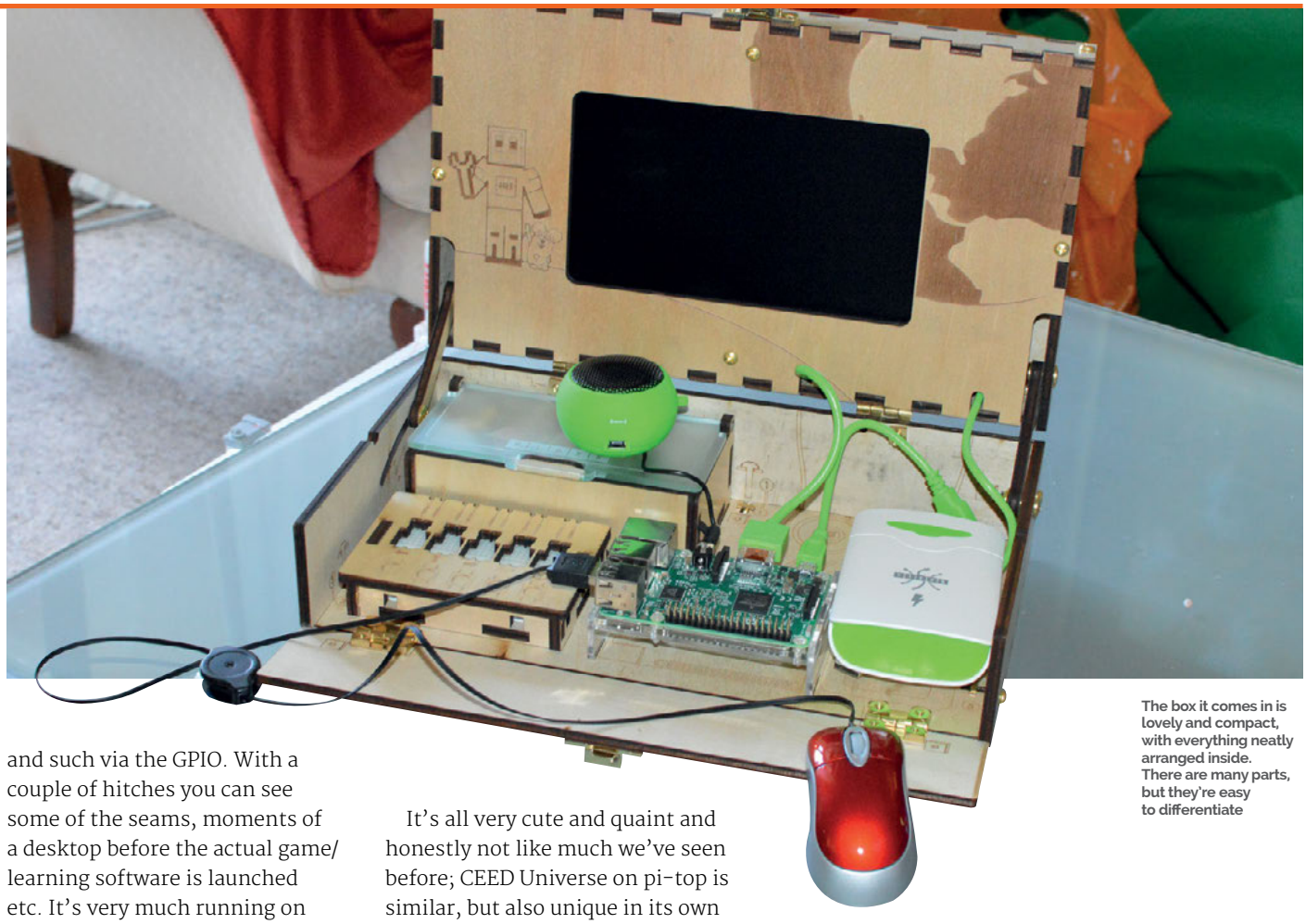


£216 / $285

pi-top.com

playpiper.com

£227 / $299



The box it comes in is lovely and compact, with everything neatly arranged inside. There are many parts, but they're easy to differentiate

and such via the GPIO. With a couple of hitches you can see some of the seams, moments of a desktop before the actual game/learning software is launched etc. It's very much running on Raspbian, but you'll never see it through normal use.

Know your craft

PiperCraft is the name of this game, a modded Minecraft Pi that gives you challenges to complete and in the process teaches you some real-world physical computing. Each section is presented by machinima-style cut scenes, presumably filmed in full Minecraft, which are also fully voiced in an adorable fashion. Guide PiperBot to save Earth from Mars, with only a witty assistant and many Minecraft blocks to help you. There are multiple levels and apparently more are being made, which will be free to download as they become available; people can also create levels and share them.

It's all very cute and quaint and honestly not like much we've seen before; CEED Universe on pi-top is similar, but also unique in its own way beyond just being 'gamified computing education'.

Let's return to the concept of it as a laptop, though. As we've said before, the version you're supposed to build and play with is not really a proper Pi laptop in a traditional sense. You don't have a keyboard, for starters. However, it can easily be modified to be a more normal laptop. You can take out the Piper SD card and make a normal Raspbian one for yourself. The screen connects via HDMI, so it doesn't require any extra software to get running. And if you take out the little component chest and the breadboard, there's enough space to store a little USB or Bluetooth keyboard within the case. The version that's shipping to consumers will come with a Pi 3 so you can connect to wireless, so really it's very little effort to do a 'conversion' if you wish.

It's a really fun, excellent kit. The build, the game, and the possibilities for it are great, even if it's perhaps more suitable for younger kids than the 'all ages' for which it's being marketed.

Last word

The price may be a little steep, but it's a really fun educational computer kit that should really impress those who love Minecraft and building stuff. You can also take it almost anywhere!

★★★★★

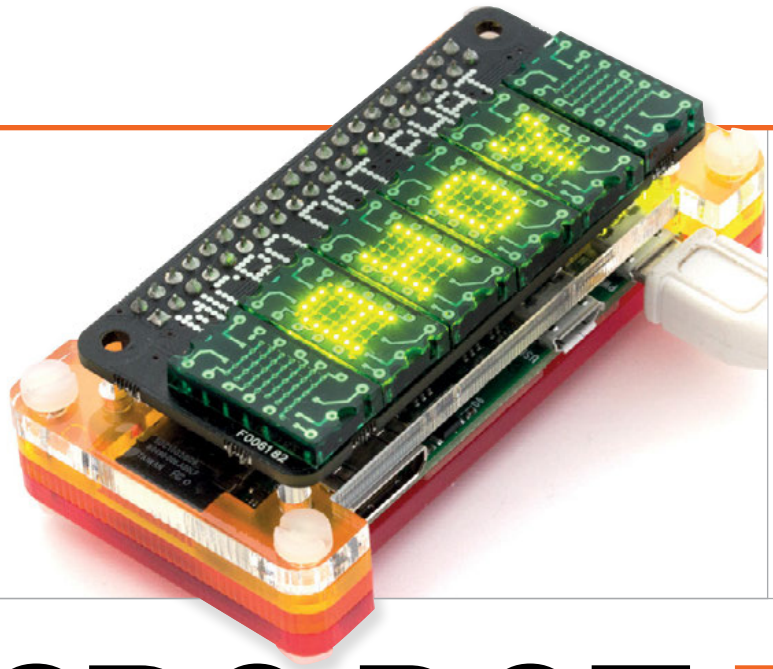
magpi.cc/2cfq70b

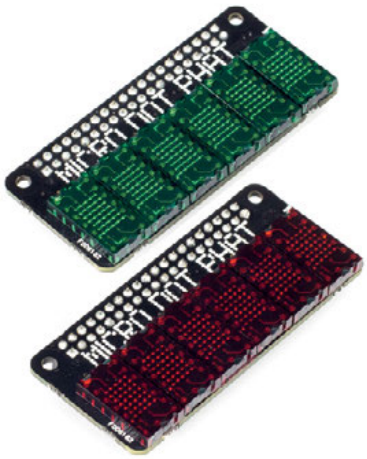
£22 / $30

Maker Says

"An unashamedly old-school LED matrix display board"

Pimoroni





MICRO DOT PHAT

A versatile mini LED display with retro appeal

The LTP-305 LED matrices used with the Micro Dot pHAT boast a substantial heritage, having been introduced in 1971 by Texas Instruments. Now manufactured by Lite-On, they come in green and red varieties and feature 5×7 pixels plus a decimal point. Up to six can be mounted on the Micro Dot pHAT, included in the full kit (£22) or purchased separately for £5 per pair (the bare pHAT is just £8), so you could opt for fewer to suit your project's needs. You'll need to warm that soldering iron up, as the full set requires connecting 118 pins: 13 each for the matrices, plus a standard 40-pin female header. Since each matrix has a leg missing on one side, mirrored on the board, there's no chance of accidentally inverting them. You'll want to ensure they're sitting flush, though.

With the soldering done, it's simply a matter of installing the software with a single terminal command. This loads the Micro Dot's own Python library, plus optional example scripts, although for some reason we ended up having to download them manually. Running the **flash.py** script is recommended first, to check all the pixels are working. Other code examples demonstrate the display's considerable capabilities and possible uses, including an excellent digital clock, animated bar graph, sine wave, and scrolling text – horizontal and vertical. The comprehensive Python library enables you to light individual pixels; it also includes options to alter brightness to fade text in and out, and use a tiny text mode to write smaller characters, rotated 90°.

While the display's high number of small pixels results in well-defined digits and letters, which look excellent when shown one per matrix, horizontally scrolling text isn't quite so easily read due to the gaps between matrices. Other than that, however, this is a very versatile retro-style display and certainly a cut above the standard seven-segment alternative. And since each pair of matrices is driven by an IS31FL3730 chip, talking to the Pi via I²C on addresses 0x61, 0x62, and 0x63, you should be able to use the display alongside many other boards, such as the Enviro pHAT.

Related

SCROLL PHAT

While not quite so fancy, this all-in-one 11×5 array of white LEDs is easier to assemble and ideal for scrolling text.

£10 / $13

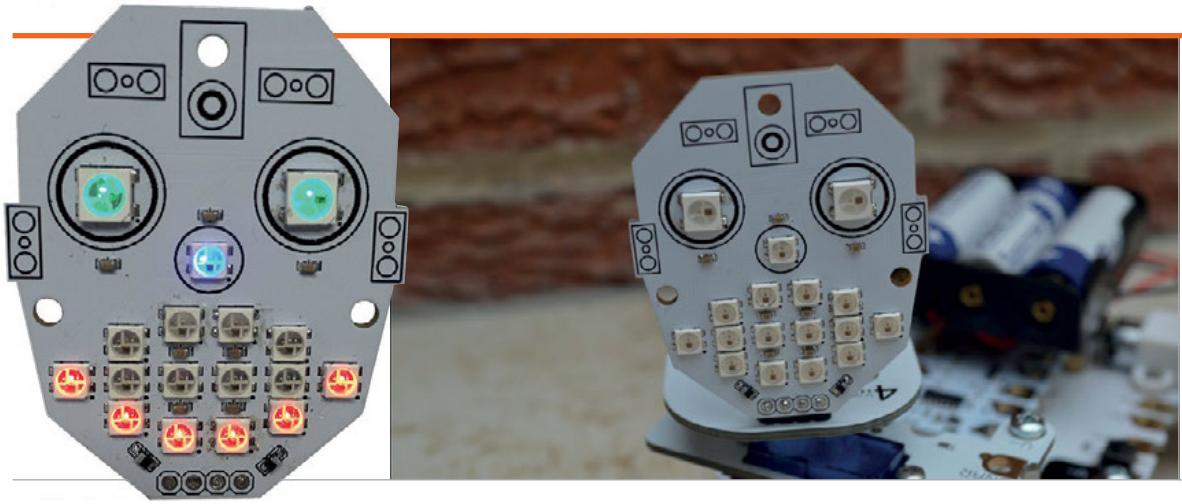
magpi.cc/2c38LH3

Last word

Apart from the slight difficulty in reading horizontally scrolling text due to the gaps between matrices, this is an excellent, highly versatile retro-style LED display. Superior to seven-segment LED displays, it renders characters in great detail and could come in useful for all manner of projects. You might even be able to play simple games on it, like Pong.

magpi.cc/2dXcipD

£7 / $9



Maker Says

"Add emotion and fun to your electronic creation"

4tronix

MCROBOFACE

This bright light-up face will add character to your projects

Launched via Kickstarter, McRoboFace is a PCB board with 17 WS2812B RGB LEDs, also known as NeoPixels. These are fully addressable and arranged in the shape of a face. At full power, they're blindingly bright and, while their intensity is adjustable via software, we'd advise purchasing the optional diffuser kit to soften the effect; the frosted acrylic diffuser is easily fitted to the front using three nylon screws, nuts, and spacers.

Either way, you'll need to solder on the supplied four-pin right-angled header to connect it to your Raspberry Pi. It can also be driven by many other microcontrollers, including micro:bit, Arduino, Codebug, BeagleBone, Crumble, and ESP8266. When using it with the Pi, you have two options. The first method is to connect it via a Picon Zero, using output 5 set to WS2812B. Since the Picon Zero also features two H-bridge motor drivers, it's an easy way to create a wheeled robot with an expressive face at the front.

Alternatively, you can hook the McRoboFace up directly to the Pi's GPIO pins 5V and GND, along with GPIO 18 (the PWM pin) for precision control of the NeoPixels. While requiring a few extra setup steps, this method works perfectly well; no voltage level shifting is needed, as the pixels can be controlled using 3.3V quite happily. Incidentally, the fourth McRoboFace pin is a digital out for daisy-chaining with other NeoPixel displays.

The Pi connection method will determine the Python programming method for controlling McRoboFace. Again, a little more setup is required when using the GPIO pins directly, including the importing of the neopixel (rpi-ws281x) library. It's not a major hurdle, however, as you can just adapt the example code from the GitHub repo (**magpi.cc/2dxooY3**).

Controlling the NeoPixels is easy enough, as they're numbered on the PCB: 15 and 16 for the eyes, 14 for the nose, and the rest for the mouth. Since they're all fully addressable, you can adjust the RGB shade and brightness of each precisely. This makes it possible to create some very impressive fade and colour cycle effects. Using Python lists also enables you to easily change several pixels at once for facial expressions.

Last word

McRoboFace is an inexpensive and fun way to add a bit of character to your robots or other creations with facial expressions, or as a general NeoPixel light display. You could even hook it up to an audio input, as Robin Newman did (magpi.cc/2dxqZ4k), to 'sing' along to music!

★★★★☆

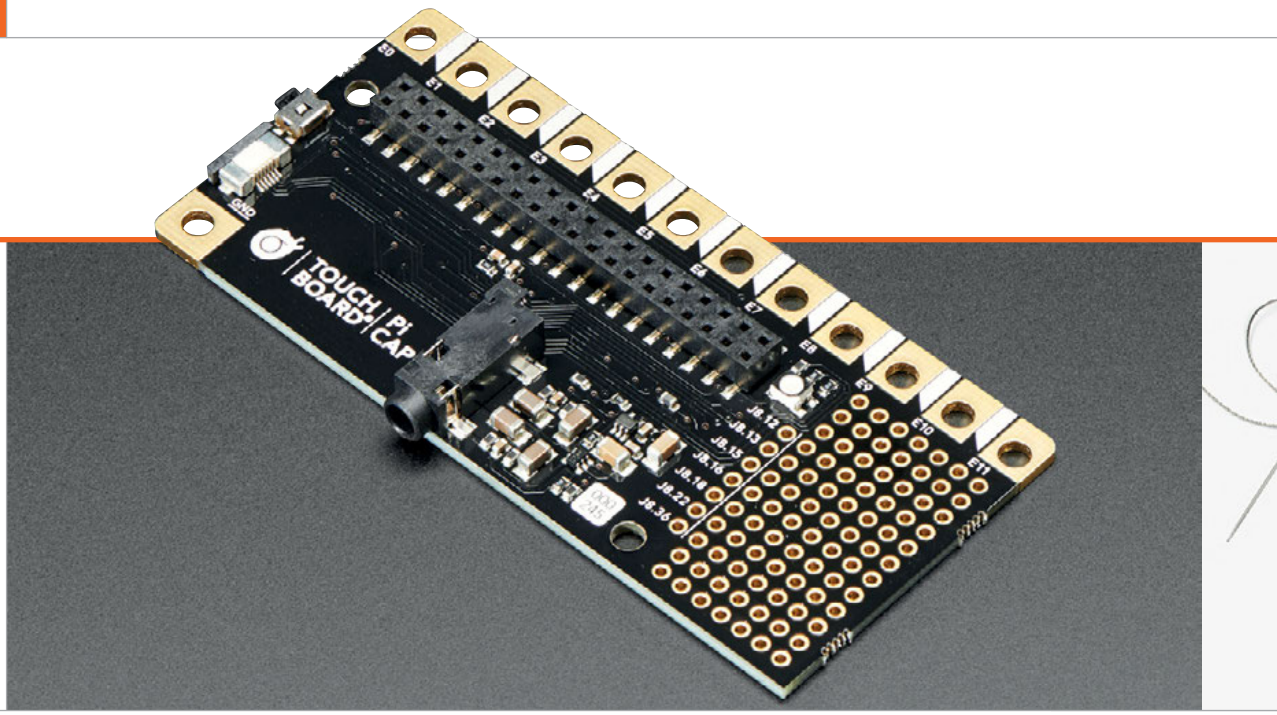
Related

NEOPIXEL RING

Available in various sizes, from 12 to 60 NeoPixels, these chainable rings are an alternative to the standard NeoPixel strips.

From £6 / $8

magpi.cc/2dXaWLy



Maker Says

"Connect your Raspberry Pi to the physical world"

Bare Conductive

PI CAP

A HAT that adds some interesting features to a Raspberry Pi. Just how well does it fit?

Bare Conductive is one of those cool things we love to see stuff made with. The conductive paint can be used in some amazing creative builds and we love seeing people make things with it and post pictures and videos online.

To expand the uses of the paint, Bare Conductive the company has created its own special HAT for the Raspberry Pi, called the Pi Cap. HAT, Cap? The name is a bit more than just a pun, as one of the board's most interesting features is the addition of capacitive touch buttons/pads. We'll get to that, though – first, let's talk about the design.

The Pi Cap works very much like a standard HAT, sitting on top of your Raspberry Pi and granting you immediate access to more functions through the use of special software. Unlike most HATs, it hangs over the sides of the Raspberry Pi, though this is a deliberate design choice which allows easier access to some of its features. While it's designed with the Pi Zero in mind (the parts of the board that don't overhang fit snugly over the Zero's form factor), it will work on any other Pi model with a 40-pin GPIO.

The board comes pre-soldered so you can use it out of the box. You can put it straight onto a Raspberry Pi from there, although this does require some degree of software setup. The process is well documented on the website (**magpi.cc/2eKcB5C**), and you should be able to get it all set up within half an hour.

Internet of caps

With the Pi Cap on and ready to use, you have access to the aforementioned capacitive touch pads. These are the large gold connectors on the long edge, which can also be used to connect to wires and which are ideal for painting on with the Bare Conductive paint. Next to these is a large prototyping area with a GPIO breakout. There's also a physical button and an RGB LED attached to the board.

It's all really quite appealing. It adds a decent amount of useful functionality for education and for creating some interesting projects (you can find Capong in our last issue, but there are others on the firm's website under Suggested Tutorials: **magpi.cc/2eKcB5C**). We especially like the little breakout area, which is useful in any project. However, the capacitive pads are also excellent and the slightly larger holes in the connectors make them pretty good for wearables as well.

Related

RASPIO PRO HAT

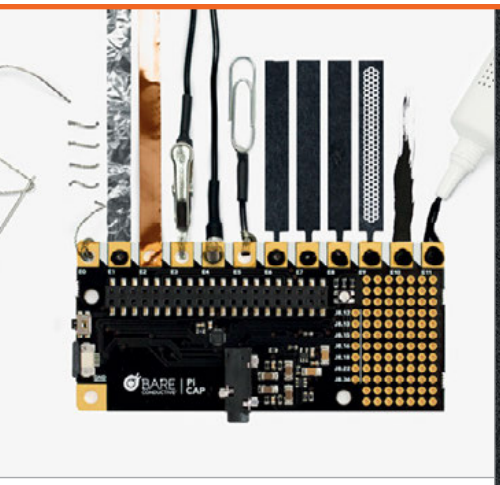
Although it doesn't quite have as many functions as the Pi Cap, it's still a great prototyping board for the Pi.

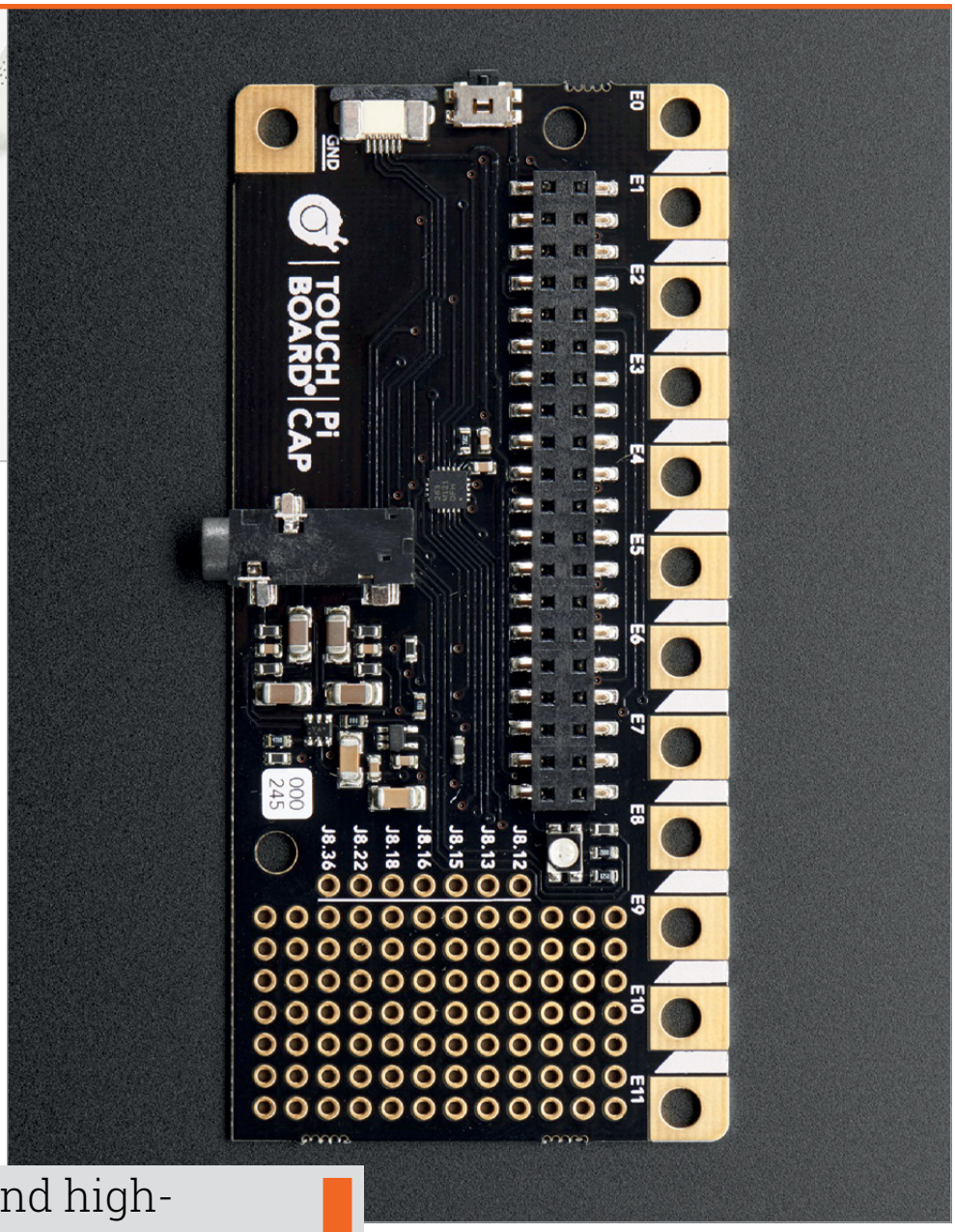
£14 / $18

rasp.io/prohat

magpi.cc/2dDDdXP

£28 / $37





Programming the Cap is pretty simple and you can do so in a number of languages, including the standard Python and C++, although you can also control it via Processing. Again, refer to our Capong tutorial from last issue for a sneak peek into how that works. There's also an interactive introduction and examples that come with the code so you can try some lower-barrier-to-entry tests; they're enough to get your head around what's going on at least.

Thinking cap

We do like the build of the Pi Cap. It's very robust and high-quality, possibly even more sturdy than the Pis themselves, which is quite the accomplishment. All the components are very small and have a very small form factor, meaning you will be unlikely to snap them off easily. There's even a high-quality audio jack installed on the board – it's perfect for the Pi Zero and is a little better than the 3.5mm jack on the Pi 3 as well.

> "It's very robust and high-quality, possibly even more sturdy than the Pis themselves"

We'd love to see an activity kit come with the Pi Cap in the future, with some components and perhaps a book to get you started with some fun projects. At the moment, though, it's still a pretty great board on its own, and maybe something to look into with Christmas coming up.

Last word

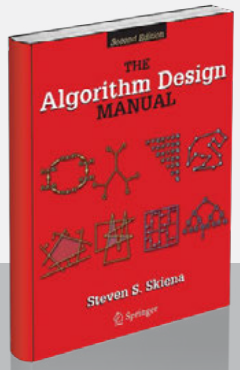
A neat little board with a lot of potential, adding some fairly unique features to the Pi. It would be better in a kit, but it's still great in its own right.

RASPBERRY PI BESTSELLERS

ALGORITHMS

Improve your handling of computational problems with a thorough grounding in algorithms

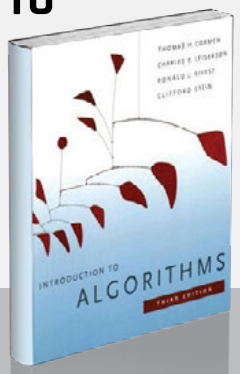
THE ALGORITHM DESIGN MANUAL



Author: Steven S Skiena
Publisher: Springer
Price: £55.07
ISBN: 978-1848000698
magpi.cc/1VDJchi

Skiena tackles algorithms with an engineer's mindset; lacking the mathematical rigour (and page count) of the MIT Press book (see below), it wins out on real-world examples, making a most practical introduction.

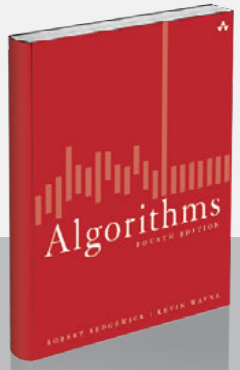
INTRODUCTION TO ALGORITHMS



Authors: Thomas H Cormen, Charles E Leiserson, Clifford Stein & Ronald L Rivest
Publisher: MIT Press
Price: £44.95
ISBN: 978-0262533058
magpi.cc/1VDJt3l

Perhaps the definitive reference work and tutorial on algorithms, but you'll need a good grasp of mathematics, as well as some programming experience, to get the most out of it.

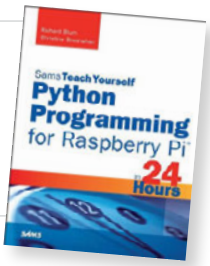
ALGORITHMS



Authors: Robert Sedgewick & Kevin Wayne
Publisher: Addison Wesley
Price: £55.99
ISBN: 978-0321573513
magpi.cc/1VDJrch

Sedgewick's popular book – which has its own two-part Coursera MOOC – has clear explanations and diagrams, and useful Java implementations which should translate to your preferred language (but check errata).

SAMS TEACH YOURSELF PYTHON PROGRAMMING FOR RASPBERRY PI IN 24 HRS



Author: Richard Blum & Christine Bresnahan
Publisher: SAMS
Price: £21.99
ISBN: 978-0672337642
magpi.cc/1VDJyV6

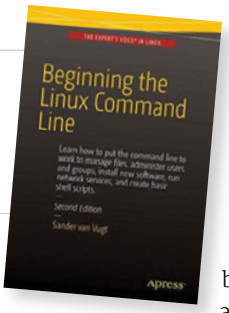
"Making Python the official programming language of the popular Raspberry Pi was genius," write the authors in the introduction. Some measure of astuteness must also be ascribed to Pearson's Sams imprint for realising a couple of dozen tutorial chapters of typical length made the equivalent of a day – albeit a somewhat demanding one – and creating the 'Teach Yourself … in 24 Hours' series. Now Blum and Bresnehan, both experienced sysadmins, have updated and revised their fine introduction to programming in Python on the Pi.

Aimed squarely at those new to both Python and Pi, the first three 'hours' cover setup for both. That done, the next four hours are spent on programming fundamentals, building simple scripts, then advancing to logic flow control. Useful introductions along the way include the NumPy module, iterating over lists, and dealing with errors.

A ten-hour chunk of 'Advanced Python' covers topics like lists and tuples, creating functions, and object-oriented programming, as well as exception handling and an introduction to regular expressions. Next is GUI programming with Tkinter and Pygame, 'business programming' (network, database, and web server), and multimedia and GPIO projects round off a very useful guide. Recommended for any beginner with 24 hours to spare.

Score ★★★★☆

BEGINNING THE LINUX COMMAND LINE



Author: Sander van Vugt
Publisher: Apress
Price: £26.50
ISBN: 978-1430268307
magpi.cc/1VDJPHL

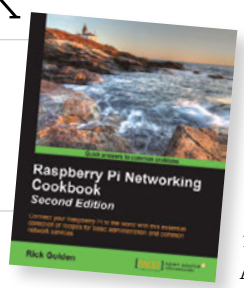
Every user of Raspbian, or any other GNU/Linux distribution, becomes in a small way a systems administrator. Most tasks are best accomplished via the command line, and van Vugt's excellent introduction is very much aimed at practical administration, whether the reader has a Pi, a server they're programming, or they're a Windows sysadmin taking on Linux tasks.

Providing plenty of background along the way – which should help new users understand the 'why' along with the 'how' of the command line – the author starts with several practical commands which give the user useful control of her system, with users, files, and disks all covered well in the first three chapters. The chapter on working with text files starts with mastering vi – the one editor found on all Unix computers. Further chapters are usefully task-oriented.

Steering a middle path between quick introduction and comprehensive reference, *Beginning the Linux Command Line* gives the reader practical information without overwhelming with detail, yet still fills in background where context is necessary for real understanding. The final chapter, on shell scripting, is a good example of the author's well-paced tutorial approach, and leaves the reader prepared for tackling many admin tasks.

Score ★★★★☆

RASPBERRY PI NETWORKING COOKBOOK



Author: Rick Golden
Publisher: Packt
Price: £28.99
ISBN: 978-1785280214
magpi.cc/1VDKs3U

Despite some bottlenecks in networking through Ethernet and USB-connected network interfaces, the Pi 2 is, as Golden points out in his preface, every bit as powerful as a network server from the late 1990s. This being so, despite 20 years of Moore's Law growth for modern network servers, the Pi 2 (and now the Pi 3) can be used for many network tasks to "share files, host websites, create internet access points, and analyse network traffic."

With chapters collecting recipes both by theme and by difficulty, there are entry points in this work for beginner and intermediate users. The early chapters also include enough information on setup, system administration, and maintenance, to enable those new to the GNU/Linux command line to learn the ropes.

Recipes for file sharing enable you set up the Pi as a network file server for home or office. The Advanced Networking chapter covers firewalls, web servers and wikis, wireless access points, and network analysis tools. Internet of Things goes from GPIO setup, through GrovePi, to web.py and controlling devices through a webpage. Lastly, there's computing over the network with clustering – and setting up your own Raspberry Pi "supercomputer". A valuable introduction and reference.

Score ★★★★☆

ADAPTIVE WEB DESIGN

Author: Aaron Gustafson
Publisher: New Riders
Price: £21.99
ISBN: 978-0134216140
adaptivewebdesign.info

It's a full house of second editions this month, and Gustafson's is a welcome update to this high-level yet practical guide to the why and how of adaptive web design. The approach here is more than adding responsive design for mobile browsers. Progressive enhancement is the mantra, building site features on a foundation of content that's accessible to all browsers and users.

From GitHub.io project pages to WordPress blogs, it's almost inevitable that makers end up responsible for websites, and thus for web design, so this could be the book to make you think more seriously about the subject. Aimed at developers, project managers, and web professionals, it's broad enough to be approachable for all, yet deep enough to leave you with a practical idea of what needs to be done to make your site(s) a pleasant and informative experience for all visitors.

Gustafson starts squarely with content – sadly an afterthought for many designers, but the only thing everyone needs from a site. Markup, visual styling, and interactions are then added to a common base of universally accessible content, ensuring each user gets the best site for her browser, device, and screen size. Practical, thoughtful, and a good read, too.

Score ★★★★☆

ESSENTIAL READING: ROBOT BUILDING

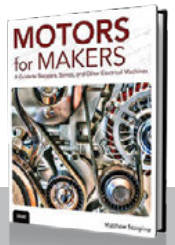
Classics and recent releases to help you get your robotic project moving

Raspberry Pi Robotics Essentials

Author: Richard Grimmett
Publisher: Packt
Price: £19.99
ISBN: 978-1785284847
magpi.cc/1VDKDw8

Build a walking robot and add six degrees of freedom, plus sensors, vision, and path planning.

Motors for Makers



Author: Matthew Scarpino
Publisher: Que
Price: £21.99
ISBN: 978-0134032832
motorsformakers.com

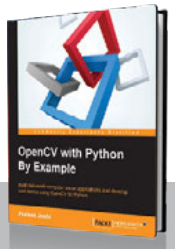
Detailed and practical guide to using motors with the Pi and other boards, with code and vehicle projects.

Robot Operating System (ROS): The Complete Reference

Author: Anis Koubaa (Editor)
Publisher: Springer
Price: £149.00
ISBN: 978-3319260525
magpi.cc/1VDKR6k

27 chapters, both practical and academic, giving comprehensive coverage of open-source middleware for autonomous robots.

OpenCV with Python By Example



Author: Prateek Joshi
Publisher: Packt
Price: £31.99
ISBN: 978-1785283932
magpi.cc/1VDKVmE

Well-regarded guide to the practical side of computer vision algorithms, that enables the reader to understand the problems – and solutions.

I Robot

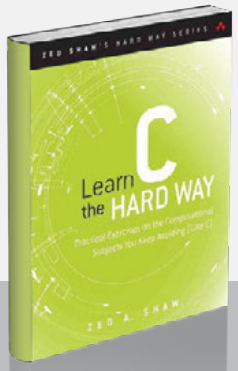
Author: Isaac Asimov
Publisher: Harper Voyager
Price: £7.99
ISBN: 978-0007532278
magpi.cc/1VDL9uf

Robots that think like us are still science fiction, but Asimov's pioneering stories ask essential philosophical (and relevant) questions.

RASPBERRY PI BESTSELLERS

C A new generation of C tutorials for a new generation of programmers

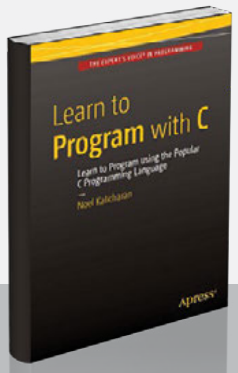
LEARN C THE HARD WAY



Author: Zed Shaw
Publisher: Addison Wesley
Price: £24.99
ISBN: 978-0321884923
magpi.cc/1TQkpD1

Zed Shaw's famous 'type all of this in yourself' method of instilling learning, brought to C. Strongly opinionated but, if you get on with it, a practical introduction to C basics.

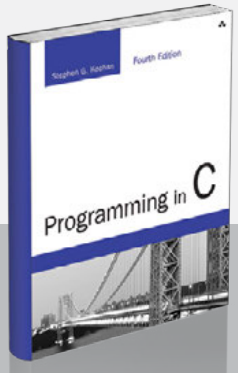
LEARN TO PROGRAM WITH C



Authors: Noel Kalicharan
Publisher: Apress
Price: £17.50
ISBN: 978-1484213728
magpi.cc/1TQkJBu

A comprehensive C tutorial for non-programmers that – despite missing appendices on setting up a compiler – wins friends through the excellent clarity of the author's explanations.

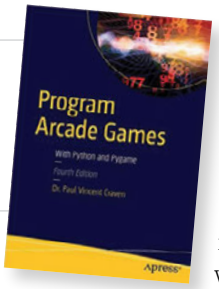
PROGRAMMING IN C, 4TH EDITION



Authors: Stephen G Kochan
Publisher: Addison Wesley
Price: £30.99
ISBN: 978-0321776419
magpi.cc/1TQkIxs

Comprehensive, a classic (now on its fourth edition), and very detailed: serious but readable. If you struggle with pointers, this could be the book to give you that 'aha' moment.

PROGRAM ARCADE GAMES, WITH PYTHON AND PYGAME



Author: Dr Paul Vincent Craven
Publisher: Apress
Price: £29.99
ISBN: 978-1484217894
magpi.cc/1W02yNA

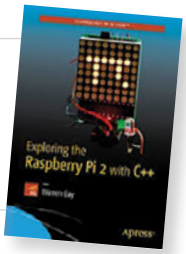
Based on Craven's popular **programarcadegames.com** website, and now in its fourth edition, this book does a superb job of fitting programming concepts – and Python learning – to building several games with the ever useful Pygame library. Programming lessons are carefully introduced throughout each game or game-related project – starting with type correctness in the opening calculator example.

Aimed at younger readers, but accessible to all, Craven's teaching experience shows through in both tone and pace: this is a book that you will not finish quickly, but should keep you motivated through the journey. Along the way, everything from functions and lists, through searching and sorting, to sprites and array-backed grids will be gradually absorbed, until the reader has the skills to write their own games.

There is plenty of code in this book – more than we'd normally expect to see – but Apress's production is excellent here, with pages looking well balanced, and beginners unlikely to be unnerved by the sight of so much Python! Plentiful exercises, including a whole chapter at the end revisiting every project in the book, drive the lessons deep. Well written, well developed, and – despite eschewing the self-conscious whimsy of some other tutorials – very enjoyable to work through.

Score ★★★★★

EXPLORING THE RASPBERRY PI 2 WITH C++



Author: Warren Gay
Publisher: Apress
Price: £29.99
ISBN: 978-1484217382
magpi.cc/1TQkeaS

C++ is a powerful and complex beast, which does little to protect you from your mistakes, and is not something we often see in a Pi project. Nevertheless, if you've learnt the basics, you can build your skills while experimenting with your favourite single-board computer, thanks to this interesting tour of the Pi, its hardware, and some useful C++ programs.

Download the code from **magpi.cc/1TQkoyY** and you'll get useful utilities like gp, which manipulates and reads the GPIO interface, and pipwm, for handling PWM (pulse-width modulation) signals. The powerful pispy captures GPIO data and in the book is used to build an oscilloscope. Practical electronics feature in circuit designs for debouncing input signals.

A C++ primer feels slightly misplaced three quarters of the way in, but you can skip forward to that point at any time. Lastly, a multicore web server highlights the Pi 2's abilities, then leaves the challenge of improving and finishing the project to the reader. Appendices on GPIO classes, and other classes – such as Matrix, which document those used earlier in the book – round off a book which may be essential, if you want to work on your Pi with C++.

Score ★★★★☆

LEARNING PYTHON DESIGN PATTERNS

Author: Chetan Giridhar
Publisher: Packt
Price: £25.99
ISBN: 978-1785888038
magpi.cc/1TQkVRj

When Kent Beck and Ward Cunningham began applying the concept of design patterns, espoused by architect Christopher Alexander, to programming, OOP (object-oriented programming) was still a relatively tiny niche. As it grew over the next decade, design pattern use spread, and basic patterns were collected in the famous Gang of Four book. These proven development paradigms speed up programming, and well-defined components and interfaces make for scalable and maintainable code.

With many of today's intermediate Python programmers not coming from a formal Java or C++ background – and many patterns being workaraounds for the limitations of those two languages – what's needed is an introduction from the perspective of dynamic scripting languages like Python. Giridhar provides this with a practical, Python-based look at several well-known patterns, from Observer to MVC patterns.

Following a useful introduction to OOP concepts and design patterns, we start with the Singleton design pattern, then progress through several patterns – all with Python code, and practical examples, as well as a discussion weighing up the advantages and disadvantages. Lastly, AntiPatterns tells the reader "what we shouldn't do as architects or software engineers." Concise, thoughtful, and a useful stepping stone on your Python journey.

Score ★★★★☆

INTRODUCING GO: BUILD RELIABLE, SCALABLE PROGRAMS

Author: Caleb Doxsey
Publisher: O'Reilly
Price: £16.50
ISBN: 978-1491941959
magpi.cc/1TQkZk9

Go may be a difficult word to Google for, but the language has many positives – bringing 40 years of advances to a C-style language, from concurrency to memory safety. The key feature is simplicity, even at the cost of missing out on features taken for granted in other languages.

To match the Go language's simplicity, this is a concise, efficient introduction to a concise and efficient language, and very much in the tradition of K&R. Indeed, you'll find yourself taking that classic off your shelf to check half-remembered references as you spot them in Doxsey's Go introduction. Types, variables, and control structures make for a gentle start, with the opening chapters accessible for relatively inexperienced programmers. Exercises at the end of each chapter ensure that the reader thinks through each topic covered.

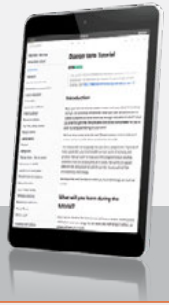
'Arrays, Slices, and Maps' – we were very pleased to see an Oxford comma in the chapter title – and 'Functions' manage to cram a lot of idiomatic Go into explanations which, like the language, know what to leave out. Concentration is required for the pared-back coverage of further Go features, then chapters on packages, testing, and concurrency leave the reader well prepared to tackle some real-world challenges in Go.

Score ★★★★☆

ESSENTIAL READING: DJANGO

The Django web framework is the comprehensive, and Pythonic, answer to your website needs

Django Girls Tutorial



Author: Django Girls
Publisher: Online tutorial
Price: Free
ISBN: N/A
magpi.cc/1WptxT1

From total beginner to fully working blog site with Django – an online version of the Django Girls workshop.

Beginning Django CMS

Author: Nigel George
Publisher: Apress
Price: £22.99
ISBN: 978-1484216705
magpi.cc/1WptBlT

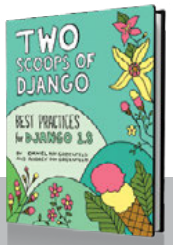
George assumes no Django knowledge, and gets you up and running – a welcome alternative to the occasionally opaque official docs.

Django By Example

Author: Antonio Melé
Publisher: Packt
Price: £28.99
ISBN: 978-1784391911
magpi.cc/1WptIOn

Useful collection of real-world websites, from social sites to online shops. Good integration of other technologies.

Two Scoops of Django



Author: Daniel Roy Greenfeld & Audrey Roy Greenfeld
Publisher: Two Scoops Press
Price: £31.12
ISBN: 978-0981467344
magpi.cc/1WptMho

High-level, low-level, practical, philosophical, opinionated, essential – a reference that makes "Python and Django as fun as ice cream."

High Performance Django

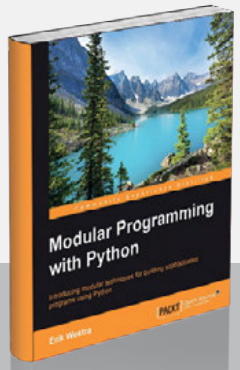
Author: Peter Baumgartner & Yann Malet
Publisher: CreateSpace
Price: £28.23
ISBN: 978-1508748120
magpi.cc/1WptNl3

"A repeatable blueprint for building and deploying fast, scalable Django sites" with lessons beyond Django sites.

RASPBERRY PI BESTSELLERS PACKT PYTHON

The best of Packt Python books will hugely broaden your knowledge

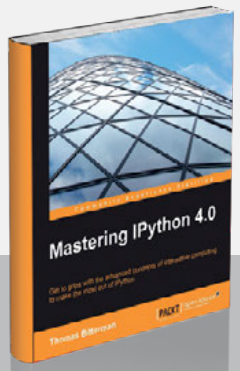
MODULAR PROGRAMMING WITH PYTHON



Author: Erik Westra
Publisher: Packt
Price: £25.99
ISBN: 978-1785884481
magpi.cc/2aUBKho

Get organised with this succinct guide to making your code modular, which takes in Python's extensive import system, testing your modules, and even preparing your modular code for sharing on GitHub.

MASTERING IPYTHON 4.0



Authors: Thomas Bitterman
Publisher: Packt
Price: £31.99
ISBN: 978-1785888410
magpi.cc/2aUC5QZ

Get interactive with IPython, not just as a rich workbook interface to scientific computing, but for developing for parallel and high-performance computing. The book covers testing and working with R, Julia, and JavaScript.

PYTHON: REAL-WORLD DATA SCIENCE

Authors: Dusty Phillips et al.
Publisher: Packt
Price: £49.18
ISBN: 978-1786465160
magpi.cc/2aUBZsl

Packt's comprehensive curated course combines works to give you 1,250 pages of intensive data science learning and practical Python coding, taking in NumPy, Matplotlib, Redis, and MongoDB along the way.

BUILDING THE WEB OF THINGS

Author: Vlad M Trifa & Dominique D Guinard
Publisher: Manning
Price: £21.99
ISBN: 978-1617292682
magpi.cc/2aUEsTC

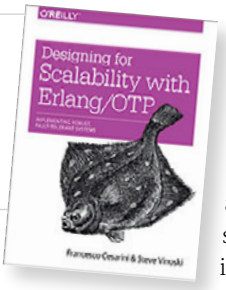
Competing standards and fragmentation – IT's traditional curse – have led to the Internet of Things (IoT) being more a collection of isolated Intranets of Things. Guinard and Trifa's solution is to integrate the fragmented parts with the most successful application layer of them all, the web, using its loose coupling and simply defined programming model as the basis of clean web APIs to build a scalable Web of Things (WoT).

The book, accessible to anyone with basic programming and web skills, is split into two parts. The first introduces the basics: the concept, a device to work on (enter the Raspberry Pi), and using JavaScript and Node.js to glue things together. A hands-on walkthrough in chapter two gets readers comfortable with using the Pi as a remote, web-connected device.

The second section also combines the theoretical and the practical, as APIs and protocols are introduced then used to build interactive WoT projects, and the reader is drawn from data security to scalable physical mash-ups of devices. As long as competing IoT devices and networks can be interacted with through the web, at least through some gateway, all things are possible. This will ready you for tomorrow, while others are still arguing over standards.

Score ★★★★★

DESIGNING FOR SCALABILITY WITH ERLANG/OTP



Author: Steve Vinoski & Francesco Cesarini
Publisher: O'Reilly
Price: £33.50
ISBN: 978-1449320737
magpi.cc/2aUDTta

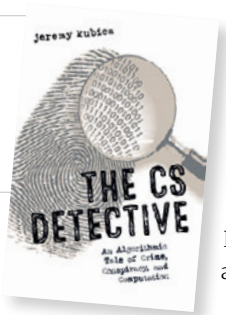
Writing this as a sequel to O'Reilly's *Erlang Programming*, veterans Cesarini and Vinoski deliver the ideal next step to anyone who's completed any introductory work on the language and is ready to tackle a project that demands the distributed language's key benefits: scalability, reliability, and availability.

The introduction helps to define the problem space, and the tools and libraries available, as well as the principles of the OTP environment. It's followed by an Erlang refresher, or an introduction for those brave enough to start their Erlang journey here. Next, design patterns and behaviours: client server examples are developed, broken into parts, packaged into library modules, and migrated to OTP-based generic server behaviour. Then it tackles finite-state machines and event handlers, using a straightforward telephony example.

Next, there's monitoring and handling errors with supervisors, packaged into the building blocks of applications, and then non-standard behaviours and building robust applications. This is hard going for readers, as something of a shift in thinking is involved to turn out programs in such a form, but this book will help you understand the whys and hows of OTP. Treating the full trade-offs of developing, deploying, and working with code in scalable, distributed applications makes up a very useful final section.

Score ★★★★☆

THE CS DETECTIVE



Author: Jeremy Kubica
Publisher: No Starch
Price: £12.99
ISBN: 978-1593277499
nostarch.com/searchtale

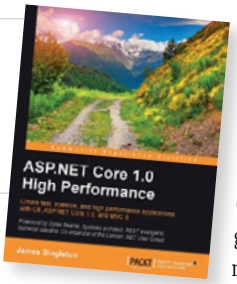
"Meet Frank Runtime. Disgraced ex-detective. Hard-boiled private eye. Search expert." Search expert? Yes, Runtime uses search algorithms, in a novel designed to introduce computational thinking to a wider audience. Although most useful to learner programmers of all ages – each chapter ends with lecture notes on the concepts covered therein – the detective stories are entertaining enough to stand on their own for anyone who'll get some of the references.

Runtime, the loner who doesn't follow the rules, is a familiar figure in fiction, and a Sam Spade-style gumshoe in a pre-industrial world is found everywhere from the 1999 computer game Discworld Noir, to Lindsey Davis's ancient Roman detective Falco. The key to making it work is to keep the humour light and the prose terse, which Kubica does. Take a look at his popular Computational Fairy Tales blog if you'd like a preview of his style.

Thanks to courses like Police Procedures and Data Structures, Runtime is able to find the best search algorithm; everything from best-first and depth-first search, to iterative deepening, parallelising, and binary search, is covered in this entertaining and educational read that should give you enough background to pursue your learning further.

Score ★★★★☆

ASP.NET CORE 1.0 HIGH PERFORMANCE



Author: James Singleton
Publisher: Packt
Price: £34.99
ISBN: 978-1785881893
magpi.cc/2aUG4x2

From 'Why Performance is a Feature', the first chapter, this is a book that encourages caring about how your code performs, to the ultimate benefit of the end user, using profiling to eliminate bottlenecks in C# applications on MS's latest web application framework. Singleton's introduction to getting the best performance on .NET Core 1.0 is not your average web application development book; performance implications of architecture are weighed, with the Raspberry Pi explicitly considered. Yes, the Pi running .NET, and not necessarily with Mono.

.NET Core, unlike traditional Microsoft products, is open-source and cross-platform. In the spirit of this, it's not an MS-centric book; other platforms (Mac, Linux, and of course the Pi), other services (RabbitMQ recommended as far better than Microsoft Message Queuing), and other tools are given a fair examination, and many so-called ALT.NET choices are recommended for working with the new ASP.NET.

After measuring, optimising, and even searching for bottlenecks in the network stack, the author gives a good look at the downsides of your improvements: there are always trade-offs, and the burden of managing complexity and caching and debugging issues is considered. Essential reading for anyone working with ASP.NET Core 1.0.

Score ★★★★☆

ESSENTIAL READING: PROCESSING

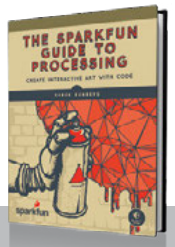
Learn to code with the open-source language designed for the visual arts

Make: Getting Started with Processing

Author: Casey Reas & Ben Fry
Publisher: Maker Media
Price: £17.99
ISBN: 978-1457187087
magpi.cc/2aUHSpE

Very highly regarded introduction to working with Processing, teaching core programming concepts to coding newbies.

The Sparkfun Guide to Processing



Author: Derek Runberg
Publisher: No Starch
Price: £21.50
ISBN: 978-1593276126
nostarch.com/sparkfunprocessing

Project-based intro that oozes creativity, supported by a strong educational framework.

Learning Processing 2nd Edition

Author: Daniel Shiffman
Publisher: Morgan Kaufmann
Price: £30.99
ISBN: 978-0123944436
learningprocessing.com

Well-regarded and comprehensive intro, updated for compatibility with Processing 3 with new chapters on video, sound, data visualisation, and networking.

Welcome to Processing 3

Author: Daniel Shiffman
Publisher: N/A
Price: Free
ISBN: N/A
vimeo.com/140600280

Inspiring look at what's new in Processing 3 (more online resources are linked from **processing.org**).

Processing: A Programming Handbook for Visual Designers

Author: Casey Reas & Ben Fry
Publisher: MIT Press
Price: £55.95
ISBN: 978-0262028288
magpi.cc/2aUL2d4

Covering Processing 2.0 and 3.0, and updated for the new syntax, the definitive reference from Processing's co-founders.

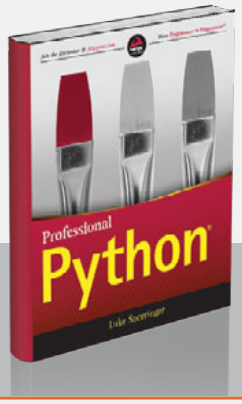
RASPBERRY PI BESTSELLERS

WROX PROFESSIONAL

Take your nascent coding skills up to the professional level with Wrox's popular Professional Guides.

PROFESSIONAL PYTHON

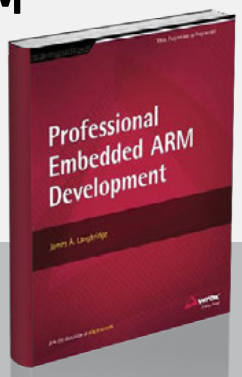
Author: Luke Sneeringer
Publisher: Wrox
Price: £33.99
ISBN: 978-1119070856
magpi.cc/2cCfozG



One of the best intermediate Python books – this one really fills the gaps in your knowledge after your first tutorials and projects. Read our review in *The MagPi* 42.

PROFESSIONAL EMBEDDED ARM DEVELOPMENT

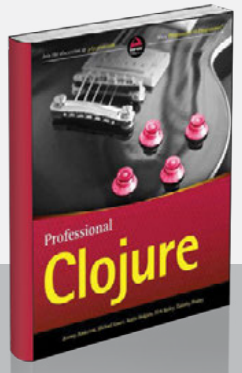
Authors: James A Langbridge
Publisher: Wrox
Price: £33.99
ISBN: 978-1118788943
magpi.cc/2cCfJm8



A comprehensive introduction to Assembly language development on ARM-based boards like the Raspberry Pi, with plenty of background and information on essential tooling, though lacking updates on most recent processors.

PROFESSIONAL CLOJURE

Authors: Jeremy Anderson, Michael Gaare, Justin Holguín, Timothy Pratley & Nick Bailey
Publisher: Wrox
Price: £33.99
ISBN: 978-1119267270
magpi.cc/2cCf3gz



A book that makes you think, from the first chapter's dive into thoughtful code-led examples, and covers web services, testing, and performance. Brings you closer to functional thinking.

RASPBERRY PI USER GUIDE

Author: Eben Upton & Gareth Halfacree
Publisher: Wiley
Price: £16.99
ISBN: 978-1119264361
magpi.cc/2cCfdoi

The unofficial 'official' guide reaches a fourth edition, reflecting the rapid pace of Raspberry Pi development, but remains focused on the opportunity to learn the creative act of programming. Given that aim, the bare-bones board needs a manual, particularly for users and families with little experience of GNU/Linux, coding, or physical computing.

Part I, The Board, will get you started; there's chapters on the range of boards, connecting up, using Linux, troubleshooting (with its own chapter, as befits a serious manual), network configuration, the config tool, and advanced configuration. Only parts of this will be needed by most users, but when you do need to dip into this reference, Tip boxes and the occasional warning combine with clear listings and tables, and just enough screenshots, to help anyone get up and running.

The chapter on setting up a web server has been dropped – possibly reflecting both the diversity of web server options and the simplicity of setting up common configurations – but media centre and productivity chapters remain. The short programming chapters (Scratch, Python, Minecraft) are model introductions, with good pointers to further reading. Hardware and physical computing round off a reference which will become as essential as its three predecessors.

Score ★★★★☆

ELECTRONICS FOR KIDS

Author: Øyvind Nydal Dahl
Publisher: No Starch
Price: £17.99
ISBN: 978-1593277253
magpi.cc/2cCf7wN

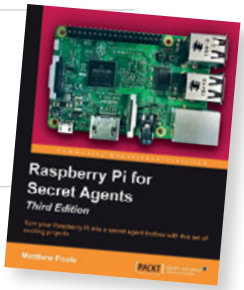
Looking for a book which teaches electronics practically through projects, without neglecting the theory and component knowledge, and does it in a way that will keep young readers interested, without talking down to them? Here it is. In the introduction, Joe Grand talks about the "hacker mindset – solving problems with unconventional solutions, pushing the limits of technology, harming no one, and learning through constant questioning and experimentation." This book will help children to embrace that curiosity.

Its clear and logical progression gives a good grounding in the basics of electronics, reinforced through practical projects. Clear diagrams throughout complement a text full of explanations and project-based learning. The first project, an intruder alarm for your room, is delightfully hands-on, with an aluminium foil strip on the door and a guitar string as trigger wire. Each project features a troubleshooting section.

Electromagnets, motors, shake generators, fruit batteries, destroying an LED, flashing a light, soldering, a transistor-based touch switch, and a sunrise wake-up alarm: these short projects lead you through discrete electronics. Then it explores integrated circuits with 555 timer-based musical projects, digital electronics with simple games, and a great introduction to logic gates and memory circuits, culminating in a reaction game. Enlightening and fun.

Score ★★★★★

RASPBERRY PI FOR SECRET AGENTS



Author: Matthew Poole
Publisher: Packt
Price: £23.99
ISBN: 978-1786463548
magpi.cc/2cCfOWM

Learning by doing goes even better when fun and motivation combine in mini-projects: here the projects are spy gadgets or simple pranks, and this update over the previous edition (reviewed in issue 32) adds in the miniaturised Pi Zero for even better gadgets, as well as the Pi 3. It's an interesting collection of projects, which reaches areas of Linux and computing not touched upon by many similar titles.

The first chapter introduces the Pi and setting up, but jumping to chapter 2 for the audio projects there you get an introduction to the Linux sound architecture, and then using its components for a number of pranks in bite-size projects. Video follows, both with the Pi Camera and controlling a TV through the HDMI interface. Then the Pi goes 'off-road' with some "stealthy reconnaissance missions".

From radio jamming to tracking the Pi, projects are fairly software-centric, and therefore require little in the way of other components until the final chapter's GPIO-based projects, including a laser trip wire and an LED matrix to display secret messages. Along the way, readers can pick up tips about real random number generation, along with useful ways of connecting the Pi to the world, like SMS gateways.

Score ★★★★☆

CREATING BLOGS WITH JEKYLL

Author: Vikram Dhillon
Publisher: Apress
Price: £27.99
ISBN: 978-1484214657
magpi.cc/2cCeUcO

If you're fed up with WordPress plugin problems and slow page loads, you've probably thought about a static blog. Static blogging, with its philosophy of leveraging your coding knowledge and letting almost nothing get in the way of the writing, is a great way of getting back control of your publication platform. This book is about much more than simply setting up a static blog, starting with somewhat laboured chapters of background on how the internet got where it is, but it contains useful material of real interest.

Jekyll's competitors, including the Python-powered Pelican, get a fair examination, and technologies necessary for static blogging – Markdown, Git, and styling tools like Bootstrap, Foundation, SASS, and LESS – are introduced, before Jekyll itself is installed and examined in a concise roundup.

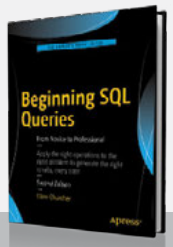
The Projects section presents interesting use cases which walk through the practicalities in a more applied way and, after skimming earlier sections, are probably where most readers will spend their time. This section combines the practical (tags, Git, theming, Mailchimp, gem, Bundler, etc.) with food for thought on platforms for open debate, open research, and open healthcare. Walk through the examples to gain a real appreciation of Jekyll's potential.

Score ★★★★☆

ESSENTIAL READING: SQL

Relational databases remain the essential but unglamorous workhorses behind the Web of Things

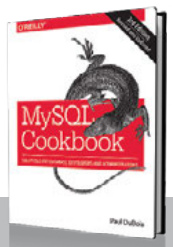
Beginning SQL Queries: From Novice to Professional



Author: Clare Churcher
Publisher: Apress
Price: £19.99
ISBN: 978-1484219546
magpi.cc/2cCgwmY

Great coverage of all the key topics in querying SQL databases, in a reasonably beginner-friendly style.

MySQL Cookbook



Author: Paul DuBois
Publisher: O'Reilly
Price: £56.99
ISBN: 978-1449374020
magpi.cc/2cCfpUt

DuBois gives the right balance between great examples and clear explanations of the theory behind them.

Introduction to Databases

Author: Jennifer Widom
Publisher: Stanford OpenEdX
Price: Free
ISBN: N/A
magpi.cc/2cCh3Fu

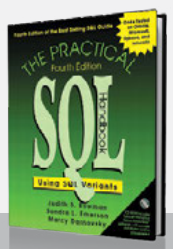
One of Stanford's three inaugural MOOCs, now split into self-paced 'mini-courses', covering different aspects of databases.

PostgreSQL Server Programming

Author: Usama Dar, Hannu Krosing, Jim Mlodgenski & Kirk Roybal
Publisher: Packt
Price: £30.99
ISBN: 978-1783980581
magpi.cc/2cCguLO

Very good guide to working with PostgreSQL in Python, Perl, Tcl, C, and C++, as well as PLpgSQL.

The Practical SQL Handbook



Author: Sandra L Emerson, Judith S Bowman & Marcy Darnovsky
Publisher: Addison Wesley
Price: £43.99
ISBN: 978-0201703092
magpi.cc/2cCg1tk

This edition – after 15 years in print – remains one of the best SQL references, whatever implementation you use.

The MagPi ESSENTIALS

LEARN | CODE | MAKE

OUT NOW IN PRINT

ONLY £3.99

from

store.rpipress.cc

The MagPi ESSENTIALS — From the makers of the official Raspberry Pi magazine

GET THEM DIGITALLY:

Available on the App Store

GET IT ON Google Play

THE *Official*

RASPBERRY PI PROJECTS BOOK

VOLUME 3

The Official Raspberry Pi projects book is back again with incredible projects, guides, and product reviews based around the microcomputer phenomenon that is the Raspberry Pi. See why educators and makers adore the credit card- sized computer that's used to make robots, retro consoles, and even art.

200 PAGES OF PI

- Step into the world of coding with Raspberry Pi
- Be inspired by amazing community projects
- Make your own creations with incredible guides
- Learn about the very best accessories for your Pi

raspberrypi.org/**magpi**