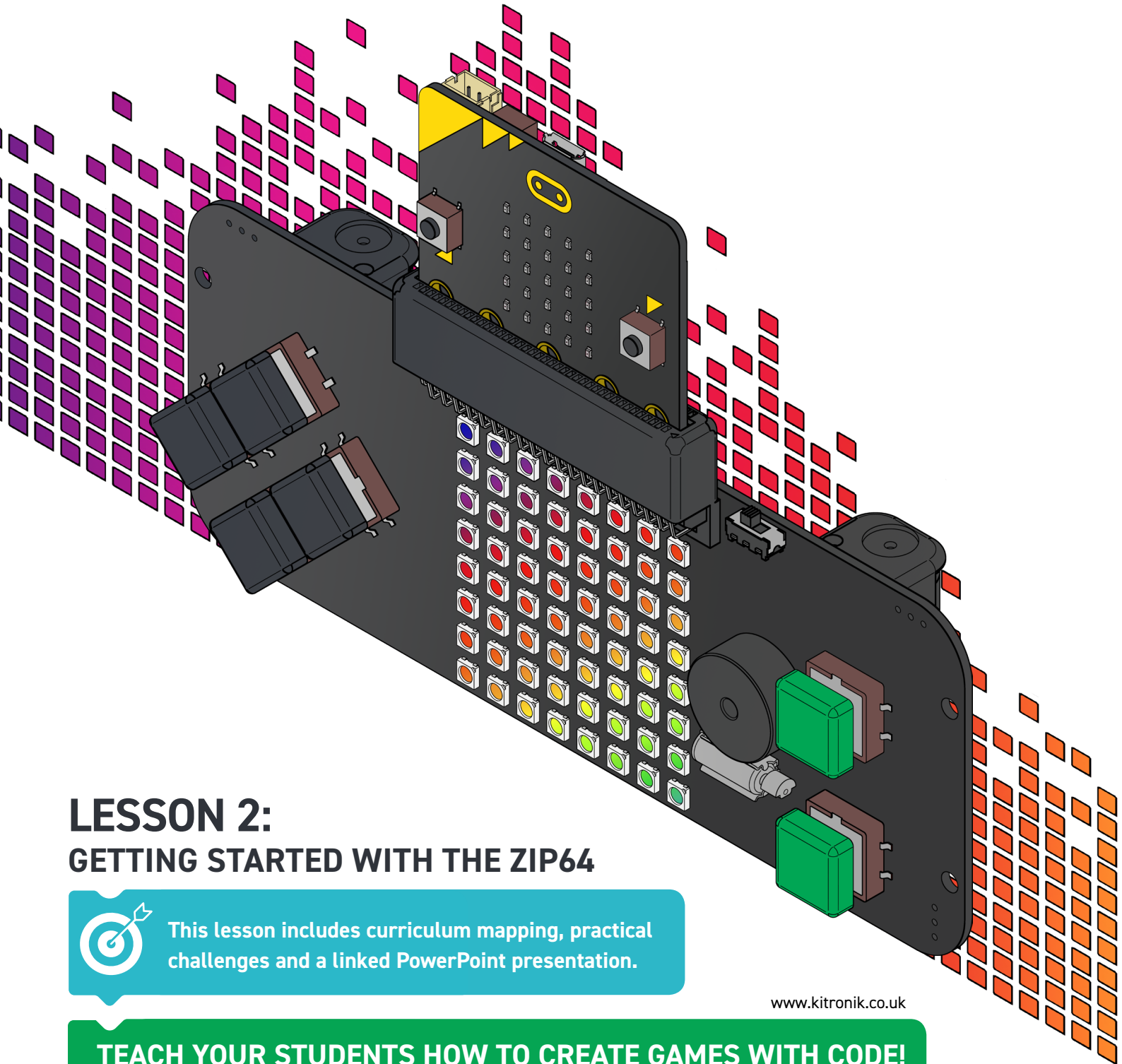


# THE TEACHERS LESSON GUIDE TO THE :GAME ZIP 64



## LESSON 2: GETTING STARTED WITH THE ZIP64



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

[www.kitronik.co.uk](http://www.kitronik.co.uk)

**TEACH YOUR STUDENTS HOW TO CREATE GAMES WITH CODE!**

# GETTING STARTED WITH THE ZIP64



This is the second lesson for students in Key Stage 3 to the Kitronik :GAME ZIP64 controller. The lesson involves using the buttons on the ZIP controller alongside the lights. It also uses the buzzer and motor on the controller.

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 Output challenges 1 – 4. Just do challenges 5 and 6 where the buzzer and motor are turned on and off in the same button press. This will also save your ears as the buzzer keeps going forever in Challenge 1 until they've done Challenge 2!

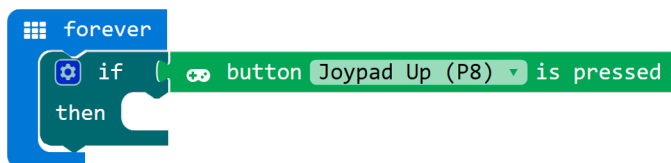
## Suggested lengthening:

Add multiple actions to the button presses, e.g. when I press Fire Button 1 the buzzer sounds, the vibrating motor goes off and the matrix displays a smiley face. Then it all clears after 5 seconds. The more blocks they add to a button press the better as they will be using techniques to shorten these sequences later. So, again, the slower work they do now, the more they will appreciate efficient coding techniques.

## 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/](http://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.



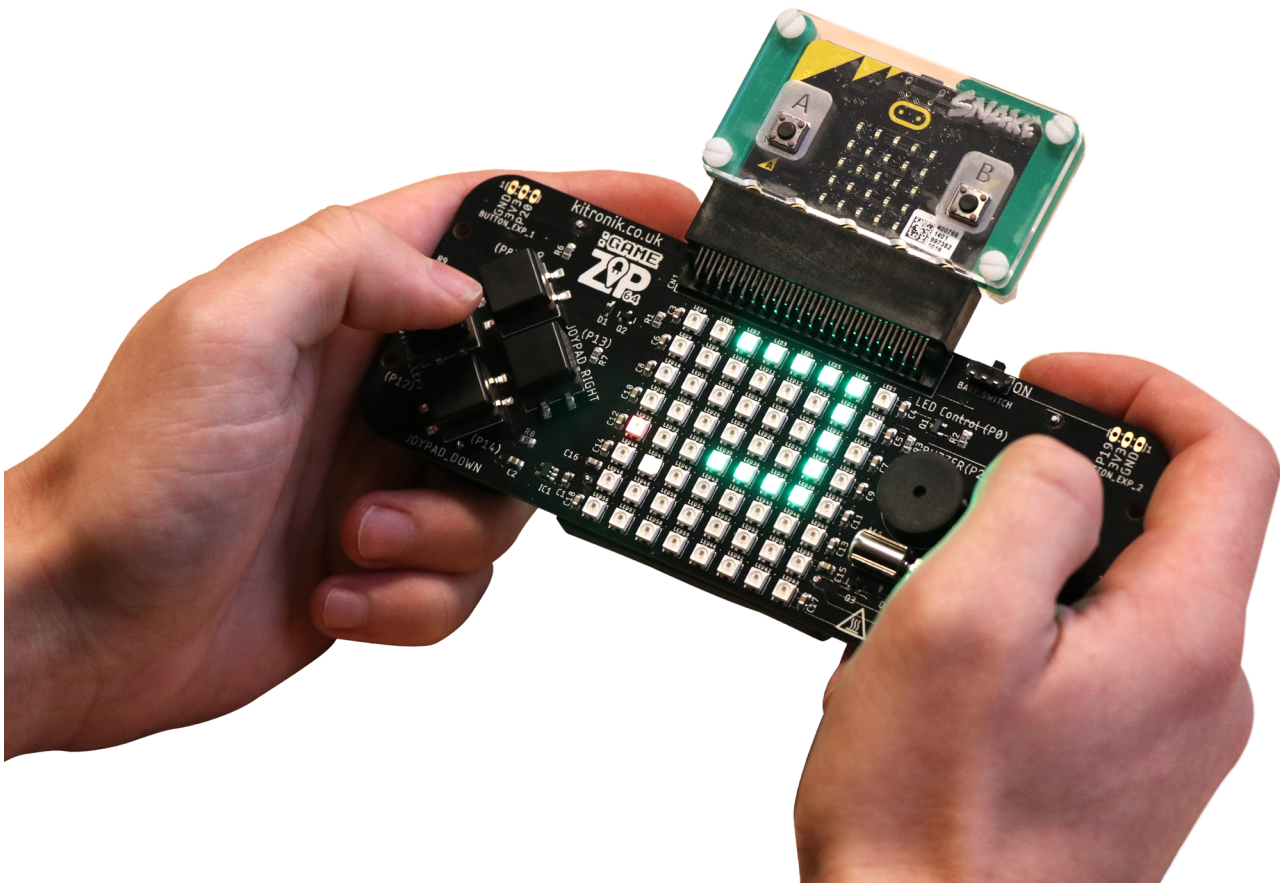
Teacher writes the text in this box on the board.



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



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



# GETTING STARTED WITH THE ZIP64

## INTRODUCTION



This week we're going to look at the buttons on the zip controller.  
You will continue to work in pairs (reminder of pairs if needed).



### Curriculum mapping

Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems.



### The Controller

Use the PowerPoint to look closer at 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 the buttons. Write down what they are called and how they are connected.



Write this grid on the board (or display the slide of the presentation) and ask students for their answers.

Button	How it's connected
Joypad Up	P8
Joypad Down	P14
Joypad Left	P12
Joypad Right	P13
Fire SW 1	P15
Fire SW 2	P16



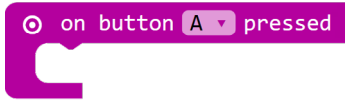
Students to record the Pins in their notes/workbook.



## LESSON 2

# GETTING STARTED WITH THE ZIP64

The micro:bit knows when a button is pressed. Just like button A press we did last week:



Let's change the lights on the ZIP64.



### Algorithm

When I press Fire SW 1:  
Make all the lights go blue.



We will use the other buttons as well, let's just get used to these ones first.

## Main Lesson

### Let's get coding!



### Curriculum mapping

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

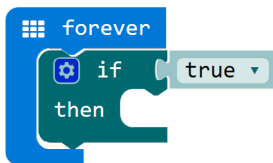
Demonstrate how to detect button presses using blocks:



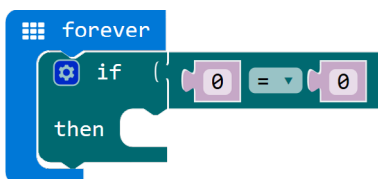
From Basic:



From Logic:



From Logic:

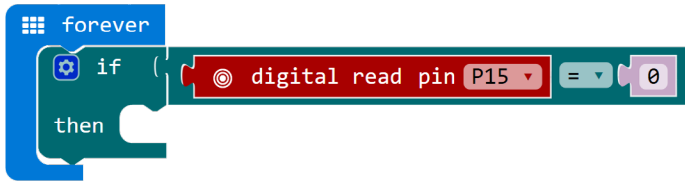


Continued on next page >

## LESSON 2

# GETTING STARTED WITH THE ZIP64

From Advanced > Pins:



From Neopixel:



## Challenges



Once you've demonstrated how to detect button presses using blocks, give the students some challenges to test their ability to control the Buttons on the controller. Perhaps keep the animation of how you added the digital read blocks and the logic blocks going on the screen.



### Algorithms

#### Input Challenge 1

When I press Fire SW 1:  
Clear the 64 lights matrix  
Show a smiley face on the micro:bit

#### Input Challenge 2

When I press Fire SW 2:  
Clear the micro:bit  
Show your character from last week

#### Input Challenge 3

When I press Up:  
An up arrows shows on the micro:bit

#### Extension

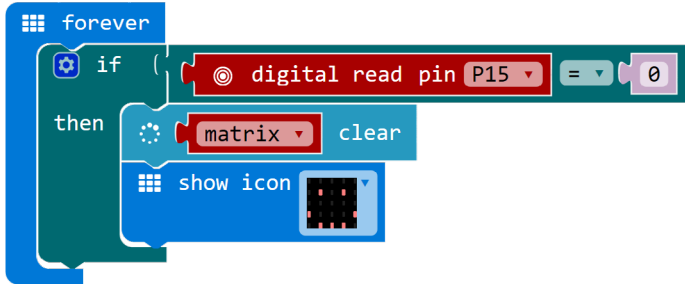
When I press Down: A down arrow shows  
When I press Left: A left arrow shows  
When I press Right: A right arrow shows

Give the students a lot of time to create the code. Challenge 3 is using new buttons. Encourage the students to refer to their table of buttons they wrote down earlier for Pin numbers.

# GETTING STARTED WITH THE ZIP64

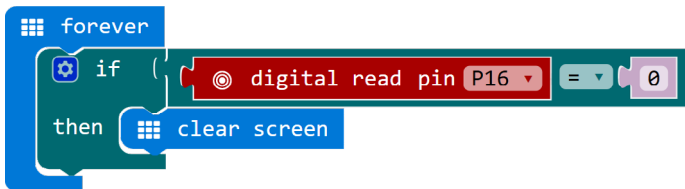
Answers:

Input Challenge 1:

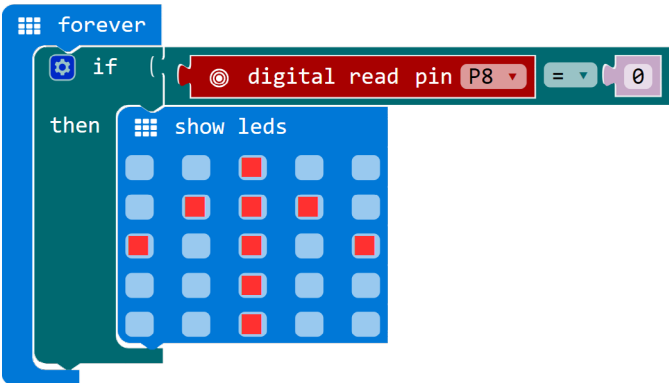


Input Challenge 2:

They will need to recreate their character from last week (or a new one).



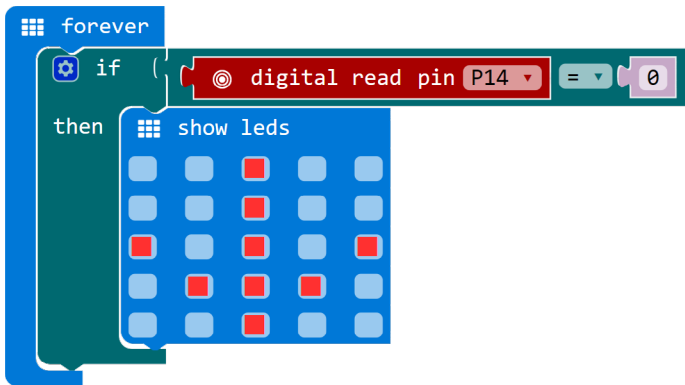
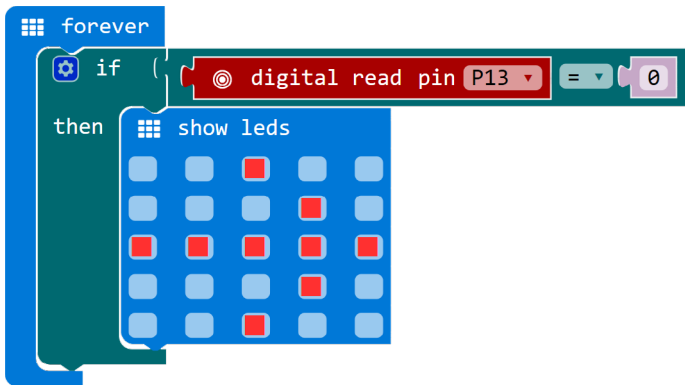
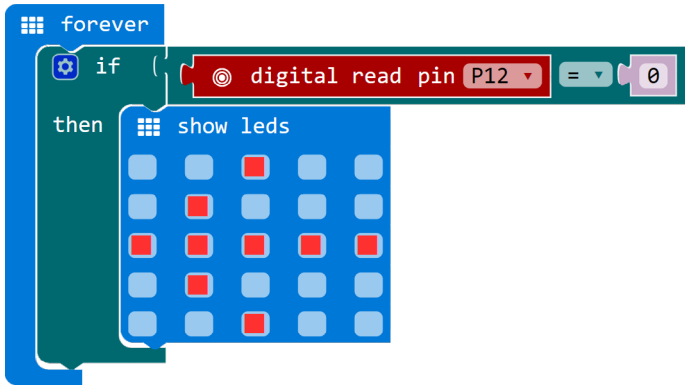
Input Challenge 3:



Continued on next page >

# GETTING STARTED WITH THE ZIP64

Extensions:



There are two more items on the board we haven't used yet. What are they?

**Answer:** The buzzer (P2) and the vibrating motor (P1).



Write these on your table below the buttons.



Students to record the Pins in their notes/workbook.

# GETTING STARTED WITH THE ZIP64



These are **OUTPUTS** on the controller.  
The buttons are inputs.

How do we think we control the outputs?

**Hint:** How did we control using the buttons? Digital read input.

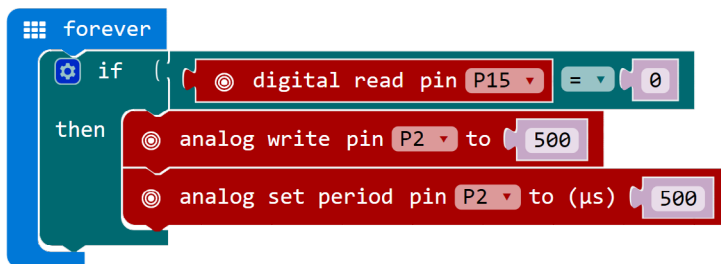
**Answer:** Digital write output? Yes and kinda yes!

The motor is a digital output. So setting its output to 1 will turn it on and setting its output to 0 will turn it off.

The buzzer is a digital device but we use analogue code to control it. The mechanical setup of this buzzer means you must send multiple signals to it for it to make a sound. The crystal expanding when current runs through it will only make 1 click. To make a constant sound we need to keep sending signals. To do this we use analogue blocks.



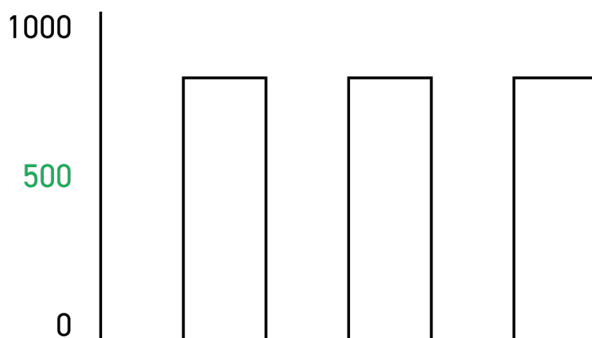
This code sets the buzzer going when Button Fire\_SW\_1 is pressed (challenge x below).



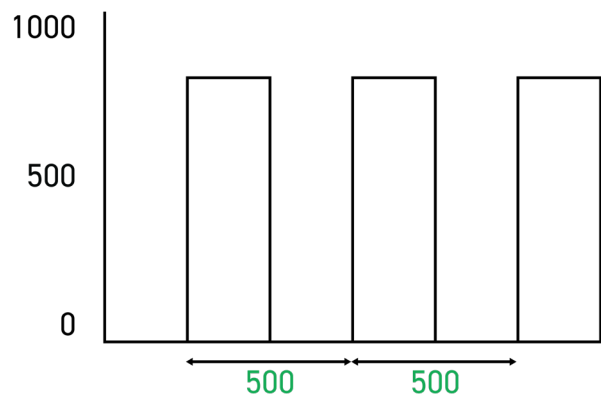
The first block is the average pulse being sent to the buzzer so it ensures it spends as much time on as it does off.



The second block is the time between each on/off.



analog write pin P2 to 500



analog set period pin P2 to (µs) 500



# GETTING STARTED WITH THE ZIP64

Challenge the students to turn on the buzzer/vibrating motor.



## Algorithms

### Output Challenge 1

When I press Fire Button 1:  
The Buzzer will sound

### Output Challenge 2

When I press Fire Button 2:  
The Buzzer will stop

### Output Challenge 3

When I press Up:  
The vibrating motor will start

### Output Challenge 4

When I press Down:  
The vibrating motor will stop

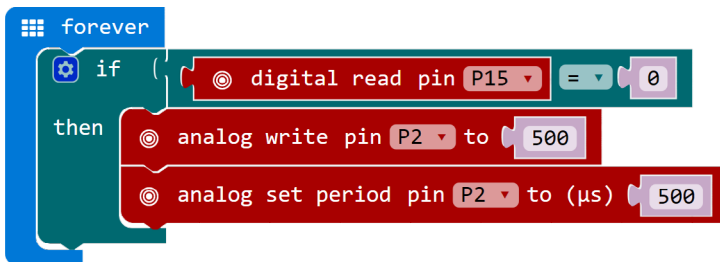
### Extension 1

When I press Left  
The Buzzer will sound for 3 seconds

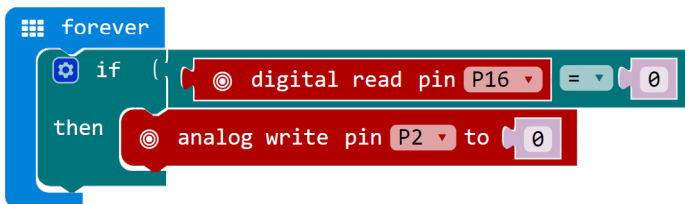
### Extension 2

When I press Right:  
The Motor will run for 3 seconds

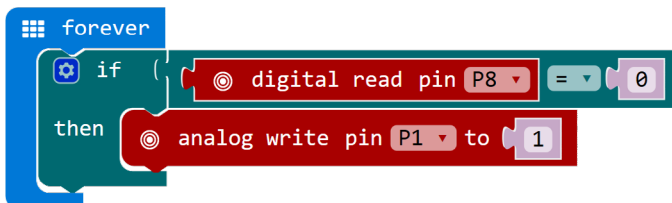
### Output Challenge 1:



### Output Challenge 2:



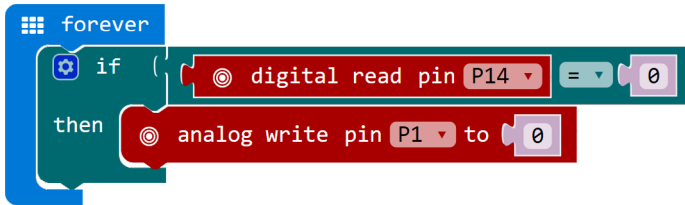
### Output Challenge 3:



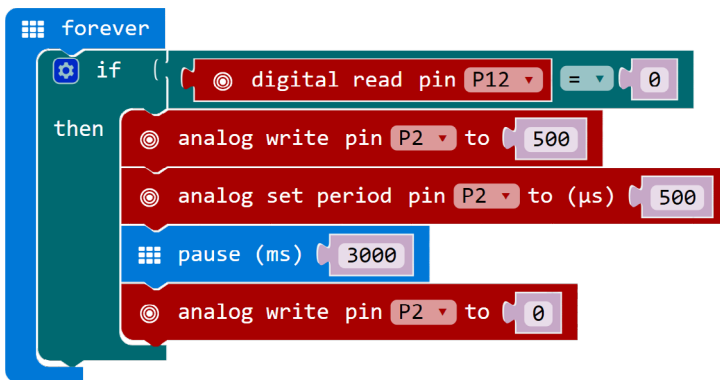
Continued on next page >

# GETTING STARTED WITH THE ZIP64

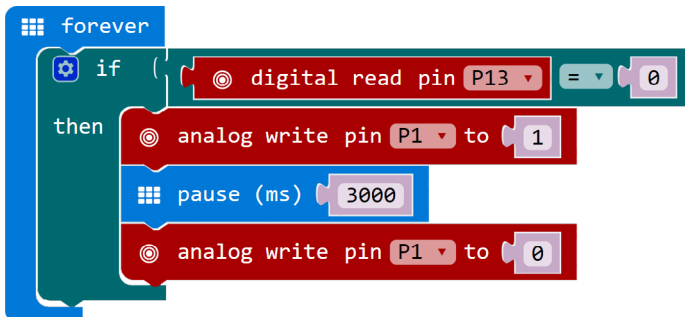
## Output Challenge 4:



## Output Extension Challenge 1:



## Output Extension Challenge 2:



Before the end of the lesson:



Ask the students to take notes.

They will need a copy of the algorithm and their code for each challenge they've completed today. Take screenshots of their code and explain it.

## Summary

Students learnt how to read and write digital input using the buttons and motor on the ZIP controller. They learnt about analogue output and the difference between digital and analogue outputs by coding them differently. The students are building up the skills they will need to create a game on the controller.



This is the second lesson for students in **Key Stage 3** for the **Kitronik :GAME ZIP64** controller for **BBC micro:bit**. The lesson involves using the buttons on the ZIP controller alongside the lights. It also uses the buzzer and motor on the controller. 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](http://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.

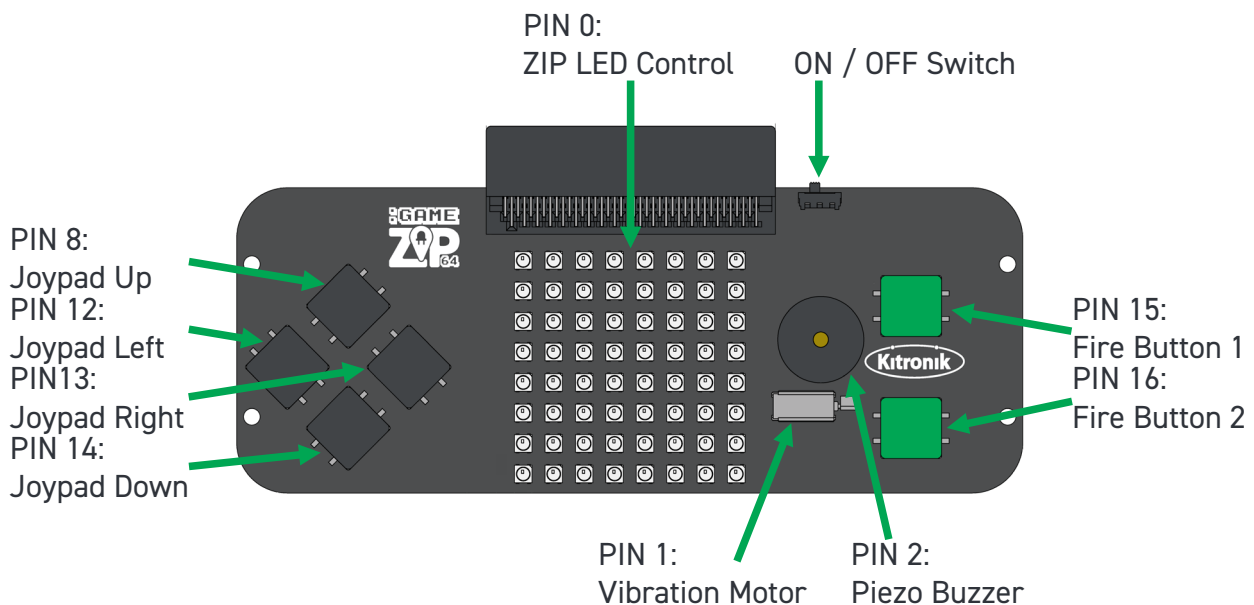
It features on-board sound, 4 directional buttons, 2 fire buttons, and haptic feedback. All features are fully programmable. Breakout points are also included allowing for shoulder buttons or I2C devices to be added, as well as the use of additional LEDs.

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](http://www.kitronik.co.uk)

E: [support@kitronik.co.uk](mailto: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)