LESSON 33

Microbit Traffic Lights

40 minutes | Designing and Building for the Future

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 LESSON

- Open makecode.com yourself and search the Stop:bit extension so you know the flow before pupils hit it, and check the school filter allows the site.
- One device per pupil or pair, each with a Microbit and a Kitronik traffic lights kit attached.
- Have a screwdriver or check the kit screws are tight, as loose screws cause flickering lights in step 3.
- Confirm you can download and send code to a Microbit on your setup, as this happens repeatedly.

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. Walk pupils to makecode.com together and pause until every screen shows a new, empty project before anyone moves on. Nobody catches up later.
- 2. Add the Stopbit extension. Stress that they click the traffic lights kit picture, not just any search result. Once the Kitronik STOP:bit category appears in the toolbox, have them point to it so you can spot who missed it.
- 3. Test all the lights. Before any programming, get all three lights confirmed working with A, B and A+B. Any flickering means loose screws, fix it now rather than mid-sequence when they will blame their code.
- 4. What is a sequence? Use the sock-then-shoe example, then ask pupils what breaks if you swap the order. Land the point that a computer runs instructions in the exact order given, nothing more.
- 5. Program the sequence using on/off. Make them delete the test code first. Watch the timings: five seconds green, one yellow, two red. Have them count the seconds out loud against the lights to verify.
- 6. Program the sequence using state. Draw out that this produces the same result as step 5 by a different method. Ask which they found clearer and why, so state versus on/off becomes a real comparison.

WHAT TO WATCH FOR

- Pupils blame their code when a light flickers or fails, when the real cause is a loose screw or connection on the kit. Separate hardware from code early. If a light misbehaves, check the physical connection first before touching the blocks.
- Pupils put the blocks in the wrong order but expect the lights to cycle correctly, assuming the Microbit will sort it out. Return to the sock-and-shoe example. The computer runs instructions top to bottom exactly as written, so wrong order means wrong sequence.
- Pupils confuse the wait time with when a light is on, so a light stays lit during the following light's time. Have them trace each light: turn it on, wait, then check they turned it off before the next colour starts.

DIFFERENTIATION

- Emerging: Sit beside them for the extension step, the point most get stuck, and confirm the Stop:bit category is in their toolbox before they build anything. Let them copy the on/off sequence block by block from the board without worrying about the state method yet.
- Developing: Ask them to say the sequence aloud before they run it, then check the lights match their words. Have them fix a deliberately wrong timing you point to, rather than only building from scratch.
- Proficient: Once both methods work, ask them to explain to a partner why state and on/off give the same result. Challenge them to change the timings to model a real junction and justify their choices.