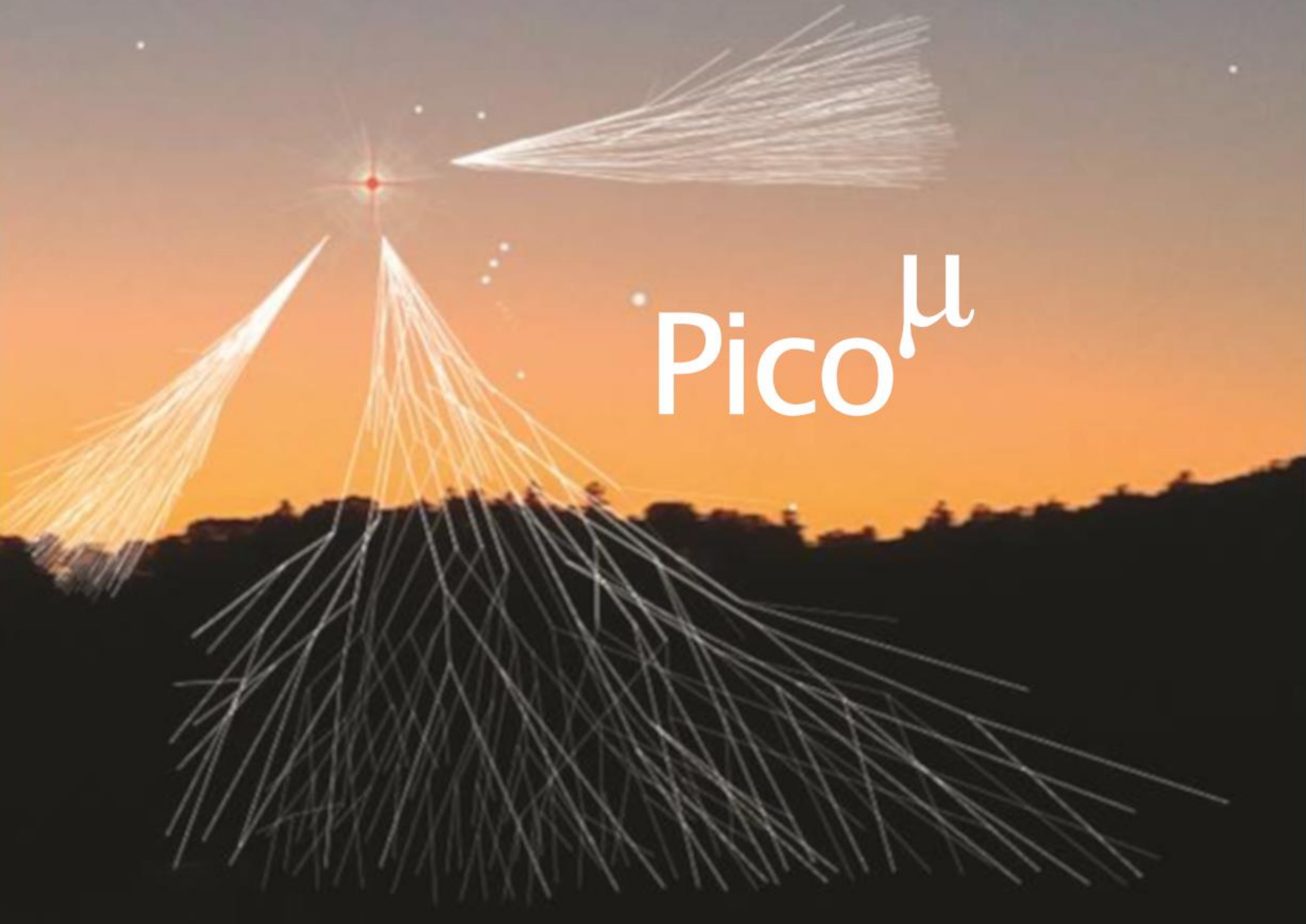
PicoMuon Detector

Setting the RTC



https://ukraa.com/

The UK Radio Astronomy Association
A charitable incorporated organisation
Registered charity in England and Wales No. 1123866

Setting the PicoMuon DS3231 RTC

Set of instructions for setting the date/time on the DS3231 RTC module of the UKRAA PicoMuon detector.

Table of Contents

The CR2032 button cell story reason…

UK Customers

A 3V CR2032 button cell is required to support the real time clock. The detector is supplied with a battery installed to UK customers.

Overseas Customers

No battery is supplied.

The latest air cargo regulations are very strict about dangerous goods such as batteries. We are unable to supply our overseas customers with a battery. We provide detailed instructions for fitting the battery and initialising the real time clock. We are also happy to provide technical support by Zoom or telephone – see Contact us.

What CR2032 button cell

Whilst we do not recommend one brand of button cell over another brand of button cell, currently we are supplying UK customers with either:

- DURACELL CR 2032 button cell (**DL2032-B2)**, or
- Panasonic CR2032 button cell. (**CR2032EL/1BP**)

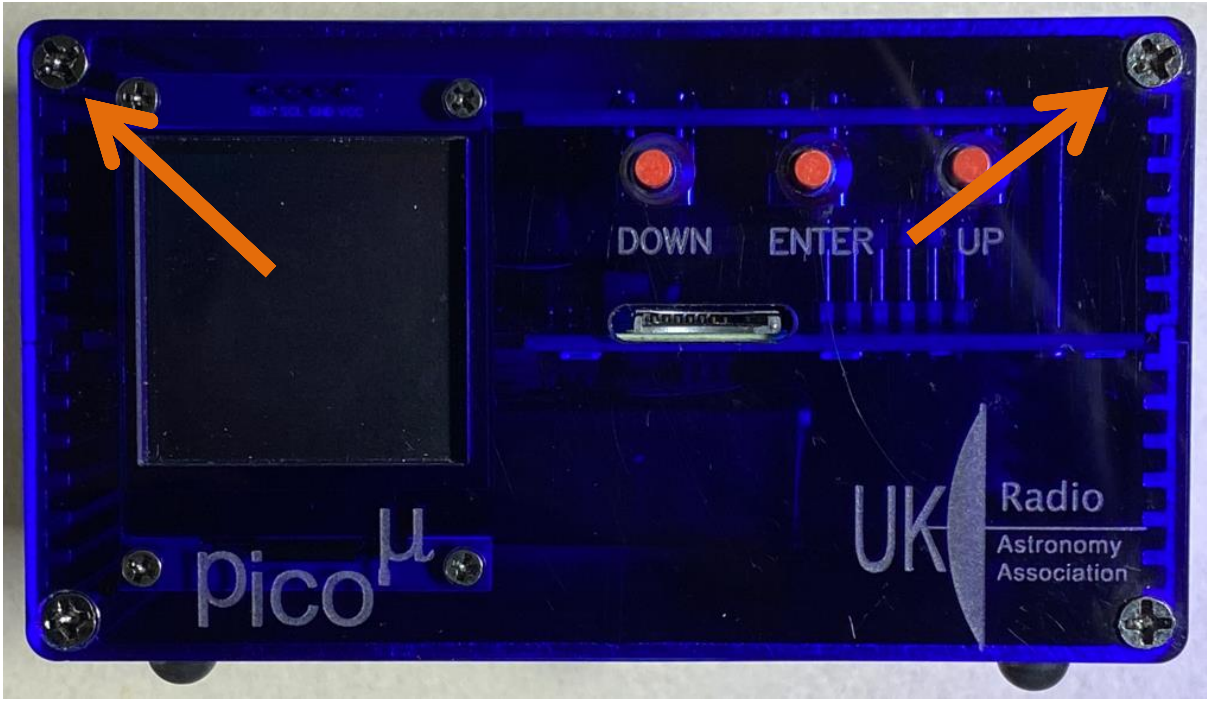
The DS3231 RTC module has provisions for the use of a rechargeable CR2032 button cell when connected to an external supply source.

It should be noted that there is often a difference in button cell capacity between non-rechargeable (about 250mAh) and rechargeable (about 70mAh) – hence if you plan to have the unit unplugged for a length of time, it may be more suitable to have a non-rechargeable button cell installed.

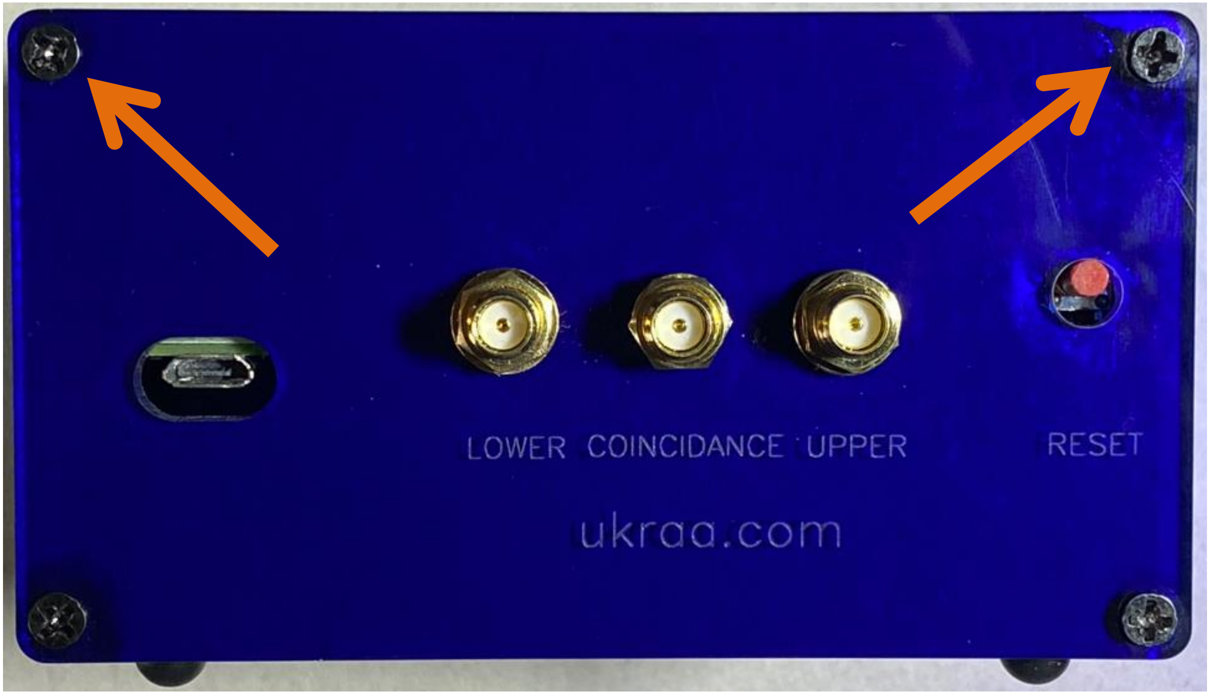
It should also be noted that there is often a difference in button cell nominal voltage between non-rechargeable (3V) and rechargeable (3.7V) – as the supply voltage to the DS3231 RTC module is 3V3, no testing has been undertaken on use of rechargeable CR2032 button cell within the UKRAA PicoMuon detector

Inserting a CR2032 into your PicoMuon unit

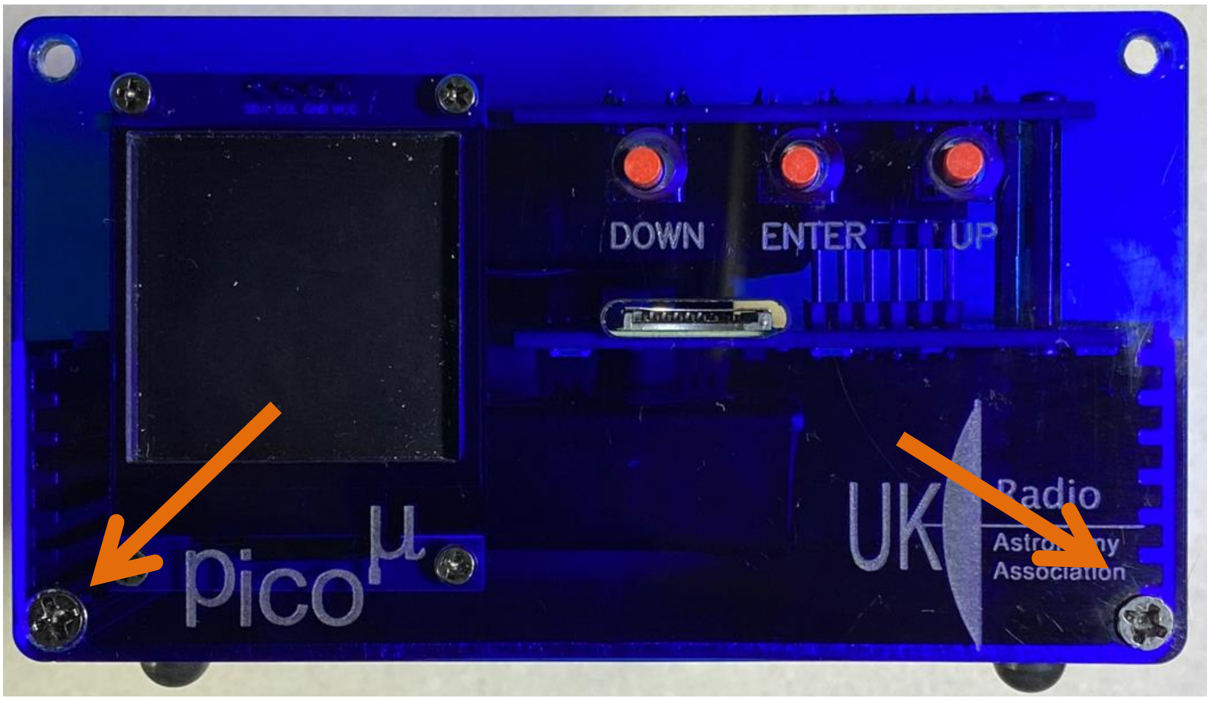
1. Remove the upper x2 case screws (nominally PH2 size) from the front panel.



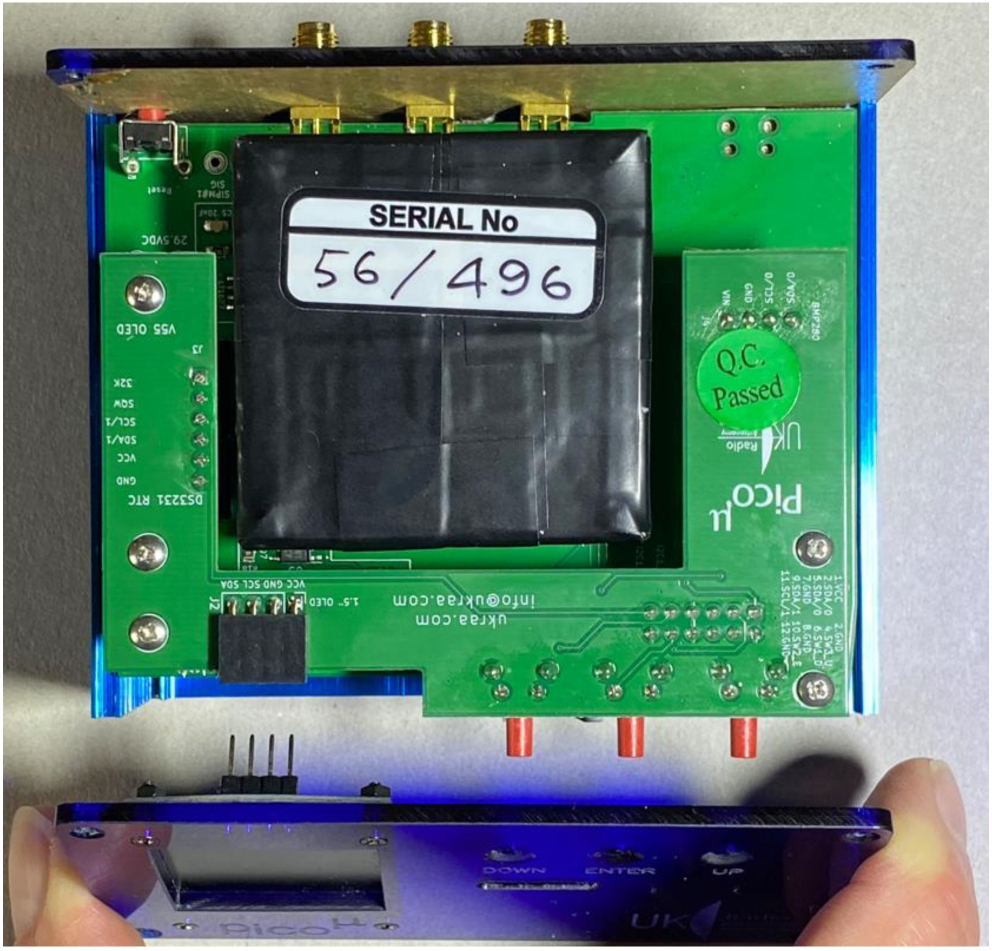
2. Remove the upper x2 case screws (nominally PH2 size) from the rear panel.



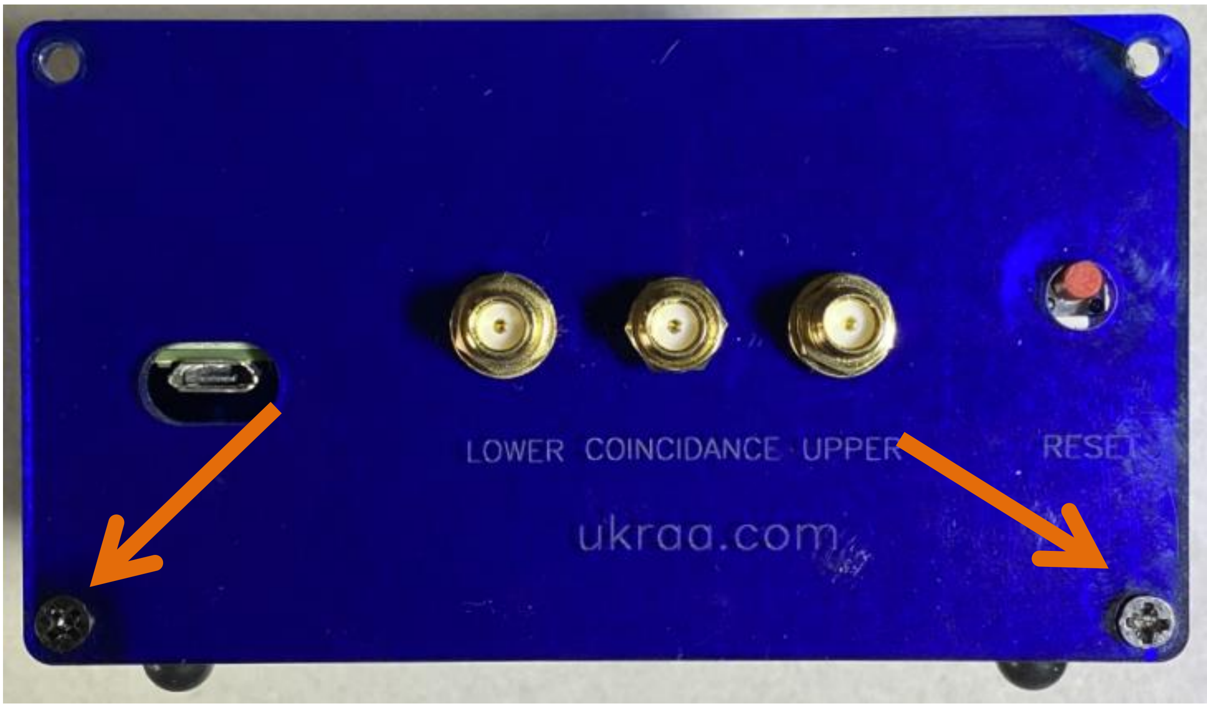
3. Remove the lower x2 case screws (nominally PH2 size) from the front panel.



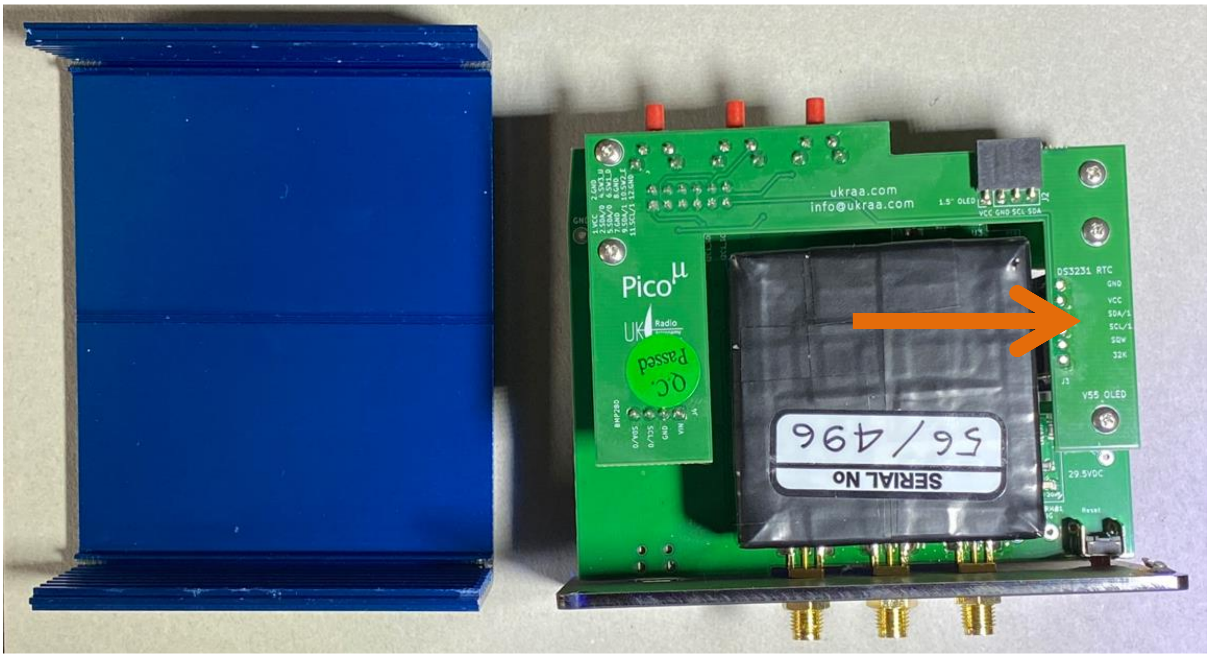
4. The front panel can now be removed from the PicoMuon detector. Pull forward to remove the 1x4 OLED pin header from the 1x4 pin socket located on the upper PCB.



5. Remove the lower x2 case screws (nominally PH2 size) from the rear panel.

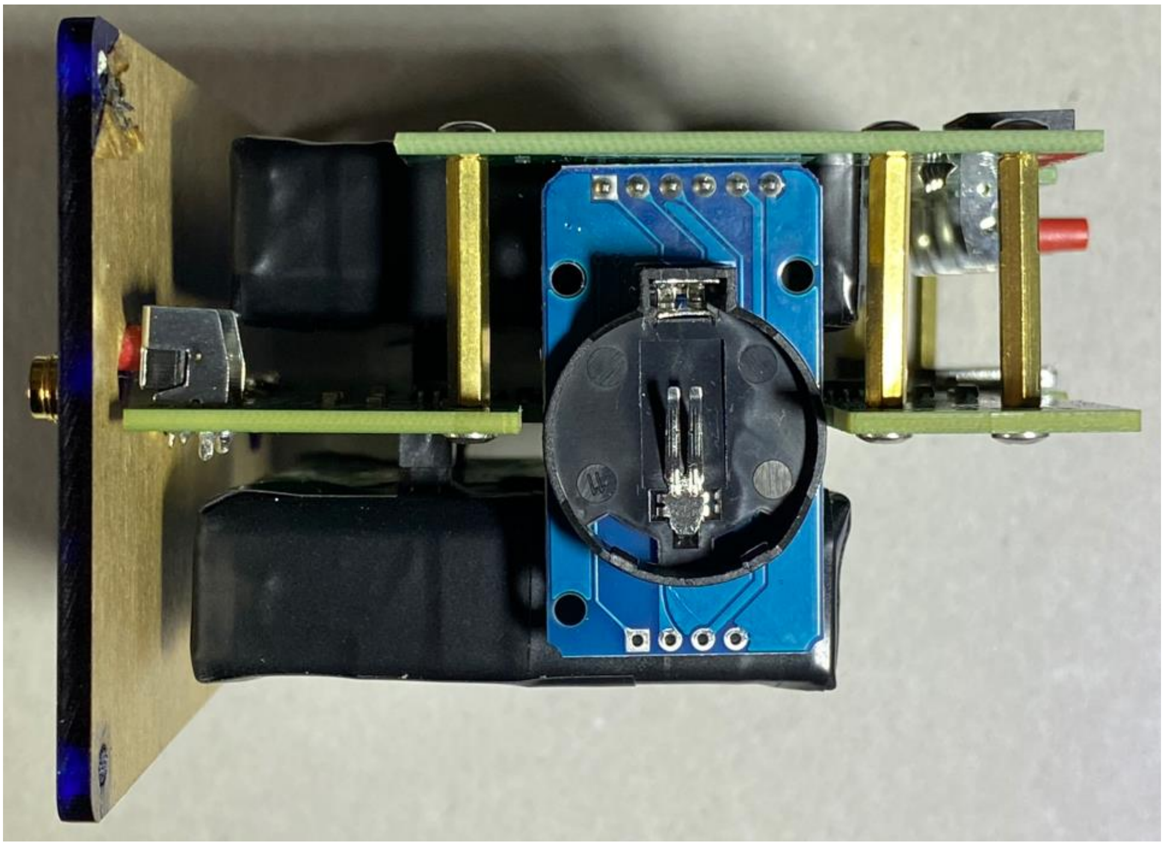


6. Now remove the PCB assemblies from the lower case.

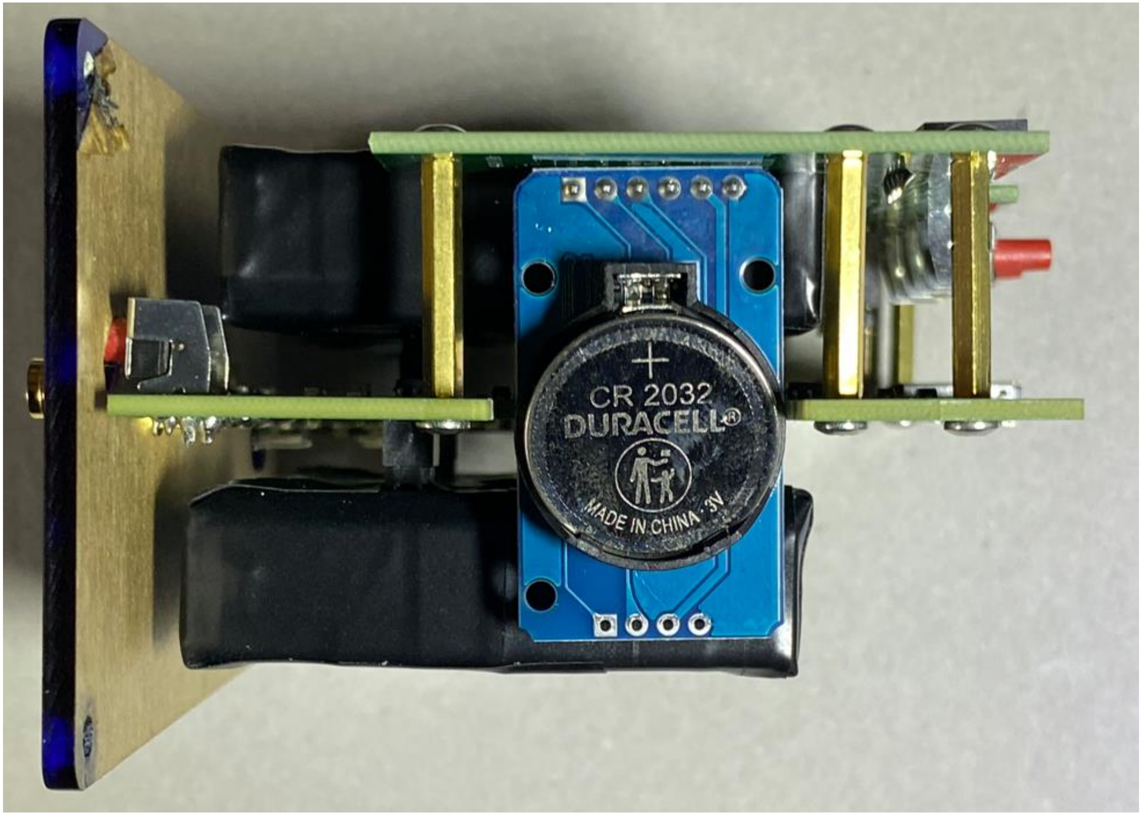


7. The DS3231 RTC module is located to the right of the above picture.

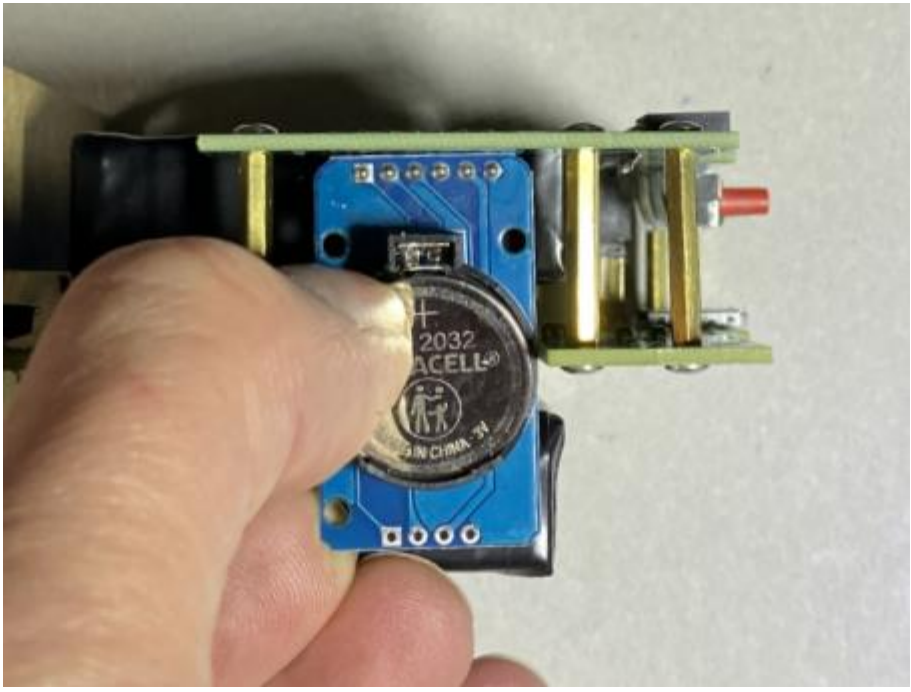
8. Rotate your PicoMuon detector to gain access to the DS3231RTC module battery holder.



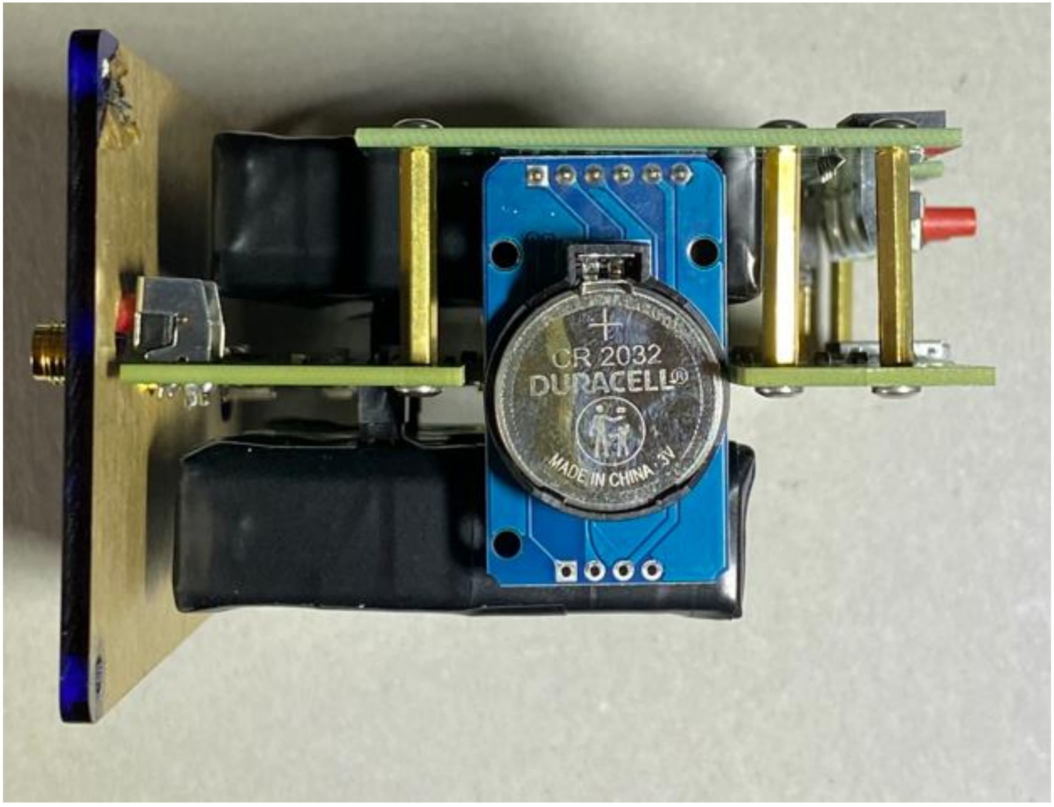
9. Carefully place your CR2032 button cell into the DS3231 RTC button cell holder. Ensure that the + terminal of the button cell is facing out – towards you.



10. Apply pressure to the top of the CR2032 button cell, while supporting the bottom of the DS3231 RTC module, until the CR2032 button cell is fully seated in the DS3231 RTC battery holder – you will hear a click.



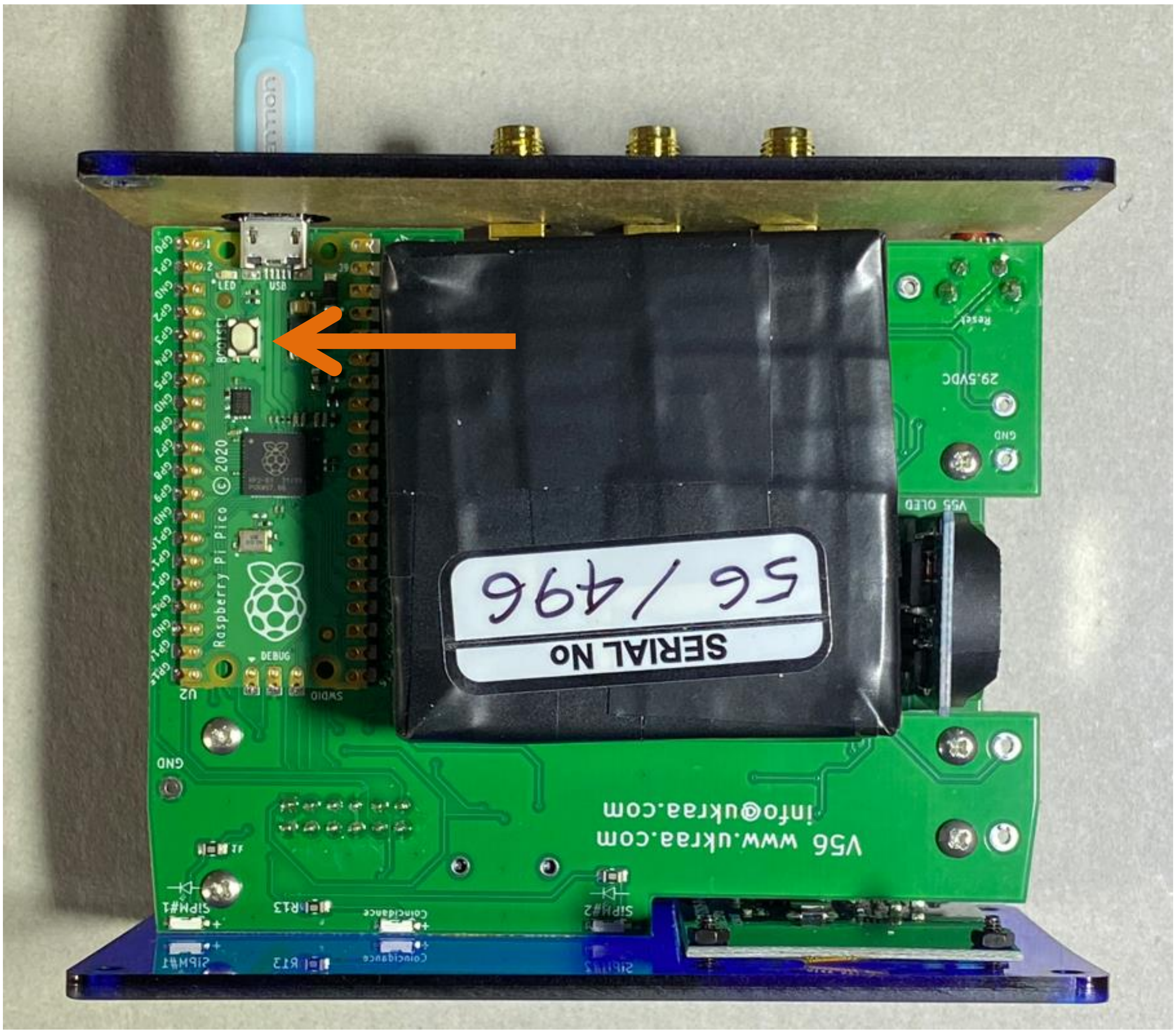
11. Review and ensure that the CR2032 button cell is fully seated into the DS3231 RTC module battery holder.



12. Replace the front panel by inserting the 1x4 OLED pin header into the 1x4 pin socket located on the upper PCB.

13. Insert the supplied micro USB cable into the RPi Pico micro USB socket.

14. Rotate the unit so that the RPi Pico is accessible. You will need to use the **BOOT SEL** button located on the RPi Pico.



Setting the RTC on PicoMuon unit

Thonny IDE

This section requires the use of Thonny IDE.

Thonny IDE can be obtained from the following URL: https://thonny.org/

No instructions are given on installing Thonny IDE – there are numerous instructional videos available through YouTube on undertaking this task.

microSD card

The PicoMuon detector is supplied with a microSD card that contains, not only commissioning files/data but also, all necessary MicroPython files to check DS3231 RTC module date/time and set DS3231 RTC module date/time.

On the supplied microSD card there are a couple of folders of note:

- **MicroPython UF2** – this holds the current MicroPython UF2 bootloader for the RPi Pico.
- **PicoMuon MicroPython** – this holds the MicroPython code needed to check and set the DS3231 RTC date time.

Preparing the PicoMuon unit for RTC date/time

The PicoMuon detector is supplied with the necessary compile C code to enable detection and recording of events from the two scintillator assemblies.

To set the DS3231 RTC date/time we are going to install MicroPython and then run two MicroPyton scripts.

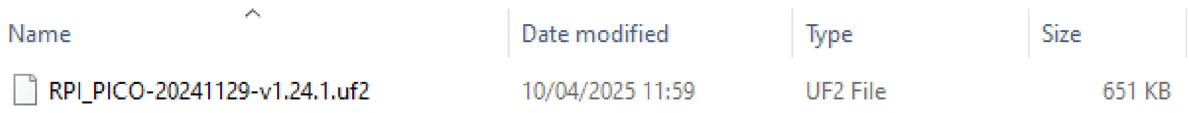
The following instructions are for a Windows based PC.

To prepare the PicoMuon detector to be able to load MicroPython undertake the following.

1. Copy the **MicroPython UF2** folder from the microSD card to your desktop PC.

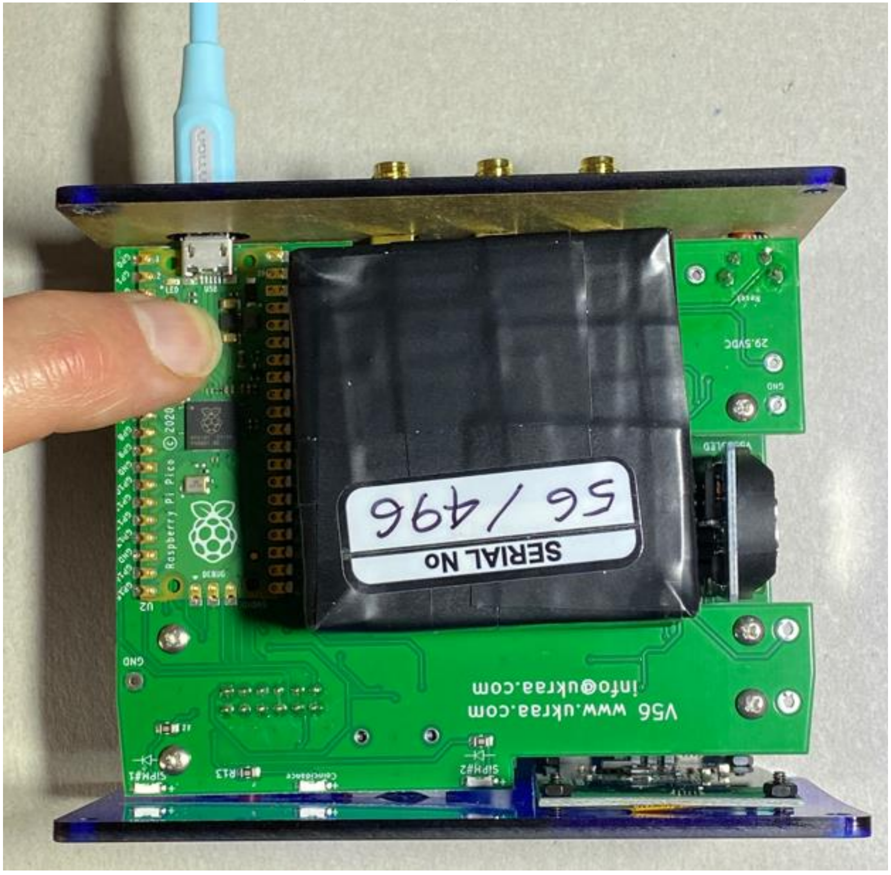
2. Copy the **PicoMuon MicroPython** folder from the microSD card to your desktop PC.

3. Open the **MicroPython UF2** folder on your desktop PC. You will see a single UF2 file.



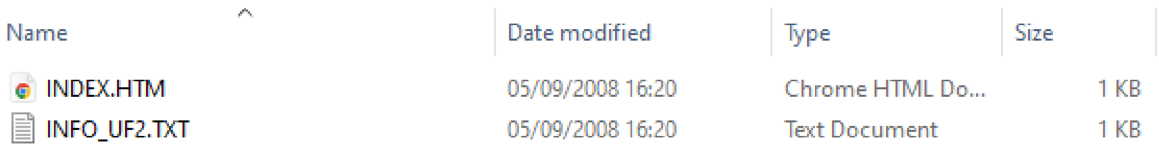
Name	Date modified	Type	Size
RPI_PICO-20241129-v1.24.1.uf2	10/04/2025 11:59	UF2 File	651 KB

4. While pressing the **BOOT SEL** button on the RPi Pico, insert the USB A connector into a suitable USB port on your desktop PC.



5. Release the **BOOT SEL** button once the USB cable is plugged into your desktop PC.

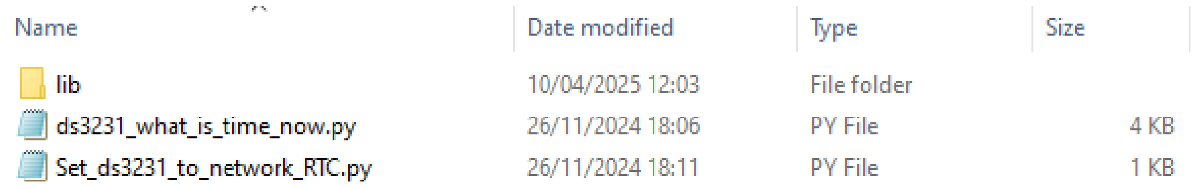
6. Open a second Windows Explorer and navigate to the RPI-RP2 drive, and open the drive. You will see two files.



Name	Date modified	Type	Size
INDEX.HTM	05/09/2008 16:20	Chrome HTML Do...	1 KB
INFO_UF2.TXT	05/09/2008 16:20	Text Document	1 KB

7. From the **MicroPython UF2** folder drag and drop the RPI_**PICO-xxxxxxx-vx.xx.x.uf2** file onto the **RPI-RP2** folder. This will install MicroPython bootloader onto your Pico. As a result your RPI-RP2 drive will disappear from the Window Explorer drive listing. This indicates that MicroPython bootloader was successfully installed onto the Pico.

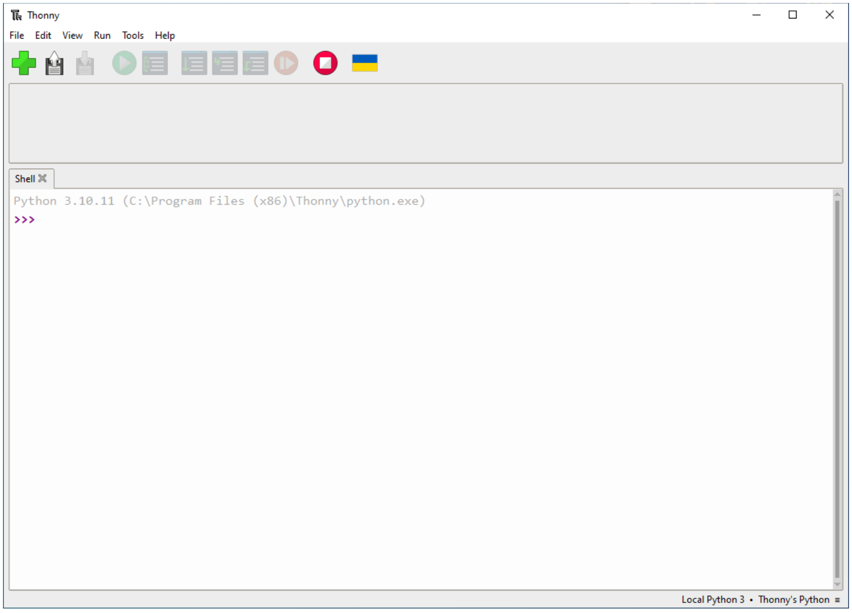
8. Open the **PicoMuon MicroPython** folder on your desktop PC. You will see two files and a folder.



Name	Date modified	Type	Size
lib	10/04/2025 12:03	File folder	
ds3231_what_is_time_now.py	26/11/2024 18:06	PY File	4 KB
Set_ds3231_to_network_RTC.py	26/11/2024 18:11	PY File	1 KB

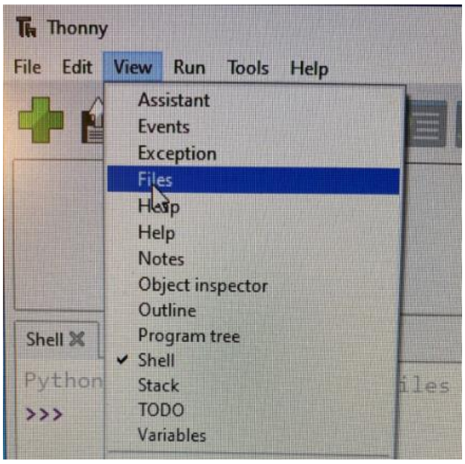
9. Run **Thonny IDE**.

10. When first opening Thonny you may have a screen that looks like this…

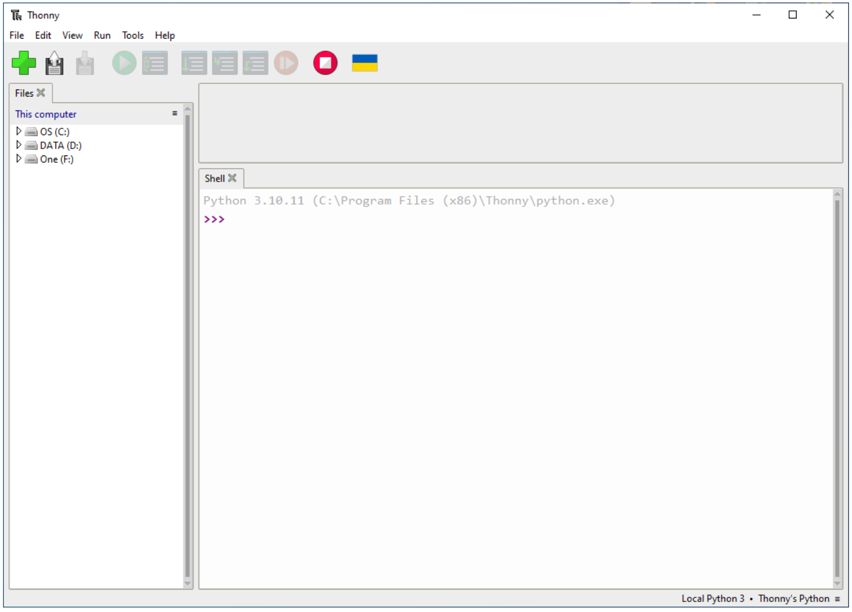

Thonny
File Edit View Run Tools Help
Shell
Local Python 3 • Thonny's Python

```
Python 3.10.11 (C:\Program Files (x86)\Thonny\python.exe)
>>>
```

11. We need to be able to access the **File** panel – to do this select **View** from the top ribbon bar and from the drop down menu select **Files**.

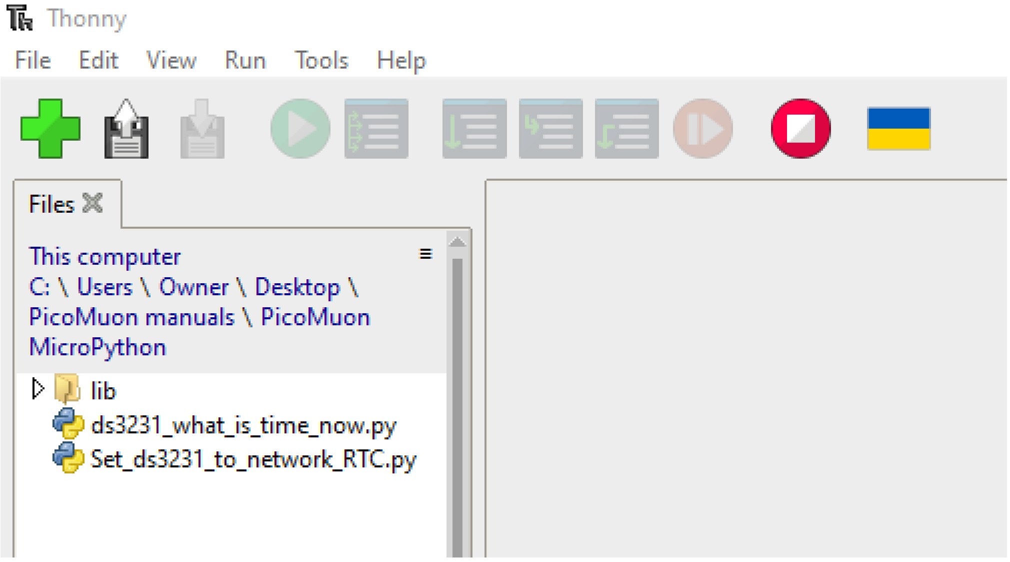


12. Thonny will now look like this…


Thonny
File Edit View Run Tools Help
Files
This computer
OS (C:)
DATA (D:)
One (F:)
Shell
Local Python 3 • Thonny's Python

```
Python 3.10.11 (C:\Program Files (x86)\Thonny\python.exe)
>>>
```

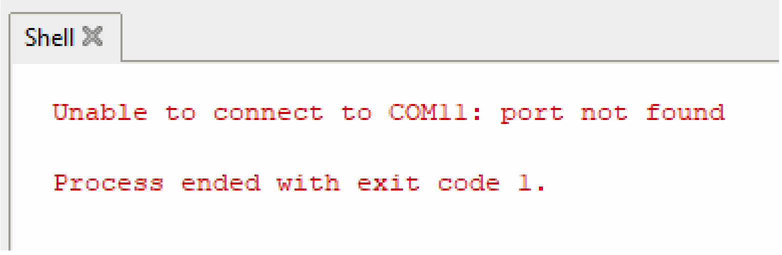
13. Within the **Files** panel on the left hand side of **Thonny IDE**, navigate to where you have stored the PicoMuon MicroPython folder on your desktop PC.



This computer
C: \ Users \ Owner \ Desktop \ PicoMuon manuals \ PicoMuon MicroPython

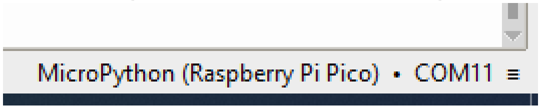
- lib
- ds3231_what_is_time_now.py
- Set_ds3231_to_network_RTC.py

14. Looking at the **Shell** panel towards the lower right of **Thonny IDE**, you should see that there is a red message that no connection to a Pico is made.



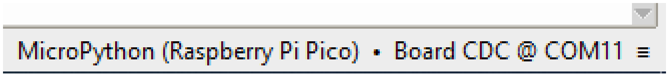
```
Unable to connect to COM11: port not found

Process ended with exit code 1.
```

15. Moving your mouse to the lower right hand corner of **Thonny IDE**, you should see something similar to the following.



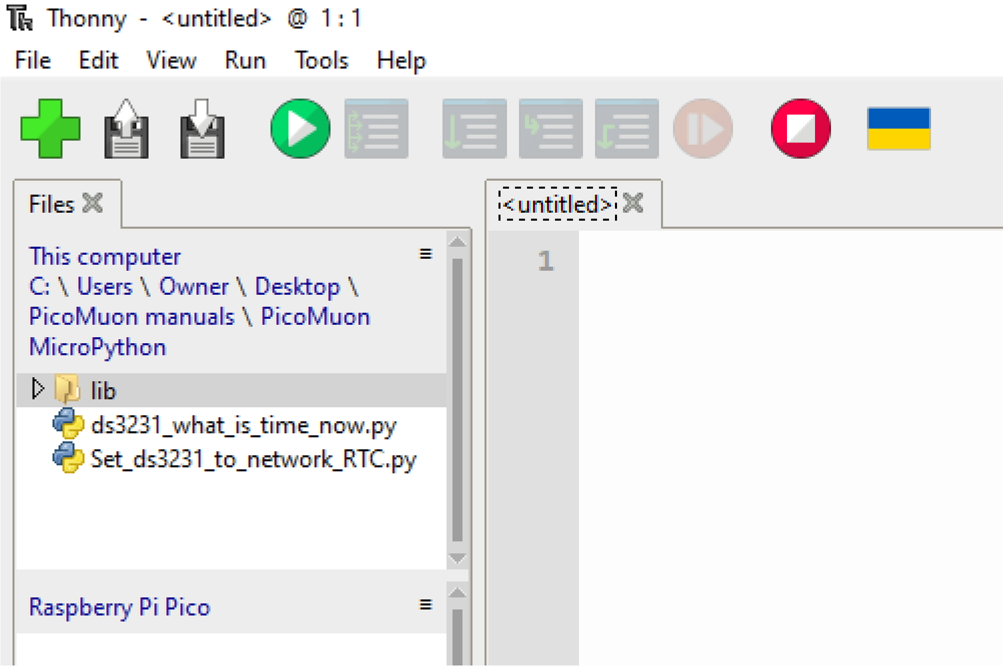
MicroPython (Raspberry Pi Pico) • COM11 ≡

16. If you left click with your mouse on this text you will get a listing of available Pico's that have MicroPython bootloader installed that **Thonny IDE** can connect to. Selecting your Pico COM port, you should see something similar to the following.



MicroPython (Raspberry Pi Pico) • Board CDC @ COM11 ≡

17. The **File** panel will now show your RPi Pico as a separate area.



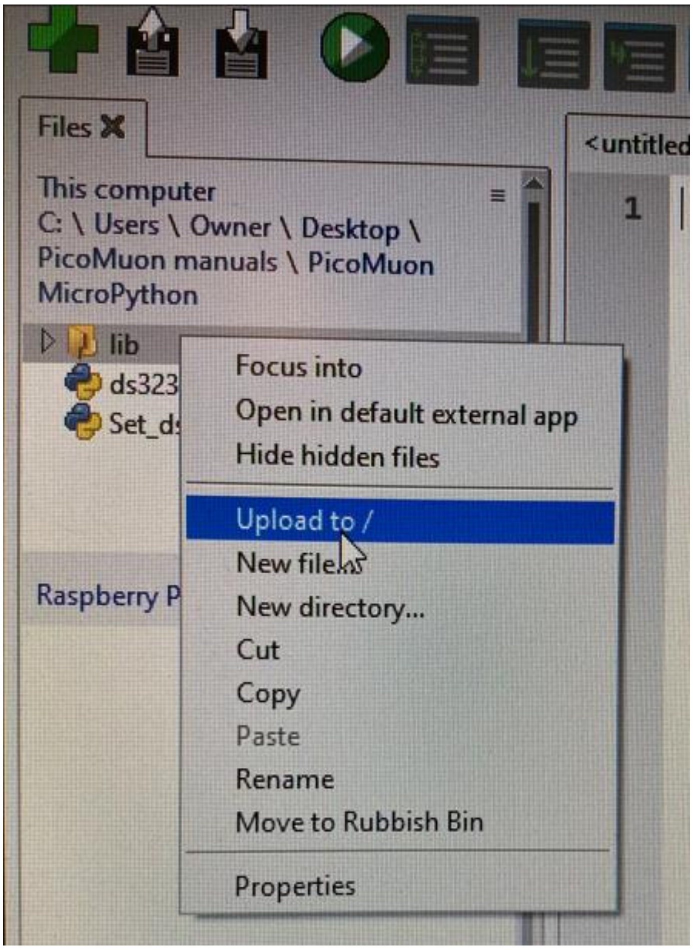
Thonny - <untitled> @ 1 : 1

This computer
C: \ Users \ Owner \ Desktop \ PicoMuon manuals \ PicoMuon MicroPython

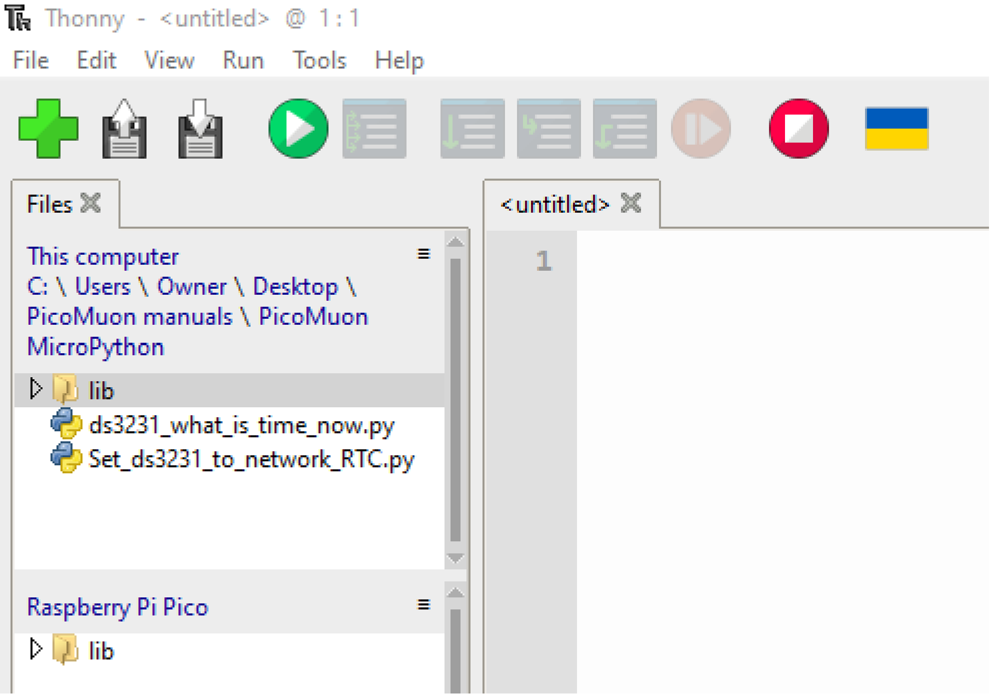
lib
ds3231_what_is_time_now.py
Set_ds3231_to_network_RTC.py

Raspberry Pi Pico

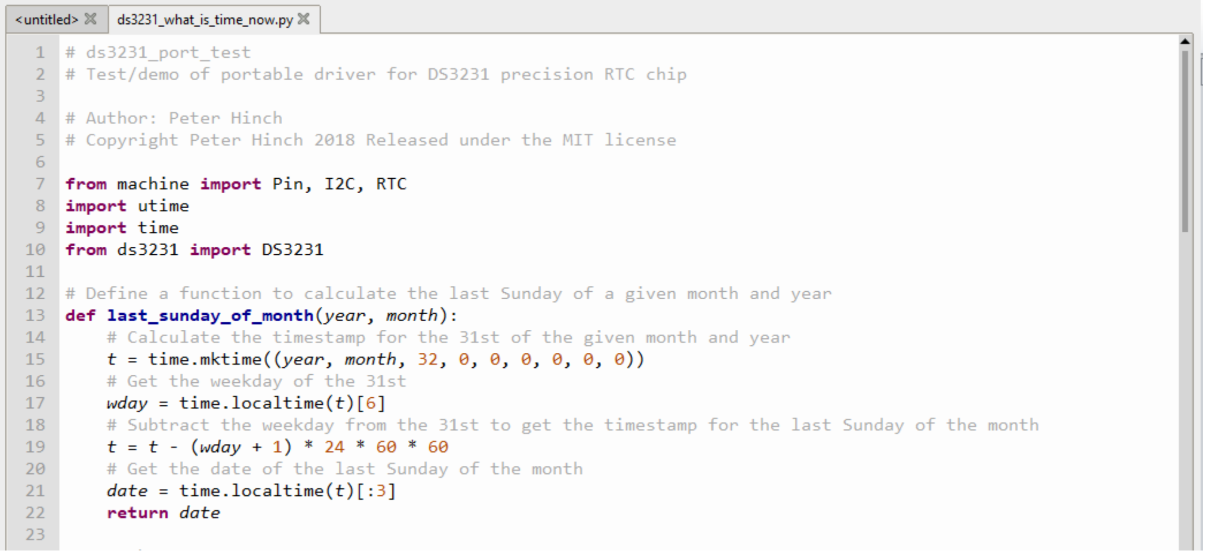
18. Highlight the **lib** folder and right click your mouse. This will bring up a menu screen. You want to select **Upload to /**. This will automatically upload the **lib** folder to the Pico.



19. The **File** panel will ow show you RPi Pico with a lib folder.


Thonny - <untitled> @ 1 : 1
File
Edit
View
Run
Tools
Help
Files
<untitled>
1
This computer
C: \ Users \ Owner \ Desktop \ PicoMuon manuals \ PicoMuon MicroPython
lib
ds3231_what_is_time_now.py
Set_ds3231_to_network_RTC.py
Raspberry Pi Pico
lib

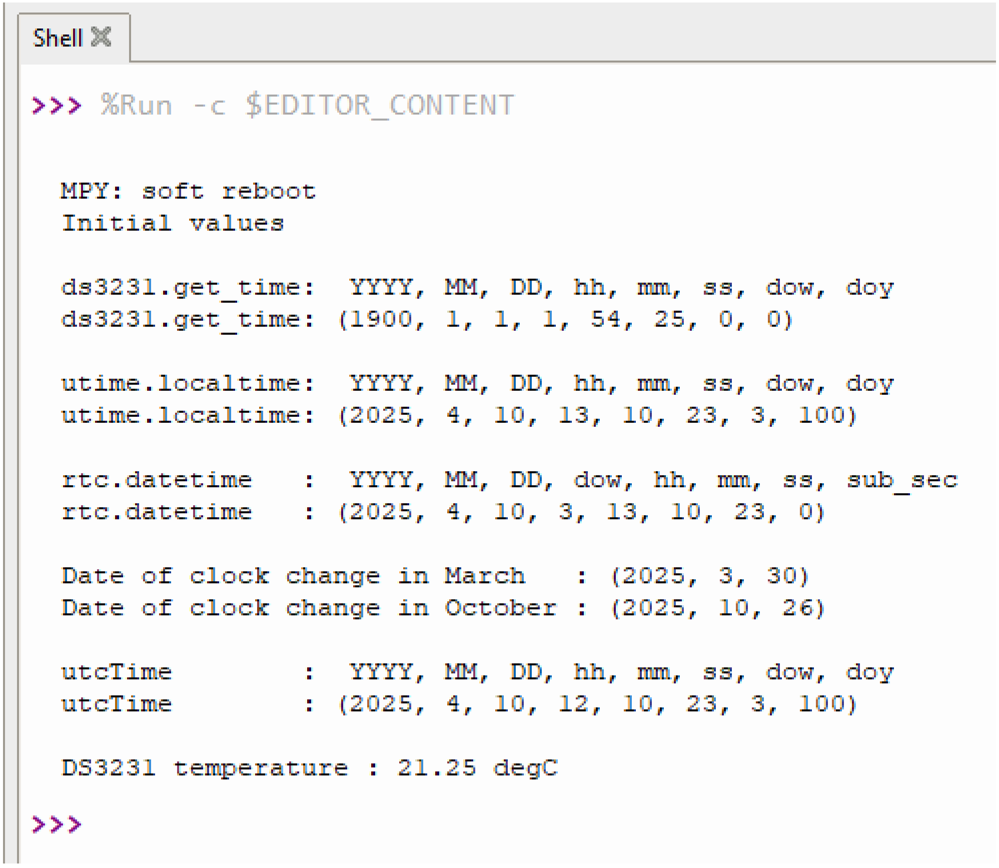
20. Select **ds3231_what_is_time_now.py** from the **Files** panel and double click on it. This will bring the code into the **Thonny IDE** main panel.



```
# ds3231_port_test
# Test/demo of portable driver for DS3231 precision RTC chip

# Author: Peter Hinch
# Copyright Peter Hinch 2018 Released under the MIT license

from machine import Pin, I2C, RTC
import utime
import time
from ds3231 import DS3231

# Define a function to calculate the last Sunday of a given month and year
def last_sunday_of_month(year, month):
    # Calculate the timestamp for the 31st of the given month and year
    t = time.mktime((year, month, 32, 0, 0, 0, 0, 0))
    # Get the weekday of the 31st
    wday = time.localtime(t)[6]
    # Subtract the weekday from the 31st to get the timestamp for the last Sunday of the month
    t = t - (wday + 1) * 24 * 60 * 60
    # Get the date of the last Sunday of the month
    date = time.localtime(t)[:3]
    return date

```

21. Run the code using the ▶ icon on the **Thonny IDE** menu ribbon.

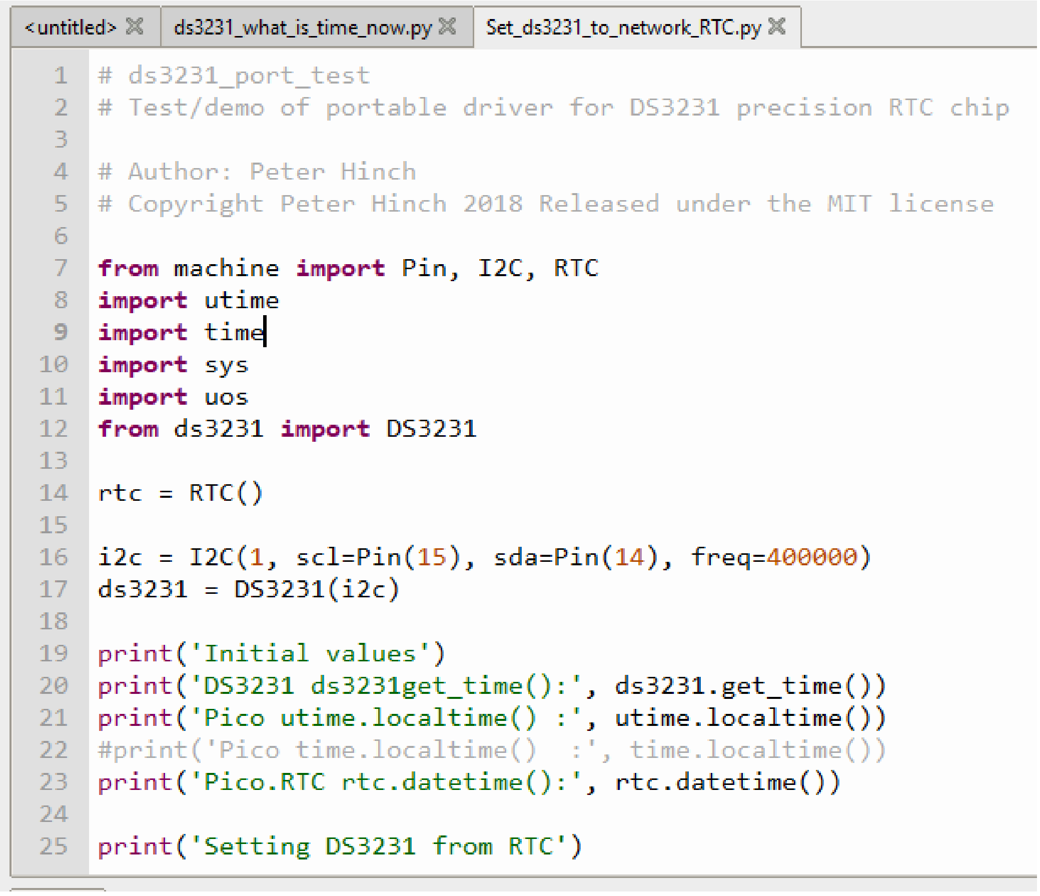
22. The output from the code will be displayed in the **Shell** panel.



```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Initial values

  ds3231.get_time:  YYYY, MM, DD, hh, mm, ss, dow, doy
  ds3231.get_time: (1900, 1, 1, 1, 54, 25, 0, 0)

  utime.localtime:  YYYY, MM, DD, hh, mm, ss, dow, doy
  utime.localtime: (2025, 4, 10, 13, 10, 23, 3, 100)

  rtc.datetime    :  YYYY, MM, DD, dow, hh, mm, ss, sub_sec
  rtc.datetime    : (2025, 4, 10, 3, 13, 10, 23, 0)

  Date of clock change in March    : (2025, 3, 30)
  Date of clock change in October : (2025, 10, 26)

  utcTime         :  YYYY, MM, DD, hh, mm, ss, dow, doy
  utcTime         : (2025, 4, 10, 12, 10, 23, 3, 100)

  DS3231 temperature : 21.25 degC

>>>
```

23. The Pico is picking up the date/time through the **Thonny IDE**, whilst the DS3231RTC date/time is the base values when a button cell is first inserted.

24. Select **Set_ds3231_to_network_RTC.py** from the **Files** panel and double click on it. This will bring the code into the Thonny IDE main panel.



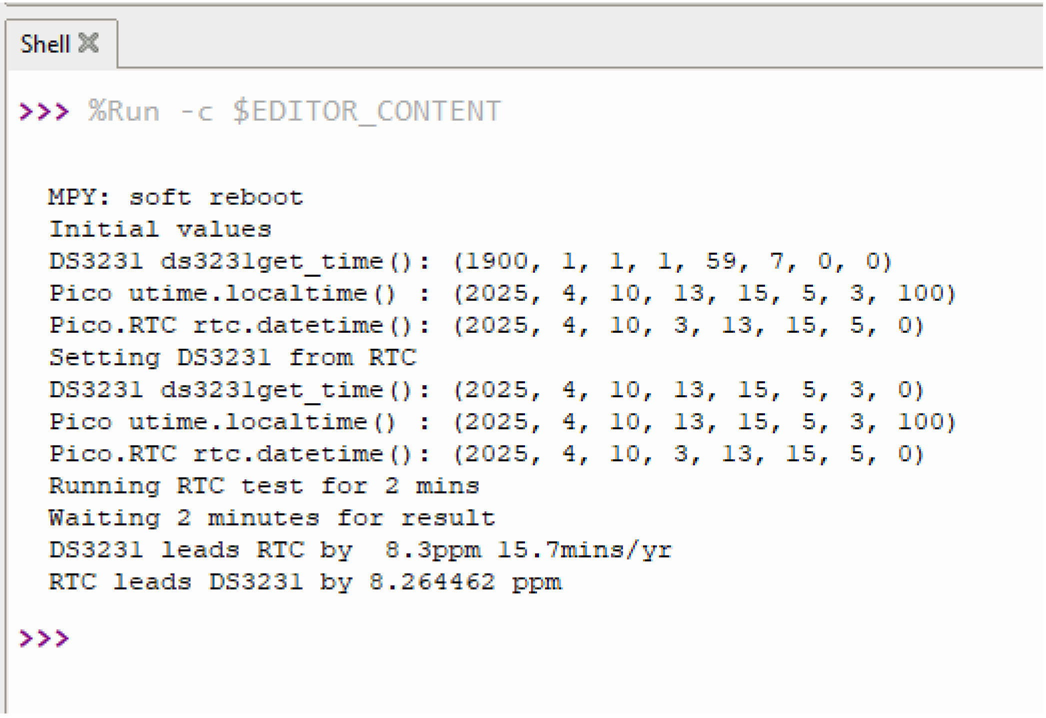
```
# ds3231_port_test
# Test/demo of portable driver for DS3231 precision RTC chip

# Author: Peter Hinch
# Copyright Peter Hinch 2018 Released under the MIT license

from machine import Pin, I2C, RTC
import utime
import time
import sys
import uos
from ds3231 import DS3231

rtc = RTC()

i2c = I2C(1, scl=Pin(15), sda=Pin(14), freq=400000)
ds3231 = DS3231(i2c)

print('Initial values')
print('DS3231 ds3231get_time():', ds3231.get_time())
print('Pico utime.localtime() :', utime.localtime())
#print('Pico time.localtime()  :', time.localtime())
print('Pico.RTC rtc.datetime():', rtc.datetime())

print('Setting DS3231 from RTC')
```

25. Run the code using the ▶ icon on the **Thonny IDE** menu ribbon.

The code will set the DS3231 RTC date/time to your current network date/time and then perform a test of the DS3231 RTC date/time against your network date/time.

This test will take 2 minutes to complete.

26. The output from the code will be displayed in the **Shell** panel.



```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Initial values
  DS3231 ds3231get_time(): (1900, 1, 1, 1, 59, 7, 0, 0)
  Pico utime.localtime() : (2025, 4, 10, 13, 15, 5, 3, 100)
  Pico.RTC rtc.datetime(): (2025, 4, 10, 3, 13, 15, 5, 0)
  Setting DS3231 from RTC
  DS3231 ds3231get_time(): (2025, 4, 10, 13, 15, 5, 3, 0)
  Pico utime.localtime() : (2025, 4, 10, 13, 15, 5, 3, 100)
  Pico.RTC rtc.datetime(): (2025, 4, 10, 3, 13, 15, 5, 0)
  Running RTC test for 2 mins
  Waiting 2 minutes for result
  DS3231 leads RTC by  8.3ppm 15.7mins/yr
  RTC leads DS3231 by 8.264462 ppm

>>>
```

You can see that the DS3231 RTC has been set to the correct date/time.

You can see a typical expected error for the DS3231 RTC of about 16 minutes per year.

27. Setting of the DS3231 RTC date/time is now complete.

28. You now have a PicoMuon detector with the correct date/time on the DS3231 RTC, but no software to actually detect events. Follow the next section to fully get your PicoMuon detector working.

Install compiled code to your PicoMuon unit

microSD card

The PicoMuon detector is supplied with a microSD card that contains, not only commissioning files/data but also, the compiled C code that will enable your PicoMuon detector to detect and process ionising events recorded from the scintillator assemblies.

On the supplied microSD card there is a folder of note:

- **PicoMuon UF2** – this holds the compiled C code needed to run your PicoMuon detector.

Preparing the PicoMuon unit for recording events

You will have removed the compile C code to enable detection and recording of events from the two scintillator assemblies in setting the date/time of the D3231 RTC module.

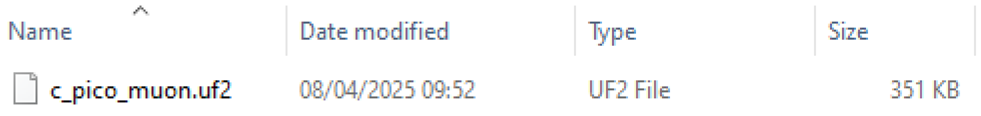
You therefore need to reinstall the compiled C code to your PicoMuon detector.

The following instructions are for a Windows based PC.

To prepare the PicoMuon detector to be able to load MicroPython undertake the following.

1. Copy the **PicoMuon UF2** folder from the microSD card to your desktop PC.

2. Open the **PicoMuon UF2** folder on your desktop PC. You will see a single UF2 file.



Name	Date modified	Type	Size
c_pico_muon.uf2	08/04/2025 09:52	UF2 File	351 KB

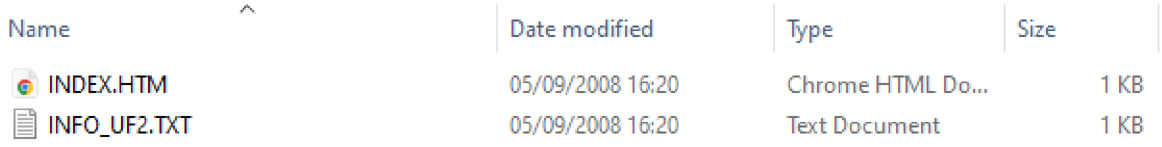
3. With the PicoMuon detector still connected to your desktop PC via its supplied USB cable.

4. Press and hold the **RESET** button located from the rear panel of the PicoMuon detector and, at the same time, press and hold the **BOOT SEL** button on the RPi Pico.

5. Then release the **RESET** button, while still holding the **BOOT SEL** button on the RPI Pico.

6. Now release the **BOOT SEL** button on the RPi Pico. , insert the USB A connector into a suitable USB port on your desktop PC.

7. Open a second Windows Explorer and navigate to the **RPI-RP2** drive, and open the drive. You will see two files.



Name	Date modified	Type	Size
INDEX.HTM	05/09/2008 16:20	Chrome HTML Do...	1 KB
INFO_UF2.TXT	05/09/2008 16:20	Text Document	1 KB

8. From the **PicoMuon UF2** folder drag and drop the **c_pico_muon.uf2** file onto the **RPI-RP2** folder. This will install the compiled C code to the PicoMuon detector. As a result your **RPI-RP2** drive will disappear from the Window Explorer drive listing. This indicates that compiled C code was successfully installed onto the Pico.

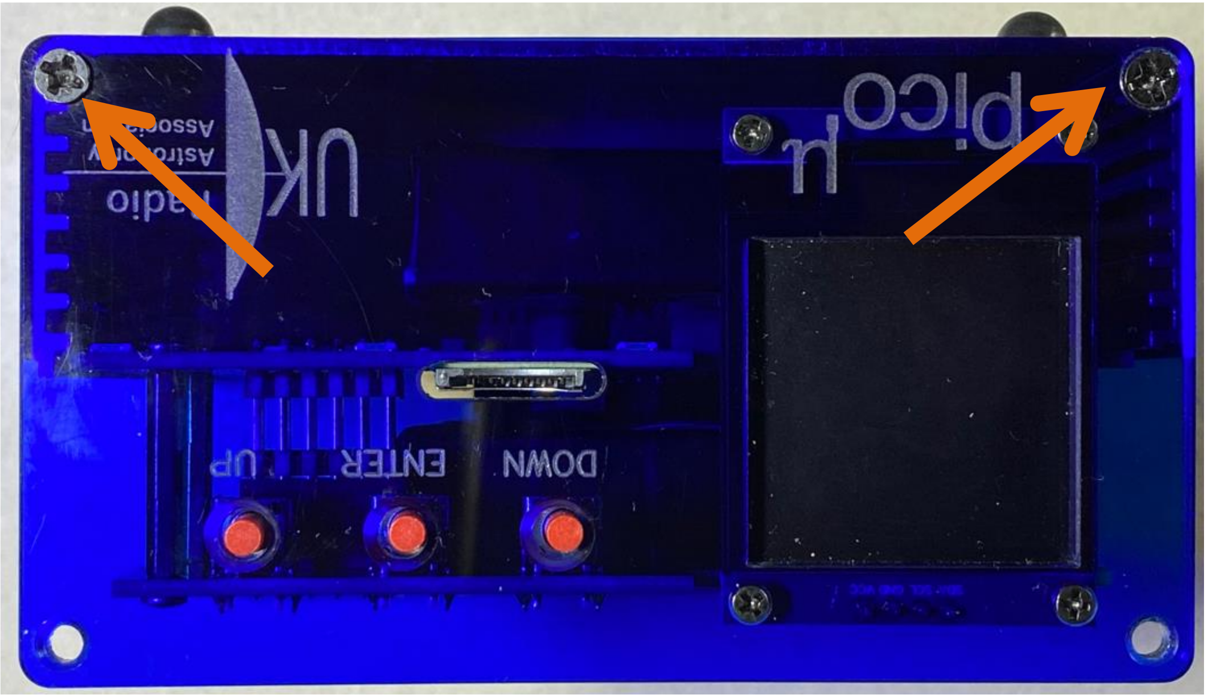
9. The detector should now be fully functioning.

10. Unplug your detector from your desktop PC.

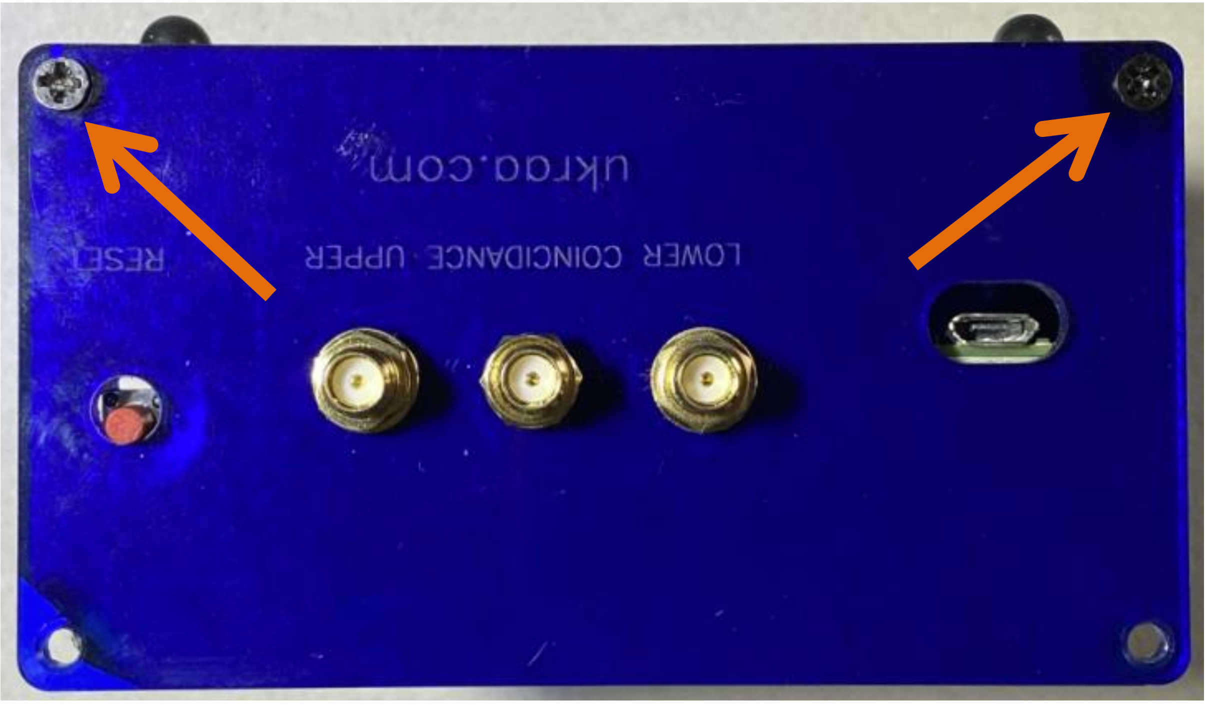
Re-casing your PicoMuon unit

1. Taking the lower case section, reposition it between the front and rear panels of the PicoMuon detector.

2. Replace the lower x2 case screws (nominally PH2 size) to the front panel.



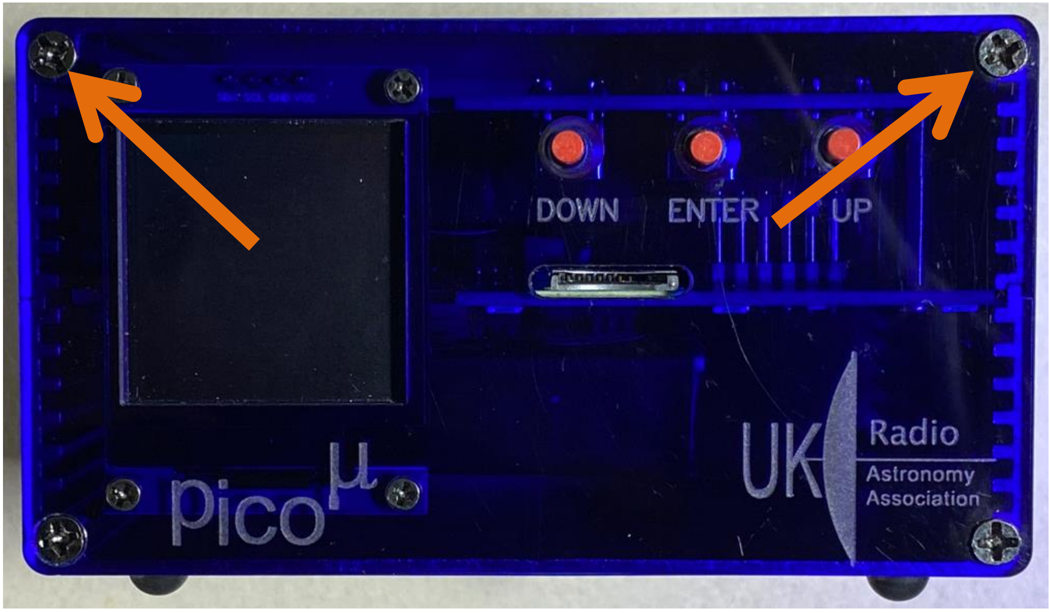
3. Replace the lower x2 case screws (nominally PH2 size) to the rear panel.



4. Rotate the PicoMuon detector.

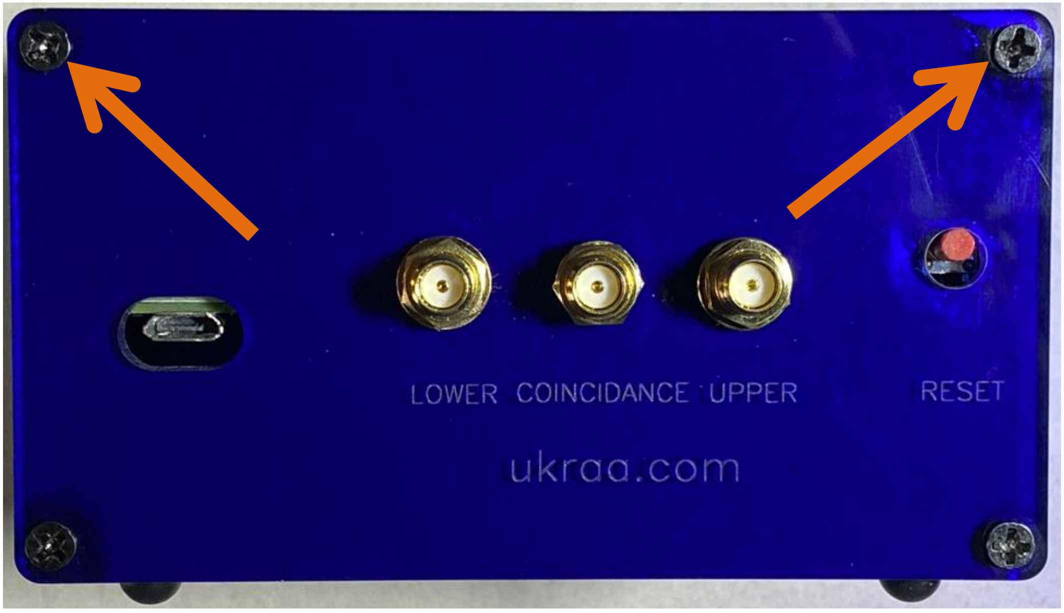
5. Replace the lower case section, reposition it between the front and rear panels of the PicoMuon detector.

6. Replace the upper x2 case screws (nominally PH2 size) to the front panel.



7.

8. Replace the upper x2 case screws (nominally PH2 size) to the rear panel



9. Your detector is now complete – enjoy.

License

Contact us

Please send an e-mail to picomuon@ukraa.com

UK Radio
Astronomy
Association