STEM ARDUINO PBL — PROJECT 1 | TIER 1: FOUNDATION

# Smart LED Control System

Full PBL Lesson Plan, Assessment Rubric & Teaching Resources

*Grades 7–9 • 2 Weeks • SDG 3, SDG 10 •*

| PROJECT SNAPSHOT | |
|---|---|
| **Driving Question** | "How can we use programmable lighting to create an accessible home environment for elderly and disabled users?" |
| **Real-World Angle** | Remote-controlled lighting for accessibility — enabling people with limited mobility to control their environment independently without physical effort |
| **Grade Level** | Grades 7–9 (suitable as first Arduino project; no prior programming experience required) |
| **Duration** | 2 weeks (10 × 45-minute lessons, or equivalent) |
| **SDG Connections** | SDG 3 — Good Health & Well-Being \| SDG 10 — Reduced Inequalities |
| **Key Components** | Arduino UNO · IR receiver (TSOP38238) · IR remote · LEDs (red, green, blue) · 220Ω resistors · Breadboard · jumper wires |
| **Core Skills** | Input → Processing → Output model · Digital and PWM signals · IR signal decoding · Conditional logic (if/else) · Brightness control with analogWrite() |
| **21st C. Skills** | Empathy and user-centred design · Critical thinking · Collaboration · Communication to a non-technical audience |
| **Public Audience** | Elderly residents, disability support workers, or school community / parents |
| **Entry Products** | User empathy map + accessibility needs analysis |
| **Final Products** | Working programmable LED device + user guide + showcase presentation |

| CURRICULUM STANDARDS COVERAGE | |
|---|---|
| **Science / Physics** | Electrical circuits (series/parallel, resistance, voltage, current) · Light as electromagnetic radiation · Visible spectrum and colour mixing (RGB) · Introduction to digital vs analogue signals |
| **Technology / ICT** | Input → Processing → Output model · Programming logic (variables, conditionals, loops, functions) · Iterative hardware-software development · Systems thinking at the component level |
| **Mathematics** | Ohm's Law calculations (V = IR, selecting resistor values) · PWM duty cycle as a percentage · Mapping brightness values (0–255 scale) · Reading and interpreting a technical datasheet |

| | |
|---|---|
| **Design Technology** | Human-centred design process · Accessibility and universal design principles · Prototype → test → iterate cycle · Evaluating a design against user requirements |
| **Literacy** | Technical writing (user guide) · Persuasive communication (accessibility case) · Oral presentation to a non-technical audience · Reading and annotating circuit diagrams |

## Why This Is a Genuine PBL Project — Not Just a Workshop

Project 1 is the gateway to the entire STEM Arduino progression. Done well, it establishes the PBL habits — empathy-first design, inquiry before instruction, iteration over perfection, and authentic audience — that every subsequent project builds on. Done poorly, it becomes a recipe-following exercise that produces working circuits but no learning. **The distinction lies entirely in facilitation.**

| PBL ELEMENT | HOW IT APPEARS IN THIS PROJECT |
|---|---|
| **Centrality** | The lighting-for-accessibility challenge organises the entire two weeks. Students learn IR decoding, PWM, and conditional logic because they need those tools to answer the driving question — not as isolated exercises. |
| **Driving Question** | The question foregrounds the user (elderly and disabled people), not the technology. Students must understand human need before they understand IR remote protocols. This ordering is deliberate and non-negotiable. |
| **Constructive Inquiry** | Students investigate: which IR codes does their specific remote send? Why does a PWM value of 128 produce half-brightness? What resistor is needed for their LED? None of these answers are given upfront. |
| **Student Autonomy** | Teams choose which buttons control which behaviours, how many brightness levels to include, and how to present to their audience. The teacher defines the outcome (accessible lighting device) — students define the implementation. |
| **Realism / Authenticity** | Lighting inaccessibility is a genuine daily challenge for millions of people. The public audience — ideally a care worker, elderly resident, or disability advocate — brings authentic accountability that peer review cannot replicate. |

## Week-by-Week Lesson Calendar

| WEEK 1: ENTRY EVENT & INQUIRY | | | |
|---|---|---|---|
| **Lesson** | **PBL Phase** | **Learning Activities** | **Teacher Facilitation Notes** |
| **L1** | **Entry Event** | • Launch: Show 3-minute video of an elderly person struggling with wall light switches during the night<br>• Share data: 1 in 6 people globally live with a disability; limited mobility is the most common form<br>• Pose the Driving Question — students discuss in pairs, then share with class<br>• Record student 'Need to Know' questions on class board — do not answer them yet | → Do NOT show an Arduino or LED yet — the need must precede the technology<br>→ Listen for empathy responses: 'That looks really dangerous' — these are your entry points<br>→ Ask: 'What would make this person's life meaningfully easier?' Not: 'What could we build?' |

| | | | |
|---|---|---|---|
| **L2** | **Entry Event** | • User empathy mapping: Who is our user? What do they see, hear, feel, need, fear?<br>• Accessibility needs analysis: research 2–3 real conditions affecting lighting use (e.g. arthritis, stroke, MS, visual impairment)<br>• Compare existing solutions: Clapper switches, voice assistants, commercial smart bulbs — cost, complexity, limitations<br>• Entry product: completed empathy map + 'Our user needs...' design brief | → Prompt: 'What does a £200 Amazon smart bulb system assume about the user that our device should not?'<br>→ Listen for: cost barriers, technology literacy gaps, reliability concerns, setup complexity<br>→ Formative: empathy map — depth of specific user detail, not generic 'old person' |
| **L3** | **Inquiry — Circuits** | • Mini-lesson: current, voltage, resistance — why LEDs need a resistor (V = IR calculation)<br>• Students calculate the correct resistor for their LED colour using datasheet values<br>• Build: light one LED with a 220Ω resistor on the breadboard; verify it works<br>• Explore: what happens without a resistor? (brief, supervised demonstration only) | → Ask: 'What happens to current if resistance doubles? Use the formula — don't guess.'<br>→ Expected error: students put resistor after LED rather than before — explain either position is valid, but ask why<br>→ Formative: resistor calculation shown in their notebook with working — not just the answer |
| **L4** | **Inquiry — Digital Control** | • Mini-lesson: what is a digital signal? HIGH vs LOW, 0V vs 5V, on vs off<br>• Students write first Arduino sketch: blink an LED using digitalWrite() and delay()<br>• Experiment: change delay values — what is the shortest blink they can see? (introduces PWM concept)<br>• Introduce pinMode(), setup() vs loop() — draw the execution flow on a whiteboard | → Key question: 'Why is there a setup() and a loop()? What would happen if everything was in setup()?'<br>→ Expected confusion: students put pinMode() inside loop() — show the consequence (it runs 1000×/second but works — then ask if that's good practice)<br>→ Formative: annotated code — students add a comment to every line explaining what it does |
| **L5** | **Inquiry — IR Remote** | • Mini-lesson: how does an IR remote work? (IR LED, carrier frequency, signal modulation)<br>• Wire the IR receiver to Arduino (signal, power, ground); load IRremote library<br>• Run IR receive sketch: press buttons; read hex codes in Serial Monitor; record in a table<br>• Map remote buttons to intended functions: which button = on? which = dim? which = colour change? | → Key question: 'Why does the TV remote not interfere with your Arduino? What is the carrier frequency for?'<br>→ Expected issue: library not installed — teach the Library Manager process as a real-world skill<br>→ Formative: IR code table — at least 6 buttons mapped with hex values AND intended function |

## WEEK 2: BUILD, UPGRADE & SHOWCASE

| Lesson | PBL Phase | Learning Activities | Teacher Facilitation Notes |
|---|---|---|---|
| **L6** | **Build & Iterate** | • Introduce PWM: what is it? Why does analogWrite() produce apparent brightness control?<br>• Students write code: one IR button → LED on (full brightness), another → LED off | → Ask: 'If analogWrite(9, 128) gives 50% brightness, what value gives 25%? Test your prediction.'<br>→ Expected error: using a non-PWM pin for analogWrite — Arduino silently ignores it; ask |

| | | | |
|---|---|---|---|
| | | • Add second button: LED at 50% brightness using analogWrite(pin, 128)<br>• First integration test: IR remote → Arduino → LED responds. Debug any failures. | students to find which pins have the ~ symbol<br>→ Formative: debugging log — problem observed, hypothesis, change made, result |
| **L7** | **Build & Iterate** | • Extend brightness control: add 3+ distinct levels mapped to different remote buttons<br>• Add a second LED colour (if using RGB): map colour-change to a remote button<br>• Test usability: can a team member operate the device without looking at the code?<br>• Peer circuit review: partner team checks wiring safety and button mapping logic | → Ask: 'From your user's perspective — is 3 brightness levels enough? What would they want at 2am vs reading?'<br>→ Challenge: 'Can you make the LED fade gradually rather than jump between levels?'<br>→ Formative: peer review form — wiring safety, code readability, function mapping |
| **L8** | **Upgrade Challenge** | • Introduce upgrade options (differentiated by readiness):<br>• Upgrade A: Add a third LED colour and a dedicated remote button for each — full RGB control<br>• Upgrade B: Add smooth fade effect using a for() loop and small delays between brightness steps<br>• Upgrade C: Add a 'night mode' — single press cycles through: off → dim red → off (non-disruptive for sleep)<br>• Teams plan their upgrade before building: draw the new logic flow on paper first | → Upgrade B introduces loops as a tool for animation — connect to the user: 'A hard switch is jarring at night; a fade is gentler on the eyes'<br>→ Upgrade C requires state tracking with a variable — scaffold with: 'How does the Arduino remember which mode it is in?'<br>→ Formative: upgrade plan sketch — logic flow drawn before any code is written |
| **L9** | **Prepare Showcase** | • Draft the user guide (see Handout 3) — tested with a non-technical person<br>• Prepare the showcase presentation: user story first, technology second<br>• Practice: explain the project to someone who has never heard of Arduino (use a partner from another team)<br>• Prepare live demo protocol: what if the IR signal doesn't register? Have a backup plan. | → Teach: 'Your first sentence should be about a person, not a circuit. Start with: Our user is...'<br>→ Common weakness: students explain HOW it works but not WHY the design serves the user — push them to answer the Driving Question explicitly<br>→ Formative: practice presentation run-through — does the explanation connect technology to user need? |
| **L10** | **Public Showcase** | • Showcase event: teams present to invited non-technical audience<br>• Each team: 4-minute presentation + 2-minute live demo + Q&A<br>• Audience members complete feedback forms (see Handout 4)<br>• Individual reflection completed in final 10 minutes (see Handout 5) | → Invite: care worker, elderly relative, school nurse, SENCO, or parent unfamiliar with Arduino<br>→ Teacher role during presentations: observer and timekeeper only — do not intervene<br>→ Summative: presentation rubric (criteria 4 + 5) + user guide submission (criterion 6) |

## Detailed PBL Phase Plans

## Phase 1 — Entry Event: Establishing the Human Need

| SUB-PHASE / TASK | TEACHER ACTIONS | STUDENT ACTIONS + PRODUCTS |
|---|---|---|
| **Entry Event Launch (Lesson 1)** | Show the video without framing it as a STEM problem. Ask open questions: 'What did you notice?' 'How did that make you feel?' 'What would help?' Let empathy precede design. | Watch and respond emotionally first, analytically second. Generate 'Need to Know' questions as a class. Entry ticket: 'Describe in 2 sentences what problem we are trying to solve — without mentioning LED or Arduino.' Product: Need to Know board. |
| **Empathy Mapping (Lesson 2)** | Facilitate empathy map completion. Push beyond generic responses: 'You wrote they feel frustrated — frustrated about what specifically? What triggers it?' Research accessibility conditions as a class. | Complete empathy map with specific, named user scenario. Research 2 real conditions. Compare 3 existing solutions and their limitations. Product: Empathy Map + Design Brief ('Our user needs...'). |
| **Entry Event Assessment** | Check: is the design brief specific to a person with a named condition? Or is it generic? Generic briefs ('elderly people need easier lights') will produce generic designs — redirect to specificity. | Self-assessment: 'Can I describe my user in enough detail that a stranger would know exactly who I'm designing for?' If not — go back to the empathy map. |

## Phase 2 — Inquiry: Learning What Is Needed to Build

| SUB-PHASE / TASK | TEACHER ACTIONS | STUDENT ACTIONS + PRODUCTS |
|---|---|---|
| **Circuit Fundamentals (Lesson 3)** | Pose the problem without giving the formula: 'This LED will burn out if we give it too much current. How do we limit it?' Let students find Ohm's Law as the answer, rather than receiving it as a fact. | Calculate correct resistor value from LED datasheet. Build single-LED circuit. Verify with multimeter if available. Record: what voltage drop, what current, what resistor? Product: Resistor Calculation + Verified Circuit. |
| **Digital Signals (Lesson 4)** | Ask: 'What does HIGH actually mean in volts? Measure it.' Students use a multimeter or observe the LED to confirm 5V = on, 0V = off. The blink sketch is an investigation tool, not a recipe. | Write, annotate, and modify the blink sketch. Experiment: what delay makes the LED appear solid? (introduces persistence of vision / PWM preview). Product: Annotated blink code with student comments on every line. |
| **IR Remote Investigation (Lesson 5)** | Do NOT give students the hex codes for their remote. The investigation IS the lesson. Ask: 'What do you notice about the hex codes — are they random? Are they consistent across multiple presses?' | Map minimum 6 remote buttons to hex codes. Identify: is there a pattern in the codes? (Many remotes use a consistent prefix.) Decide button-to-function mapping for the design brief. Product: IR Code Table + Button Mapping Design Decision Document. |
| **Inquiry Assessment** | Update Need to Know board. Ask: 'Which of your week 1 questions can | Peer quiz: can you explain to your partner what PWM is without using |

| | | |
|---|---|---|
| | you now answer? What new questions did the inquiry create?' Unanswered questions become the focus of build-phase debugging. | the words 'pulse width modulation'? If not — what's missing from your understanding? |

| Phase 3 — Build & Iterate: Making It Work for a Real User | | |
|---|---|---|
| **SUB-PHASE / TASK** | **TEACHER ACTIONS** | **STUDENT ACTIONS + PRODUCTS** |
| **First Integration (Lesson 6)** | Circulate with one question only: 'What does the Serial Monitor tell you?' Teach students to read the monitor as their primary debugging tool. Do not touch their circuit. | Build the full IR → Arduino → LED circuit. Test: does pressing button 1 turn the LED on? Does button 2 dim it? Debugging log required for every failure. Product: Working First Prototype + Debugging Log. |
| **Usability Test (Lesson 7)** | After the circuit works technically, ask the accessibility question: 'Would your actual user be able to use this comfortably? Could they tell the difference between the dim and medium setting in a dark room?' | Test the device from the user's perspective. Adjust brightness levels based on what is perceptually distinct, not mathematically equal (e.g. 64/128/255 may be better than 85/170/255). Product: Revised Prototype + Usability Test Notes. |
| **Peer Critique (Lesson 7/8)** | Teach the critique protocol: warm (what works and why), cool (one specific concern), hard (one concrete suggestion). Model it with an example first. Teacher participates as a critiquer too. | Give structured critique to one other team. Receive critique. Document: what feedback was received, what change was made, and what was rejected and why. Product: Critique Form + Revision Log. |
| **Upgrade Build (Lesson 8)** | During upgrade building: ask 'How will you test that this upgrade works for your user — not just that it works technically?' The test condition should reference the user's context. | Build chosen upgrade. Test in simulated user context (e.g. test the night mode in a darkened corner of the room). Document: what upgrade, how tested, what was changed. Product: Upgraded Prototype + Test Protocol. |
| **Build Assessment** | Observe: are students testing systematically or randomly changing things? Debugging quality is assessed, not just whether the circuit works. | Self-assessment: 'Describe the hardest bug you fixed. What was the symptom? What was the cause? How did you find it?' Depth of explanation is the evidence of learning. |

| Phase 4 — Upgrade Challenge: Differentiated Extension | | |
|---|---|---|
| **SUB-PHASE / TASK** | **TEACHER ACTIONS** | **STUDENT ACTIONS + PRODUCTS** |
| **Upgrade A — RGB Control (Core+)** | Guide: 'Your user said they find bright white light painful in the evening. What colour would help? How would you map that to a remote button | Add third LED or switch to RGB LED. Map minimum 3 colours to dedicated remote buttons. Test: can the user switch colour with one button press from 2 metres away? Product: Multi- |

| | | |
|---|---|---|
| | simply?' Connect every technical decision back to the user. | colour LED control with documented button mapping. |
| **Upgrade B — Smooth Fade (Advanced)** | Introduce the for() loop in context: 'Instead of jumping from 0 to 255, we want to walk there in steps. How many steps? How big should each step be?' Let students experiment with step size and delay to find a perceptually smooth fade. | Implement fade-up and fade-down using for() loops. Test: at what step size and delay does the fade look smooth to the human eye? Document the values found. Product: Fade-in/Fade-out demo with user-evaluated smoothness test. |
| **Upgrade C — Night Mode (Extended)** | Introduce state tracking: 'The Arduino needs to remember which mode it is in between button presses. What kind of variable holds the current state?' Scaffold: 'If mode = 0 then off; if mode = 1 then dim red; ...' | Implement a mode variable that cycles through states on each press of one button. Test: can a user navigate from off to dim to brighter without any visual feedback other than the LED itself? Product: Mode-cycling demo with state machine code and annotation. |
| **Upgrade Assessment** | Differentiated: Upgrade A assessed on colour mapping accuracy and reliability. Upgrade B on smoothness and the ability to explain the loop mathematically. Upgrade C on state machine logic clarity and commenting quality. | Design decision log: what upgrade chosen, why chosen (must reference the user), how tested, what was adjusted. At least one failed approach documented with what was learned. |

## Phase 5 — Showcase: Authentic Audience & Honest Reflection

| SUB-PHASE / TASK | TEACHER ACTIONS | STUDENT ACTIONS + PRODUCTS |
|---|---|---|
| **User Guide (Lesson 9)** | Coach on audience: 'Your user guide reader has never seen an Arduino and may never want to. They just need to use the device. Write for them, not for your teacher.' | Write user guide. Test with a real non-technical person (another teacher, parent, younger sibling). Revise based on their confusion points. Product: User Guide (tested + revised). |
| **Showcase Prep (Lesson 9)** | Ask: 'If your user was sitting in the audience, would your presentation make them feel seen and understood? Or would they feel like a props in a technology demonstration?' This reframe often transforms presentations. | Prepare 4-minute presentation. First 90 seconds: the user's story and the problem. Final 90 seconds: what the device does and how it helps. Middle: the design journey, not just the result. Product: Presentation outline. |
| **Public Showcase (Lesson 10)** | Introduce the audience before teams present. Remind students: questions from the audience are a gift, not a threat. Stay calm, answer with evidence from your work. | Present to real audience. Demonstrate live device. Field questions. Record 2 pieces of feedback that surprise them. Product: Completed showcase + stakeholder feedback forms. |
| **Reflection (Lesson 10)** | Allow 10 quiet minutes for individual written reflection. Do not facilitate this | Complete Handout 5 reflection. At minimum: one technical learning, one |

| | | |
|---|---|---|
| | — let students think without prompting. | design decision they would change, one thing the audience taught them that testing did not. Product: Individual Reflection Document. |

## Project 1 Assessment Rubric

Distribute this rubric on Lesson 1 so it shapes students' work throughout the project, not just at the end. *Level 3 (Proficient) is the expected standard for all students. Level 4 represents work that could be shared as a model example for future cohorts.*

| CRITERION | 1 — Beginning | 2 — Developing | 3 — Proficient ✓ | 4 — Exemplary |
|---|---|---|---|---|
| **Technical Functionality (25%)** | Circuit does not function; LEDs do not light; IR receiver not responding; major wiring or code errors prevent any operation. | Circuit partially functions; LEDs respond to some IR commands but behaviour is inconsistent, brightness control is absent or incorrect, or remote buttons trigger wrong outputs. | Circuit functions reliably; IR remote controls LED on/off correctly for at least two buttons; at least two brightness levels achieved with analogWrite(); circuit is safe and components correctly rated. | System exceeds brief; includes three or more distinct brightness levels, colour-change modes via remote, smooth fade transitions using loops or delay(), and correct resistor calculations documented. |
| **Understanding of Input → Processing → Output (20%)** | No understanding demonstrated; student cannot identify which component is the input, which is the output, or explain what the Arduino does. | Surface understanding; student can label components as input/output but cannot explain the signal flow or what the processing step involves. | Clear understanding; student explains the complete signal path: IR receiver captures signal → Arduino decodes hex value → conditional logic routes output → LED responds via digital or PWM pin. | Deep understanding; student explains PWM duty cycle mathematically (e.g. 50% duty cycle = analogWrite(9,128)), connects IR carrier frequency to receiver sensitivity, and relates their circuit to a real commercial product (e.g. smart bulb internals). |
| **Accessibility Design & User Empathy (20%)** | No connection to accessibility context; project built as a circuit exercise with no reference to the user or their needs. | Mentions elderly or disabled users but does not demonstrate specific design decisions driven by user needs. | Design decisions clearly connected to identified user needs; student can explain why specific remote buttons were chosen (e.g. large buttons for arthritic hands), why brightness levels matter (e.g. night-time vs reading), and who the specific user is. | Deep empathy; student conducted or references research on specific conditions (e.g. limited fine motor control, photosensitivity, visual impairment); design choices are justified with reference to universal design principles; proposes a real deployment scenario with named user or organisation. |

| | | | | |
|---|---|---|---|---|
| **Collaboration & Team Process (15%)** | Limited contribution; student relied on others to build and code; team process not evident in work or reflection. | Contributed to some aspects; needed prompting to participate; limited evidence of shared decision-making. | Consistent equitable contribution; roles shared and rotated; team resolved disagreements productively; incorporated feedback from critique sessions into revised design. | Led or co-led team effectively; facilitated peer critique sessions; documented design decisions and the reasoning behind changes; actively supported struggling teammates. |
| **Communication & Showcase Presentation (20%)** | Presentation absent, incomplete, or unclear; audience not considered; no live demonstration. | Presentation covers the project but uses technical jargon unexplained to the lay audience; demo attempted but unreliable. | Clear, audience-appropriate presentation; Input → Processing → Output explained without jargon; live demo works reliably; accessibility impact of the design articulated; user guide provided. | Compelling presentation that tells the user's story first and the technology second; handles questions from non-technical audience confidently; user guide is usable by someone with no technical knowledge; suggests realistic next steps for deployment. |
| **Written User Guide (20%)** | User guide absent or a list of components only; no instructions a real user could follow. | User guide present but written for a technical reader; instructions assume Arduino knowledge; no troubleshooting guidance. | User guide written for a non-technical elderly or disabled user; includes plain-English setup steps, button-by-button remote guide, and a simple troubleshooting section. | User guide is accessible-design quality; uses large font guidance, high-contrast layout suggestions, plain language at reading age 10–12; includes quick-reference card for the remote; tested with a real non-technical reader and revised based on feedback. |

| CRITERION | WEIGHTING | ASSESSED WHEN |
|---|---|---|
| **Technical Functionality** | 25% | Lessons 6–8 (formative) + Lesson 10 (summative demo) |
| **Understanding of Input → Processing → Output** | 20% | Lesson 4–5 (formative quiz) + Lesson 10 (Q&A) |
| **Accessibility Design & User Empathy** | 20% | Lesson 2 (empathy map) + Lesson 10 (presentation) |
| **Collaboration & Team Process** | 15% | Ongoing teacher observation + Lesson 7 peer critique |
| **Communication & Showcase Presentation** | 20% | Lesson 9 (practice) + Lesson 10 (summative) |

| | | |
|---|---|---|
| **Written User Guide** | 20% (see note) | Lesson 10 final submission |
| **TOTAL** | **120% → normalised to 100%** | **Portfolio submitted Lesson 10** |

*Note: Weightings sum to 120%. Divide the student's raw total by 1.2 to get a percentage score out of 100, or redistribute per your school's system.*

# Student Handouts

## Handout 1 — 'Need to Know' Starter Questions

**WHAT DO I NEED TO KNOW TO ANSWER THE DRIVING QUESTION?**

You cannot design an accessible lighting system without understanding your user AND your tools. Research these before touching the hardware.

**About the User:**

- What specific conditions affect a person's ability to use a traditional light switch? (research at least 2)
- What does a person with limited grip strength find most difficult about existing lighting controls?
- What lighting conditions do elderly people prefer at different times of day? (find a source)
- What do existing solutions (clapper switches, voice assistants, smart bulbs) cost, and who cannot access them?

**About the Electronics:**

- What is a resistor and why does an LED need one? (look up: Ohm's Law, LED forward voltage)
- What is the difference between a digital signal and an analogue signal?
- What does PWM stand for and how does it make an LED appear dimmer without actually reducing the voltage?
- What is an IR receiver? What does 'IR' stand for and what range of the spectrum does it use?
- What is a hex code and why does the Arduino receive IR signals as hex values?

**About the Design:**

- What is 'universal design'? How does it differ from designing specifically for disabled users?
- What makes a remote control easy to use for someone with limited fine motor control?
- How will you test whether your device actually works for your intended user?

***Add your own questions:***

1. _____

2. _____

3. _____

## Handout 2 — IR Remote Code Investigation Table

**IR REMOTE CODE MAPPING — complete this during Lesson 5**

Open the Serial Monitor (9600 baud) and press each button on your remote. Record the hex code that appears. Press each button 3 times to confirm it is consistent.

| Button | Button Label / Symbol | Hex Code Received | Intended Function in Our Device |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| 10 | | | |

**Reflection:** Did you notice any pattern in the hex codes? What does that suggest about how the remote was programmed?

_____

_____

## Handout 3 — User Guide Template

**YOUR USER GUIDE — write for your user, not your teacher**

*Your guide must be usable by someone who has never seen an Arduino and never wants to. Test it with a real non-technical person and revise it before submission.*

**1. What this device does (2–3 sentences, plain English)**

- No technical jargon. Write as if explaining to a grandparent.
- Example: 'This small box connects to your lights and lets you control them from your armchair using a remote control. No wiring or installation needed.'

**2. What you will need**

- List: the device, the remote control, a power source (USB cable + plug or battery pack)
- Photo or clear sketch of the device and remote with labels

**3. Getting started (numbered steps)**

- Step 1: Plug in the device using the USB cable.
- Step 2: Wait 3 seconds for the green light to appear (means it is ready).
- Step 3: Point the remote at the device from up to 5 metres away.
- Continue with your specific button functions...

**4. Button guide (table)**

- Make a simple table: Button label / symbol → What it does
- Use symbols or colours if the button has no text label
- Keep the language simple: 'Full brightness' not 'PWM value 255'

**5. Troubleshooting**

- The light does not respond: try pointing the remote more directly at the black sensor; check the USB power is connected
- The light flickers: the power connection may be loose; check the cable
- Add 1–2 more problems your own testing revealed

**6. Important notes**

- Maximum remote control range: ___ metres (test and record this)
- The device should not be left near water or heat sources

**Usability test record:** Who tested your guide? _____ What confused them? _____ What did you change as a result? _____

## Handout 4 — Showcase Audience Feedback Form

**AUDIENCE FEEDBACK FORM (for invited guests)**

Team name / number: _____ Date: _____

**1. Did you understand who this device is designed for?**

*Could you picture the person this would help? Was their story told clearly in the presentation?*

Rating: 1 — Needs work 2 — Adequate 3 — Good 4 — Excellent

Comments: _____

**2. Would this device actually be useful to that person?**

*Based on what you saw — would it genuinely make their daily life easier? What would make it more useful?*

Rating: 1 — Needs work 2 — Adequate 3 — Good 4 — Excellent

Comments: _____

**3. How easy was the device to use?**

*Could you operate it without any instructions? Which parts were intuitive and which were confusing?*

Rating: 1 — Needs work 2 — Adequate 3 — Good 4 — Excellent

Comments: _____

**4. How clear was the explanation?**

*Did the team explain what the device does without using technical jargon? Could a non-technical person follow it?*

Rating: 1 — Needs work 2 — Adequate 3 — Good 4 — Excellent

Comments: _____

**5. What would you improve?**

*One specific suggestion for the team — about the device, the guide, or the presentation.*

Rating: 1 — Needs work 2 — Adequate 3 — Good 4 — Excellent

Comments: _____

**Overall: Would you want to use this device or recommend it to someone you know?** Yes / No / With changes

Reason: _____

## Handout 5 — Individual Reflection Prompts

**REFLECTION — think carefully, write honestly**

Complete this individually at the end of the showcase. Be specific — reference actual moments, measurements, or conversations from your project. 'I learned about circuits' is not an acceptable response.

**Technical learning:**

*Describe one electronic concept you misunderstood at the start of this project. What was your misconception? What experience or experiment changed your thinking?*

_____

_____

_____

**Design decision:**

*Identify one specific design decision (a button mapping, a brightness level, a layout choice) that you would change if you started again. Why? What evidence from your testing supports this?*

_____

_____

_____

**User empathy:**

*How did thinking about your specific user change what you built? Give one concrete example of a feature you included — or excluded — because of your user, not because of what was technically easy.*

_____

_____

_____

**Audience feedback:**

*What did the audience reaction during the showcase teach you that peer testing during class did not? Give one specific moment or comment.*

_____

_____

_____

**Connection to the driving question:**

*In 3 sentences: how does your finished device answer the driving question? Be honest — does it fully answer it, partially, or does it expose new problems the question didn't anticipate?*

_____

_____

_____

**Looking forward:**

*This is Project 1 of 10. Which skill from this project do you most want to develop further? How would you approach it differently in the next project?*

_____

_____

_____

## Teacher Reference: Common Misconceptions & Facilitation Moves

*Project 1 is the most important project in the sequence to get right. Students who leave it with strong inquiry habits, empathy instincts, and debugging confidence will build on those throughout Projects 2–10. Students who leave it having followed a recipe will struggle increasingly as complexity grows.*

| STUDENT MISCONCEPTION | CORRECT UNDERSTANDING | TEACHER FACILITATION MOVE |
|---|---|---|
| **'Any resistor will do — the LED still lights up'** | Without a correctly rated resistor, the LED draws excessive current, runs hot, and eventually burns out. The symptom (LED lights up) masks the problem (LED lifespan reduced from 50,000 hours to minutes). The LED appears fine until it suddenly fails. | *Ask: 'Your LED lights up — good. Now touch it with the back of your finger. Is it warm? How long do you think it will last like that?' Then have them calculate the correct resistor and compare. Record the temperature difference.* |
| **'analogWrite() sends a lower voltage to dim the LED'** | analogWrite() sends a full 5V signal, switched on and off very rapidly (490–980 Hz on most Arduino pins). The LED actually flickers too fast for the eye to see — the perceived dimness is an average of on-time vs off-time (duty cycle). No voltage reduction occurs. | *Ask: 'Measure the voltage on the PWM pin with a multimeter while the LED is at 50% brightness. What do you read?' They expect ~2.5V; they get something between 0V and 5V depending on timing. This surprise is the learning moment.* |
| **'The IR remote sends the LED command directly to the LED'** | The IR remote sends an encoded light signal to the receiver. The receiver decodes it to a digital signal sent to the Arduino. The Arduino runs code that interprets the signal and decides what to send to the LED. Three separate steps — Input, Processing, Output — with no direct connection between remote and LED. | *Draw the signal chain on the board with gaps: IR Remote → [?] → Arduino → [?] → LED. Ask: 'What travels across each gap? Is it the same kind of signal in both gaps?' Students often say 'electricity' for both — probe until they distinguish IR light from digital logic signal.* |
| **'Once it works, we're done'** | Technical function is one of six rubric criteria. A circuit that works but cannot be operated by the intended user, cannot be explained to a non-technical audience, and has no user guide has met only one of six criteria. Students often anchor on the circuit working as the definition of completion. | *After the first successful IR-controlled LED: 'Congratulations — you've completed about 15% of this project.' Show the rubric explicitly. Ask: 'Could your user operate this right now? Could they explain it to a friend? Is there a guide they could follow?' The answer to all three is almost always no.* |
| **'We should use pin 13 — it has a built-in LED'** | Pin 13's built-in LED is useful for debugging but it shares the pin with an onboard resistor that affects external circuits. For LED brightness control, it behaves inconsistently. More importantly, pin 13 is not a PWM-capable pin on some Arduino models — critical for analogWrite(). | *Ask: 'Find the pins marked with a tilde (~) symbol on the Arduino. What does that symbol mean?' Then: 'Is pin 13 one of them?' Students discover PWM pins through investigation. The ~ symbol becomes a permanent reference point for the rest of the project sequence.* |
| **'If the code compiles, it's correct'** | The compiler checks syntax only — that the code is valid Arduino/C++. It cannot check logic. A condition written as if (results.value = 0xFFA25D) instead | *Tell students nothing. Instead, introduce the Serial Monitor as a mandatory second step: 'Compile → Upload → Open Serial Monitor →* |

| | | |
|---|---|---|
| | of == will compile without error but will never correctly identify the IR code (it assigns rather than compares). Logic errors are invisible to the compiler. | *Check the output matches your expectation.' Make Serial.println() their first debugging instinct, not their last resort.* |

## Differentiation Strategies

| LEARNER PROFILE | SUPPORT SCAFFOLDS | EXTENSION CHALLENGES |
|---|---|---|
| **Learners needing additional support** | Pre-wired LED-resistor circuit to start from · Colour-coded wiring guide (red = power, black = ground, yellow = signal) · Code with fill-in-the-blank conditionals · IR hex codes pre-identified for their specific remote · Pair with a stronger partner for circuit work, independent for empathy map and user guide | Core functionality only: 2 brightness levels + on/off · User guide with sentence starters provided · Verbal presentation accepted in addition to or instead of written guide · Empathy map partially pre-populated with condition information |
| **On-track learners** | Standard project brief · BIE-style check-ins at each phase · Peer critique at Lesson 7 · All 6 rubric criteria targeted | Minimum 3 brightness levels + on/off · Complete one upgrade (A, B, or C) · Full user guide tested with non-technical reader · 4-minute presentation with live demo to real audience |
| **Advanced learners** | Minimal scaffolding · Self-directed inquiry for resistor calculation and IR investigation · Design own button mapping logic from scratch · Encouraged to identify a real local user to design for | All three upgrades · RGB LED with full colour mixing · Fade transitions between all brightness levels · User guide in large-print, high-contrast accessible format · Prototype tested with a real elderly or disabled user and revised based on their feedback · Propose installation in a real location (school, home, care facility) |

## Materials List & Approximate Costs

| COMPONENT | QTY (per team) | UNIT COST (USD approx.) | NOTES |
|---|---|---|---|
| Arduino UNO (or Nano) | 1 | $4–8 | UNO recommended for first project — easier pin identification and larger breadboard clearance |
| IR receiver module (TSOP38238 or similar) | 1 | $0.50–1.50 | Buy the module (with resistor + capacitor built in) not the bare component |
| IR remote control (NEC protocol) | 1 | $1–3 | Any standard remote works; NEC protocol most common and best-supported by IRremote library |

| | | | |
|---|---|---|---|
| Red LED (5mm) | 1 | $0.10–0.20 | Standard 5mm, 2V forward voltage, 20mA max |
| Green LED (5mm) | 1 | $0.10–0.20 | As above |
| Blue LED (5mm) | 1 | $0.10–0.20 | As above — blue typically needs 3.2–3.4V; adjust resistor calculation |
| RGB LED (common cathode) — for Upgrade A | 1 | $0.30–0.80 | Optional; replace 3 separate LEDs if doing Upgrade A |
| 220Ω resistors (1/4W) | 3–4 | $0.05 each | Students calculate required value — 220Ω is a safe starting approximation for most 5mm LEDs at 5V |
| Breadboard (half-size) | 1 | $1–2 | One per team |
| Jumper wires (assorted M-M) | 15–20 | $0.50 pack | Pre-cut kits save setup time significantly |
| USB cable (Type A to Type B) | 1 | Usually included | For Arduino connection; ensure data cable not charge-only |
| TOTAL per team (core kit) | — | ≈ $8–16 | One of the lowest-cost projects in the sequence — good budget starting point |