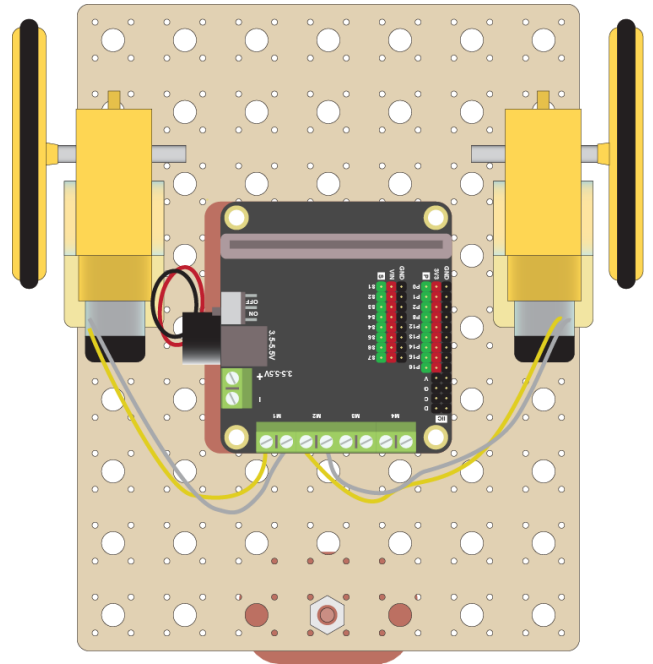
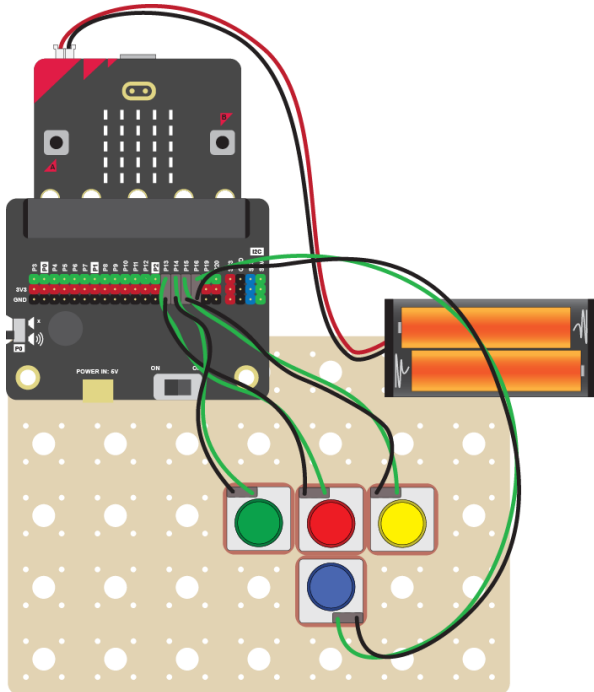


Build a Remote Control Robot

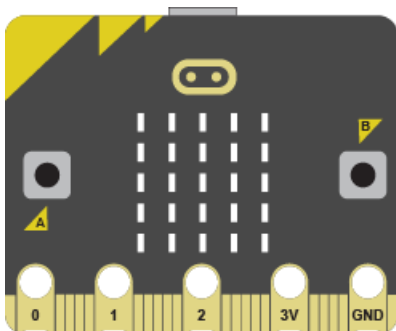
Project 1.04

In this workshop you will make a remote controller which can be used to control the robot you built in a previous workshop.



For the remote control you will need another Microbit. So you will have two microbits, one in the controller and one in the robot. We will use the radio feature of the Microbit to send messages from the controller to the robot:

Controller Microbit



Messages:

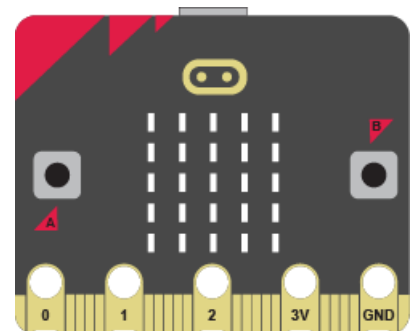
F for Forwards

L for Left

R for Right

B for Backwards

Robot Microbit

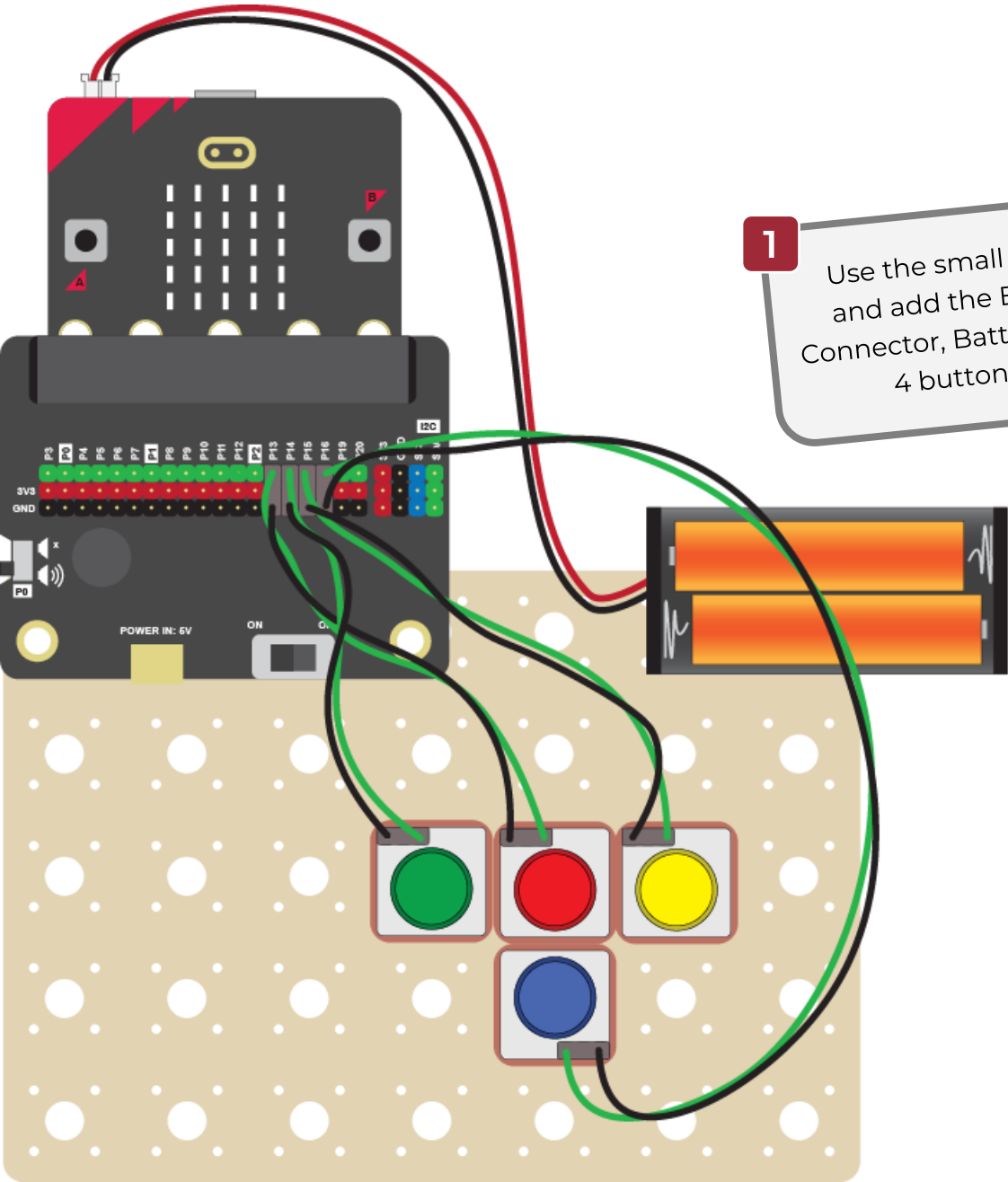


What to do

- If you haven't already done so, build the robot by referring to the previous worksheet (just build it, don't code it).
- Then follow this worksheet to remotely control the forward movement of your robot.
- Finally, attempt the coding challenge to get the left, right and backwards movements of your robot working.

Assemble the Controller

Assemble the Parts



1 Use the small board and add the Edge Connector, Battery and 4 buttons

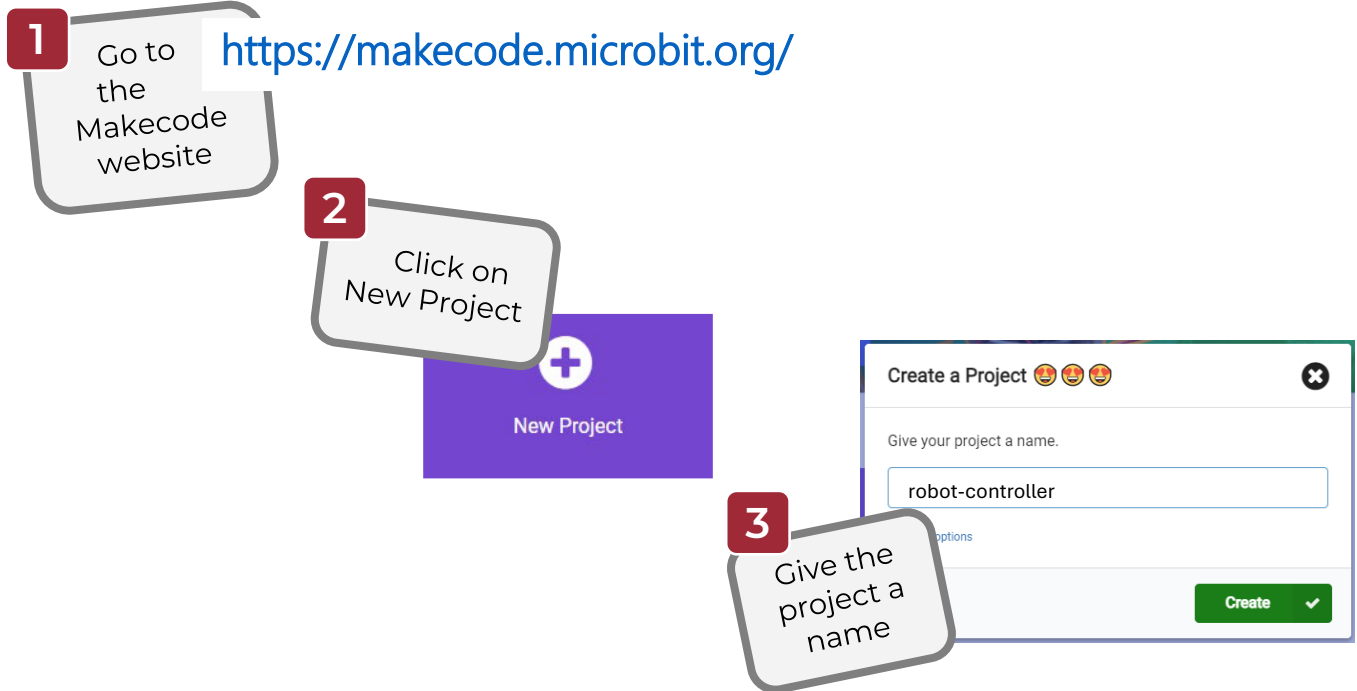
2 Wire up the buttons as follows using GS wires

Component	Microbit Connections	Purpose
Red button	P13	Forwards
Green button	P14	Left
Yellow button	P15	Right
Blue button	P16	Backwards

Code the Controller 1

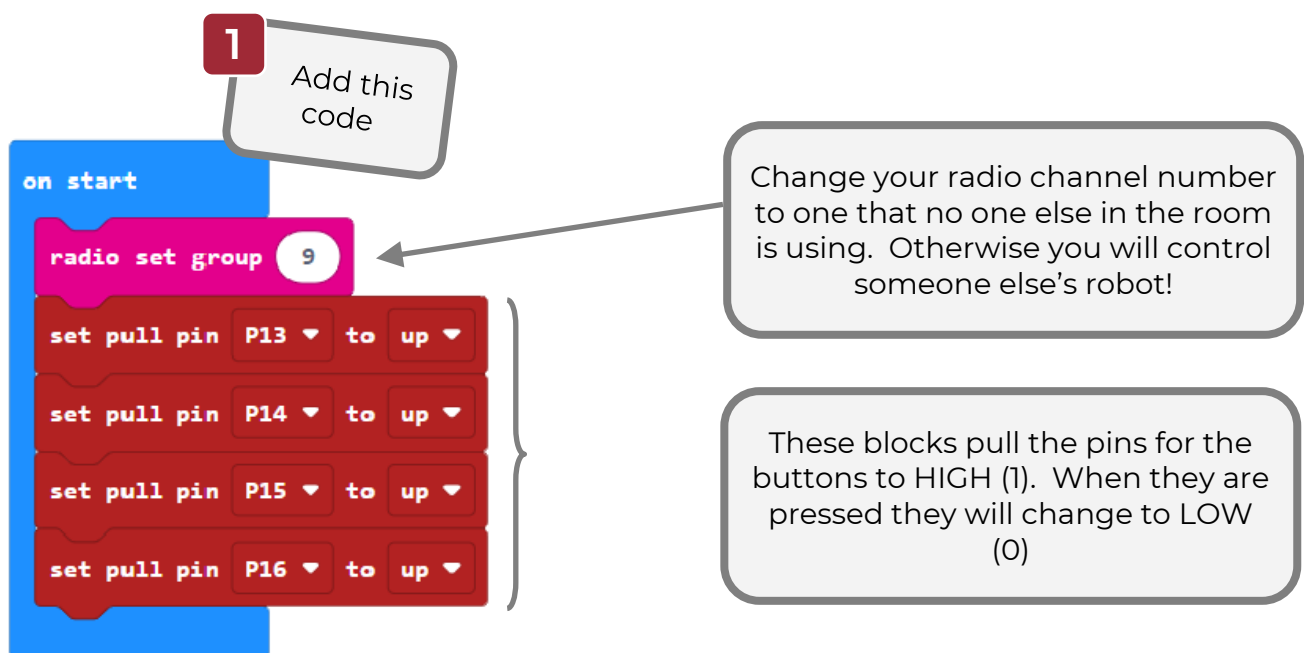
Tutors: Don't use this sheet if ready-coded controller microbits used

Create a Project for the Controller



Set up the Radio Channel and Buttons

First set up the radio channel and the buttons on your controller. The Microbit's radio will be used so your controller can send messages to your robot.



Respond to the F Button

Now get your controller to send a message "F" when the forward button is pressed.

1 Add this code

This block checks if the button connected to pin 13 has been pressed

This block will send the message "F" to your robot

This block will display a dot showing that you pressed the forward button

If no button was pressed send an "S" to your robot to ask it to stop moving

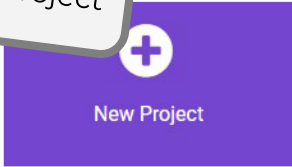
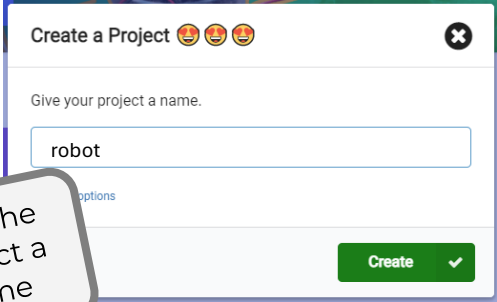
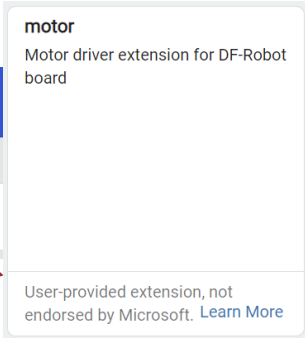
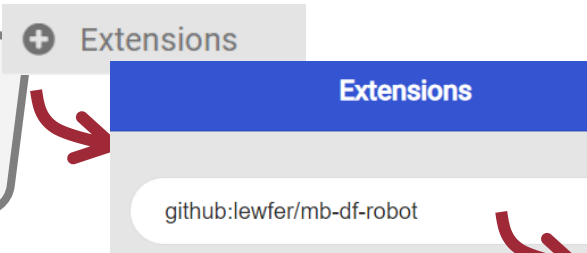
Wait a little bit so that your robot doesn't get sent too many messages!

2 Download the code to the controller Microbit

Download


Code the Robot 1

Create a Project for the Robot

- 1 Go to the Makecode website
<https://makecode.microbit.org/>
- 2 Click on New Project

- 3 Give the project a name

- 3 Add the motor driver extension


Set up the Radio Channel

Set up the radio channel (so your robot can receive messages to your controller).

- 1 Add this code


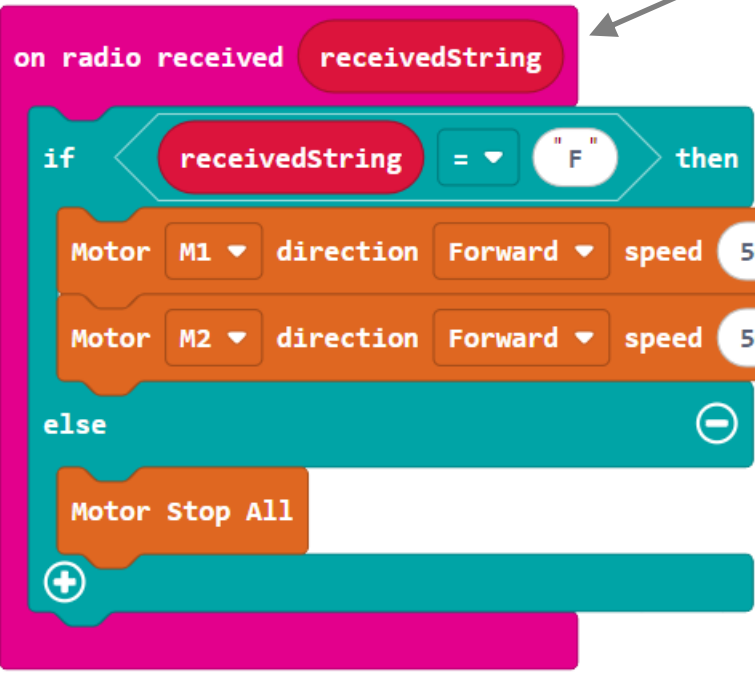
Change your radio channel number to the one you used for your controller, so that the messages from your controller are received by your robot

Code the Robot 2

Respond to the F Message

Now get your controller to receive the "F" message from the controller and move forwards.

1 Add this



The code consists of the following blocks:

- on radio received** block with **receivedString** as the message.
- if** block with **receivedString** **=** **"F"** **then**.
- Inside the **if** block:
 - Motor** **M1** **direction** **Forward** **speed** **50**
 - Motor** **M2** **direction** **Forward** **speed** **50**
- else** block with **Motor Stop All**.

2 Download the code to the robot Microbit

Download

This block is only run when your robot receives a message from the controller

This block checks if the message received was an "F"

Move the robot forwards

If any other message is received, stop the robot

Your challenge!

The code you have on your controller and robot allows you to only control the forward movement of the robot. That's not very useful!

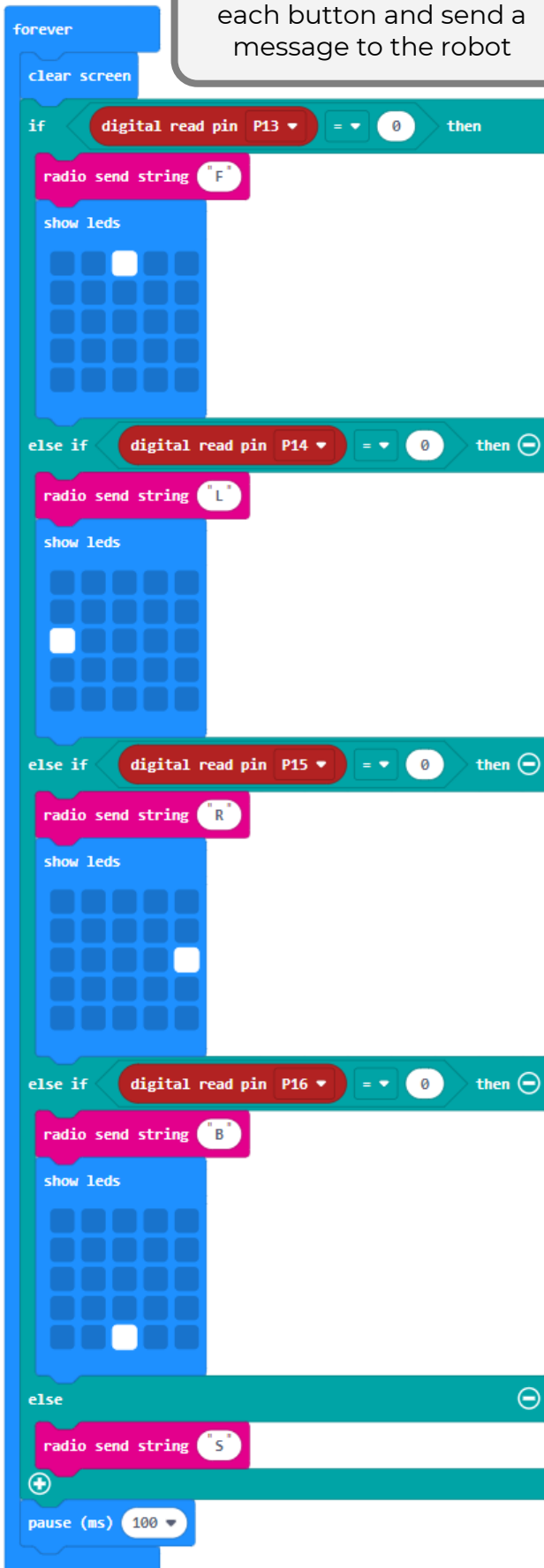
Can you get other movements working: left, right and backwards?

Hint: You will need to change the code on both the controller and robot. The controller must send different messages, such as "L", "R" and "B" for left, right and backwards. The robot must then respond with the correct movement.

Solution

Controller

You need to respond to each button and send a message to the robot



Robot

You need to respond to each message and move the robot accordingly

