

Lesson 1: Connecting Crumbles

Year 5 – Programming – Selection in physical computing



Lesson 1: Connecting Crumbles

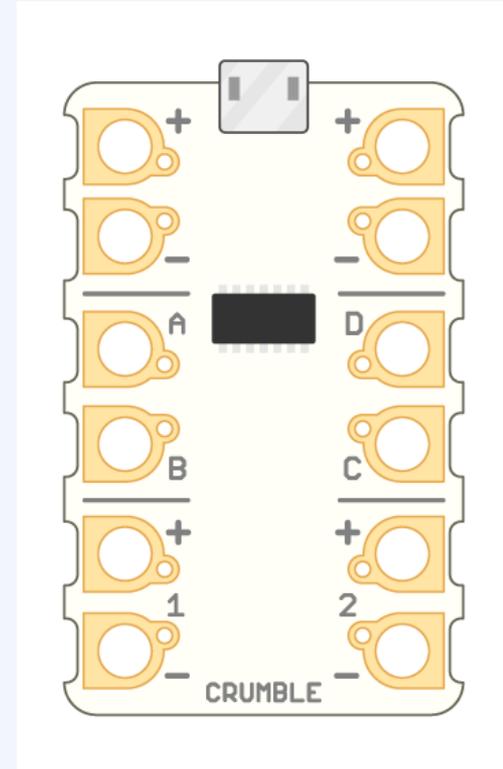
To control a simple circuit connected to a computer

- I can create a simple circuit and connect it to a microcontroller
- I can program a microcontroller to make an LED switch on
- I can explain what an infinite loop does

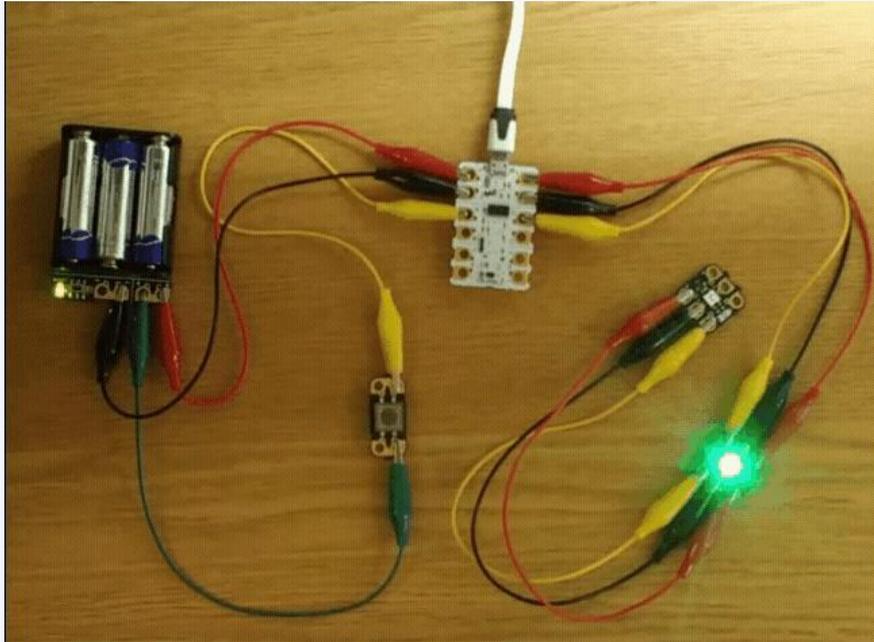
Crumble controller

A microcontroller is a small device that can be programmed to control components that are connected to it.

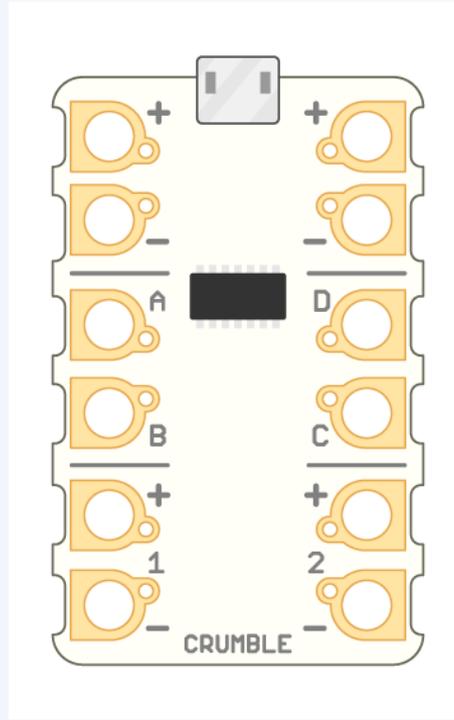
The microcontroller that you will be using is a Crumble controller. You will program a Crumble to control outputs and respond to inputs.



Crumble controller



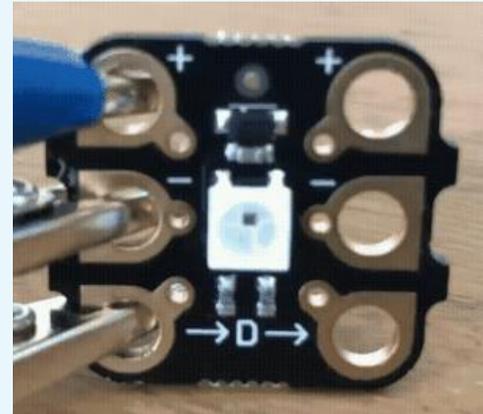
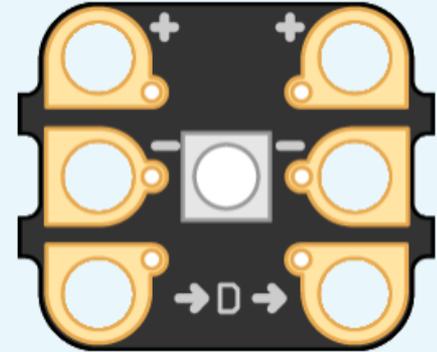
Making observations and asking questions



Connecting a Sparkle

A Sparkle is a multi-colour LED designed to work with the Crumble.

The crocodile clip leads connect the Sparkle to the Crumble. These connections provide power to light the LED and data for its colour.

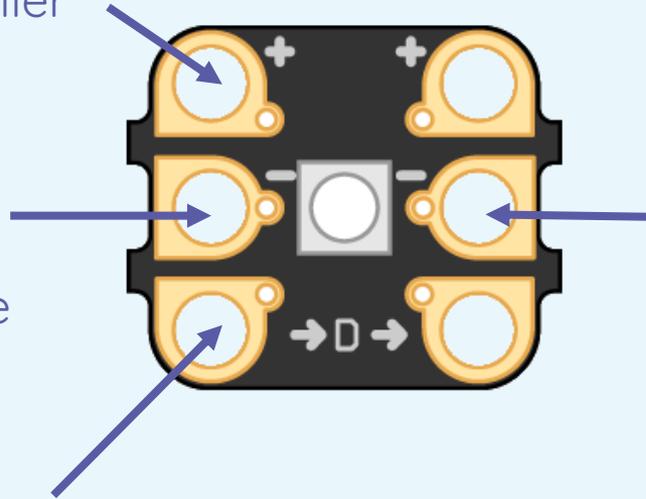


Connecting Sparkle

Connects to a positive power (+) pad on the Crumble controller

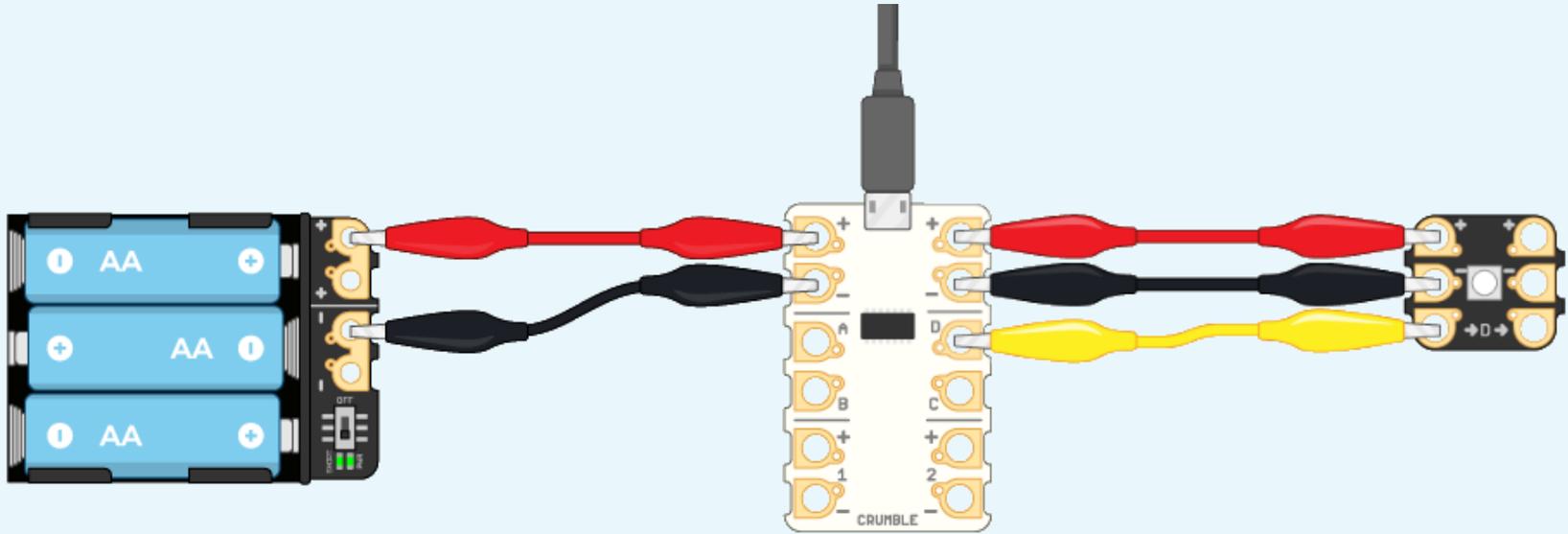
Connects to a negative power (-) pad on the Crumble controller

Connects to the D pad on the Crumble controller



The pads on this side are used to connect other Sparkles

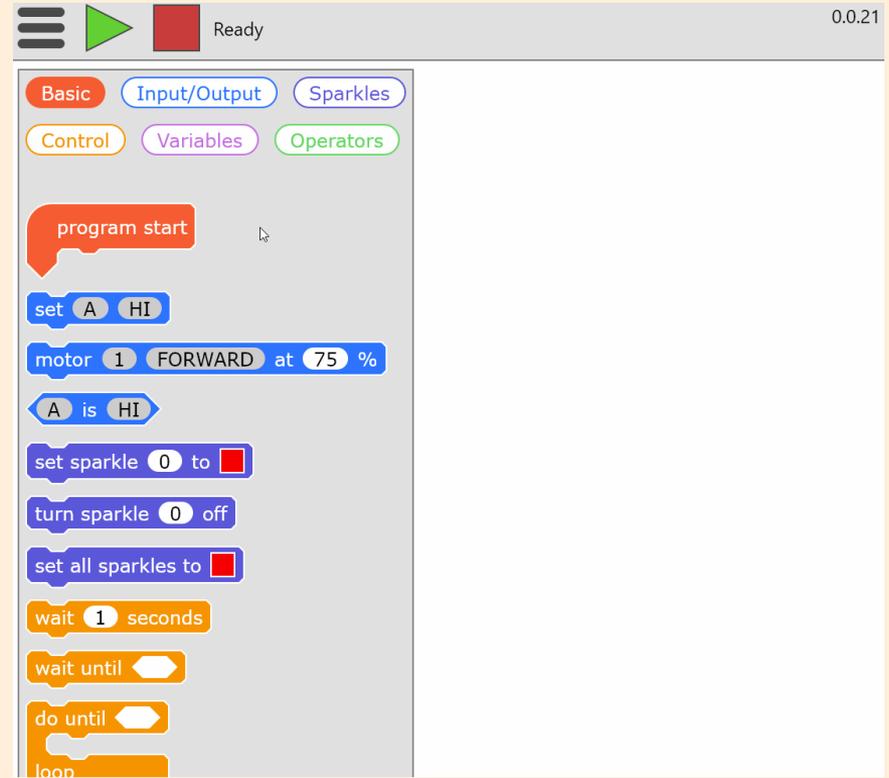
Connecting circuits



Connect your Crumble using the guide above. The Sparkle will flash white six times when you've connected it correctly.

Programming Crumbles

To create a Crumble program drag the blocks from the side panel to the main coding area.



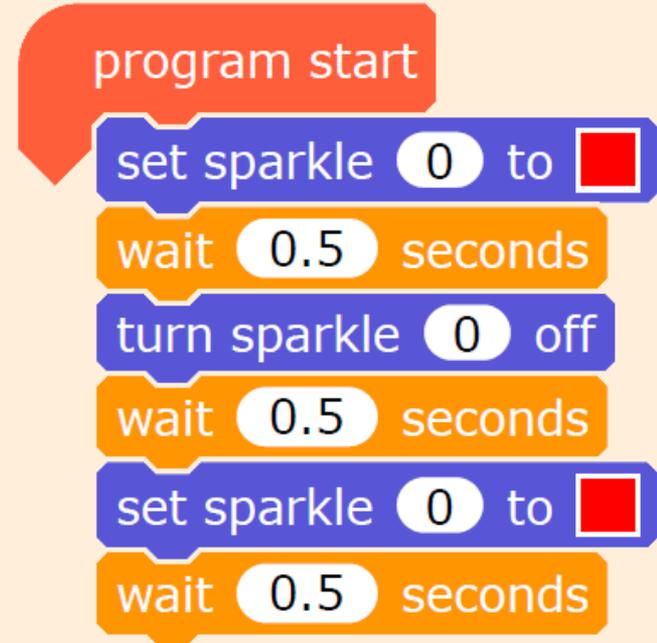
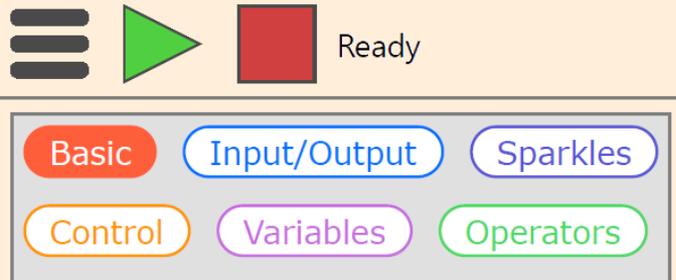
The screenshot displays the Crumble programming interface. At the top, there is a status bar with a menu icon, a green play button, a red stop button, the text "Ready", and the version number "0.0.21". Below the status bar is a side panel with several category tabs: "Basic" (red), "Input/Output" (blue), "Sparkles" (blue), "Control" (orange), "Variables" (purple), and "Operators" (green). The main coding area contains the following sequence of blocks:

- program start (red)
- set A HI (blue)
- motor 1 FORWARD at 75 % (blue)
- A is HI (blue)
- set sparkle 0 to [red square] (blue)
- turn sparkle 0 off (blue)
- set all sparkles to [red square] (blue)
- wait 1 seconds (orange)
- wait until [empty] (orange)
- do until [empty] (orange)
- loop (orange)

Programming Crumbles

Create this program in the Crumble software.

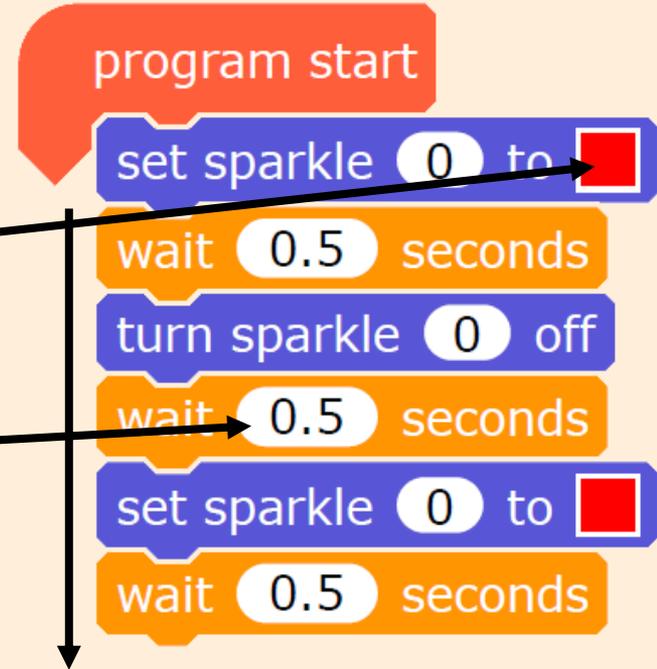
After you've checked your Crumble is connected to your computer, press the green play button.



Programming Crumbles

You can modify the program to change the Sparkle lights. You can:

- Set the Sparkle to a different colour
- Wait for a different length of time
- Flash a different number of times



Programming Crumbles

Sparkle flashes:	How many times:	Sparkle stays on and off for:	Achieved
	3	0.5 seconds	
	3	1 second	
	3	2 seconds	
	4	2 seconds	
	4	0.5 seconds	
	5	0.5 seconds	
	3	2 seconds	
	2	1.5 seconds	

Program your Crumble to make the Sparkle flash in different ways

Programming Crumbles

The screenshot shows the Scratch programming environment. The 'Script' area contains the following code blocks:

- program start
- set A HI
- motor 1 FORWARD at 75 %
- A is HI
- set sparkle 0 to [red color swatch]
- turn sparkle 0 off

The 'Code' area shows a zoomed-in view of the 'set sparkle 0 to [red color swatch]' block.

Setting a Sparkle's colour

The screenshot shows a zoomed-in view of a Scratch script with the following code blocks:

- program start
- set sparkle 0 to [red color swatch]
- wait 1 seconds

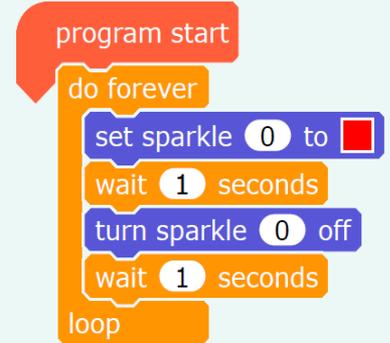
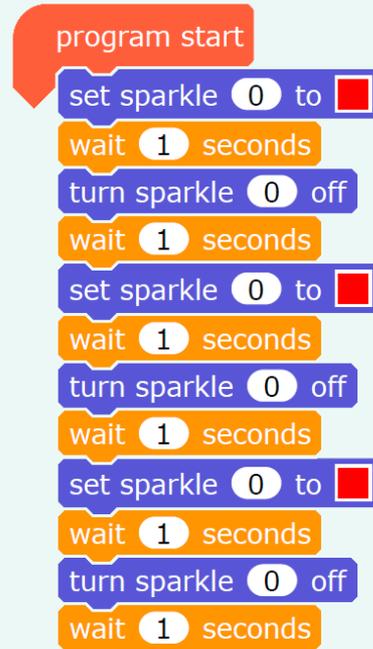
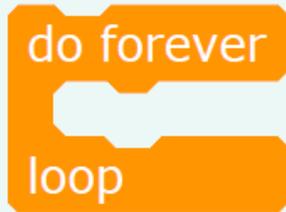
A mouse cursor is positioned over the '1' in the 'wait 1 seconds' block, indicating it is being edited.

Changing a wait command

Forever flashing

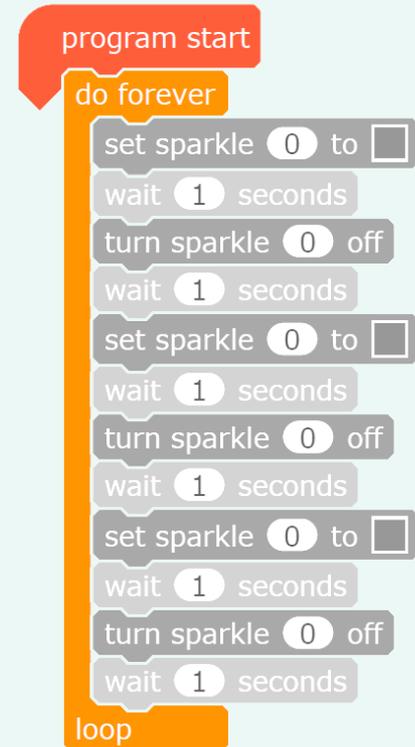
You might want to repeat some or all of the commands in your program. You can do this using a repeat block.

This block repeats the commands inside it forever.



Forever flashing

Create your own Crumble program so that your Sparkle flashes a colour pattern continuously.



```
program start
do forever
  set sparkle 0 to 
  wait 1 seconds
  turn sparkle 0 off
  wait 1 seconds
  set sparkle 0 to 
  wait 1 seconds
  turn sparkle 0 off
  wait 1 seconds
  set sparkle 0 to 
  wait 1 seconds
  turn sparkle 0 off
  wait 1 seconds
loop
```

The image shows a sequence of code blocks for a 'do forever' loop. The blocks are: 'set sparkle 0 to ', 'wait 1 seconds', 'turn sparkle 0 off', 'wait 1 seconds', 'set sparkle 0 to ', 'wait 1 seconds', 'turn sparkle 0 off', 'wait 1 seconds', 'set sparkle 0 to ', 'wait 1 seconds', 'turn sparkle 0 off', and 'wait 1 seconds'. The loop is enclosed in a 'do forever' block, and the entire program starts with a 'program start' block and ends with a 'loop' block.

Debugging



```
program start
set sparkle 0 to ■
wait 0.5 seconds
turn sparkle 0 off
wait 0.5 seconds
```

A Scratch script starting with a red 'program start' block. It contains four blocks in sequence: a blue 'set sparkle 0 to' block with a red square icon, an orange 'wait 0.5 seconds' block, a blue 'turn sparkle 0 off' block, and another orange 'wait 0.5 seconds' block.



```
program start
do forever
  set sparkle 0 to ■
  wait 0.5 seconds
  turn sparkle 0 off
loop
```

A Scratch script starting with a red 'program start' block. It contains a yellow 'do forever' loop block. Inside the loop are three blocks: a blue 'set sparkle 0 to' block with a red square icon, an orange 'wait 0.5 seconds' block, and a blue 'turn sparkle 0 off' block. The loop ends with a yellow 'loop' block.

Why don't these programs produce a continuously flashing Sparkle?

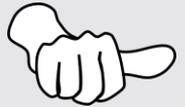
How confident are you? (1-3)

- I can create a simple circuit and connect it to a microcontroller
- I can program a microcontroller to make an LED switch on
- I can explain what an infinite loop does

3 - Very confident



2 - Unsure



1 - Not confident



Next lesson

In this lesson, you...

Built a simple circuit using a microcontroller and connected it to a computer

Programmed a microcontroller to light a Sparkle in different ways

Used repetition in the form of an infinite loop

Next lesson, you will...

Connect additional components to the microcontroller

Program a microcontroller to control more than one output

Use repetition in the form of count-controlled loops