## 

| **Lesson 5 - Etch a Sketch** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Learning Aims:** In this final lesson students will create a simple drawing console based on the old school **Etch a sketch** drawing tablet. Using all the assembled components together to **consolidate** their learning.  **Learning Objectives:**  * Learn how to combining inputs and outputs together to mimic real world systems * Learn how to remove “shake” from an analogue input device | | | | | |
| **Key Words** | | ADC, analogue, PWM, Duty Cycle, Frequency, Potentiometer | | | |
| **Preparation before the lesson:**  Print required handouts  In addition it's **highly recommended** that teachers undertake these activities prior to the lessons. | | | | | |
| **Assessment opportunities** | | Correct implementation of coding task | | | |
| **Lesson Resources** | | [Starter Activity](https://docs.google.com/document/d/1LzlrSrHnnbC-0IZ1aNx8VF1ehebmWfXz/edit?usp=drive_link&ouid=114120863087517164066&rtpof=true&sd=true)  [Circuit diagram](https://docs.google.com/document/d/14V4wlbmY8XJajO2Tp8BUEGIQKvYHFYFw/edit?usp=sharing&ouid=114120863087517164066&rtpof=true&sd=true)  [Self assessment sheet](https://docs.google.com/document/d/1pMenaIPvHyF_4ygCUGobJHrRh-_ZKIM8/edit?usp=drive_link&ouid=114120863087517164066&rtpof=true&sd=true)  [Challenge worksheet](https://docs.google.com/document/d/1JwwzkAPWabGBFKJY9O-MoHa6PjjwDV_H/edit?usp=drive_link&ouid=114120863087517164066&rtpof=true&sd=true)  [Lesson 5 ppt](https://docs.google.com/presentation/d/1eTf62K4WzETOsvJeqDV_uL01sZlbU9zX/edit?usp=drive_link&ouid=114120863087517164066&rtpof=true&sd=true) | | | |
| **Teachers Guides:** | | |  | | |
|
| **NC Links** | | | * design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems. * understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems * use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; | | |

## Lesson 5

| Starter activity (Slide 2)  5mins | Starter activity to introduce the “old school” Etch a Sketch drawing console. Most students will have come across these but likely have no idea how they work.  **Explain how it works show the short video clicp**  The Etch a Sketch has a stylus mounted on a pair of orthogonal rods, one horizontal and the other vertical. The rods are each connected to a knob via steel wire. Their combined motion allows the stylus to move anywhere within its plane. The stylus scrapes powder away from the glass as it moves, creating a line. watch a short part of the clip [Etch A Sketch – Learn the basics!](https://youtu.be/SsoZU6S8Ofc?si=_6lRogJsPA43t2gZ) |
| --- | --- |
| **Lesson Aims**  (Slides 4–9)  10 mins | Explain to students that today's lesson they will create an e-version of the etch a sketch. Although the original version was not electronic it is easy using the skills they have learnt to build the e- version |
| **Activity 1**  5mins | Students should use the circuit worksheet to complete the wiring for the Etch a Sketch. This should be almost complete from the previous lessons activities. They should however check that wiring is still in place and has not come loose. The circuit is now complex and ensuring connections are in place and refreshing their memory of the connections will help them later with debugging. |
| **Activity 2** | PRIMM predict what will happen when the code is executed  provide students with the code worksheet and allow time for students to work through it in pairs. Students should annotate the code they can identify. This is a larger program and students should be encouraged to become comfortable with the code so that they feel confident tackling the challenge.  They should be able to recognise much of the code from previous lessons. Bring students back together and walk through the code encouraging them to identify what they can recall before filling in the gaps  **overview of the code**  Import necessary modules from the MicroPython library  **from machine import Pin, I2C, ADC**  **from ssd1306 import SSD1306\_I2C**  **from time import sleep**  # Set up I2C for communication with the OLED display  **i2c = I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)**  **oled = SSD1306\_I2C(128, 64, i2c)**  **oled.fill(0) # Clear the OLED display**  Here, an I2C object (`i2c`) is created with specified GPIO pins for the data (SDA) and clock (SCL) lines. Then, an OLED display object (`oled`) is initialised with a resolution of 128x64 pixels using the SSD1306\_I2C driver. The `fill(0)` method is used to clear the OLED display by setting all pixels to black (0).  # Set up ADC (Analog-to-Digital Converter) for reading values from potentiometers  **x\_adc = ADC(0)**  **y\_adc = ADC(1)**  Two ADC objects (`x\_adc` and `y\_adc`) are created to read analog values from two different channels (0 and 1). These channels are typically associated with physical pins on the Raspberry Pi Pico.  # Initialize variables to store the last cursor position  **last\_x\_cursor = 0**  **last\_y\_cursor = 0**  Variables `last\_x\_cursor` and `last\_y\_cursor` are initialised to store the previous cursor positions.  # Set up a button (button\_left) to clear the OLED screen  **button\_left = Pin(14, Pin.IN, Pin.PULL\_DOWN)**  A button is configured using GPIO pin 14 (`Pin(14)`) as an input (`Pin.IN`) with a pull-down resistor enabled (`Pin.PULL\_DOWN`). This means that when the button is not pressed, the pin is pulled down to a logical low state.  # Function to map a value from one range to another  **def convert(x, in\_min, in\_max, out\_min, out\_max):**  **return (x - in\_min) \* (out\_max - out\_min) // (in\_max - in\_min) + out\_min**  This function (`convert`) takes a value `x` and maps it from the range `[in\_min, in\_max]` to the range `[out\_min, out\_max]`. It is used to map analog readings to cursor positions on the OLED display.  # Display "Etch a Sketch" at the top of the OLED screen  **oled.text("Etch a Sketch", 0, 0)**  The text "Etch a Sketch" is displayed at the top-left corner of the OLED display.  `  # Main loop to continuously read values from potentiometers and update the OLED display  **while True:**  **# Read analog values from potentiometers and map them to cursor positions**  **x\_cursor = convert(x\_adc.read\_u16(), 0, 65535, 125, 3)**  **y\_cursor = convert(y\_adc.read\_u16(), 0, 65535, 62, 2)**    **print(y\_cursor, x\_cursor) # Print cursor positions** (for debugging)  # Remove cursor shake: adjust as needed to smooth lines  **if abs(x\_cursor - last\_x\_cursor) < 2:**  **x\_cursor = last\_x\_cursor**  **if abs(y\_cursor - last\_y\_cursor) < 2:**  **y\_cursor = last\_y\_cursor**  # Create a dot at each cursor position on the OLED display  **oled.text(".", x\_cursor, y\_cursor)**  **oled.show() # Update the OLED display**  # Small sleep to allow for screen refresh (adjust as needed)  **sleep(0.001)**  # Keep track of the cursor's last position  **last\_x\_cursor = x\_cursor**  **last\_y\_cursor = y\_cursor**  # Check if the button is pressed to clear the screen  **if button\_left.value() == 1:**  **oled.fill(0) # Clear the OLED display**  **oled.show()**  # Update the OLED display to reflect the clearing  The main loop continuously reads analog values from the potentiometers, maps these values to cursor positions, and displays a dot at each position on the OLED display. It includes a mechanism to reduce cursor shake and a small delay for screen refresh. The loop also checks if the button is pressed to clear the OLED display if needed. The loop runs indefinitely (`while True`).  Provide students with the starter code worksheet and ask students to identify the code they recognise annotating as they go.  Have a working Pico example ready to demonstrate the execution and if possible project the working demonstration for students. A video is provided if this is not possible.  Reiterate that there is very little code that they have not already experienced and that today's lesson is an opportunity to consolidate what they have learnt. Explain to students the purpose of removing shake as the screen is so small when movement is converted even a slight movement of the POT which could be caused by vibration would result in a move of the pixel on the screen. This would mean that lines are not smooth. This snippet of code aims to prevent that shake. The sample ignores movements of less that 2 on the pot. Students should experiment with this should they have shake on their completed projects.  # **Remove cursor shake: adjust as needed to smooth lines**  if abs(x\_cursor - last\_x\_cursor) < 2:  x\_cursor = last\_x\_cursor  if abs(y\_cursor - last\_y\_cursor) < 2:  y\_cursor = last\_y\_cursor |
| **Activity 3**  30mins | Provide students with the step by step guide worksheet to complete the code. **For less confident coders you may wish to supply some of the code. I would suggest supplying up until the While True main loop** as it is the **conversion function** and the **OLED set up** is where the students tend to make the most errors entering the code. Providing this will reduce the cognitive load and the debugging required and ensure that students are able to complete the activity within the time available.  For students that finish the Etch a Sketch there is a paddle game challenge, a simple PONG style game. This is usually a very popular game with students. It is however slightly more complex than the Etch a Sketch. Which is why it is not the main outcome. |
| **Plenary**  5 mins | **Evaluation of the unit**  Get students to complete a WWW EBI of their own performance during the unit. Reflecting on how they feel they have progressed throughout the unit. What would help them to move forward in future programming units? |
| **Homework** |  |