Example: Introduction to Physical Computing

Learning Area: Technology Mandatory	**Topic area:** Digital Technologies	**Unit of Work:** Microcontroller Project
Stage of Learner: Stage 4	**Lesson Number:** 1	**Time:** 1 hour

Outcomes	**Assessment**	**Students learn about**	**Students learn to:**
Designs algorithms for digital solutions and implements them in a general-purpose programming language **(TE4-4DP)**	Digital exit slip	Prototyping with microcontrollers	Implement and modify programs with user interfaces involving branching, iteration and functions in a general-purpose programming language. **(ACTDIP030)**
Cross-curriculum Priorities		**General capabilities**	
		• Critical and Creative Thinking • Information and Communication Technology (ICT) capability • Numeracy	

Quality Teaching Elements	
Intellectual Quality This refers to pedagogy focused on producing deep understanding of important, substantive concepts, skills and ideas. Such pedagogy treats knowledge as something that requires active construction and requires students to engage in higher-order thinking and to communicate substantively about what they are learning.	1.1 Deep knowledge 1.2 Deep understanding 1.3 Problematic knowledge 1.4 Higher-order thinking 1.5 Metalanguage 1.6 Substantive communication
Quality Learning Environments This refers to pedagogy that creates classrooms where students and teachers work productively in an environment clearly focused on learning. Such pedagogy sets high and explicit expectations and develops positive relationships between teacher and students and among students.	2.1 Explicit quality criteria 2.2 Engagement 2.3 High Expectations 2.4 Social Support 2.5 Students' self-regulation 2.6 Student direction
Significance This refers to pedagogy that helps make learning more meaningful and important to students. Such pedagogy draws clear connections with students' prior knowledge and identities, with contexts outside of the classroom, and with multiple ways of knowing all cultural perspective.	3.1 Background knowledge 3.2 Cultural knowledge 3.3 Knowledge integration 3.4 Inclusivity 3.5 Connectedness 3.6 Narrative

How the quality teaching elements you have identified are achieved within the lesson.

Teaching element	Indicators of presence in the lesson
1.5	Lessons explicitly name and analyse knowledge as a specialist language (metalanguage), and provide frequent commentary on language use and the various contexts of differing language uses.
2.4	There is strong positive support for learning and mutual respect among teachers and students and others assisting students' learning. The classroom is free of negative personal comment or put-downs.
3.5	Lesson activities rely on the application of school knowledge in real-life contexts or problems, and provide opportunities for students to share their work with audiences beyond the classroom and school.

Lesson preparation:

- Download, install and test Arduino prior to the first lesson. https://www.arduino.cc/en/Main/Software (complete the Get.Connected steps from the quick-start guide, pg. 6-8)
- Download ThinkerShield program code files. https://maas.museum/learn/thinkershield/
- Book computer lab or laptop trolley in your school. It is recommended learners work in pairs as it makes troubleshooting easier. Each pair of learners will require a separate ThinkerShield kit and computer.
- A time-conscious teacher may consider setting What's.On.Board (pg.9) and What's in an Arduino program (pg. 12) as pre-reading as a flipped classroom exercise. This will allow for more time in class for experimentation.
- Email thinkershield@maas.museum for additional support with hardware/software.

Time	Teaching and learning actions
5	**Introduction to lesson** Learners will be given a brief introduction to the world of physical computing by using Arduino and the ThinkerShield. "**Arduino is an open-source electronics prototyping platform** based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments. **Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors and other actuators.** The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (eg Flash, Processing, MaxMSP)." - https://www.arduino.cc/en/Guide/Introduction **Tip:** Try not to frontload learners with information at this stage. **Sentences in bold are important points.** Get learners achieving outcomes as soon as possible. This increases motivation as well as highlights any hardware/software issues earlier rather than later. Additionally, you may want to show a quick video about the Arduino and ThinkerShield as an introduction. https://www.youtube.com/watch?v=jTAwgkfPL-g
5	**What's.On.Board?** Ask learners to examine the ThinkerShield. The ThinkerShield is a shield that easily connects external electronic components to the Arduino. See if learners can recognise any of the components on the shield already. The teacher can briefly point out any components not listed by students. Each component includes a "D" or an "A" and number. The letters refer to whether the component is a digital pin or an analog pin. The number refers to which pin on the Arduino the component is connected to. ● GND (Ground)used for external circuits ● VCC (Power)used for external circuits ● Push button ● External digital pin connections (D2, D4-D6) used for external circuits

	● Detachable buzzer jumper ● Piezo Buzzer (D3) ● Potentiometer (A5) ● Light Dependent Resistor (LDR, A4) ● External analog pin connections (A0-A3) used for external circuits ● Reset button ● External power socket (on Arduino board) ● 6 x Green LEDs (Light Emitting Diodes, D8-D13) ● USB port (on Arduino board) **Tip:** The teacher may like remind learners that some of the components are already labelled!
5	**Get.Connected** Learners will complete steps 3 - 4 from the quick-start guide (pg. 6-7). The teacher may lead this activity directly from the quick-start guide or step-by-step on the projector. 3. Connect your ThinkerShield to your Arduino and computer. 4. Select the board and serial port. Learners will need to repeat this process at the start of every lesson when using the Arduino. The teacher should encourage learners to remember the process, as all Arduinos require the similar setup process. **Tip:** The teacher may like to go through this process together with the class the first time.
5	**In.A.Blink** Once the board is connected it is a good idea to make sure everything is working properly by uploading a working sketch straight away. Learners will complete step 6 from the quick-start guide (pg. 7). 6. Launch! Upload the In.A.Blink sketch from the example code files. Ask learners to identify what has happened on the board and why. (LED D13 has lit up because the first line of the sketch has addressed D13) **Tip:** Challenge learners to light up a different LED without further instruction. (For example, change 'int led = 13;' to 'int led = 12;')

10	**What's in an Arduino Program.** As a class, go through the basics of the Arduino program structure of In.A.Blink. The quickstart guide covers these following aspects of code in detail (pg. 12-13): ● Comments (Digital post-it notes) ● Keywords (Words recognised by the program) ● Constants (Words that stay the same like HIGH, LOW) ● void setup(){} (Code that runs once setting the pinmode to input or output) ● void loop(){} (Code that runs repeatedly telling the ThinkerShield what to do) ● Semicolon (Usually put at the end of each line unless it is a conditional statement like 'if') ● Curly braces (Code must be put between the code to work) ● Debugger area (Used to find and solve errors) ● Status bar (Shows which board the computer is connected to) ● Sketch (The open Arduino program) ● Verify program (Used to check your code for errors before you upload) ● Upload program (Transfer the code onto the Arduino and ThinkerShield) ● Serial monitor (Records serial connections from the Arduino to the computer) Point out examples as you go through the code. You may not need to point out every example the first time, for example, the serial monitor can be introduced later. The Arduino website has a reference page that explains all of the correct terminology. It may be worth reading. - https://www.arduino.cc/reference/en/ **Tip:** Reassure learners! There is a lot to remember and it will take some time to understand and process. Reinforce patterns wherever possible, for example: 1. create a variable at the top. 2. Set the pinMode in 'void setup'. 3. Write in your loop function. **Tip 2:** Learners will make simple mistakes such as spelling errors and missing punctuation. The more learners understand the structure of the language the easier it will be for students to problem solve their own errors in the future.
10	**Even Flashier (Extension)** After learners have got a basic idea of the structure, encourage them to get even flashier. Challenge students to get two LEDs flashing by using the same pattern as before.

	1. Create a variable at the top. 2. Set the pinmode in 'void setup'. 3. Write your loop function in 'void loop'. Give learners some time to experiment. Some learners will come up with ideas like: ● int led = 8,9; ● int led = 8-9; ● int led = 8; int led = 9; None of these will work! Encourage students to read their errors. The common error they will receive will be " redefinition of 'int led'". This means they can not give each LED the same name. They must give each LED a new name like led1 and led2. **Tip: Fun activity -** Is there two learners with the same name, for example, Jo? Ask "Jo" to stand up. Both learners should stand up. Ask learners how I can define which Jo I wish to stand up. (Give Jo a nickname: Jo1 and Jo 2 or Jo C and Jo T)
10	**Assessment** Wrap up the lesson by reflecting on what the learners have achieved today. **Option A: Exit slip email** You may want to create a digital exit slip with questions like the ones below: ● What is one coding term you can explain and demonstrate? ● What was something you found challenging? ● What was something you found easy? ● If you were the teacher, what would you do next? (A great one to gauge where the learner wants to go to and gets them to think about their learning journey). **Option B: Kahoot** You can create a kahoot quiz, discussion or survey to assess student learning. To see how it works, watch the video below: https://www.youtube.com/watch?v=PlXpKHH5kh0 **Option C: Socrative** Another option would be to use Socrative to collect student responses. It is similar to Kahoot. To see how it works, watch the video below:

	https://www.youtube.com/watch?time_continue=168&v=EGr53IA91MU

Other considerations.

Differentiation:

Complete the table below by inserting additional ways to differentiate the lesson to meet the specific learning needs of students across the full range of abilities.

Literacy	Learners with lower literacy levels may benefit from using the pre-existing activity files from the example code. The example includes both completed and semi-completed example sketches. This may help learners to understand the concepts before they attempt to tackle computer literacy. There is also a way to use Arduino with scratch (a visual drag and drop programming language), however, at the moment it is a little bit tricky to use easily. http://s4a.cat/
Gifted and Talented	Learners who are gifted and/or talented who benefit from working independently may like to work directly from the book at their own pace. Teachers may like to ask their learners to record their progress and turn it into a digital portfolio.

Australian Professional Standards for Teachers:

Complete the table below by inserting the AITSL graduate standards that you are demonstrating and indicates the evidence from this lesson that should comply with the standard.

Graduate Standards	Evidence within this lesson
1.5.2	This lesson suggests various ways the content can be differentiated for low literacy learners as well as gifted and talented students.
	This lesson uses effective teaching strategies to integrate

2.6.2	ICT into learning and teaching programs to make content relevant and meaningful.
4.5.2	This lesson incorporates strategies to promote the safe, responsible and ethical use of ICT in learning and teaching.

Resources Attached:

Links

- **Arduino**
  - https://www.arduino.cc/
  - https://www.arduino.cc/reference/en/

- **ThinkerShield**
  - https://maas.museum/app/uploads/2016/10/GetOnWithIt-v1-2-ONLINE.pdf
  - https://thinkershield.maas.museum/