LESSON 5

Microbit Traffic Lights

40 minutes | Building Basic Robotic Systems

WHAT THIS LESSON DOES

In this step-by-step lesson, you'll create a Microbit project, add the Stopbit extension, and test all the lights. You'll learn about sequences in coding and program your traffic lights to display a sequence using on/off and state methods. You'll test your sequences to ensure they're working correctly.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand and apply the process of creating a new Microbit project.
- Learn to add and utilise the Stopbit extension for programming traffic lights kit.
- Gain skills in testing and troubleshooting the functionality of the lights.
- Comprehend the concept of 'sequence' in coding and apply it to program traffic lights.
- Develop proficiency in programming the sequence of traffic lights using on/off and state methods.

BEFORE THE BELL

- Count out one micro:bit, one Kitronik STOP:bit traffic lights kit and one USB cable per student or pair. Check kits the day before: lights seated, terminal screws tight, micro:bits charged or battery packs fresh.
- Open makecode.microbit.org on the school network and confirm students can create a project and download to a micro:bit.

Microbit Traffic Lights

HOW THE LESSON RUNS

Create a new Microbit project
Add the Stopbit extension
Test all the lights
What is a sequence?
Program the sequence using on/off
Program the sequence using state

TEACHING MOVES

- 1. Create a new Microbit project. Get everyone to makecode.microbit.org and a blank micro:bit project before you move on. Wait until every screen shows the empty workspace.
- 2. Add the Stopbit extension. Show Extensions, search stopbit, and click the traffic lights kit image. Check that Kitronik STOP:bit now appears in the toolbox before they code.
- 3. Test all the lights. After they flash the test code, have them press A, B and A+B in turn. If a light flickers or stays dark, tighten the terminal screws before blaming the code.
- 4. What is a sequence? Say the shoe steps out of order so they hear why order matters. Then read the traffic sequence aloud: green 5s, yellow 1s, red 2s.
- 5. Program the sequence using on/off. Have them delete the test code first. Stress that each light must be set on or off explicitly, with pauses, inside forever. Check timings on a real kit.
- 6. Program the sequence using state. Show that one state block sets the whole pattern (Stop, Get Ready, Go, Ready to Stop). Ask how this differs from switching each light by hand.

WHAT TO WATCH FOR

- Students think the program worked because it downloaded, even when the light order or timings are wrong. Watch the kit through one full cycle together. Call out green 5s, yellow 1s, red 2s and match what they see to the blocks.
- A light flickers or fails and they rewrite code instead of checking the hardware. Pause coding. Check the STOP:bit seating and tighten the screw terminals, then reflash the same test code.
- When using on/off, they turn a light on and forget to turn the others off, so two lights show at once. For each step of the sequence, ask which lights should be on and which off, then set all three explicitly.

DIFFERENTIATION

- Emerging: Pair them with someone whose kit already responds to A and B, and stay with them through adding the Stopbit extension and the first flash. Let them complete the button test and the on/off sequence only; the state method can wait.
- Developing: Prompt them to read their forever block aloud against the green-yellow-red timings before they download. Have them explain to you why each on/off step needs the other lights set off.
- Proficient: Ask them to rebuild the same timings using only state blocks and compare how many blocks each method needs. Have them teach another pair the difference between on/off and state using their working kit.