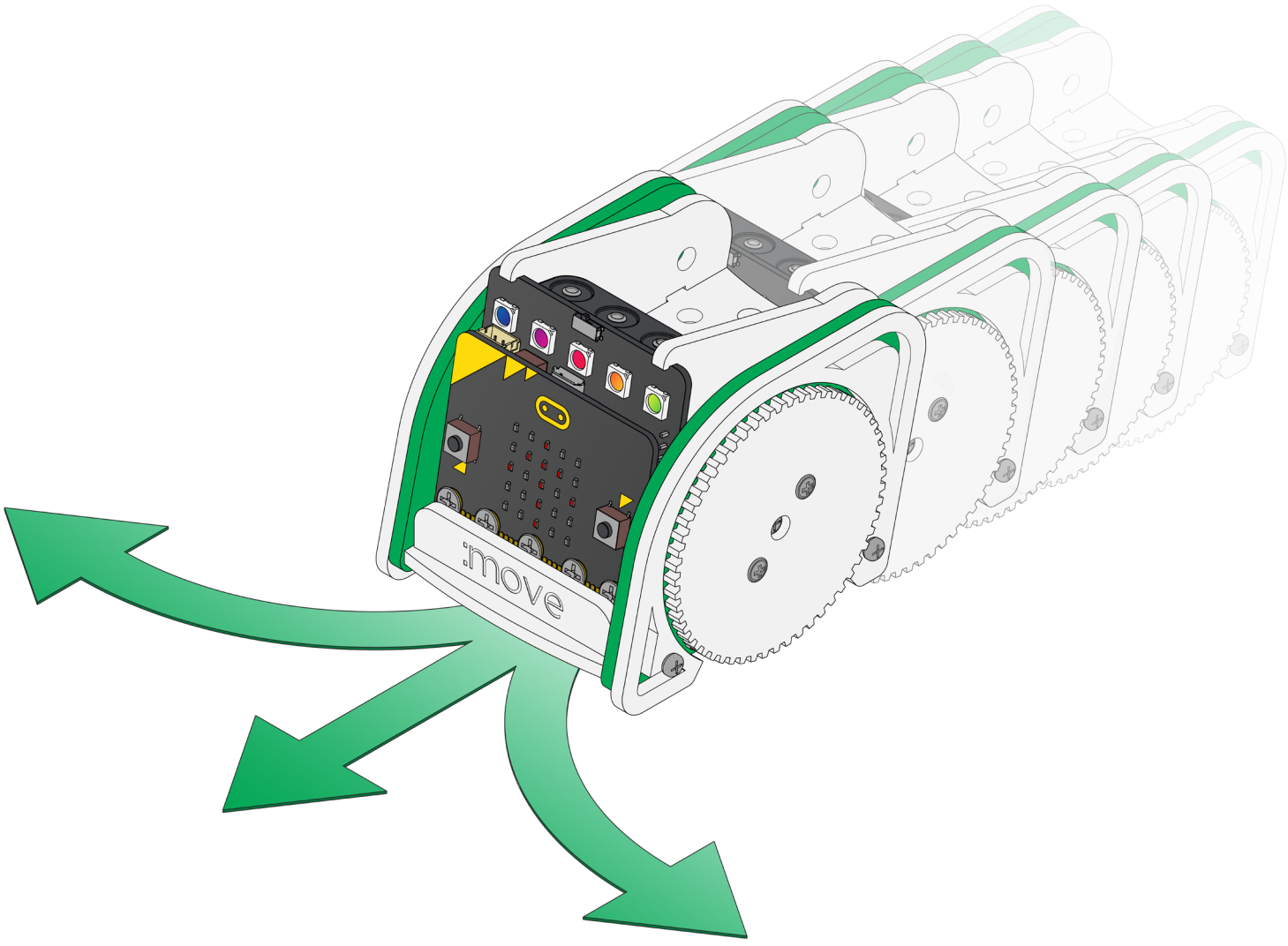
Kitronik

THE TEACHERS LESSON GUIDE TO THE :MOVE MINI



LESSON 1:
MOVEMENT AND LIGHTS

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

www.kitronik.co.uk

TEACH YOUR STUDENTS HOW TO CONTROL A ROBOT WITH CODE!

Kitronik

LESSON 1

MOVEMENT AND LIGHTS

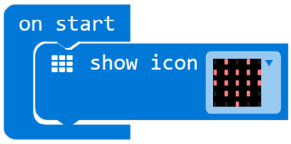
This is an introductory lesson for Key Stage 3 students to robotics. The lesson involves the teacher discussing how we make robots move: first in plain English, then through algorithms, then through code. The teacher will write the algorithms on the board and the students will create the code using the algorithm as a guide. Recommended ratio of students to robot is 6:1

Classroom setup

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

- Pen & Paper
- A computer/laptop with a USB port and Internet access
- A :MOVE mini
- A BBC micro:bit
- 3 x AAA batteries (included with :MOVE mini)
- A micro USB cable

Position the assembled robots in the centre of each group of 6 students. Make sure there is no robot code on the micro:bits so they don't run off the table when a button is pressed. A sample piece of code you could have:



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 and lengthened if needed.

Suggested shortening

Remove Challenge 2: lights. Instead of a specific path, you could create an obstacle course.

Suggested lengthening:

Give the students a path to follow, e.g. draw a map on the board or instructions: forwards for 10 seconds, left, right, forwards, reverse and ask them to code it using the robot. Ask the students to write the algorithm first. The students can work in their pairs and compete to create the most accurate solution.

Differentiation

For younger or less able students there is differentiation in the lesson plan.
The robot can be controlled using servo commands like “Servo write P1 to 120” OR it can be controlled using simple blocks like “Turn left 90 degrees”. See **www.kitronik.co.uk/blog/kitronik-custom-makecode-editor-servo-blocks** for more details. All students will understand how robots move through the algorithm but only the most able/older students will code the servos directly.

KEY

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

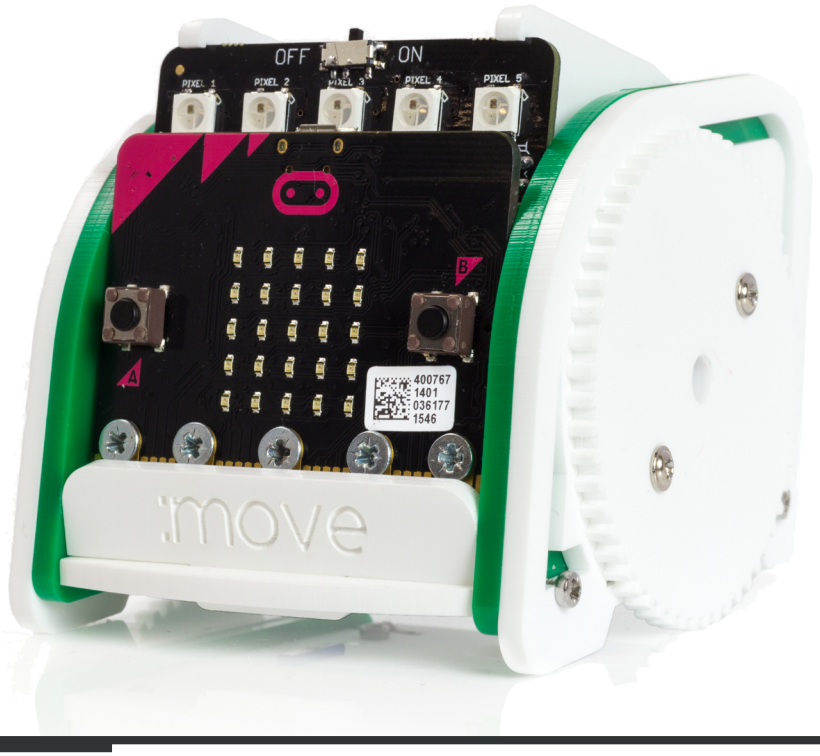
Where this content maps onto the curriculum.

Teacher writes the text in this box on the board.

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

Bug Alert.

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



LESSON 1

MOVEMENT AND LIGHTS

INTRODUCTION

In your groups, look at your robot: What does the robot have that a real car has?

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

Students: On paper, list the items and what they do using the following headings:
(The following is an example... there is no mirror!)

Item	Function
Mirror	To help the driver see around the car

Expected/encouraged answers: Wheels that help move the car.
Expand the answers to include moving the car in different directions – left, right, reverse.

Question: How does a car turn left?
Answer: the driver turns the steering wheel and the wheels swivel to the left together.
Question: Can this robot's wheels swivel together?
Answer: No! This robot moves more like a tank. To turn left: the right wheel spins forward, the left wheel doesn't/spins backwards.

Teacher: Write an algorithm on the board for turning left. This will help them with the code later.

Turning left
Turn right motor on
Turn left motor off
Pause for 2 seconds
Turn right motor off

Question: How does the robot turn right?
Answer: The left wheel spins forward, the right wheel doesn't.

LESSON 1

MOVEMENT AND LIGHTS

Turning right
Turn left motor on
Turn right motor off
Pause for 2 seconds
Turn left motor off

Question: How does a car go forward?
Answer: Both wheels spin forward together. Look back at the left and right algorithm and add "forwards" to every time you turn a motor on.

Going forwards
Turn left motor on forwards
Turn right motor on forwards
Pause for 5 seconds

Question: How does a car go in reverse?
Answer: Both wheels spin backwards together.

Reverse
Turn left motor on backwards
Turn right motor on backwards
Pause for 5 seconds

Question: What else does the robot have that a real car has?
Expected Answer: Lights for headlights.
Expand: Lights for indicating left, right, emergency lights, brake lights and reverse lights.
Question: What does the car not have? Answer: A driver! So the car has to be controlled using a computer, is this the future? What are the consequences of this?
Expected answer: Class discussion about automation, AI and social responsibility. Is it up to programmers to make cars secure so people cannot hack them & cause chaos?

LESSON 1

MOVEMENT AND LIGHTS

Curriculum mapping
Understand a range of ways to use technology safely, respectfully, responsibly and securely.

Main Lesson

Challenge 1

Let's control our robots! Let's start with going forward, left, right, and backwards. We're going to control our robots using the buttons on the micro:bit and motors called servos.

Add a button press to your algorithms on the board:

- On button A Press: Go left
- On button B Press: Go right

On Button A&B Press: Go forward then go backwards

Differentiation – higher ability

Servos work as follows:

The left motor is on Pin 2 – P2. The right motor is on Pin 1 – P1

- For Right, P1 to make it go forward you set it to any angle less than 90. The lesser, the faster.
- For Right, P1 to make it go backwards you set it to any angle greater than 90. The greater, the faster.
- For Left, P2 to make it go forward you set it to any angle greater than 90. The greater, the faster.
- For Left, P2 to make it go backwards you set it to any angle less than 90. The lesser, the faster.
-

A table might help:

Right	P1	Forwards	Fast	0
Right	P1	Forwards	Slow	80
Right	P1	Backwards	Fast	180
Right	P1	Backwards	Slow	100
Left	P2	Forwards	Fast	180
Left	P2	Forwards	Slow	100
Left	P2	Backwards	Fast	0
Left	P2	Backwards	Slow	80

MOVEMENT AND LIGHTS

Help the students by filling in the servo commands next to the algorithm.

Turning left
On button A Press:

Turn right motor on forwards	Set Servo Pin 1 to 0
Turn left motor off	Set Servo Pin 2 to 90
Pause for 2 seconds	Pause for 2 seconds
Turn right motor off	Set Servo Pin 1 to 90

Turning right
On button B Press:

Turn left motor on forwards	Set Servo Pin 2 to 180
Turn right motor off	Set Servo Pin 1 to 90
Pause for 2 seconds	Pause for 2 seconds
Turn left motor off	Set Servo Pin 2 to 90

End of differentiation

BUG ALERT!
Before we code our robots and drive them off the table – can you spot a problem with two of our algorithms? (The hint is in the phrase "drive them off the table")

Question: When the robot is going forward or backwards, when does it stop?
Answer: Never!

Let's put a stop at the end of our algorithm after 5 seconds:

Going forwards
On button A & B Press:

Turn left motor on forwards	Set Servo Pin 2 to 180
Turn right motor on forwards	Set Servo Pin 1 to 0
Pause for 5 seconds	Pause for 5 seconds
Turn left motor on backwards	Set Servo Pin 2 to 0
Turn right motor on backwards	Set Servo Pin 1 to 180
Pause for 5 seconds	Pause for 5 seconds
Turn left motor off	Set Servo Pin 2 to 90
Turn right motor off	Set Servo Pin 1 to 90

MOVEMENT AND LIGHTS

Let's get coding

1. Open http://makecode.com
2. Select the micro:bit

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

Show the students where all the blocks they need are and ask them to get coding. They should use the algorithms on the board to guide them in creating the code.

Kitronik have provided custom blocks that are as simple as "turn left 90 degrees". Weaker/younger students could use these blocks to code their robots. These students will need to add the custom package before they can see the blocks. Follow these steps:

1. Select Advanced
2. Select Add Package
3. Type "Kitronik"
4. Select "Kitronik-servo-lite"

This adds the menu item Kitronik to your menu

NOTE: Students don't need the complicated algorithm but creating it is a good exercise for all abilities/ages.

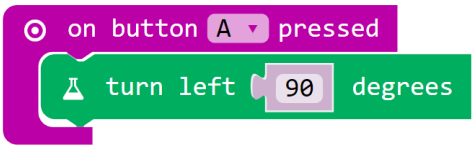
Continued on next page >

MOVEMENT AND LIGHTS

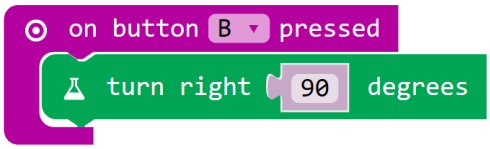
Answers

Left:

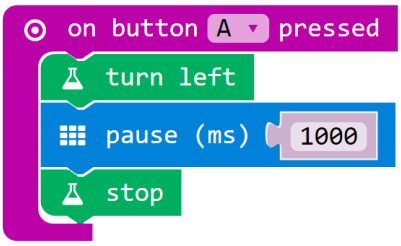
This is the simplest answer but students will find the robots aren't always accurate and they might have to adjust the angle 90.

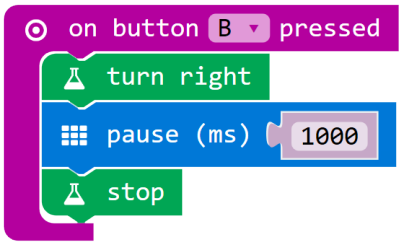


Right:

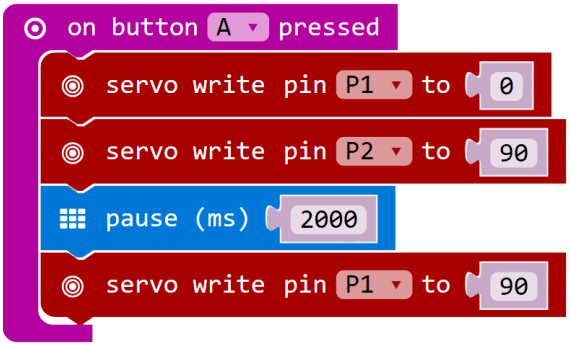


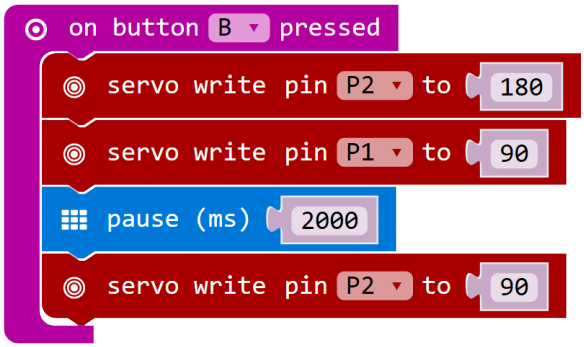
You can make the robot turn left for a second then stop it to see how far it spins in 1 second. (1 second is too long!)





Advanced:
The servo blocks are under Advanced > Pins

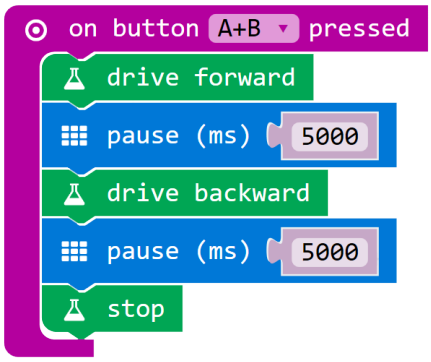


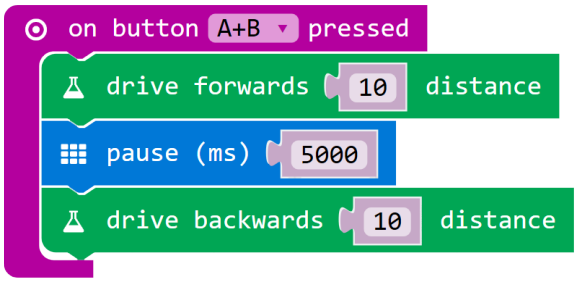


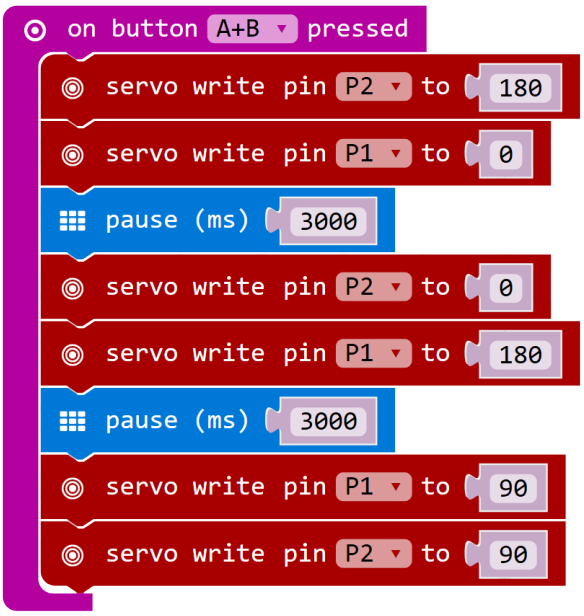
Continued on next page >

MOVEMENT AND LIGHTS

Forwards and Backwards:







Ask all the students coding left to download their code to their robot. Place the robots on the floor and ask:

1. What's it going to do?
2. Press A
3. Did it do what you expected?
4. Can we improve this code?

Ask the students to return to their desks and take notes.

They will need to write down the algorithm from the board in their notebooks and take screenshots of their code and be able to explain it.
Repeat for the groups who created the right hand turn code then the forward and backwards groups.

LESSON 1

MOVEMENT AND LIGHTS

Challenge 2

Let's add some lights! The car has 5 lights. Let's mimic a car lights – what are they?

Lights:

1	2	3	4	5
Right indicator	Headlights/ reverse lights		Headlights/ reverse lights	Left indicator

Question: What is the algorithm for an indicator light?
Answer: On, off. How many times? What lights?

Write the algorithm on the board for the left and right lights.

Right indicator:
Repeat 4 times:
Turn light 1 on
Pause
Turn light 1 off
Pause

Forwards: you need to turn some headlights on when you go forwards. Which ones, how many, how bright?

Algorithm:

Headlights:
Set brightness to 100
Turn light 2 and 4 on

Backwards: reverse lights!

Reverse:
Set brightness to 100
Turn light 2 and 4 on

LESSON 1

MOVEMENT AND LIGHTS

BUG ALERT!
Forward and reverse is the same algorithm. But reversing is VERY different to going forward. What's different?
Answer: Their location. In our case we're just going to make the colours of the lights different.
Forward will be orange, reverse will be white.

Edit the algorithms to specify a light colour, e.g. right indicator:

Right indicator:
Repeat 4 times:
Set light 1 to yellow
Turn light 1 on
Pause
Turn light 1 off
Pause

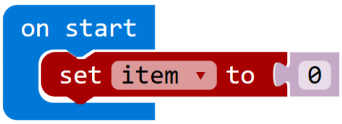
IMPORTANT: Before they can code the lights, the students will need to add the correct package:

1. Select Advanced
2. Select Add Package
3. Select "Neopixel"

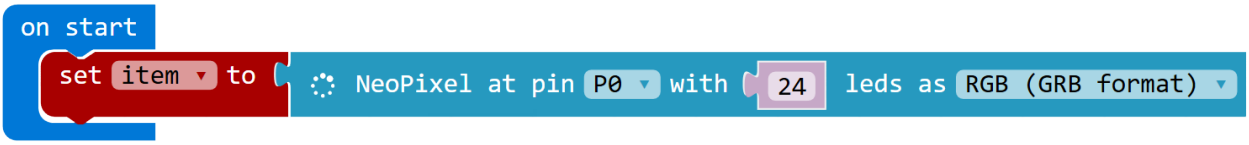
This adds the menu item Neopixel to your menu.

Setup your lights:

1. Select Variable
2. Select "set item to 0"
3. Drag this under "on start" like so:



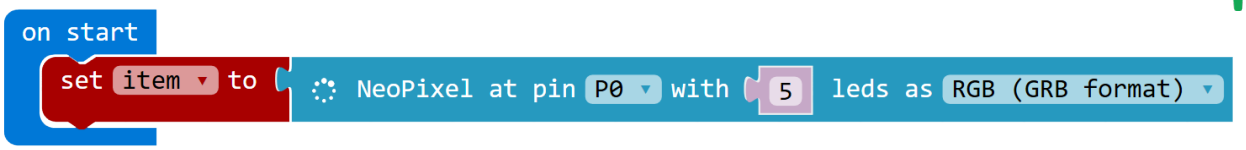
4. Select Neopixel
5. Select "Neopixel at pin P0 with 24 LEDs as RGB (GRB format)"
6. Drop it on top of the 0 of "set item to 0"



Continued on next page >

MOVEMENT AND LIGHTS

Change 24 to 5



This sets the variable item to be 5 Red, Green and Blue lights on Pin 0

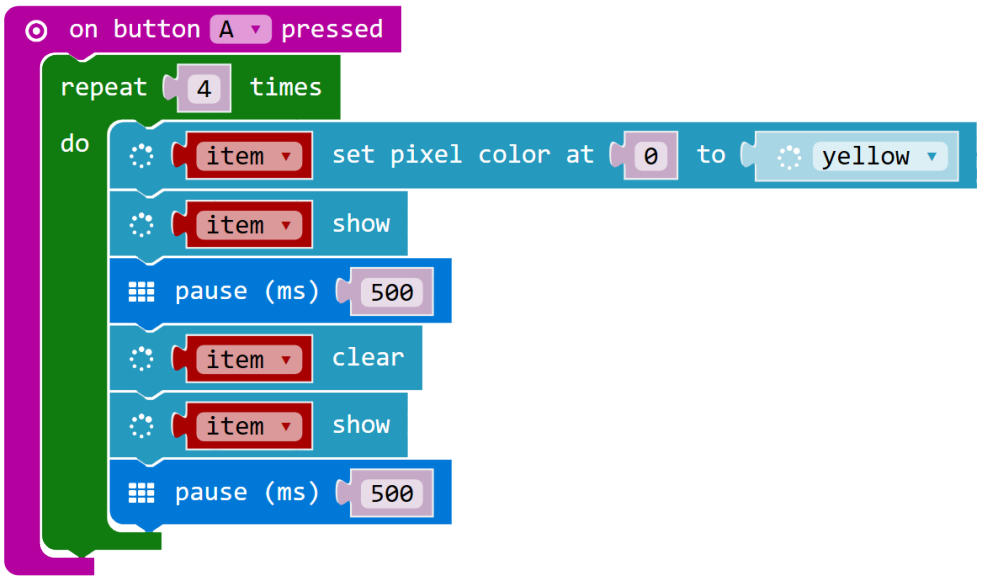
Some points to note:

- Lights are numbered 0 to 4 not 1 to 5!
- There are 1000ms in a second
- You must use the block "item show" after you set a colour. Otherwise nothing will happen!
-

Ask the students to write the code in their groups. The group that coded the left hand turn will now code the left hand indicator, etc.

Answers

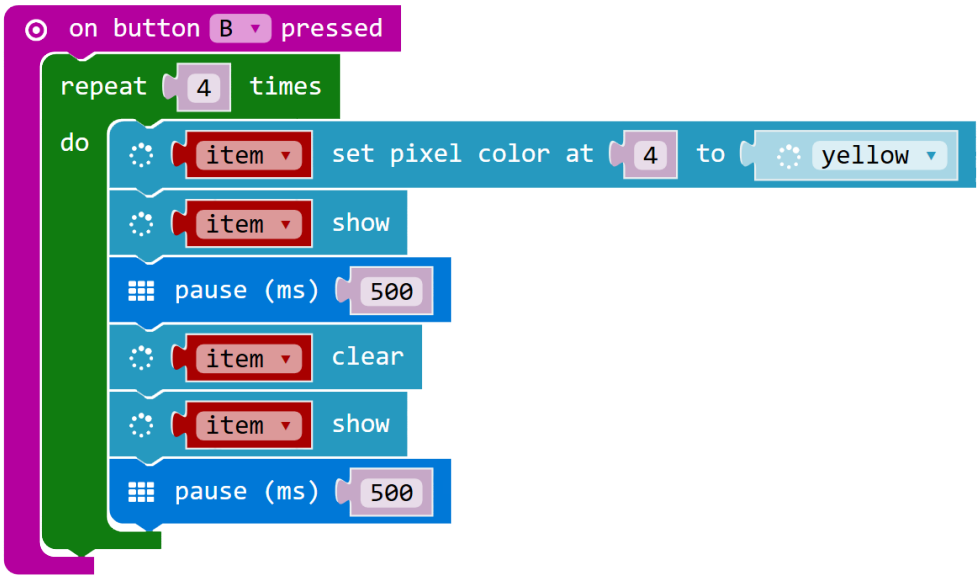
Right indicator:



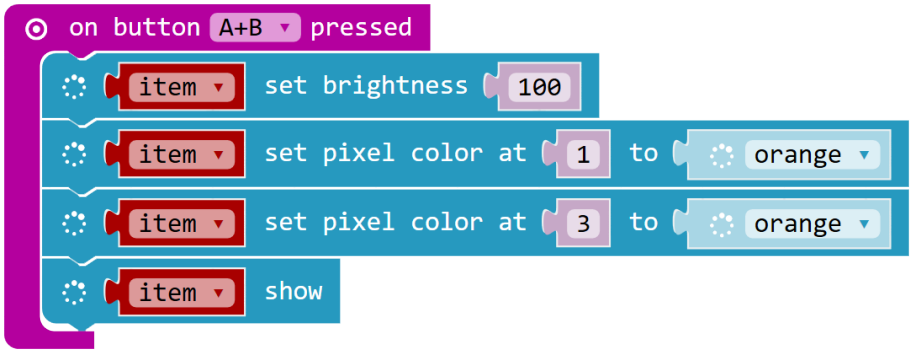
Continued on next page ›

MOVEMENT AND LIGHTS

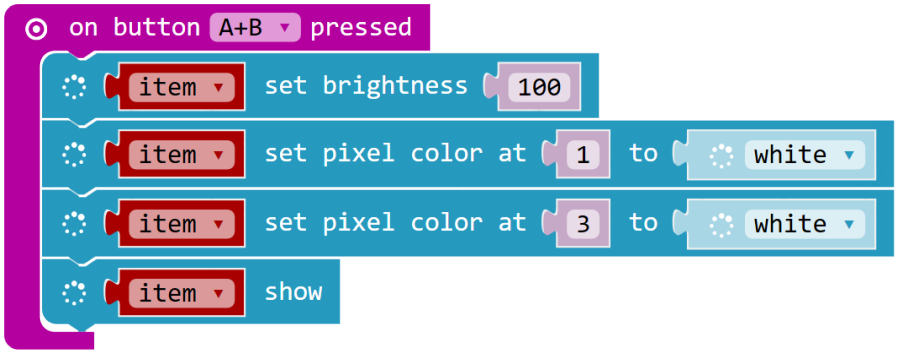
Left indicator:



Headlights:



Reverse lights:



Download and test the code once each group is finished.

LESSON 1

MOVEMENT AND LIGHTS

Evidence
With one robot between 6 students, there will be waiting between each task. As each group finishes a task ask them to take a screenshot of their code and paste it into a document. Ask them to comment the code individually.
They can also write down the algorithms from the board and make comments about them/improve them.

Summary

18

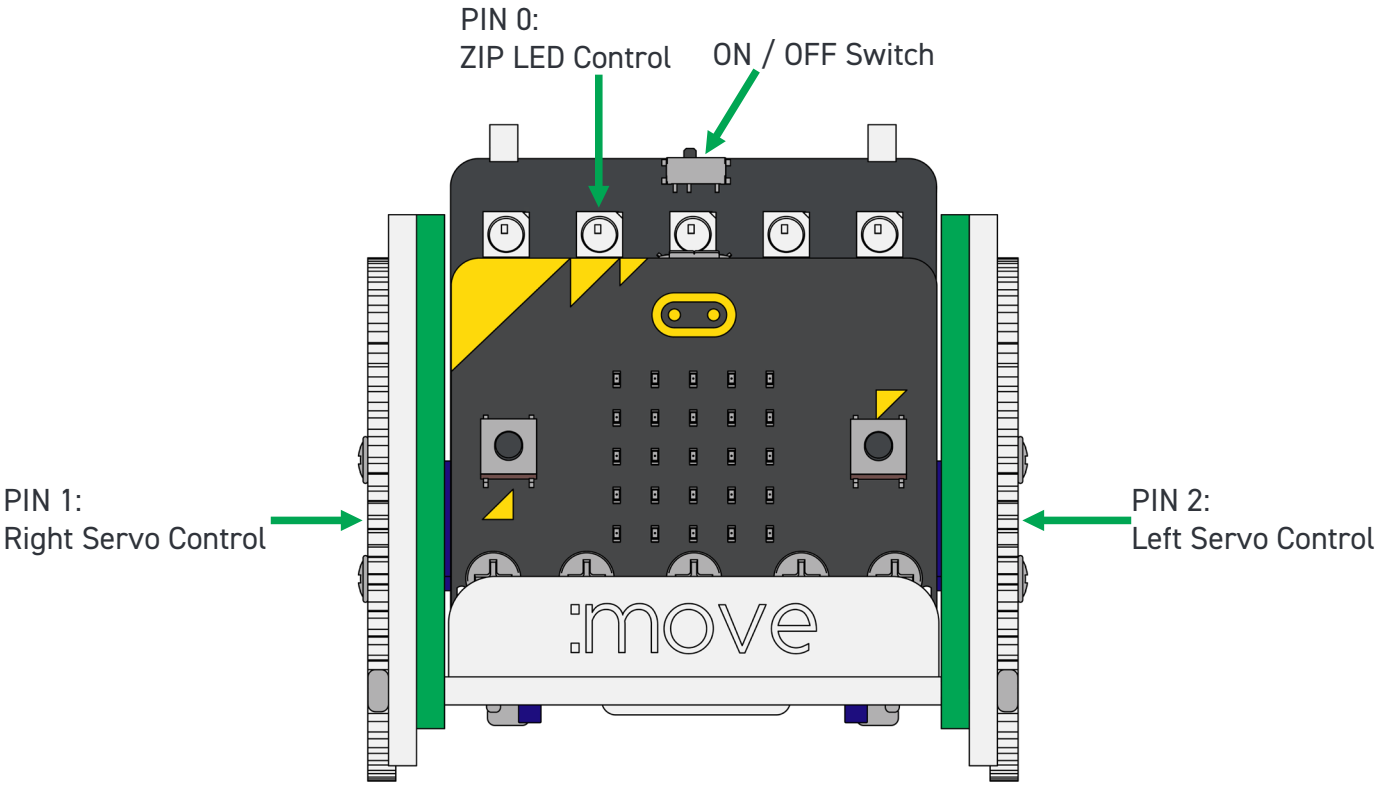
Student modelled a real world system (a car with lights) using the robots. Together you wrote and debugged algorithms, wrote and tested code and changed the code based on testing.
This is a typical project life-cycle: Design -> Use -> Evaluate
Next week we're going to go further with automation and get the robots to create drawings for us.

This is an introductory lesson for students in **Key Stage 3** for robotics with the **Kitronik :MOVE mini** for **BBC micro:bit**. The lesson involves the teacher discussing how we make robots move: first in plain English, then through algorithms, then through code. The teacher will write the algorithms on the board and the students will create the code using the algorithm as a guide. The recommended ratio of students to robot is 6:1. Please find additional lesson plans, accompanying **PowerPoint** presentations and more at **www.kitronik.co.uk/5624.**

The Kitronik :MOVE mini buggy kit for the BBC micro:bit provides a fun introduction to the world of robotics. The :MOVE mini is a 2 wheeled robot that is suitable for autonomous operation, remote control projects and more. Once built it can be coded for a variety of activities. A range of add on boards can expand the capabilities to include more advanced functionality.

LESSON REQUIRES:

- Pen & Paper
- A computer/laptop with a USB port and Internet access
- A :MOVE mini
- A BBC micro:bit
- 3 x AAA batteries (included with :MOVE mini)
- 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

kitronik.co.uk/twitter
kitronik.co.uk/facebook
kitronik.co.uk/youtube
kitronik.co.uk/google