LESSON 43

Tilt Remote Control Car

60 minutes | Designing and Building for the Future

WHAT THIS LESSON DOES

In this step-by-step lesson, you'll learn to control a Move Motor car using a Microbit as a remote controller. You'll program the Microbit to send directional commands to the car based on its tilt. You'll also create code for the car to respond to these commands, resulting in a fully remote-controlled car.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand and apply the concept of radio communication between two Microbits.
- Programme a Microbit to send specific messages based on different gestures.
- Develop the ability to programme a Move Motor car to respond to different messages received.
- Test and debug the code to ensure the car responds correctly to the remote control.
- Extend the project by adding additional features such as LED light changes.

BEFORE THE LESSON

- Two charged Microbits and one Move Motor car per pair or group, plus USB cables for download.
- Open and test https://makecode.microbit.org on class devices; confirm the site is not blocked.
- Clear floor space to drive and test the car safely once both codes are loaded.

Tilt Remote Control Car

HOW THE LESSON RUNS

- Introduction
- Program the remote
- Send message 'forward'
- Send messages 'left', 'right' & 'reverse'
- Send message 'stop'
- Download the code
- Create a new project
- Receive message 'forward'
- Receive message 'left'
- Receive message 'right'
- Receive message 'reverse'
- Receive message 'stop'
- Download the code
- Challenge

TEACHING MOVES

- 1. Introduction. Name the two roles out loud: one Microbit is the tilt remote, one sits in the Move Motor car. Check every pair has both Microbits and a car before opening MakeCode.
- 2. Program the remote. Create a new project named for the remote and add radio.setGroup(1) together. Stress that the car project must use this same group or nothing will link.
- 3. Send message 'forward'. Add the LogoDown gesture to send the string forward and show the north arrow. Have pupils tilt logo-down once so they see the arrow before adding more gestures.
- 4. Send messages 'left', 'right' & 'reverse'. Add TiltLeft, TiltRight and LogoUp the same way. Read each sent string aloud so spellings stay exact for the car code later.
- 5. Send message 'stop'. Add ScreenUp to send stop and show the No icon. Pupils lay the Microbit flat to check the icon appears.
- 6. Download the code. Download onto the remote Microbit only. Test all five gestures and confirm the matching arrow or icon before anyone starts the car project.
- 7. Create a new project. New project for the car. Add the kitronik-move-motor extension first, then set radio group 1 so it matches the remote.
- 8. Receive message 'forward'. Build the receive handler for the string forward and the forward motor action. Point pupils to the text block in Advanced so they can type the exact word.
- 9. Receive message 'left'. Expand the if with else if and add the left turn response. Keep the compared string identical to the remote's left message.
- 10. Receive message 'right'. Add the next else if for right and the turn-right motors. Glance at pairs' string spellings before they move on.
- 11. Receive message 'reverse'. Add else if for reverse the same way. Keep the chain tidy so each message has one clear motor response.
- 12. Receive message 'stop'. Add the final branch so stop halts the motors. Ask one pair to walk through the full if chain before download.
- 13. Download the code. Download to the car Microbit, power on, and test with the remote. If one direction fails, compare that message's send and receive strings side by side.
- 14. Challenge. For pairs who finish cleanly, add LED colour changes on the car for each received message so direction is visible from across the room.

WHAT TO WATCH FOR

- Remote and car use different radio groups, so messages never arrive. Check both projects show radio.setGroup(1) before download. Re-flash both Microbits after any group change.
- Send and receive strings do not match exactly (spelling or capitals), so one direction never triggers. Pick the failing direction. Read the remote sendString and the car compare text aloud character by character, then re-download.
- Pupils load remote code onto the car Microbit or mix the two boards up. Label boards remote and car before coding. After each download, test gestures alone on the remote before seating the car board.

DIFFERENTIATION

- Emerging: Work gesture by gesture on the remote: add one send, download, confirm the arrow, then add the next. Pair with a partner who holds the car checklist while they match each receive string.
- Developing: Have them prove all five remote symbols before opening the car project. When a direction fails, ask them to trace only that message on both projects rather than rewriting everything.
- Proficient: Add the challenge: different LED colours on the car for forward, left, right, reverse and stop. Ask them to explain why both Microbits must share one radio group before they extend the code.