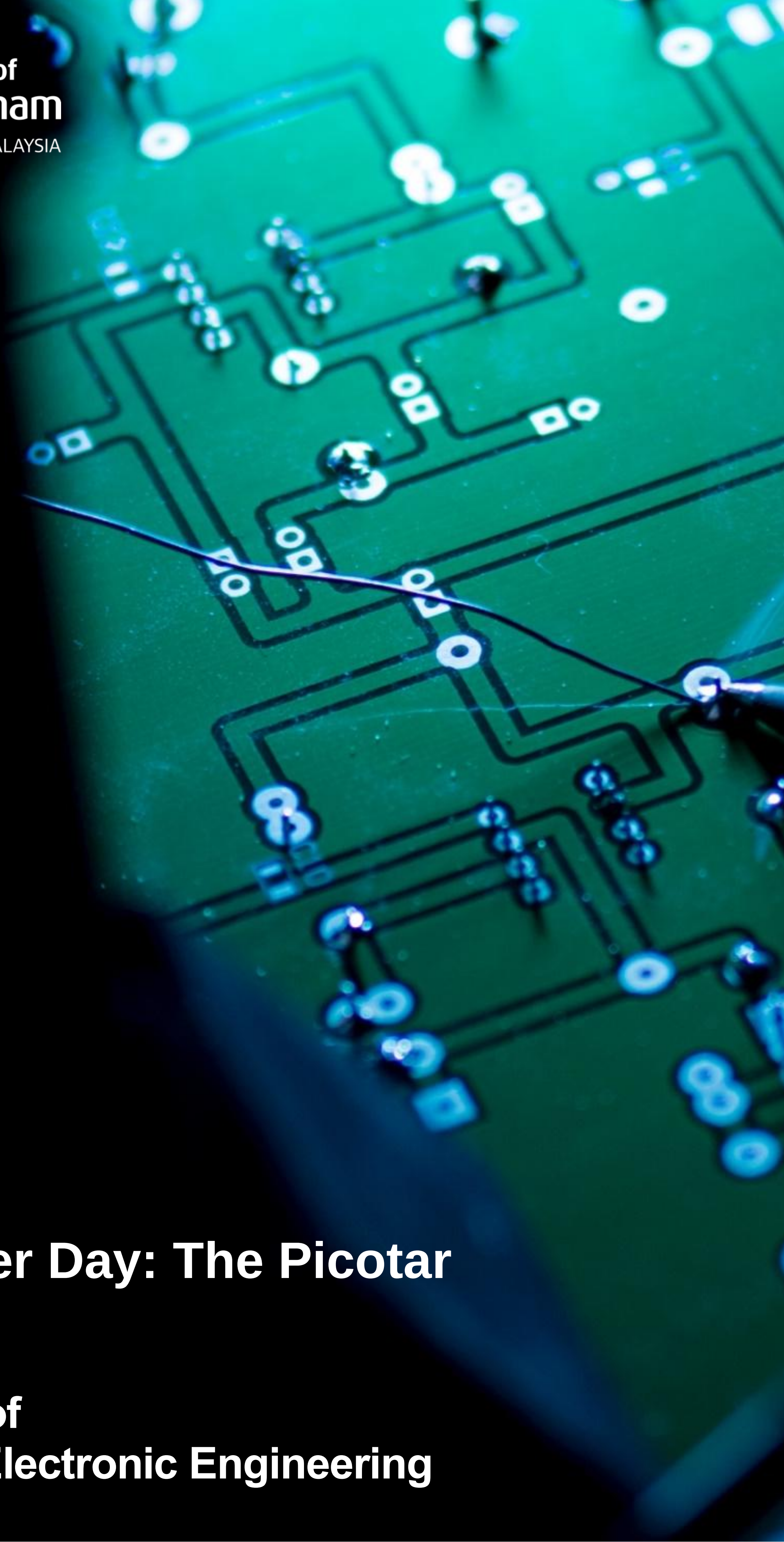
University of Nottingham
UK | CHINA | MALAYSIA



Offer Holder Day: The Picotar

Department of Electrical & Electronic Engineering

Table of Contents

1. Introduction

Welcome to the Department of Electrical and Electronic Engineering's Offer Holder Day. During this activity, you will make use of one of our project labs to complete a short design task to give you a sample of what we do here on our degree course at Nottingham.

You will be constructing a microcontroller-driven 'guitar', aptly called the 'Picotar' due to the Raspberry Pi Pico W microcontroller being used. Once built, you will be able to play several 'string' notes at varying pitch and volume, which emulates the sounds of a real guitar.

2. The Picotar Activity Overview

The aim of the activity is for you to construct an electronic guitar which makes use of different components to vary the sounds that are played. Figure 1 shows a completed Picotar, which is what you will be able to take home with you at the end of the activity.

Figure 1: The Complete Picotar

In front of you, you should have a parts kit, as shown in Figure 2. This contains several different components as well as a partially populated Picotar board.

Figure 2: The Picotar Parts Kit

2.1 The Picotar Parts Checklist

Check that you have the following components, if you are missing any, please let us know.

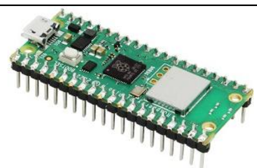
Item Description	Quantity	Notes	Photo
Partially Populated Picotar PCB	1	With the Battery Pack & Snap Lead Attached The LM386 Audio Amplifier Chip Should Already be in the Socket - Details on How to Correctly Identify the Polarity is Included in ‘3.4 The Speaker & Audio Amplifier Circuit’	
Raspberry Pi Pico W Microcontroller	1		
USB A-Micro Cable	1	Used to Power & Program the Raspberry Pi Pico W Microcontroller	
LED	3	Colours May Vary Between Red, Green & Yellow. Details on How to Correctly Identify the Polarity is Included in ‘3.7 The LED’	
330Ω Resistor	3	Details on How to Identify the Values of the Resistor are Included in ‘3.6 The Resistor’. The Image is Illustrative Only	
10kΩ Resistor	3		
Push Button	3		
Push Button Cap	3	Colours May Vary Between Red, Green & Black – These Clip onto the Top of Push Buttons	

Table 1: Picotar Parts Checklist

2.2 Visual Checklist

Your task is to finish populating the Picotar board by soldering the remaining components into the correct locations. If you have not soldered before, do not worry! Please ask one of the staff or student ambassadors for a demonstration as well as reading through *'4. Quick Guide to Soldering'.*

Figure 3 shows the partially populated board, as well as markers for each of the component locations. Tick each one off once you have soldered it in place to ensure you haven't missed anything, including pushing the Raspberry Pi Pico W microcontroller into the headers!

Once fully constructed, you will upload the example code following the process described in *'5. Programming the Picotar'.*

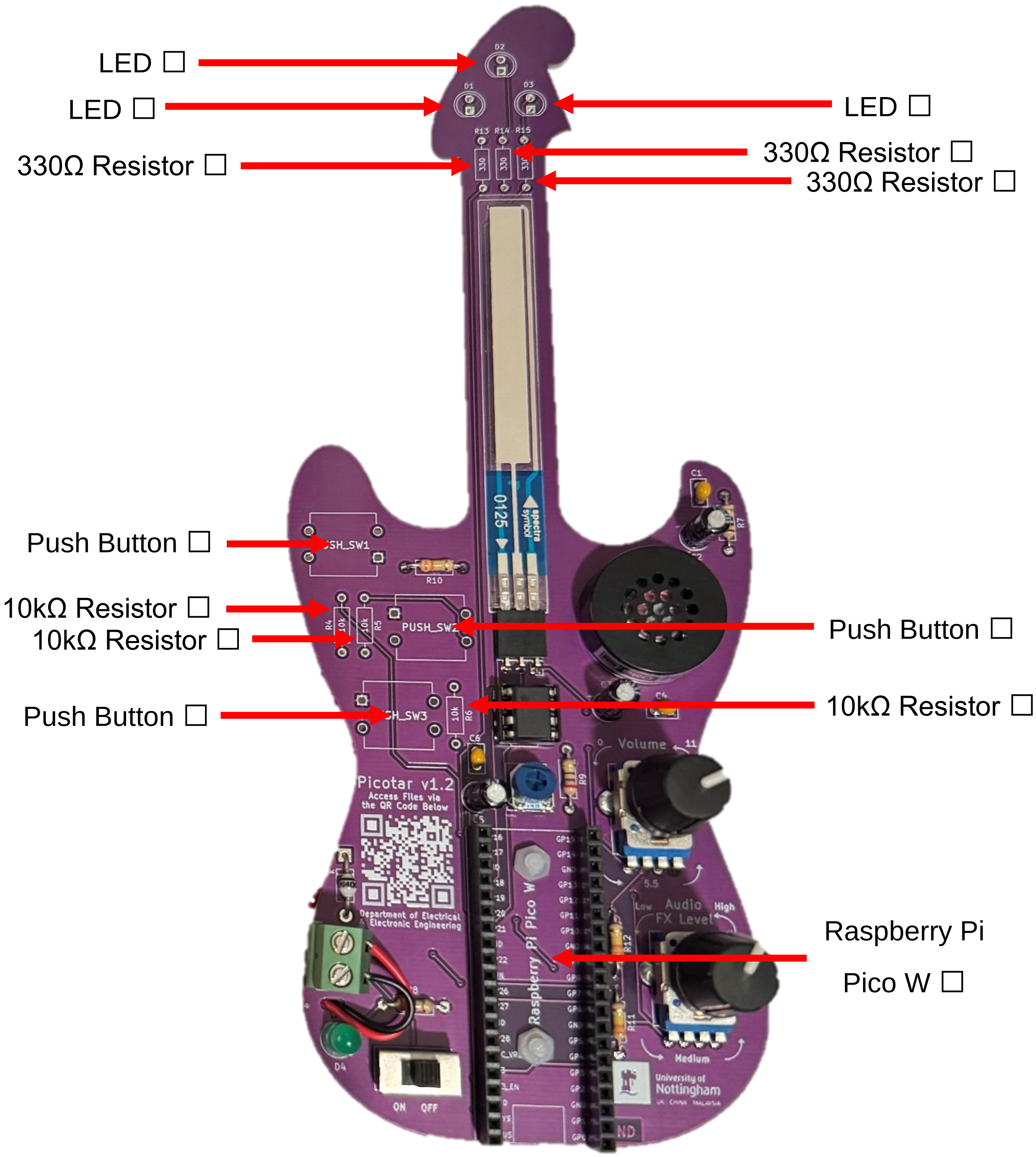


Figure 3: Partially Populated Picotar

3. Breaking Down the Picotar

This section will break down each of the key elements of the Picotar. You do not need to read and understand this to do the activity, but the following information will give you an insight into what each component does, and how it comes together into a working system. In the resistor and LED subsections, there is also useful information regarding how to identify a resistor's value and identify the polarity of an LED respectively.

3.1 The Rotary Potentiometer

Within the Picotar, there are two rotary potentiometers, as shown in Figure 4 (left) below. You will notice that each rotary potentiometer also has a knob attached which makes it easier for you to rotate each potentiometer with your fingers, as well as making use of the white mark to see the current rotational position more easily.

One potentiometer is used to control the volume of the sounds being played, and the other is used to adjust the amount of audio effects/sustain time applied to the sounds. A full list of the audio effects can be found in *'5. Programming the Picotar'.*

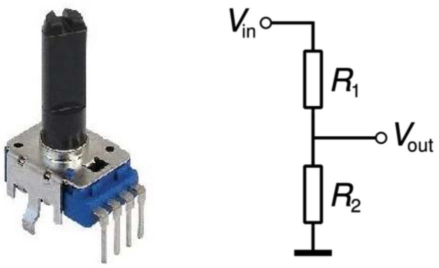


Figure 4: Rotary Potentiometer (left) & Potential Divider (right)

A rotary potentiometer is a variable resistor which proportionally divides a supplied voltage over its operational range and provides a proportional voltage output related to the position of the wiper i.e. the moving element. In our case, the voltage output is between 0V and 3.3V, with the output voltage varying depending on how much the rotary potentiometer is turned. You may be familiar with the potential divider equation, visualised in Figure 4 (right), which is:

$$V_{out} = V_{in} \frac{R_2}{R_1 + R_2}$$

Where V_{in} is the supply voltage, and R_1 and R_2 are the resistances on either side of the current rotary potentiometer position. For this particular component, the total resistance is 10kΩ. By reading the output voltage value, we can digitally interpret the amount that the potentiometer has rotated and determine whether to increase or decrease the volume or the amount of audio effects applied to the audio output.

3.2 The Linear Potentiometer

Within the Picotar, there is one linear potentiometer, as shown in Figure 5 below. This is used to adjust the pitch of a particular chord being 'played' by moving your finger along it, similar to how the strings on a real guitar are shortened and lengthened to vary the pitch.

Figure 4: 'ThinPot' Linear Potentiometer

A linear potentiometer is also a variable resistor however the 'wiper', in this case, is where your finger is on the linear potentiometer.

The same potential divider equation can be used with R_1 and R_2 being the resistances on either side of the finger position on the linear potentiometer. For this particular component, the total resistance is also 10kΩ. In the same way as the rotary potentiometer, by reading the output voltage value, we can digitally interpret the position of the finger and determine whether to increase or decrease the pitch of the sound being played.

3.3 The Push Button

Within the Picotar, there are three push buttons, shown in Figure 6, each of which is used to play a specific note. Each button has a button cap of varying colours so that each note can be more easily differentiated. If pressed in tandem with your finger being placed somewhere on the linear potentiometer, then the pitch of the note can be varied in real-time.

Figure 5: Push Button

A push button is a simple mechanical switch that completes or breaks a circuit when it is pressed. In our case, a circuit is completed when the button is pressed, and so a high signal is read by the microcontroller. This high signal is only active whilst the button is pressed, and returns to low when the button is released.

3.4 The Speaker & Audio Amplifier Circuit

Within the Picotar, there is a speaker which is used to audibly play the sound. To amplify the audio signals, an audio amplifier circuit, driven by the LM386 operational amplifier chip, is also used, as shown in Figure 7 below.

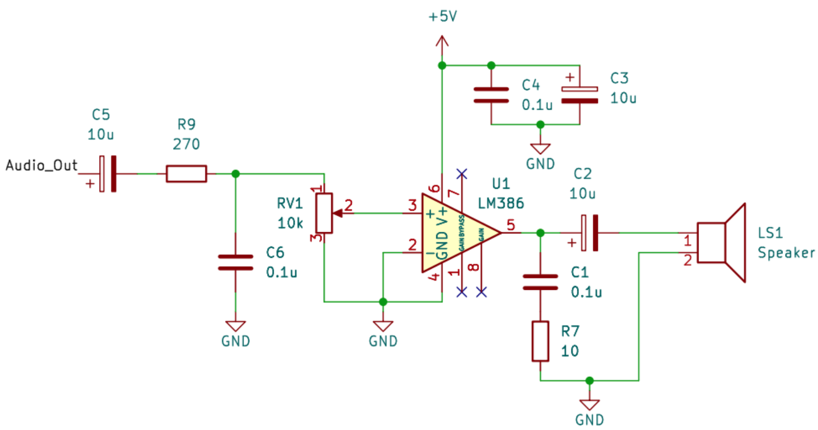


Figure 6: LM386 Audio Amplifier Circuit

Whilst the full breakdown of how this circuit works is not included here, to summarise, the key stages are as follows:

1. The audio signal is inputted to the non-inverting (+) pin of the LM386 through a variable resistor. Without this resistor, there will be a lot of noise in our audio output.
2. The gain, i.e. how much we multiply the input by to get our output, is set as the chip default of 20, so pins 1 and 8 are left unconnected.

3. There are two decoupling capacitors used on pin 6 to filter out any low-frequency noise from the power supply which otherwise would have been amplified and distorted our audio output.
4. Once amplified, the audio is outputted from pin 5 and fed into the speaker, with additional capacitors used for filtering and smoothing.

Polarity of the LM386 Audio Amplifier Chip

As part of your parts kit, you have been given an LM386 audio amplifier chip. Whilst this can only go in the black 8-way socket, it is vital that you push the chip into the socket the correct way around.

You will notice that at the top of the socket, there is a notch which indicates where the top of the chip should be placed. On the chip itself, there is a dot – this indicates the top of the chip. Therefore, the side of the chip with the dot on should match the notched side of the socket, as shown in Figure 8 below. If you are unsure, please ask.

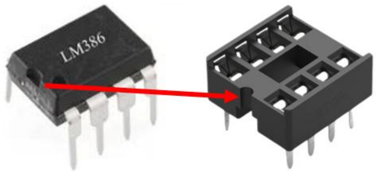


Figure 7: Correct Chip & Socket Orientation & Placement

3.5 The 'Mozzi' Audio Library

Typically, only very basic sounds can be played with a simple speaker, such as the one used on the Picotar. However, the Mozzi library is an open-source audio synthesis library that allows you to generate and manipulate waveforms in real time to output much more complex and interesting sounds.

For full details and examples of how to use Mozzi, please see *'7. Useful Links'* however, as a general overview, Mozzi makes use of the following key features:

- Oscillators; which generate different waveforms e.g. sine, square, triangle and saw-tooth waves.
- Envelopes; to control a waveform's amplitude over time.
- Filters; for shaping the signals.
- Control Signals; to modulate the sound parameters over time.
- Sampling; to play back pre-recorded audio or sampled waveforms.

3.6 The Resistor

Within the Picotar, there are a number of resistors of different resistance values, which have different purposes. These include; pull-down resistors to keep a signal low when the button isn't pressed, current limiting resistors to prevent the LEDs from being damaged and as part of the audio amplifier circuit to filter the audio signals.

Resistors, as shown in Figure 9 (left), are the most common and well-known passive electrical components. A resistor resists and limits the flow of electricity i.e. current, in a circuit, and is measured in Ohms (Ω). There are two ways in which we can determine the resistance of a specific resistor; the first is with a multimeter, which we won't use, and the second is with the resistor colour band chart, as shown in Figure 9 (right).

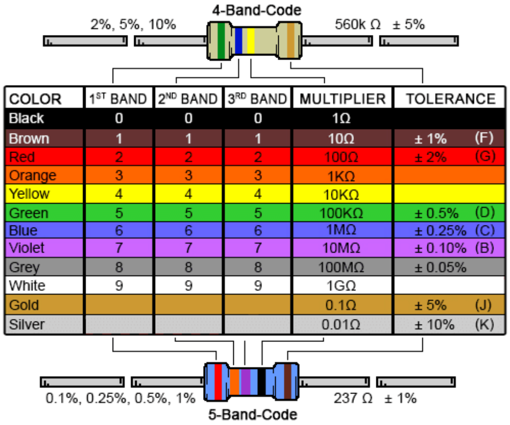


Figure 8: Example of Through-Hole Resistors (left) & Resistor Colour Band Chart (right)

Within your parts kit, you have six resistors – three of which are 330Ω, used as current limiting resistors for the LEDs and three of which are 10kΩ, used as pull-down resistors for the buttons.

- 330Ω; the bands are: Orange-Orange-Brown-Gold or Orange-Orange-Black-Black-Gold.
- 10kΩ; the bands are: Brown-Black-Orange-Gold or Brown-Black-Black-Red-Gold.

A note about resistor placement on the Picotar is that they have no polarity. This means that they can be put into the PCB in any orientation and will function the same i.e. provide the same amount of resistance.

3.7 The Light Emitting Diode (LED)

Within the Picotar, there are four LEDs of varying colours, as shown in Figure 10 (left), one of which is used to indicate power, and the other three are cosmetic, and light up when each note is played i.e. each button is pressed.

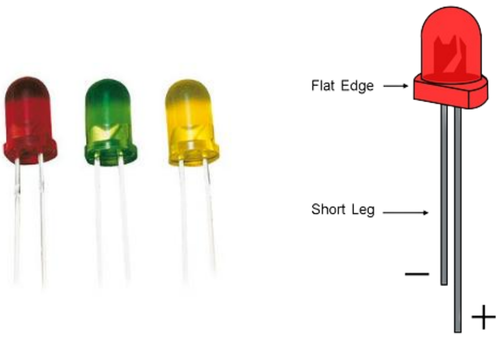


Figure 9: Light-Emitting Diodes (left) & Polairty Indicators (right)

Whilst a diode is a semiconductor device that only allows current to flow in one direction, an LED differs from a typical diode by emitting light when a current is passed through it. This means that it has polarity and must be carefully orientated in the Picotar to ensure correct operation. Typically, two observations can be used to identify the positive and negative sides of an LED, shown in Figure 10 (right). The first is that the two legs of an LED vary in length, with the longer being the anode i.e. positive, whilst the negative, i.e. cathode, is shown by the shorter leg. The second method is that on the base of the LED, there is a flat edge. This indicates the negative side and can be used when the LED has been soldered and the component legs already trimmed.

4. Quick Guide to Soldering

You do not need to have done any soldering before this activity. In the lab today there is lots of support available from both staff and current students – feel free to ask for help if you would like additional help or a demonstration of soldering.

This section is to act as a quick guide to soldering for the first time, but this is a skill that will quickly improve with practice.

It is important to remember that soldering performs two main tasks; these being:

- Providing an electrical connection in a circuit.
- Fixing a component to the circuit board, also known as a printed circuit board (PCB).

Therefore, when we solder, we need to ensure that not only is a good 'clean' solder joint made, but also that the component will remain fixed in place. For example, consider the problems if components were to 'fall off' from a board used in an industrial robot or your laptop/phone.

When soldering, we use a number of basic tools, including:

		
Soldering Iron	Soldering Iron Station & Extractor Fan	Solder
		
Goggles	Wire Cutters	Desoldering 'Solder Sucker' Pump

Table 2: Common Tools Used in Soldering

Before we start, it is always best to check your soldering iron to ensure that:

- The electrical cable is of a high standard, with no damage.
- The 'PAT' safety test is up to date.
- The solder tip is of a high standard i.e. not bent and is shiny.

If your soldering iron isn't up to scratch, let us know and we will replace it with a new one.

4.1 Preparation

The first task is to get the soldering iron warmed up, which can take a minute or two – the temperature should settle on either 375°C or 400°C depending on the solder station preset. Once the soldering iron is heated, wipe it on the gold wire wool to remove any oxidation – this will show as a dark residue. If it does not come away easily you may need to use tip cleaner which we can show you how to use if needed.

You now need to apply a small amount of solder to 'tin the tip'. The aim here is to give a shiny layer of solder as shown in Figure 11 (right).

Figure 10: Wire Wool (left) & the Tip when 'Tinned' with Solder (right)

Now it is time to place the component in the board. It is recommended you read to the end of this section before soldering any components into the Picotar board. When doing this, ensure that the component is placed flush to the board and that, if required, it is aligned/orientated correctly. Please ask if you have any queries.

You can also bend the legs of the component out slightly to help keep it in place, as shown in Figure 12.

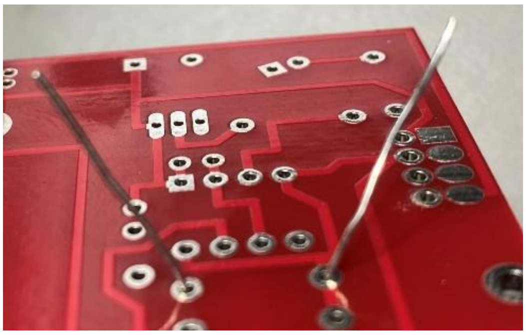


Figure 11: Component Legs Slightly Bent to Prevent the Component from Falling Out

It is important to remember that when soldering, we are aiming for 'the ideal solder joint' which resembles a cone or volcano – as shown in Figure 13.

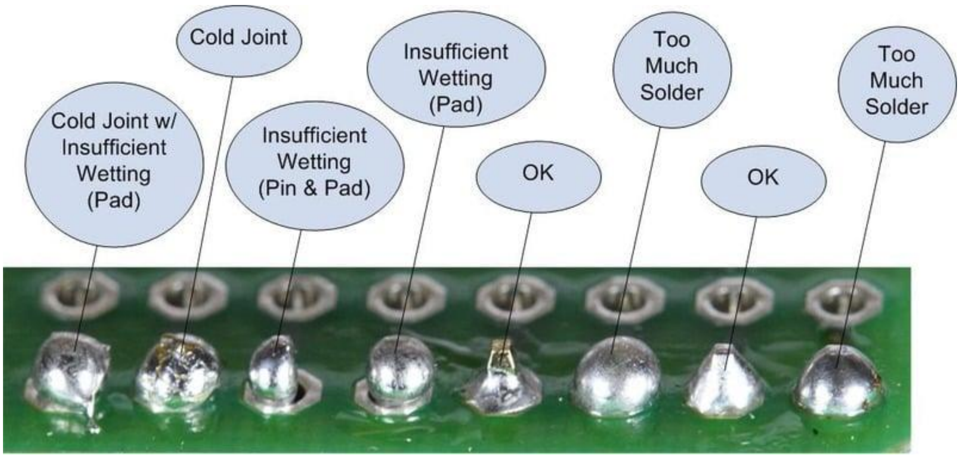


Figure 12: Solder Joint Guide

4.2 The Soldering Process

Now we know what we are aiming for, it is time to make the solder joint — this is a five-stage process consisting of:

1. **Heating the Joint:** Be sure to heat both the solder pad and the component leg or pin; a small amount of solder on the tip will help the heat flow.

2. **Apply the Solder**: Touch the end of the solder to the joint so that it contacts both the solder pad and the component leg or pin. It should melt and flow smoothly onto both the pin and the pad. If the solder does not flow, heat the joint for another second or two and try again.

3. **Let It Flow**: Keep heating the solder and allow it to flow into the joint. It should fill the hole and flow smoothly onto both the solder pad and the pin or component leg.

4. **Let It Cool**: Once enough solder has been added to the joint and it has flowed well onto both the component leg and the solder pad, remove the iron from the joint and allow it to cool undisturbed.

5. **Trim the Leg**: Use your wire cutters to trim the component leg close to the board. Please wear goggles when doing this as component legs may ‘fly off’. This step applies only to components with wire legs. It is not necessary to trim the pins on integrated circuit chips or surface mount components – but don’t worry, none of these are in your kits today.

This is pictorially shown below:

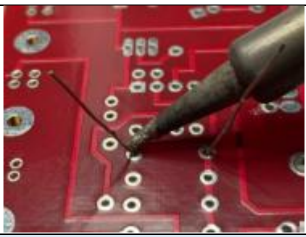
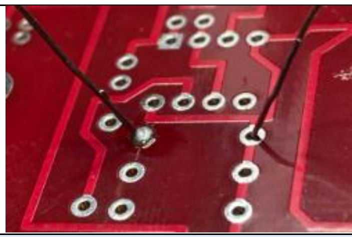
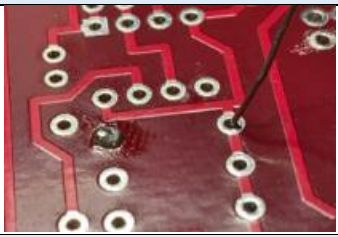
		
1. Heat the Joint	2. Apply the Solder	3. Let the Solder Flow & 4. Let the Solder Cool
		
5. Trim the Wire	A Well-Soldered Joint	

Figure 13: The Stages of Soldering

Have a go and please make use of the support on offer. Whilst soldering is a key skill in electrical and electronic engineering, it is developed through continuous practice – so whilst your first joint may not be ideal, you will see a noticeable improvement with your final few solders by the end of this activity. However, **if in doubt regarding the orientation and placement of components, it is always best to check prior to any soldering** – as it is much easier to fix any issues before the components are soldered down.

5. Programming the Picotar

5.1 The Example Codes

There are two 'ready to go' example codes available for you to upload to the Picotar. Below each example code is broken down along with the function of each component. Additionally, within each example code, once opened, you can read through the notes on how the program, and the functions, work.

Picotar_Three_Chord_String_Guitar_Example

This example code allows the Picotar to function in a similar way to an authentic guitar, with string-like sounds being played. The purpose of each component is as follows:

- The three buttons are used to play three different chord notes.
- The linear potentiometer is used to vary the pitch of a note being played – you can slide your finger up and down whilst a button is pressed to vary the pitch in real-time.
- One rotary potentiometer is used to vary the volume of the audio output.
- The other rotary potentiometer is used to vary the decay/sustain time of the audio output i.e. how long the sound plays for once the button is released.
- The three LEDs, on the Picotar head, toggle on and off depending on which buttons are pressed.

Picotar_Samples_Metronome_AudioFX_Example

This example code allows the Picotar to be set up in three different modes; a metronome, audio sample player and echo distortion. Each mode is selected by pressing each of the buttons, which lights up a corresponding LED on the Picotar head. Regardless of the mode, one rotary potentiometer is used to vary the volume of the audio output.

Metronome; where a rhythmic beat is played.

- The other rotary potentiometer is used to vary the speed i.e. beats per minute, of the metronome.
- The linear potentiometer is not used in this mode.

Audio Sample Player; where a preloaded 'bamboo' sample is played.

- The other rotary potentiometer is used to vary the playback speed of the sample.
- The linear potentiometer is not used in this mode.

Echo Distortion; where an echo distortion effect is played.

- The other rotary potentiometer is used to vary the amount of averaging applied to the audio i.e. the amount of echo and distortion on the audio output.
- The linear potentiometer is also used to vary the amount of averaging applied to the audio i.e. the amount of echo and distortion on the audio output.

5.2 Uploading the Picotar Example Code

The following section will demonstrate the process of uploading code to the Picotar. The screenshots included will show the 'Picotar_Three_Chord_String_Guitar_Example' code being uploaded, however, the process is identical with the other example code; 'Picotar_Samples_Metronome_AudioFX_Example'.

On the desktop, you should see a ‘Picotar_Three_Chord_String_Guitar_Example’ folder which contains the example code inside. Double-click the sketch ‘ino’ file to open it in the Arduino IDE, you should see a window like Figure 15 – the Arduino IDE version may differ.

Figure 14: Arduino IDE & Example Code

Within the Arduino IDE, in the top left, there is a ‘tick’ and an ‘arrow’, which are the ‘Verify’ and ‘Upload’ buttons respectively. You can verify the code without any microcontroller being connected, and this will highlight any compilation errors or verify that the code can be uploaded via a message in the output window at the bottom. You can then click the upload button to program a microcontroller once it is connected correctly, as detailed below.

As shown in Figure 15, you can see that the code has already been written for you and is ready to upload ‘as is’. The comments in grey after the ‘//’ on each line fully explain how the code works so please take the time to read through these.

Before you can upload to the Picotar, the Raspberry Pi Pico W microcontroller that we are using is not installed by default. Therefore, to install it, go to ‘File → Preferences’.

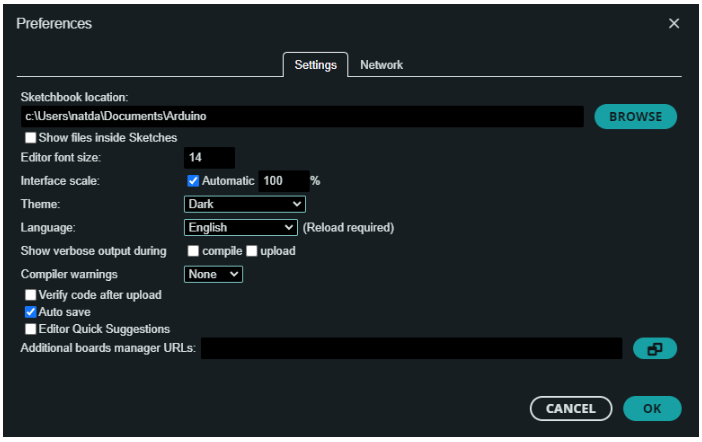


Figure 15: Preferences Window

In the ‘Additional boards manager URLs:’ box, type or paste the following – on smaller screens, you may need to scroll down to find this dialogue box:

https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json

Click ‘OK’ to save the new URL. Now go to the boards manager by clicking ‘Tools → Board → Boards Manager…’, as shown in Figure 17 below.

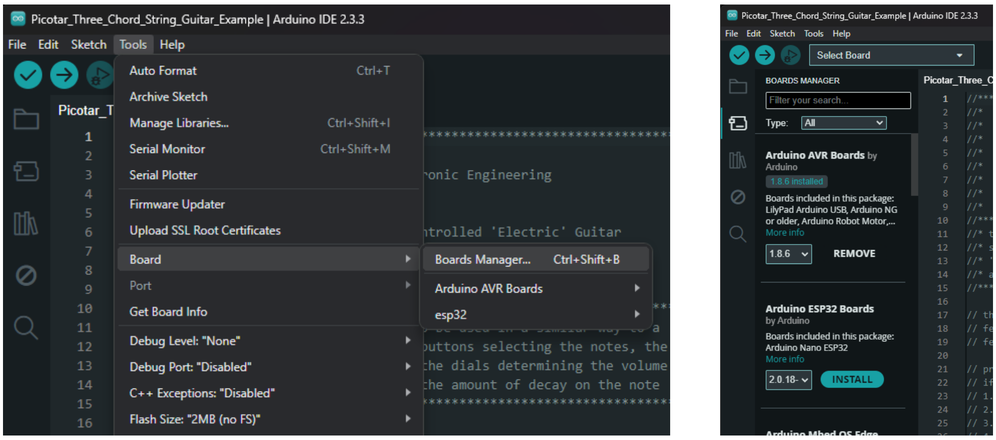


Figure 16: Boards Manager Select Menu (left) & Menu Tab (right)

Within this menu tab, type 'pico', and **install the latest version of 'Raspberry Pi Pico/RP2040/RP2350' by Earle F. Philhower, III.** This may take several minutes and will show 'Installed' when complete – installation messages for each board will appear in the output window.

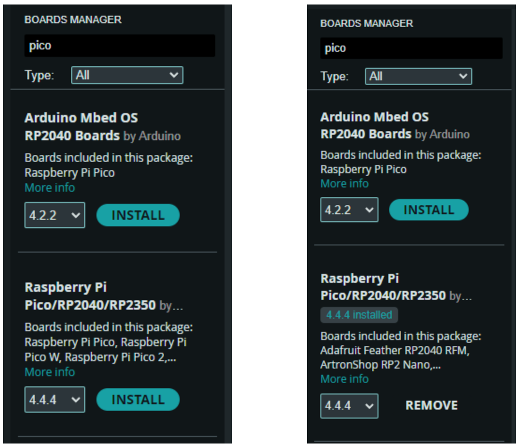


Figure 17: Boards Manager Window Pre-Installation (left) & Post-Installation (right)

To select the correct microcontroller board, select 'Tools → Board → Raspberry Pi Pico/RP2040/RP2350 → Raspberry Pi Pico W'.

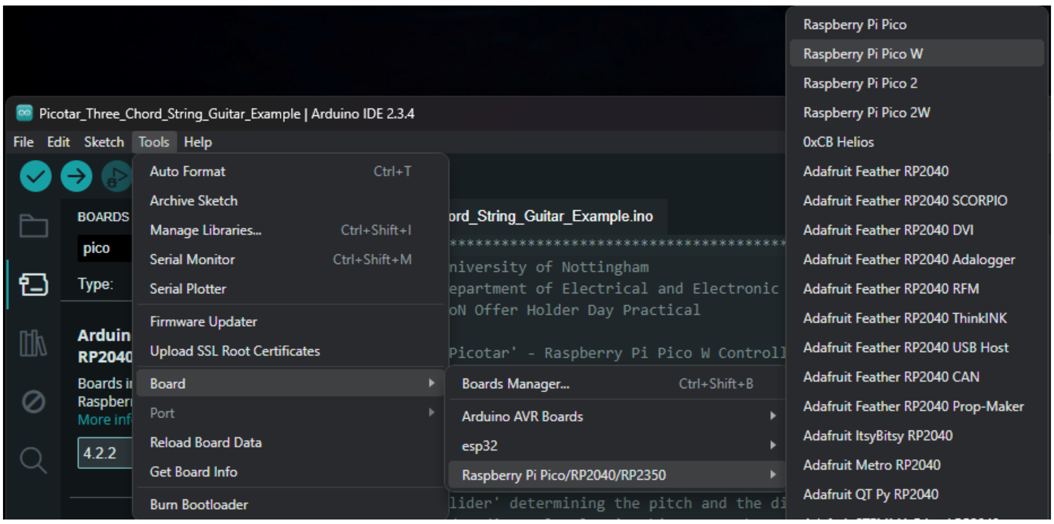


Figure 18: Selecting the Raspberry Pi Pico W Board

Now we need to install the Mozzi library before our code will compile and upload to the Picotar. To do this, open the ‘Library Manger’ by clicking ‘Sketch → Include Library → Manage Libraries…’, as shown in Figure 20 below.

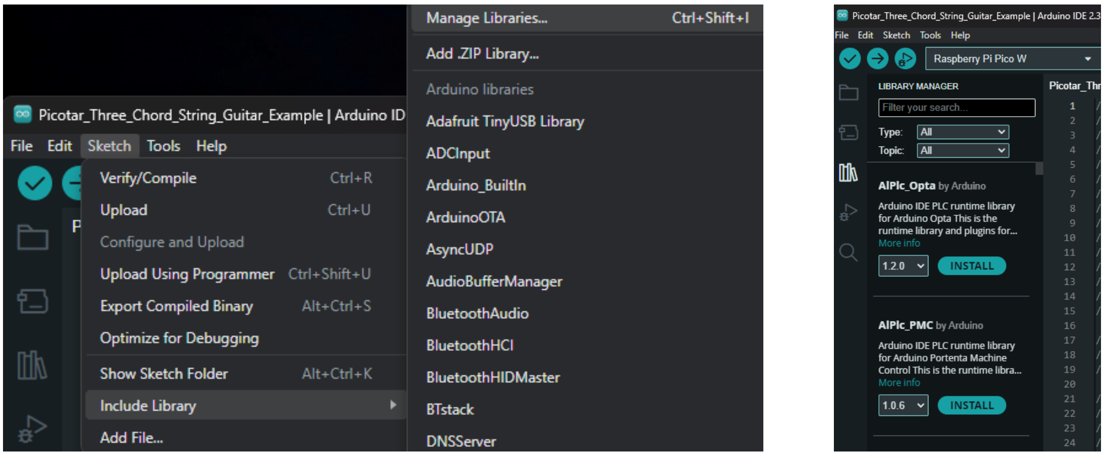


Figure 19: Library Manager Select Menu (left) & Menu Tab (right)

Within this menu tab, type ‘mozzi’ and install ‘Mozzi’ by Tim Barrass. **You must ensure that you install version 1.1.2 – do not install version 2.x as the example code will not work.** Do this by clicking the drop-down menu and selecting ‘1.1.2’. Once installed, this will show ‘1.1.2 installed’ as well as an ‘Update’ button – **do not click this nor update this library if prompted when opening the program in the future**. The output window will also display ‘Installed Mozzi@1.1.2’ once complete.

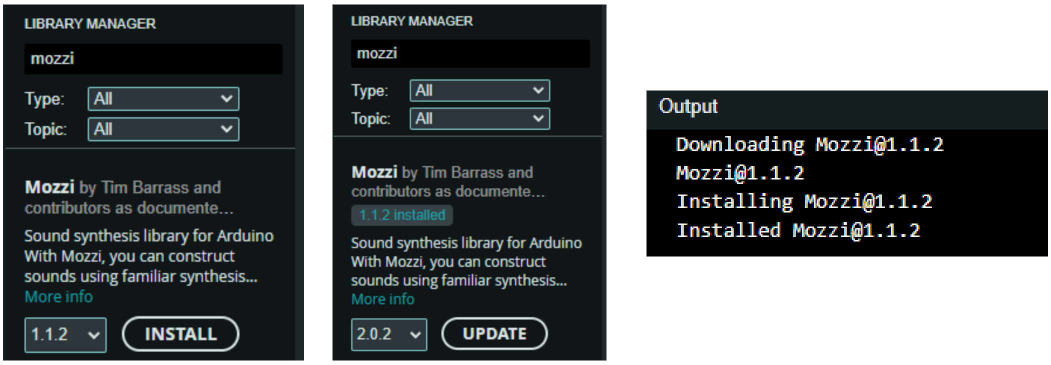


Figure 20: Library Manager Window Pre-Installation (left) & Post-Installation (middle) & Output Message (right)

Now that everything is installed, plug the Raspberry Pi Pico W microcontroller into the Picotar, and use the USB cable to connect it to the PC (or your laptop). Ensure that the correct board is selected and that you have selected the correct COM port under ‘Tools → Port’. The correct port will have the description shown in Figure 22, although the port number may be different.

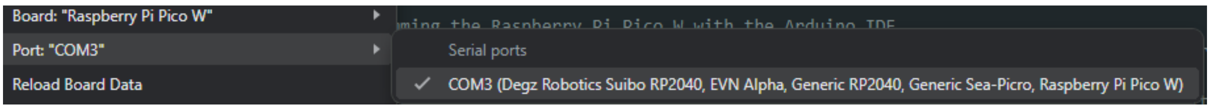


Figure 21: Raspberry Pi Pico W Port Description

Once selected, click upload – this may take several minutes. Ignore the red #warning that is displayed in the output window. Once you see a message that says ‘Wrote 685056 bytes to H:/NEW.UF2’ in the output window, you have successfully uploaded the code and your Picotar should be working – **keep the USB cable plugged in to power it!** If the Picotar isn’t working or the code hasn’t been uploaded, see the troubleshooting steps on the next page.

5.3 Troubleshooting Steps

Troubleshooting is a key engineering skill, and whilst not a complete guide, please check the following, and feel free to ask for help if it still doesn't work.

If the Code Didn't Upload…

- To check that you have the correct COM port on Windows, open your 'Device Manager' by right-clicking the Windows icon in the bottom left and selecting 'Device Manager'. Under 'Ports (COM & LPT)' you are looking for the COM port number that is listed as 'USB Serial Device'.
- Disconnect and reconnect the USB cable at both ends. You may also want to try a different USB port on the PC/laptop – don't forget to recheck the COM port number.
- If you are still stuck, please ask one of the ambassadors or staff in the lab for help.

If the Code Did Upload…

- Check all of the components which you soldered, and resolder if needed;
    - Are all of the solder joints a high quality?
    - Are the LEDs placed in the correct way round?
    - Are the correct resistor values used in the correct place?
- If all of the solder joints are good, then, with a screwdriver, rotate the blue 'trimpot', shown in Figure 23, located between the audio amplifier chip and the microcontroller. Gradually rotate this all the way in one direction, and then in the other direction, whilst pressing one of the buttons. All being well, you should hear the quality and volume of the note being played vary – stop rotating when you have found the position that delivers the best quality output.
- If you are still stuck, please ask one of the ambassadors or staff in the lab for help.

Figure 22: Blue Trimpot on the Picotar

6. What's Next?

You have now completed the Picotar activity, but what's next?

Firstly, **this document and the Picotar are yours to keep and play around with**. You can power the Picotar, either with the USB cable, or if you'd prefer a more portable guitar, you can place three standard AA batteries in the battery pack at the back and ensure that the power switch is switched on.

We would love to hear from you and see what you do with the Picotar – *'7. Useful Links'*, contains links to some resources which may be of interest, but consider the following:

- Modify the example code to play different sounds or add different audio effects, you may wish to explore the Mozzi library further; we have only scratched the surface.
- Use the Raspberry Pi Pico W microcontroller in some of your own custom projects – this board is a versatile microcontroller which can be used to drive most DIY electronics projects.

7. Useful Links

7.1 Links to the Software & Resources Used

Arduino IDE
https://www.arduino.cc/en/software

Under 'Software', download the Arduino IDE relevant to your operating system. Note that this installer will not include the board or library setup so you will need to repeat these steps.

GitHub Repository
https://github.com/University-of-Nottingham-EEE-Projects/Picotar

This contains all of the example code used today, as well as resources related to the Picotar PCB.

7.2 Useful Resources

For more information and ideas on what you could do next, please look at the following links. Please do let us know what you create, either with the Picotar or your own projects, by emailing nathaniel.dacombe@nottingham.ac.uk

Mozzi
https://sensorium.github.io/Mozzi/

Here you can find everything that you need to know about Mozzi, as well as more example code and a link to the community forums where you can see other projects using the Mozzi library.

Raspberry Pi Pico Project Ideas
https://how2electronics.com/raspberry-pi/raspberry-pi-pico-projects/

This contains a list of over 75 different Raspberry Pi Pico-related projects – have a go at these, use them for inspiration or come up with your own entirely new ideas!

8. Thank You for Visiting!

Finally, all that is left to say is thank you for visiting the Electrical and Electronic Engineering Department today!

We hope you've enjoyed this activity and we look forward to welcoming you as a student – remember to take your Picotar project and documentation home with you!