littleBits and Digital I/O

Robotics Curriculum
IT Adventures

Lesson Overview

- Learn what the littleBits topper kit is and how to incorporate it into the RVR
- Learn how to program the littleBits with the micro:bit
- Use the previous lessons to interact with the littleBits digital I/O platform and create more complex, functional robots with the RVR

littleBits Overview

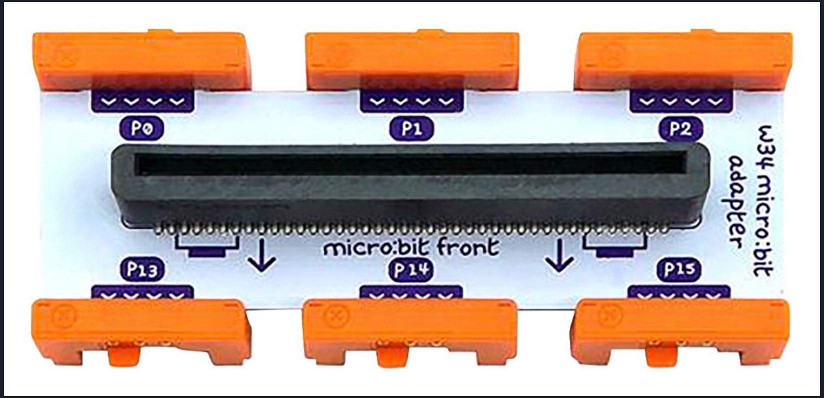
The littleBits topper kit is an extension of the micro:bit controller and contains a series of sensors for extending the functionality of the robot. These are split into the following areas, which will be explored over the next few weeks:

- Digital I/O - Digital is considered a binary on or off state, similar to a switch or button.
- Analog I/O - Analog is any input or output that can be multiple states, and usually is a range (such as from 0 to 255).
- Remote I/O - While not a fully separate category, this will be explored in coming weeks to examine how a remote control can be used to change the RVR's functionality from a distance.

This week, we will be using the digital sensors - similar to what you've used in the previous challenges on the micro:bit - to examine how to interact with the littleBits kit.

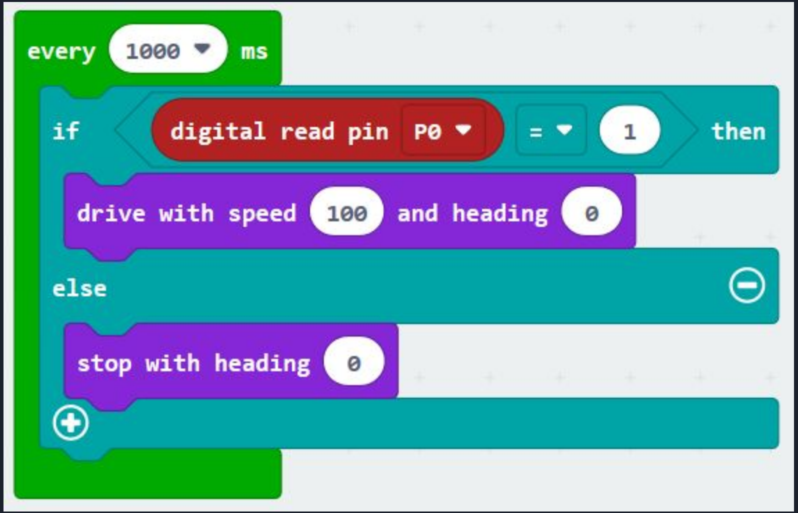
Circuit Example

For controlling the littleBits, you won’t be interacting with the components directly. Instead, you’ll be interacting with the components through the *pins* on the micro:bit. The first step to any circuit in this setting is connecting to the micro:bit adapter (as seen below). The pins P0, P1, and P2 are inputs, with P13, P14, and P15 being outputs.



Go ahead and connect a button to P0; we will use that for the example of a digital input. It should magnetically snap into place if you have it lined up correctly.

Code Example



```
every 1000 ms
    if digital read pin P0 = 1 then
        drive with speed 100 and heading 0
    else
        stop with heading 0
```

Continuing with that circuit built, this code will check if P0 is set to 1 every 1000 ms (or 1 second). If it is, then it will drive with speed 100; otherwise, the RVR will stop. Since we know that P0 is connected to the button, we can figure out that it will run when the button is pressed. The pin settings (for reading, writing, and more) can be found under “Advanced”->“Pins”. Go ahead and mess around with the pins, exploring different inputs and outputs. While we are explicitly focused on digital I/O this week (it’s either on (1) or off (0)), take a look at all the possibilities!

Primary Learning Challenge: Button Bumper Challenge

Attach the button to the front of the RVR and write a program so that the RVR will run slowly until it hits a wall, at which point it will stop and make a noise using the buzzer. While creating this program, you should be thinking about your previous lessons: what might come into play? Is there any logic in effect? What is your input and output?

Components Needed:

- littleBits Button
- littleBits Buzzer

Secondary Learning Challenge: Bulldozer Simulator Challenge

Construction equipment or other large vehicles often make a noise while backing up. For this challenge, create a program that simulates such a vehicle by doing the following:

- Using the littleBits button, back up the RVR when it is pressed and make the RVR stop when it is pressed again. This should be in an infinite loop.
- While the RVR is backing up, it should be flashing the LEDs on the micro:bit and the RVR.
- When the RVR is backing up, the littleBits sensor should be making a beeping tone, similar to a construction vehicle.