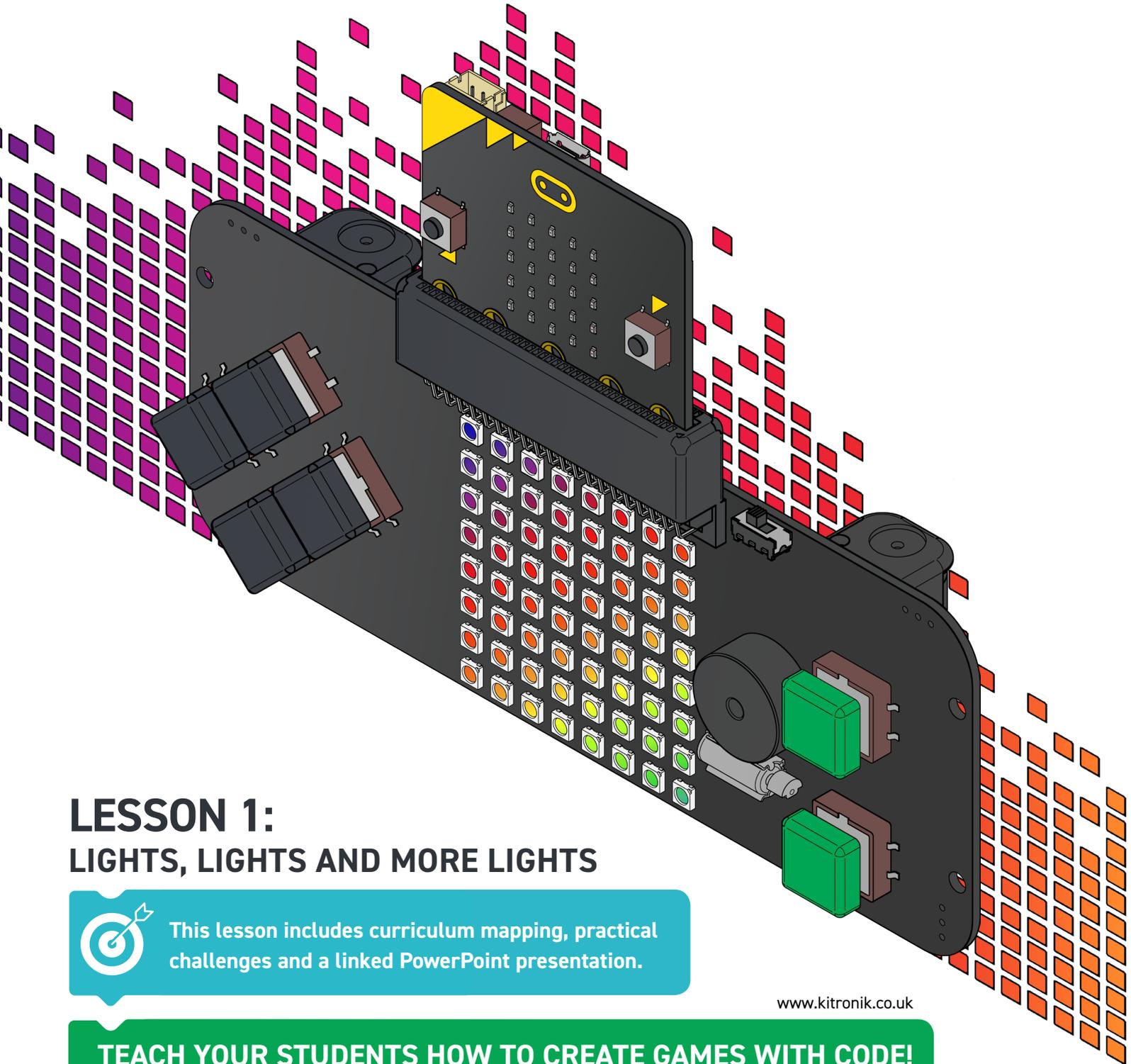


THE TEACHERS LESSON GUIDE TO THE :GAME ZIP 64



LESSON 1: LIGHTS, LIGHTS AND MORE LIGHTS



This lesson includes curriculum mapping, practical challenges and a linked PowerPoint presentation.

www.kitronik.co.uk

TEACH YOUR STUDENTS HOW TO CREATE GAMES WITH CODE!

LIGHTS, LIGHTS AND MORE LIGHTS



This is an introductory lesson for students in Key Stage 3 to the Kitronik :GAME ZIP64 controller. The lesson involves a discussion about colours through light and how we make different colours. Students will create their own 8x8 character and choose its colours using RGB codes. They will then code the micro:bit to display this character on the ZIP controllers. Recommended ratio of students to :GAME ZIP64 is 2:1.

Classroom setup

Students will work be working in pairs. They will need:

- Pen & Paper including coloured pens (grid paper)
- A computer/laptop with a USB port and Internet access
- A :GAME ZIP64
- A BBC micro:bit
- 3 x AA batteries
- A micro USB cable

The teacher will be writing on the board as well as demonstrating code on a projected board (if available).

Timings

The lesson is expected to take 1 hour. It can be shortened or lengthened if needed.

Suggested shortening

Remove Challenge 4 and/or 5: Using loops to draw lines. If students want to focus on creating their characters you can leave the loop coding until another week. It is good for students to learn the slow way of coding so they appreciate the efficiency of adding loops.

Suggested lengthening:

Ask the students to create a character that can be moved. E.g. many of them may have drawn a head given the limited number of pixels. But what about a stick man? So the next lesson they can try and animate the stick man moving. Ask the students to draw the stickman static then the stickman moving one leg, another leg, move an arm?

Differentiation

For younger or less able students there is differentiation in the lesson plan.

The controller can be controlled using the NeoPixel and digital read blocks described below. There are simpler custom blocks available to use. See www.kitronik.co.uk/blog/game-zip-64-makecode-blocks/ for more details.



KEY



Teacher asks this question to the class and discusses the answers with them.



Where this content maps onto the curriculum.



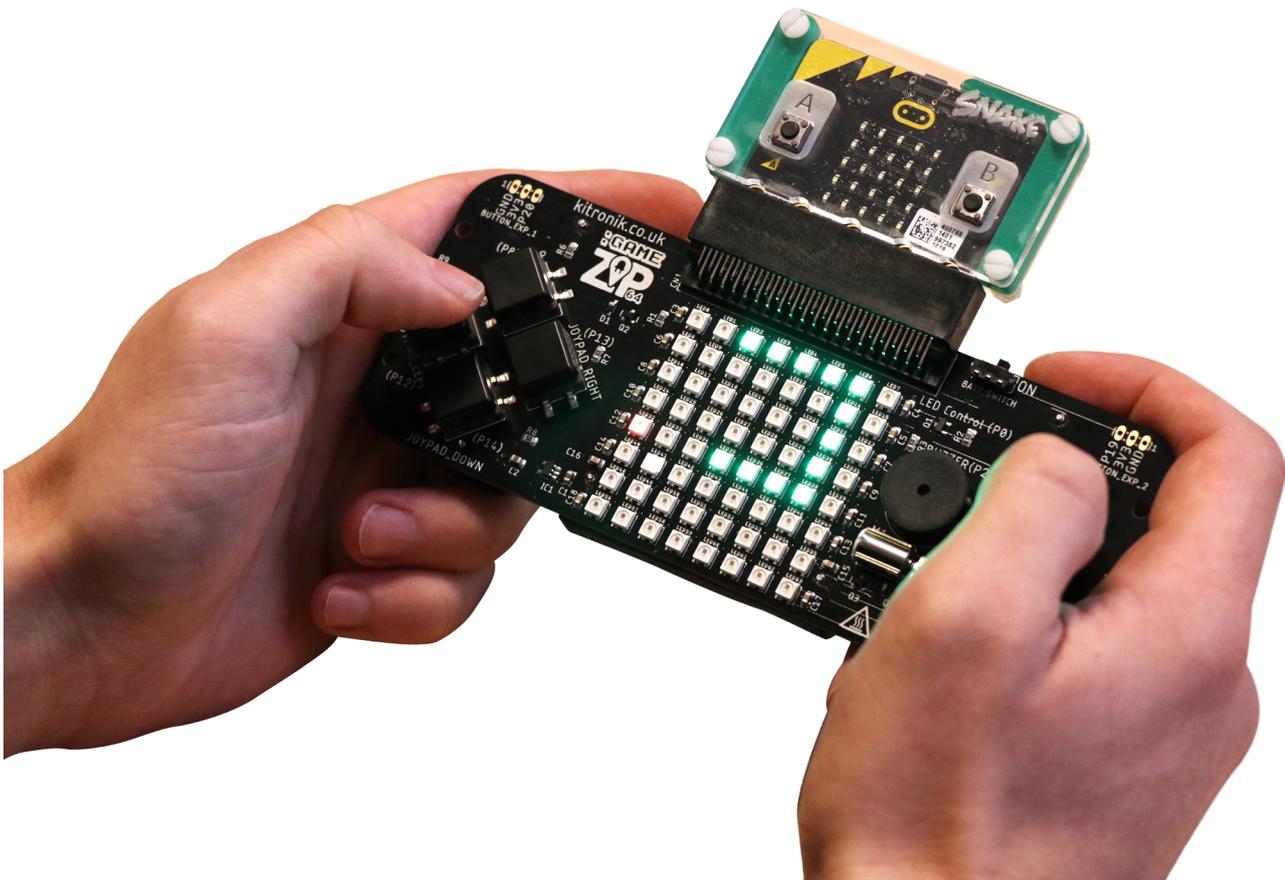
Teacher writes the text in this box on the board.



Students take notes, either on paper or in their electronic workbooks.



This part of the lesson aligns to Slide 1 of the attached PowerPoint.



LIGHTS, LIGHTS AND MORE LIGHTS

INTRODUCTION



We are going to build a game using a new piece of hardware. We will build up the skills over the next few weeks for you to be able to create this game in pairs (Assign pairs). In this lesson we will be controlling the lights on the :GAME ZIP64 controller.

Curriculum mapping
Design, use and evaluate computational abstractions that model the state and behaviour of real world problems and physical systems.



The Controller

Use the PowerPoint to describe the different parts on the controller
OR

Draw the controller on the board and label the different parts:

- micro:bit
- 2 x fire buttons
- 4 x direction buttons
- 64 red, green, blue LEDs
- Vibrating motor
- Buzzer



Students: Pick up a controller and look closely at a single light. Can you see three different squares inside each light? They are very tiny! Each one is a different colour: Red, Green and Blue.



The micro:bit sends 3 numbers to each light, a value between 0 and 255, for each colour.



What colour do you think this is?

RED	GREEN	BLUE
255	0	0

Answer: Red



What colour do you think this is?

RED	GREEN	BLUE
0	0	255

Answer: Blue



What colour do you think this is?

RED	GREEN	BLUE
255	255	0

Answer: Yellow

LESSON 1

LIGHTS, LIGHTS AND MORE LIGHTS



What colour do you think this is?

RED	GREEN	BLUE
100	0	0

Answer: A less bright red!

On screen this would be a darker red. But in lights it just means less red which is less bright.



How many colours can we create?

Answer:

$256 \times 256 \times 256 = 16,777,216!$

Main Lesson

Let's get coding!



Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems.

Let's see what these lights look like on the zip.



To code the lights we need to know more about them.



- How are they connected to the micro:bit?
Look at your controller – what does it say?
Hint: we're looking for something like Pin1, Pin2?

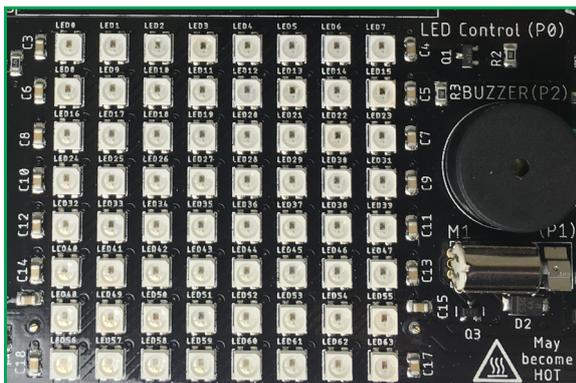
Answer: P0

(Knowing the different pins for the different parts of the controller will become important later)

- How many of them are there?

Hint: look closely at the lights again. Each one is numbered (this will become more important later)

Answer: 64 (not 63! Because we count the number 0 in computing there are 64 lights numbered 0 to 63)



Continued on next page >

LIGHTS, LIGHTS AND MORE LIGHTS

1. Ask the students to create an RGB code, e.g. 255, 0, 0 is red. Ask them to be a bit more imaginative that just one solid colour!
2. Go to <http://makecode.com>, select micro:bit blocks and demonstrate adding and setting up the :GAME ZIP 64:

Add the neopixel package

1. Select Advanced > Add Package > Select NeoPixel

Setup the neopixel grid

2. Under Variables select "Make a variable" and give it a name, e.g. matrix
3. Under Variables drag out "set item to" and add it under "on start"
4. Under NeoPixel drag out "NeoPixel at pin P0 with 24 leds as RGB (GRB format)" and add this to the end of "set item to"
 - a. Change item to matrix
 - b. Change 24 to 64
 - c. Note how it's already at P0

```

on start
  set matrix to NeoPixel at pin P0 with 64 leds as RGB (GRB format)
  
```

Write the algorithm on the board:



Algorithm

When I press button A: All the lights will go red.

This is how we change all the lights to a set colour:

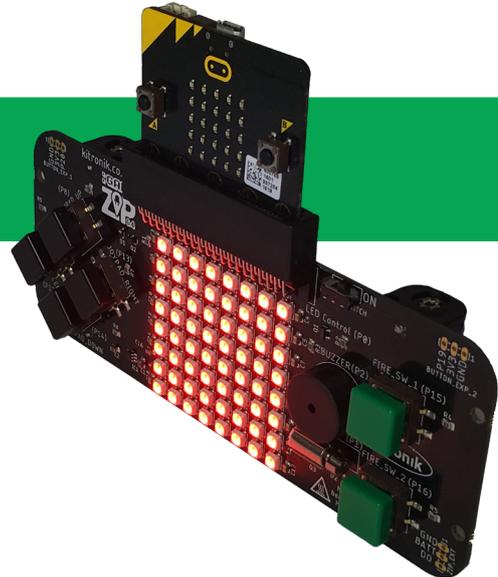
```

on button A pressed
  matrix show color red
  
```

Under Neopixel select More and add this block on top of red:

```

on button A pressed
  matrix show color ( red 255 green 255 blue 255 )
  
```



LIGHTS, LIGHTS AND MORE LIGHTS

Challenge 1



Algorithm

When I press button A: All the lights will go to an RGB code that I've chosen.

Give the students a few minutes to create the code. Don't let them download it to the micro:bit just yet.

Challenge 2



Algorithm

When I press button B:
 Light 1 will go red
 Light 17 will go blue
 Light 25 will go green
 Light 32 will go a different colour

They need this block under Neopixel > More:

```

on button B pressed
  matrix set pixel color at 0 to red
  matrix show
  
```

IMPORTANT: "set pixel colour" just sets the grid. Nothing will display until they add the block "show" afterwards. Ask the students to complete Challenge 2 and download the program to the micro:bit and test their 2 challenges.

Common Problems

They set the NeoPixels up correctly under the variable matrix but are trying to show under the variable item, e.g.

```

on start
  set matrix to NeoPixel at pin P0 with 64 leds as RGB (GRB format)

on button B pressed
  matrix set pixel color at 0 to red
  
```

This will not work - they did not add the block "matrix show" after setting all their colours.

LIGHTS, LIGHTS AND MORE LIGHTS

Challenge 3

Let's create our own characters. We have an 8 x 8 grid of lights to fill. What can we draw?



Character

Display the samples from PowerPoint
OR

Draw a simple smiley face in an 8x8 grid



Ask the students to draw out their own 8x8 character on some grid paper using coloured pens. Ask them to find the RGB codes of the colours they wish to use and create a key.

Ask the students to code their character on the micro:bit.

Give the students a few minutes to start then interrupt with a possible solution below:

Challenge 4

While drawing their characters students may draw some straight lines. Explain how we can save them time by coding these straight lines using variables and a loop.

Example:

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

We want to colour in the line 24 to 30 yellow

Algorithm 1:

Set pixel 24 to yellow
Set pixel 25 to yellow
Set pixel 26 to yellow
Set pixel 27 to yellow
Set pixel 28 to yellow
Set pixel 29 to yellow
Set pixel 30 to yellow



Algorithm 2:

Set temp to 24
Loop 7 times:
Set pixel temp to yellow
Change temp by 1



LIGHTS, LIGHTS AND MORE LIGHTS

Code:

This code colours in lights 10 to 13.



```

on button A pressed
  set temp to 10
  repeat 4 times
    do
      matrix set pixel color at temp to red
      change temp by 1
      matrix show
  
```

Challenge 4

The loop above works for horizontal lines. To draw vertical lines the students just need to change temp by 8. Ask the students to use loops and variables to create lines in their drawings.

After they've finished creating their characters:



Ask the students to take notes.

They will need to keep a record of their character drawing (take a picture or put it in their folders)
Take screenshots of their code and be able to explain it.

Summary

Student learnt about lights and how different colours are created. They created their own colour and coded it on the micro:bit. They designed and created their own character and coded it on the micro:bit. Some will have used variables and loops to code their character.

Next week the students will use the buttons on the controller to control the lights.



This is an introductory lesson for students in **Key Stage 3** to the **Kitronik :GAME ZIP64** controller for **BBC micro:bit**. The lesson involves a discussion about colours through light and how we make different colours. Students will create their own 8x8 character and choose its colours using RGB codes. They will then code the micro:bit to display this character on the ZIP controllers. The recommended ratio of students to :GAME ZIP 64 is 2:1. Please find additional lesson plans, accompanying **PowerPoint** presentations and more at www.kitronik.co.uk/5626.

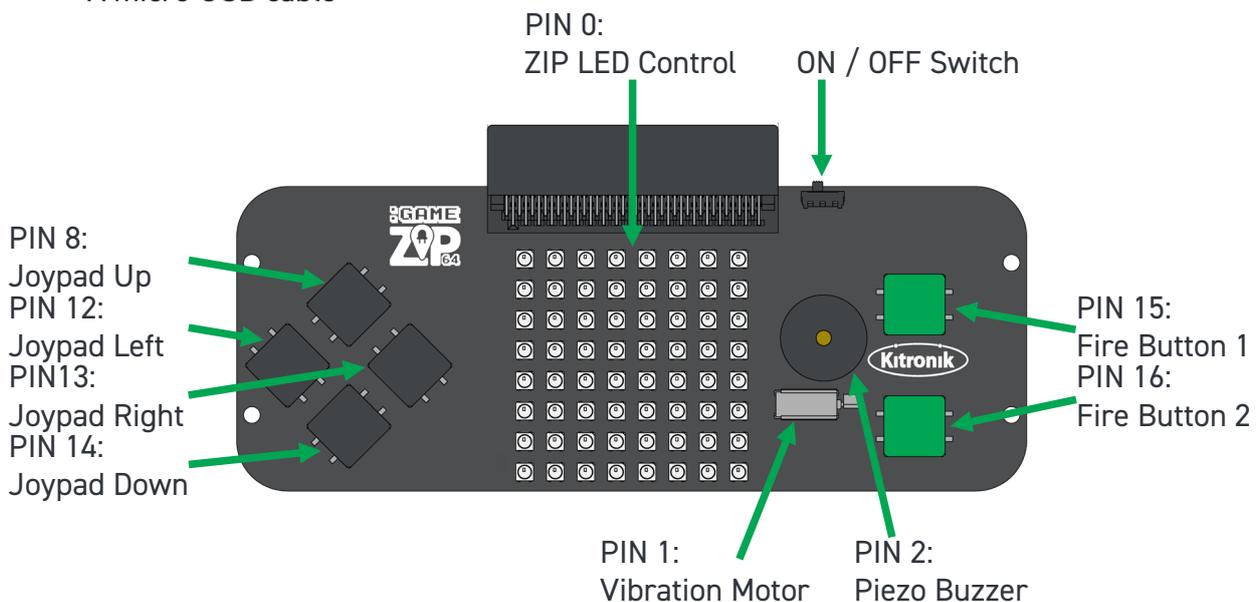
The Kitronik :GAME ZIP 64 is the ultimate retro gaming accessory for the BBC micro:bit. This all in one hand held gaming platform includes a built in 64 (8x8) individually addressable full colour ZIP LED screen.

It features on-board sound, 4 directional buttons, 2 fire buttons, and haptic feedback. All features are fully programmable. Breakout points are also included allowing for shoulder buttons or I2C devices to be added, as well as the use of additional LEDs.

All the BBC micro:bit features are still available when plugged in to the :GAME ZIP 64, so your games can still make use of the LED matrix, accelerometer etc.

LESSON REQUIRES:

- Pen & Paper including coloured pens (grid paper)
- A computer/laptop with a USB port and Internet access
- A :GAME ZIP64
- A BBC micro:bit
- 3 x AA batteries
- A micro USB cable



WARNING : Contents may inspire creativity

T: 0845 8380781

W: www.kitronik.co.uk

E: support@kitronik.co.uk

🇬🇧 Designed & manufactured
in the UK by



[kitronik.co.uk/twitter](https://twitter.com/kitronik)



[kitronik.co.uk/facebook](https://www.facebook.com/kitronik)



[kitronik.co.uk/youtube](https://www.youtube.com/kitronik)



[kitronik.co.uk/google](https://www.google.com/kitronik)