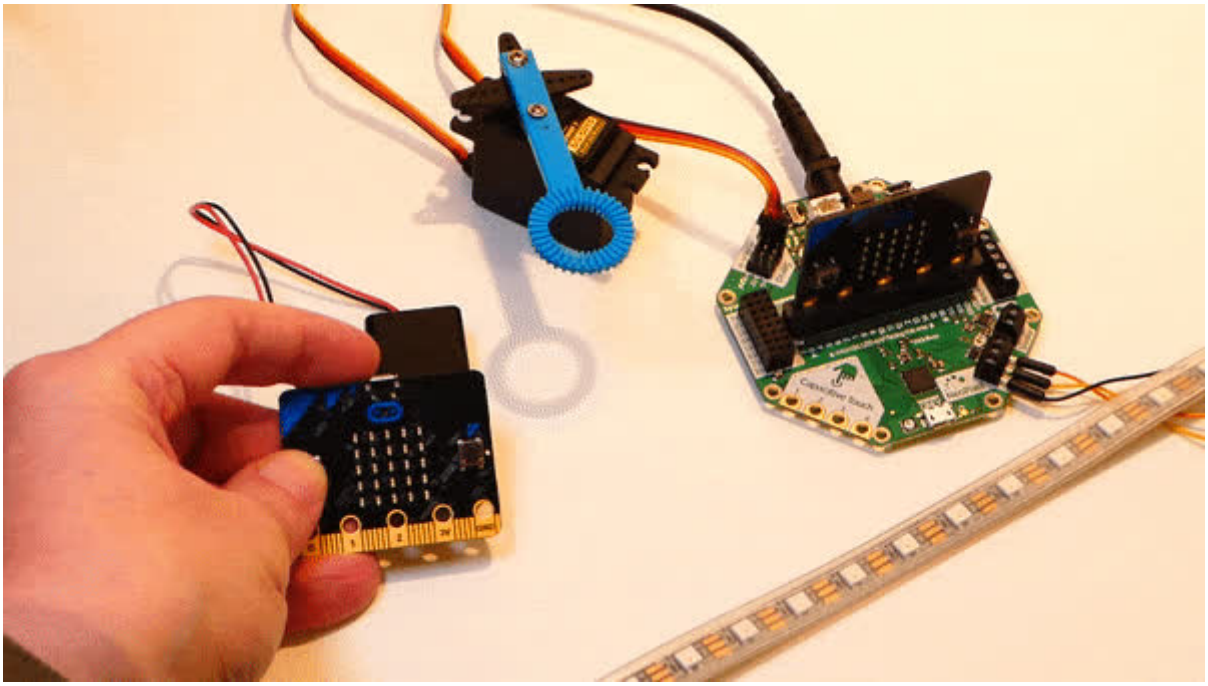
Micro:bit Radio Control of Crickit Robotics

Created by Anne Barela



https://learn.adafruit.com/micro-bit-radio-control-of-crickit-robotics

Last updated on 2024-03-08 03:22:26 PM EST

Table of Contents

Overview

The BBC micro:bit continues to gain market share in the educational space. And why not, with an LED matrix display and a built-in Bluetooth LE radio, users can do some advanced projects.

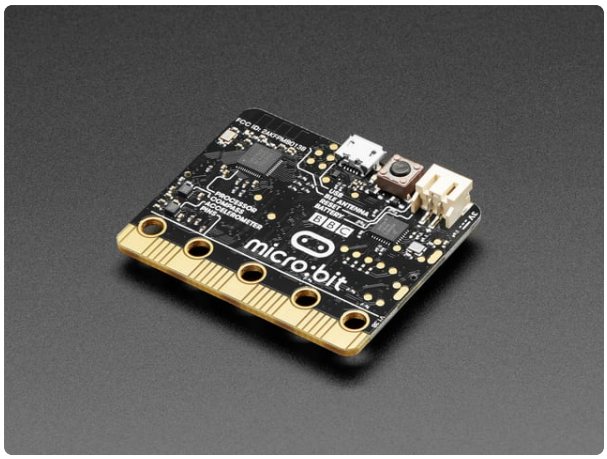
In this guide, one micro:bit will be used as a radio control. Pushing the A and B buttons will transmit commands. A second micro:bit will be connected to an Adafruit Crickit robotics controller. The receiving micro:bit will translate the received radio commands into actions on the Crickit.

Parts

The Go bundle includes a battery pack, batteries, and a USB cable for a tiny bit more than the micro:bit alone. This is good for the transmitter.

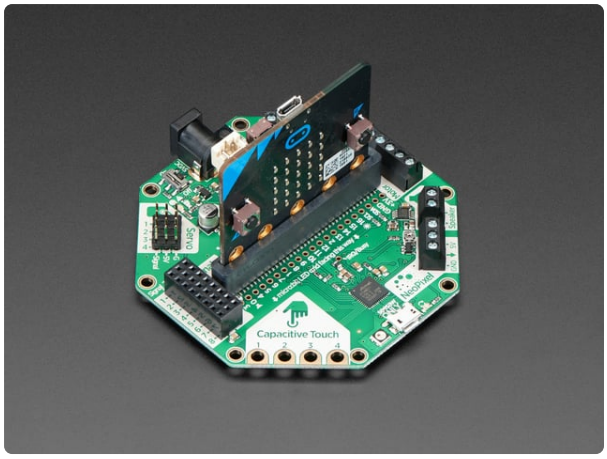
BBC micro:bit Go Bundle
Discontinued - you can grab micro:bit v2 Go Bundle - Batteries and USB Cable Included instead! The British...
https://www.adafruit.com/product/3362

The receiver is not battery powered, so the regular micro:bit pack is fine.



BBC micro:bit
The British Invasion is here! No, not music...microcontrollers! New to the USA is the newest and easiest way to learn programming and electronics - the BBC...
https://www.adafruit.com/product/3530

Adafruit Crickit for micro:bit provides robotics, sound, and light support.



Adafruit CRICKIT for micro:bit
Sometimes we wonder if robotics engineers ever watch movies. If they did, they'd know that making robots into servants always ends up in a robot rebellion. Why even go down that...
https://www.adafruit.com/product/3928

We'll need a power supply for the Crickit and some interface items. You can choose, but for this tutorial we'll use a servo and an LED NeoPixel strip.

1 x 5V 2A (2000mA) switching power — https://www.adafruit.com/product/276
Barrel connection which matches Crickit

---

1 x Micro servo — https://www.adafruit.com/product/169
A micro servo to move with Crickit

---

1 x NeoPixel Strip 30 LEDs — https://www.adafruit.com/product/2562
This version has wires that will connect to the Crickit

---

1 x USB cable - USB A to Micro-B - 3 foot long, data + power — https://www.adafruit.com/product/592
Connect your micro:bit to your computer, a longer cable if needed.

Tools

You can use your own wire stripper, but if you don't have one, this one is great.

1 x Hakko Professional Quality 20-30 AWG Wire Strippers — https://www.adafruit.com/product/527
Cuts like butter

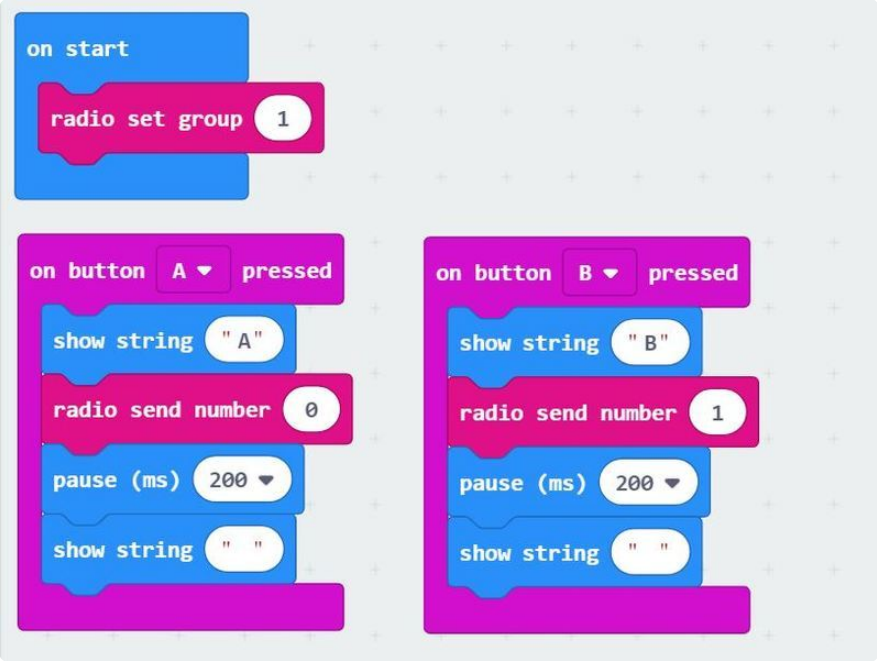
Transmitter Code

For this tutorial, we'll use Microsoft MakeCode for micro:bit. MakeCode allows for some very interactive projects with just a few connected code blocks.

Go to makecode.microbit.org (https://adafru.it/yEW) to get started. If you'd like to start learning Makecode, check the first project under **Tutorials**.

For this project, we'll use two pieces of code: one for the transmitting (remote controller) micro:bit. And another MakeCode program for the receiving micro:bit which will include blocks for using Crickit.

Transmitter MakeCode



Open this code in MakeCode for micro:bit

https://adafru.it/E9H

The radio controls are in the **Radio** block group (pink). On the start of the program, we set the `radio group` to `1`. If multiple micro:bits are in a room, you can keep them from interfering by setting unique radio groups. We do need to keep this number the same as our receiver micro:bit on the next page.

The rest of the code is in two Input blocks: `on button A pressed` and `on button B pressed`. When button A is pressed, the code within the `on button A pressed` block is run, the same for B.

The actions for both are very similar: Display the letter of the button pushed with `show string`, send a unique command (0 for A, 1 for B) via `radio send number`, `pause` a bit so you can see the letter on the display and clear the display by `show string` with a blank character.

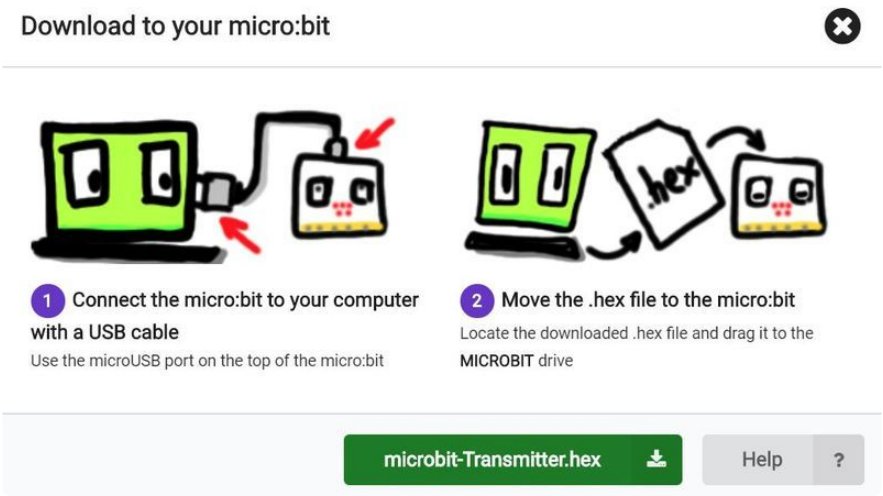
If you want to send more commands, I would suggest grabbing the `on shake` block. This allows you to control things via 11 different gestures (click `shake` and the editor will show you the range of options). The `on button` block also accepts on button `A+B`.

Load the Code

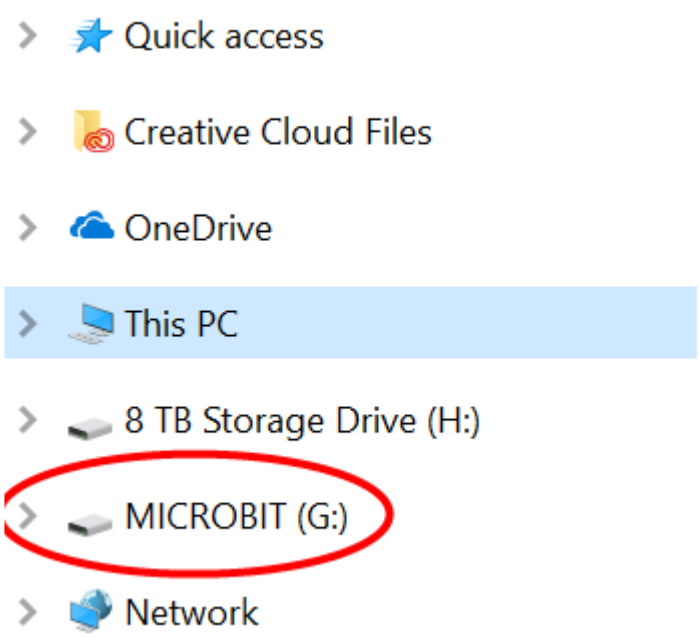
In this area of the editor, select a unique name for your code, I used **transmitter**.

If you click the download button it is similar, the editor will ask you to save the file on your computer storage.

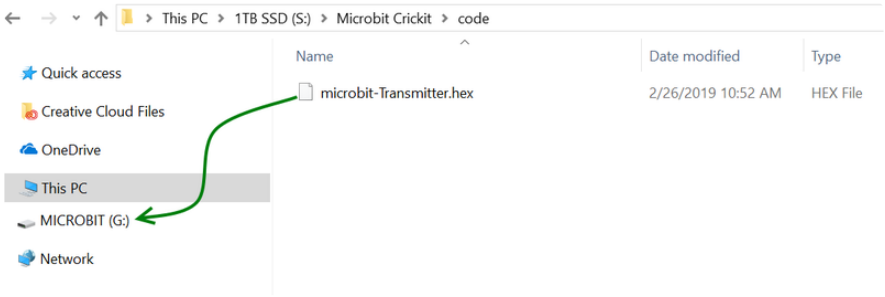
The instructions that pop up are what we follow to download the code.



Be sure the micro:bit you want to program is plugged in via a good USB data+power cable. The editor saves the code in a file named **microbit-projectname.hex**. Our project name is **Transmitter**. Go to your file explorer / finder on your computer. With the micro:bit plugged in via USB, you should see a new flash drive pop up named **MICROBIT**.

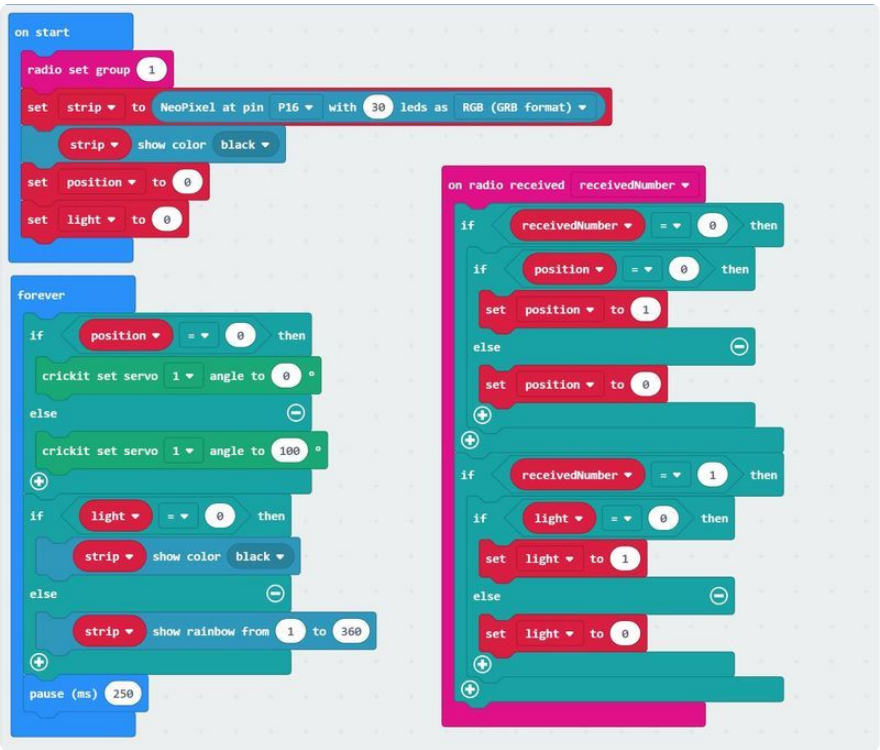


Navigate to the directory where you saved your **.hex** file. Using the mouse, drag the **.hex** file over to the **MICROBIT** drive. The micro:bit should program itself and start running your code. Press the A and B buttons and you should see the letter **A** or **B** on the LED matrix.



Receiver Code

The receiver code contains a few more blocks than the transmitter. We'll go over things to see why.



Open the Receiver code in MakeCode for micro:bit

https://adafru.it/E9I

Download

Download the Receiver code into the second micro:bit per the same method used on the previous page to load the transmitter code.

Click Edit in the upper right corner of the code display to get into the MakeCode editor. You can then save the program. Save it as **Receiver**.

What the Program is Doing

This code has two new sets of blocks:

- `strip` blocks work with NeoPixel LED strips via the NeoPixel Extension
- `crickit` blocks work with the Adafruit Crickit robotics board via the Crickit Extension

If you load the code via the green button above, MakeCode will load those extensions for you. They provide two new block groups, **Crickit** and **NeoPixel**, which provide robotics and light blocks respectively.

In `on start`, the code sets things up. It sets the `radio group` just like the transmit code. It also sets up a NeoPixel strip with 30 LEDs. Pin P16 is the pin which does NeoPixels for micro:bit on Crickit. We set the whole strip to black which turns all the LEDs off. Then we set two variables, `position` and `light`, for later use.

The other two loops work with each other to process your radio code:

- the `forever` loop acts on the values of `position` and `light` to move a servo or toggle the NeoPixels.
- the `on radio received` loop toggles the variables `position` and `light` if the right radio message is received.

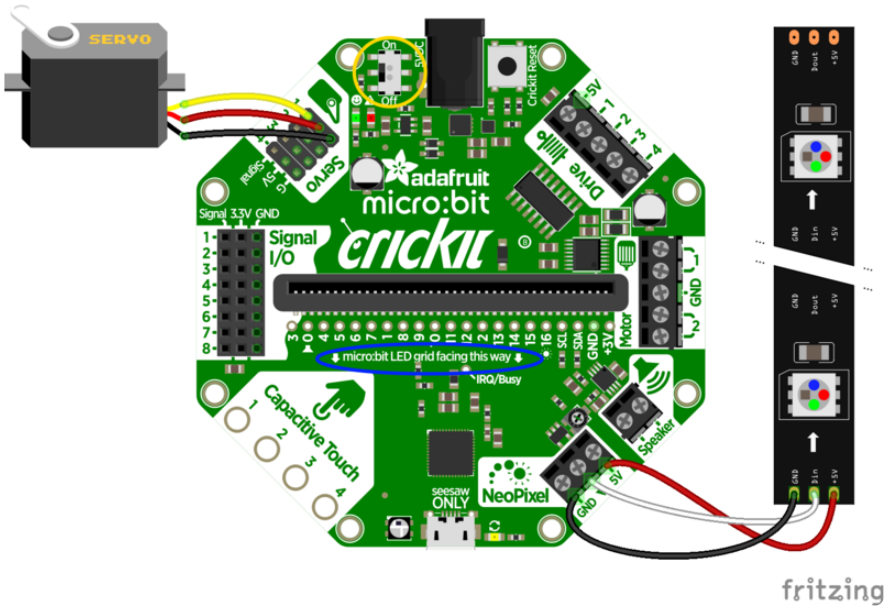
The `on radio received` loop looks for the button press messages 0 and 1 placed in the value `receivedNumber` by MakeCode. If it receives them, the loop will toggle the value of `position` (for button A presses) and `light` (for button B presses). If one of those variables is `1`, it will set it to `0` and if the value is `0` it will set it to `1`.

The `forever` loop does the work for us. It acts on the values of `position` and `light`. If `position` is 0, the angle of the servo is set to zero else it is set to 100 degrees. If `light` is set to 1 the NeoPixel strip is set to a rainbow pattern, otherwise it is turned off.

If you want to add actions, you can have the `on radio received` block look for other radio commands and set a variable. Then add an action in `forever` that acts on the variable.

Next to construct our circuit.

Wire It Up



Plug your servo motor into the Crickit **Servo** block on position 1, such that the lightest color (yellow most likely) faces away from the Crickit and the darkest color (brown or black) wire is closest to the center of the Crickit board.

Wire your NeoPixel strip to the NeoPixel block on Crickit with the black ground wire to GND, the red power wire to 5V and the white signal wire from the Crickit arrow terminal in the middle to the strip Din (Data In) pin. You'll need to strip the ends of the wires if they are not already exposed. Open each terminal block slot using a small screwdriver (the screw does not come out), insert the wire, and then tighten for a snug fit.

Plug the micro:bit you programmed as receiver into the Crickit. Note the text on the Crickit circled in blue - the micro:bit LED matrix should face that text when plugged in. Unfortunately the micro:bit can be plugged in the wrong way. Check twice the LEDs face the way the text indicates.

Put the AAA batteries in the battery pack and plug the battery pack into the transmitter micro:bit. You can press buttons A and B to ensure it is working as the display will light up A or B when the buttons are pressed. If it does not work, be sure the programming was done using the Transmitter Code page.

Plug the wall power transformer into the wall and into the Crickit black power jack. Be sure the on/off switch, circled in yellow, is set to on. A small green LED near the servo connections will light up if the board is happy. The servo may twitch, that is good.

Now we're ready to try it out.

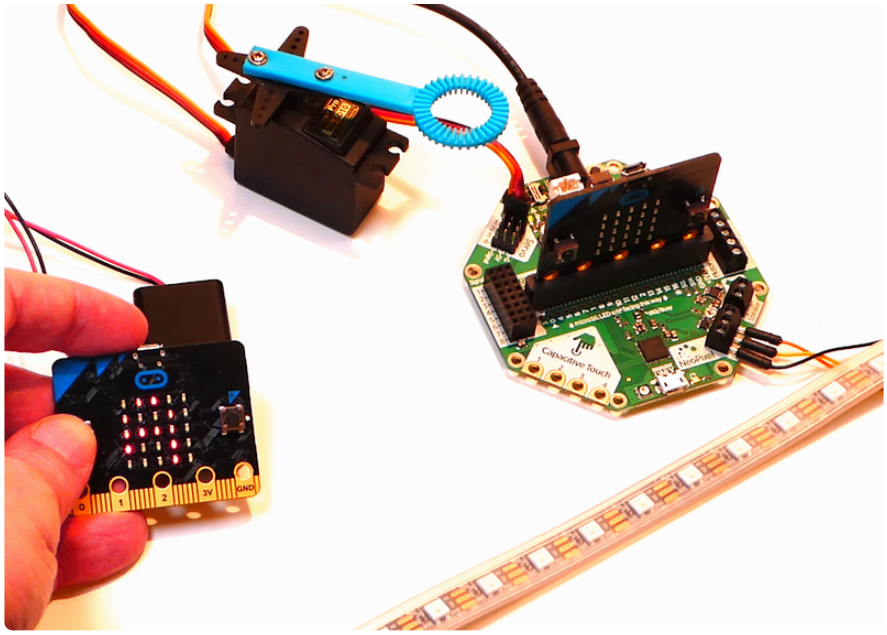
Troubleshooting

If your servos don't work, try testing your Crickit and servo with the simple programs in the guide Make it Move with Crickit (https://adafru.it/19zd). If they work there, then there is an issue elsewhere in your build.

If the simple case fails to have your servo work, check:

1. Wiring: to be sure you have the connections right. Non-Adafruit servos may have different connectors and wire colors.
2. Power: motors are power hungry. Be sure the Crickit is connected to a power source that can supply the voltage and current the servo(s) need. Also if you have multiple motors or other power hungry items like NeoPixels, size your power supply appropriately. See this guide (https://adafru.it/wbm) for NeoPixel power tips.
3. USB: If you have Crickit connected to USB, ensure it's via a known good USB cable which has both power and data lines. Power only USB cables should be avoided.

Use



If all is well, you can use your transmitter micro:bit (with battery pack) and you can press the A and B buttons and have A and B appear on the screen.

What is happening at the receiver end?

When the micro:bit + Crickit get the button A press, it moves the servo motor. One press moves it to 100 degrees, another press moves it back to zero.

For button B, one press lights the NeoPixel LED strip to a rainbow of colors, a second press turns it off.

Congratulations, you're using radio control!

Next Steps

Now that you have solid examples of a transmitter and receiver, what would you like to build? A radio-controlled robot? Mood lights for your room?

You can review the numerous examples of Crickit projects in the Adafruit Learning System (https://adafru.it/BXT) to get ideas for Crickit-based projects. Don't worry if they use CircuitPython, you can come back to MakeCode and look to implement your own version. Have fun!