

Programming Raspberry Pi Pico with Arduino IDE

November 21, 2023

Notes:

- This tutorial is based on the tutorial available at <https://randomnerdtutorials.com/programming-raspberry-pi-pico-w-arduino-ide/#Raspberry-Pi-Pico-Boards-Manager>

1 Introduction

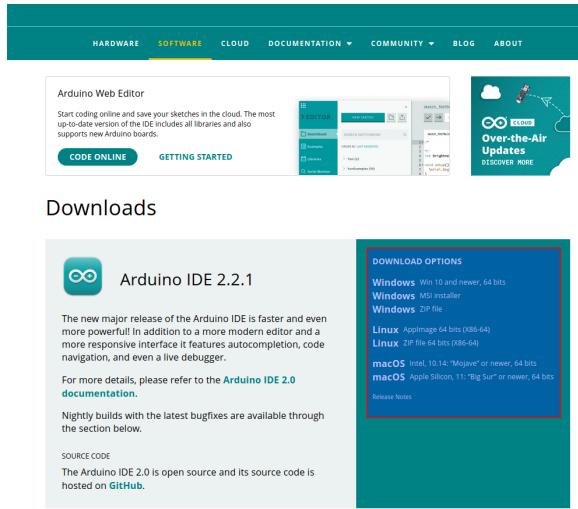
The Raspberry Pi Pico is a low-cost microcontroller board developed around the RP2040 chip by the Raspberry Pi Foundation and it can be programmed using MicroPython or C/C++ like the Arduino.

In this tutorial, you'll learn how to set up Arduino IDE to start programming your Raspberry Pi Pico and Pico W boards with C/C++ programming language. Further, all steps to connect Raspberry Pi Pico to AntiX ARQCP virtual machine are also described.

At the end of this tutorial, messages sent by Raspberry Pi Pico should be printed on a terminal running on AntiX ARQCP virtual machine.

2 Installing Arduino IDE

1. For installing Arduino IDE, first download it from <https://www.arduino.cc/en/software>.

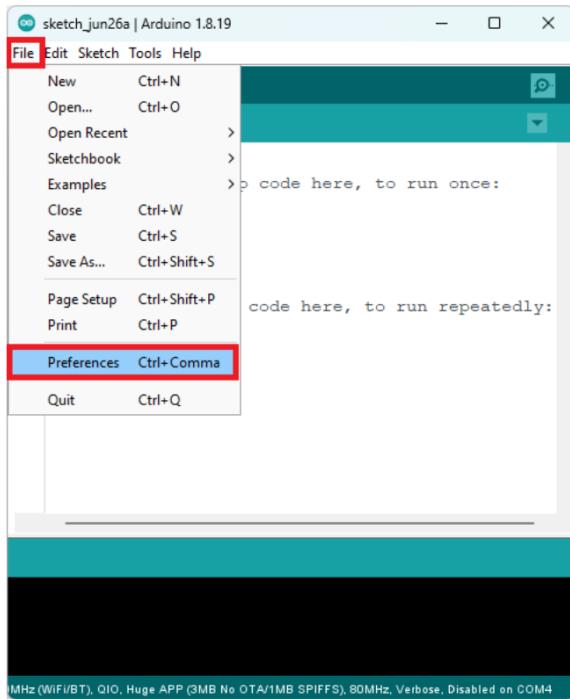


and then, install it.

There are versions for Windows, Linux, and Mac OS (Intel and Apple Silicon).

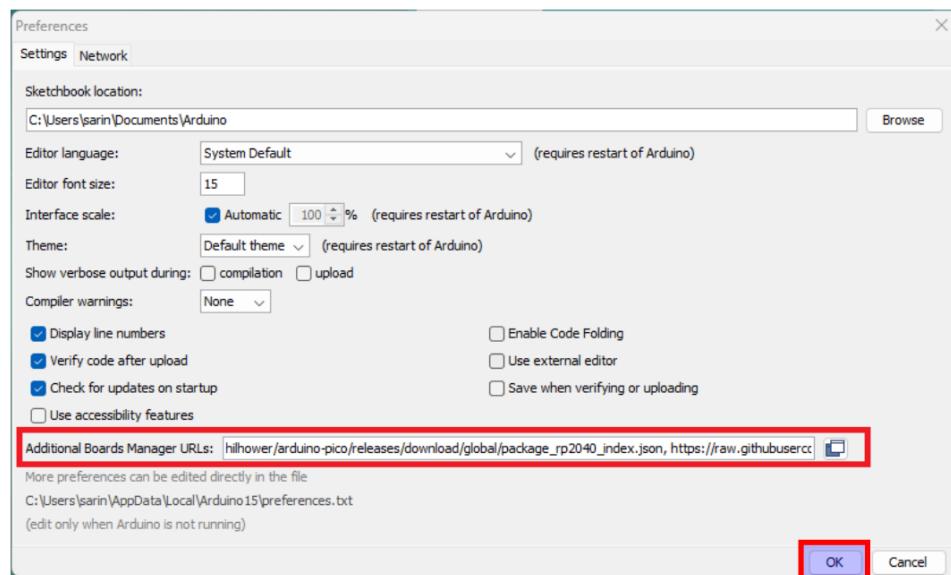
3 Adding the Raspberry Pi Pico to the Boards Manager

1. In the Arduino IDE for Windows, go to `File > Preferences`. In Mac OS, go to `Arduino IDE > Settings`....



2. Enter the following URL into the Additional Boards Manager URLs field:

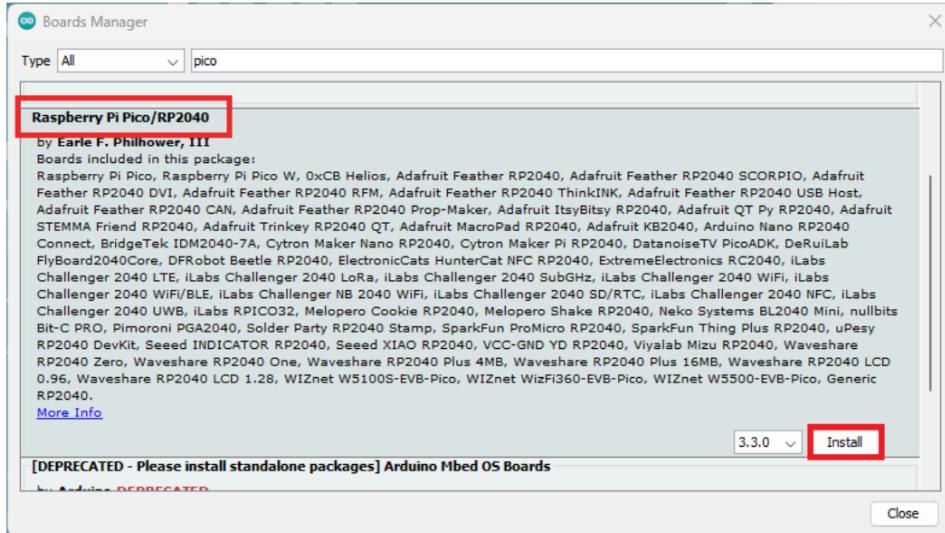
https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json



and then, click on OK.

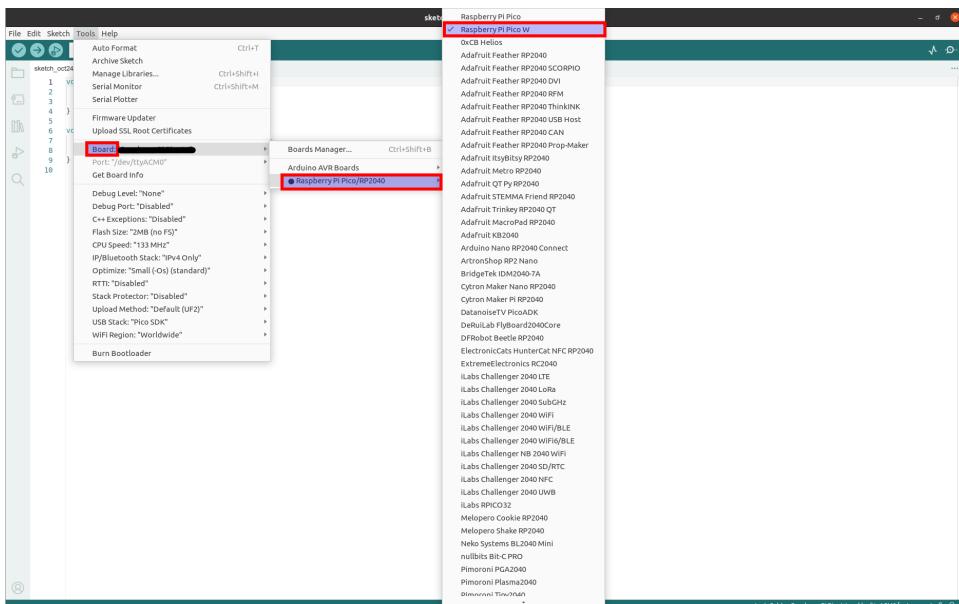
3. Open the Boards Manager. Go to Tools > Board > Boards Manager...

4. Search for pico and install the Raspberry Pi Pico/RP2040 boards.



and then, install it.

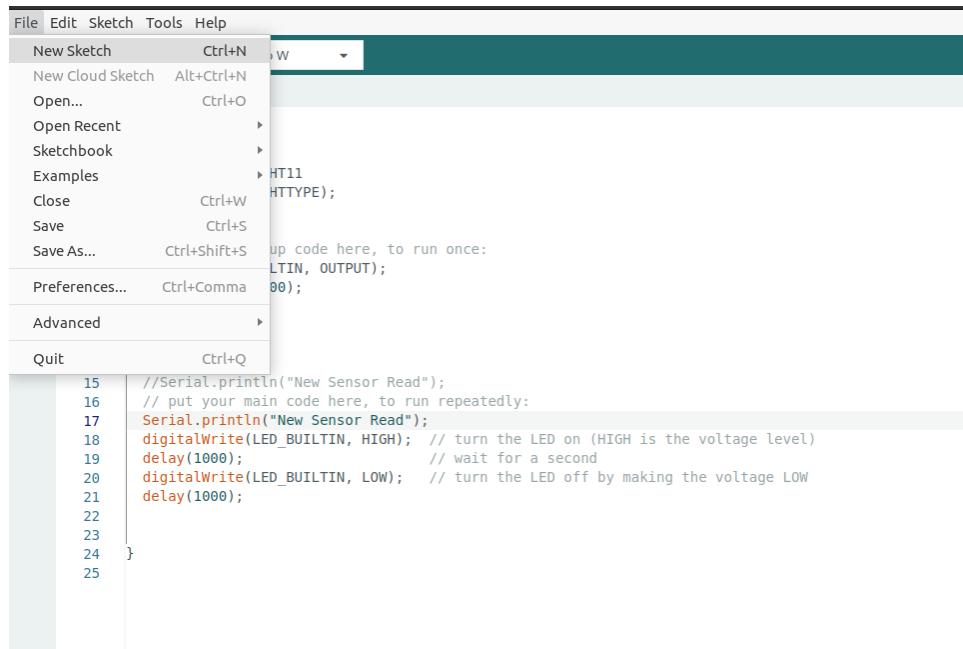
5. Go to Tools > Board > Raspberry Pi Pico/RP2040 and select Raspberry Pi Pico W



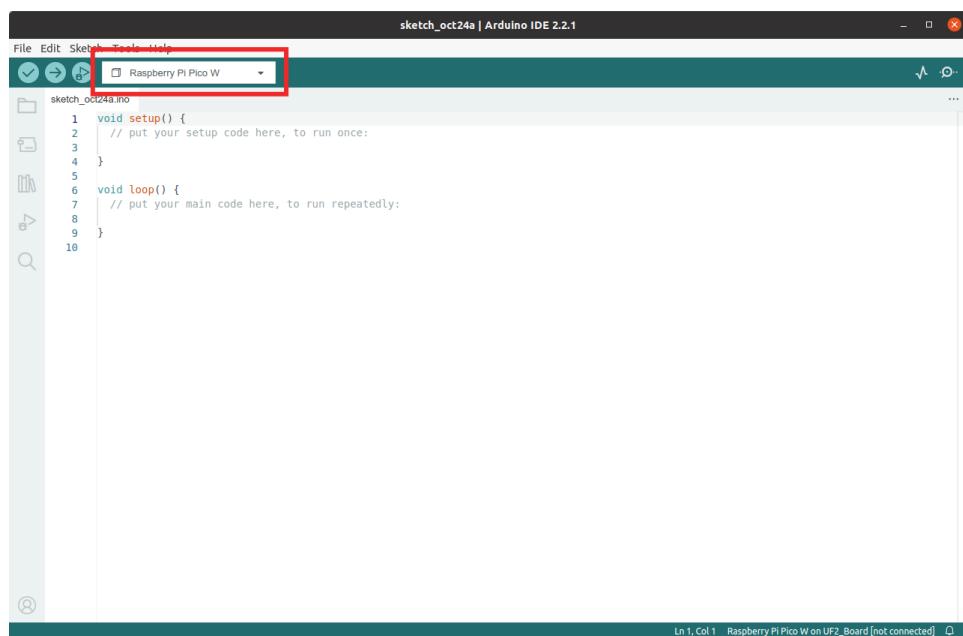
4 Programming

4.1 Create a new Sketch

1. In the Arduino IDE, go File > New Sketch



and you got:



2. Code setup and loop functions as follows:

```
void setup() {
    // put your setup code here, to run once:
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);

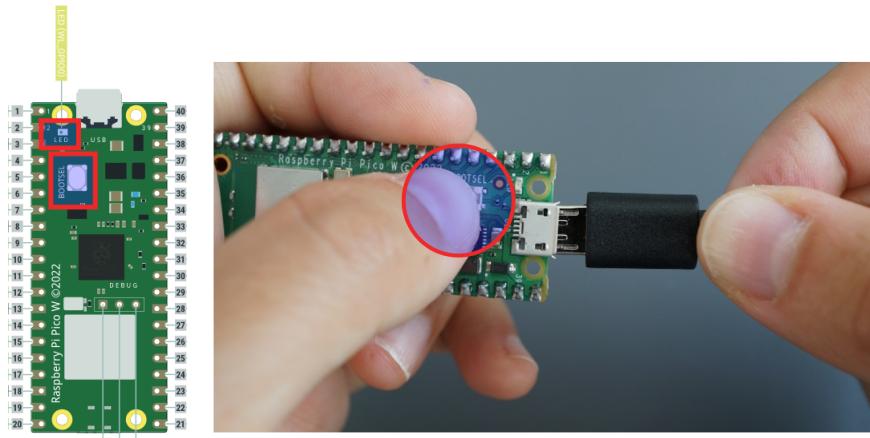
}

void loop() {
    // put your main code here, to run repeatedly:
    // turn the LED on (HIGH is the voltage level)
    digitalWrite(LED_BUILTIN, HIGH);
    // wait for a second
    delay(1000);
    // turn the LED off by making the voltage LOW
    digitalWrite(LED_BUILTIN, LOW);
    // wait for a second
    delay(1000);
}
```

4.2 Connecting the Raspberry Pi Pico in BOOTLOADER mode

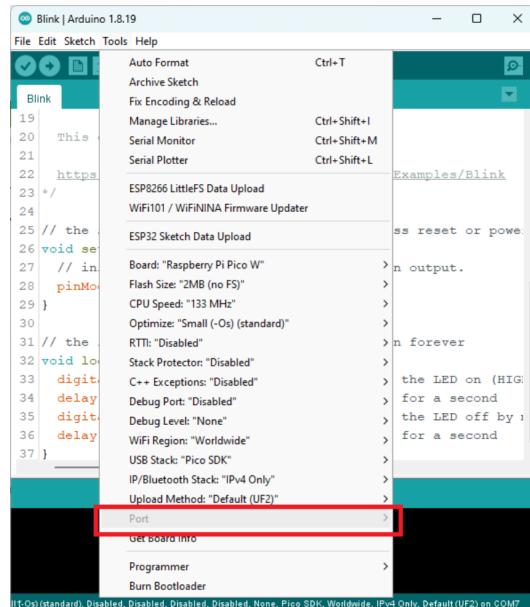
1. Set the BOOTLOADER mode

- For you to be able to upload code to the Raspberry Pi Pico, it needs to be in bootloader mode.
- For that, connect the Raspberry Pi Pico to your computer while **holding the BOOTSEL button at the same time**.

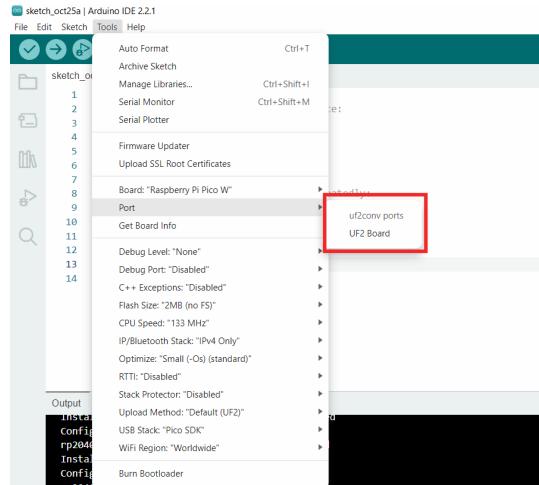


2. Select your COM port in Tools > Port.

- It may be the case that the COM port is grayed out. If that's the case, don't worry. It will automatically find the port once you hit the upload button.

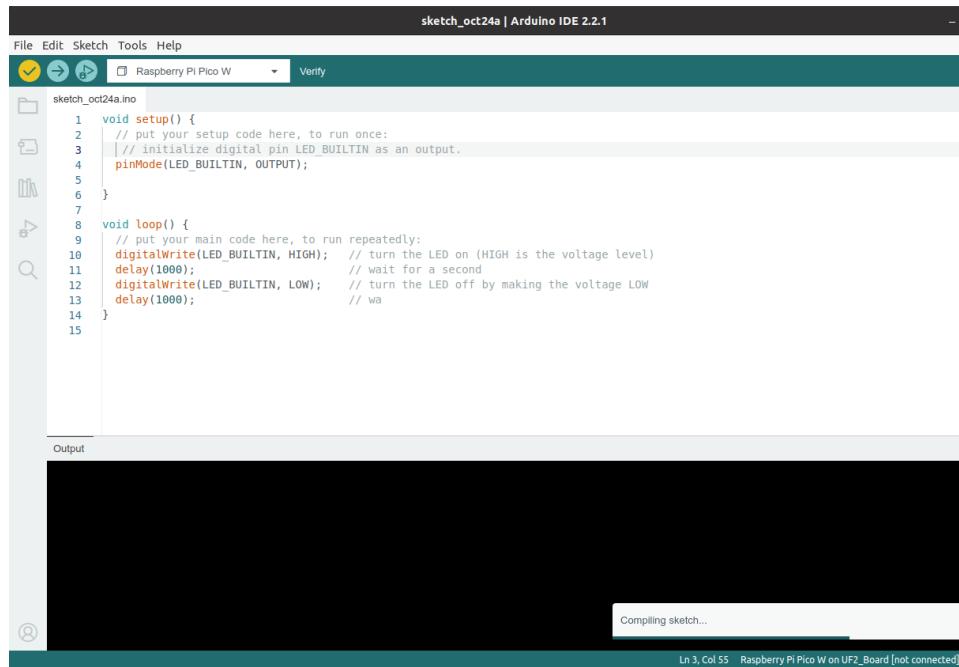


- In Mac OS, select the UF2 port (or /dev/cu.usbmodem14101 if it is already available)



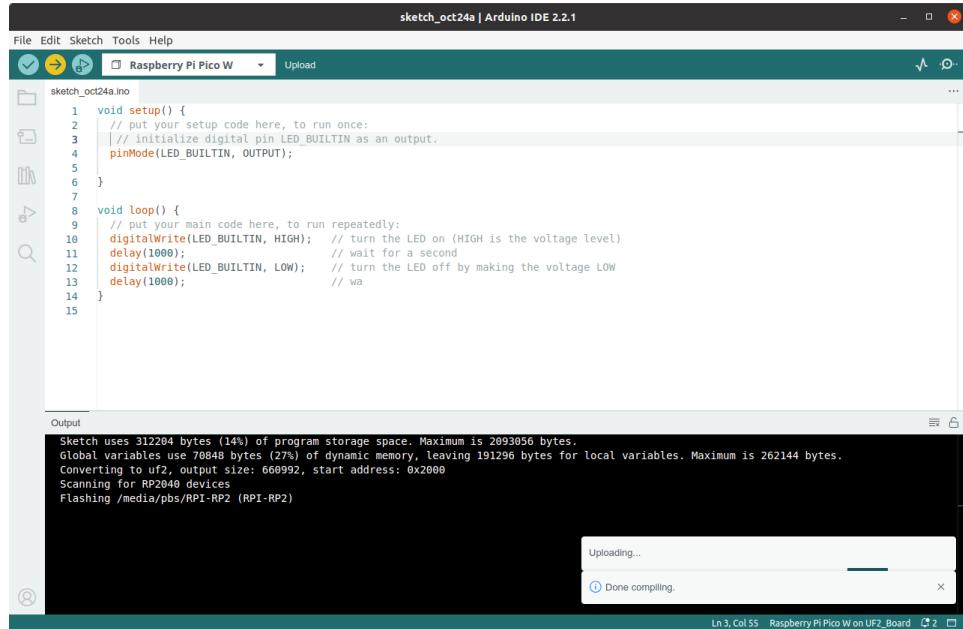
4.3 Compiling and uploading code

1. In the Arduino IDE, select Sketch > Verify/Compile (or using the first toolbar option)



2. In the Arduino IDE, select Sketch > Upload (or using the second toolbar option). In Mac OS, check if /dev/cu.usbmodem14101 port

is selected prior to upload the sketch.



The screenshot shows the Arduino IDE interface. The top menu bar includes File, Edit, Sketch, Tools, Help, and a dropdown for "Raspberry Pi Pico W". Below the menu is a toolbar with icons for Open, Save, Run, Stop, and Upload. The central workspace displays the code for "sketch_oct24a.ino":

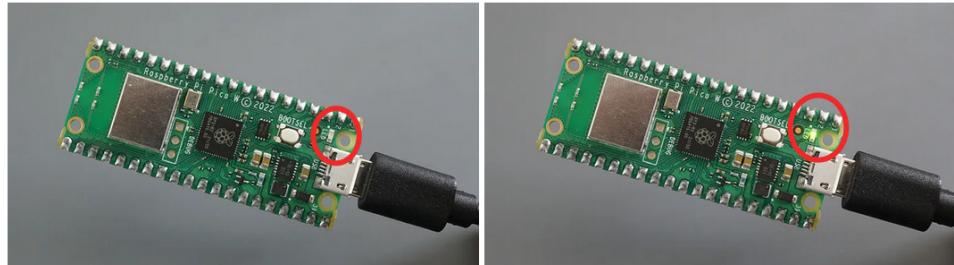
```
sketch_oct24a.ino
1 void setup() {
2     // put your setup code here, to run once:
3     // initialize digital pin LED_BUILTIN as an output.
4     pinMode(LED_BUILTIN, OUTPUT);
5 }
6
7
8 void loop() {
9     // put your main code here, to run repeatedly:
10    digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is the voltage level)
11    delay(1000);                      // wait for a second
12    digitalWrite(LED_BUILTIN, LOW);       // turn the LED off by making the voltage LOW
13    delay(1000);                      // wait
14 }
15
```

The bottom pane, titled "Output", shows the compilation and upload process:

```
Sketch uses 312204 bytes (14%) of program storage space. Maximum is 2093056 bytes.
Global variables use 70848 bytes (27%) of dynamic memory, leaving 191296 bytes for local variables. Maximum is 262144 bytes.
Converting to uf2, output size: 660992, start address: 0x2000
Scanning for RP2040 devices
Flashing /media/pbs/RPI-RP2 (RPI-RP2)
```

An "Uploading..." progress bar is shown at the bottom of the output window, with a status message "Done compiling." next to it.

3. If everything went as expected, the Raspberry Pi Pico onboard LED should be blinking every second.



4.4 Sending message to serial monitor

1. Update the sketch code as follows (red text):

```

void setup() {
    // put your setup code here, to run once:
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
    // opens serial port, sets data rate to 9600 bps
    Serial.begin(9600);

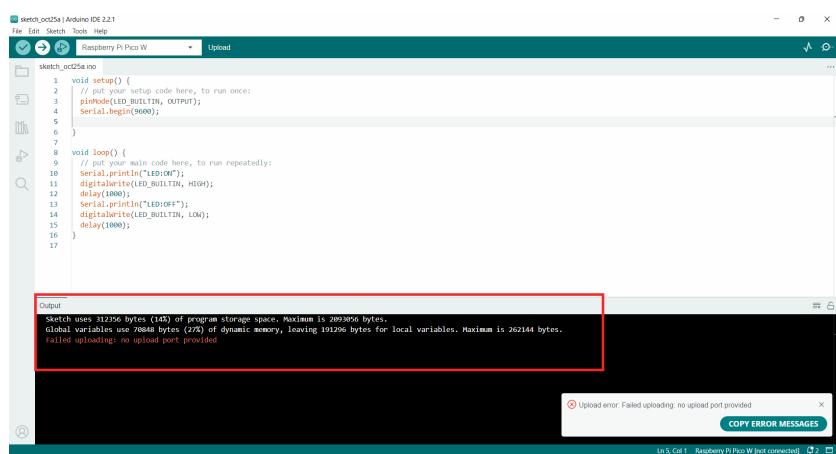
}

void loop() {
    // put your main code here, to run repeatedly:
    Serial.println("LED: ON");
    // turn the LED on (HIGH is the voltage level)
    digitalWrite(LED_BUILTIN, HIGH);
    // wait for a second
    delay(1000);
    Serial.println("LED: OFF");
    // turn the LED off by making the voltage LOW
    digitalWrite(LED_BUILTIN, LOW);
    // wait for a second
    delay(1000);
}

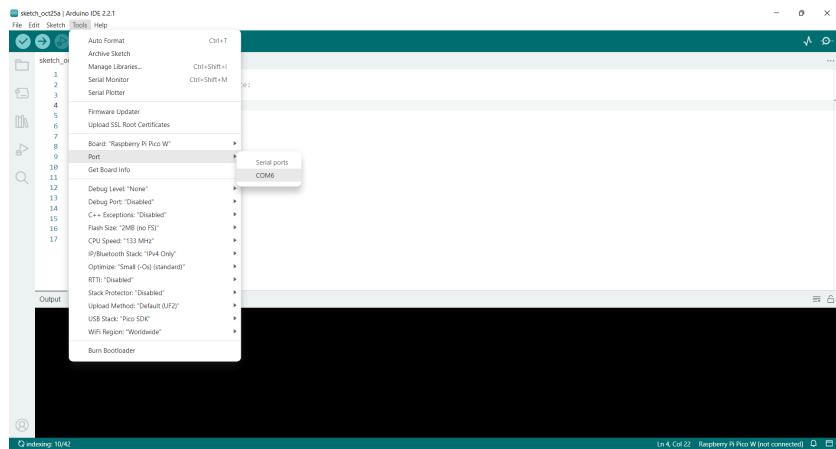
```

2. Compile and upload.

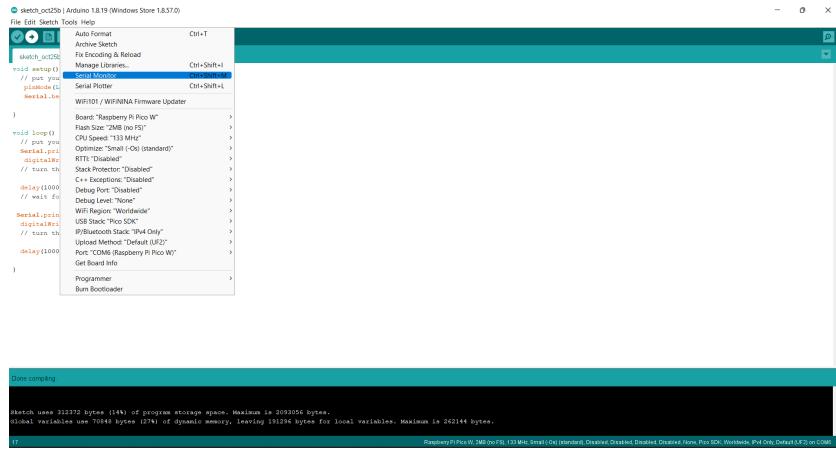
- Windows operating system:
 - If you got an error like this :



- With the board connect to the computer, close the arduino IDE, open it and then check the port by selecting Tools > Port and if it appears (Serial ports) some COM port select it.



- * Note: in this case it has assigned COM6 port, but could be other.
- Upload it again.
- Open the Serial Monitor Tools > Serial Monitor



- And you can check messages on console

```

sketch_oct2sa.ino | Arduino IDE 2.2.1
File Edit Sketch Tools Help
Raspberry Pi Pico W
sketch_oct2sa.ino
1 void setup() {
2     // put your setup code here, to run once:
3     pinMode(LED_BUILTIN, OUTPUT);
4     Serial.begin(9600);
5 }
6
7
8 void loop() {
9     // put your main code here, to run repeatedly:
10    Serial.print("LED:ON");
11    digitalWrite(LED_BUILTIN, HIGH);
12    delay(1000);
13    Serial.print("LED:OFF");
14    digitalWrite(LED_BUILTIN, LOW);
15    delay(1000);
16 }
17

```

Output | Serial Monitor x

Message (Enter to send message to Raspberry Pi Pico W on 'COM0')

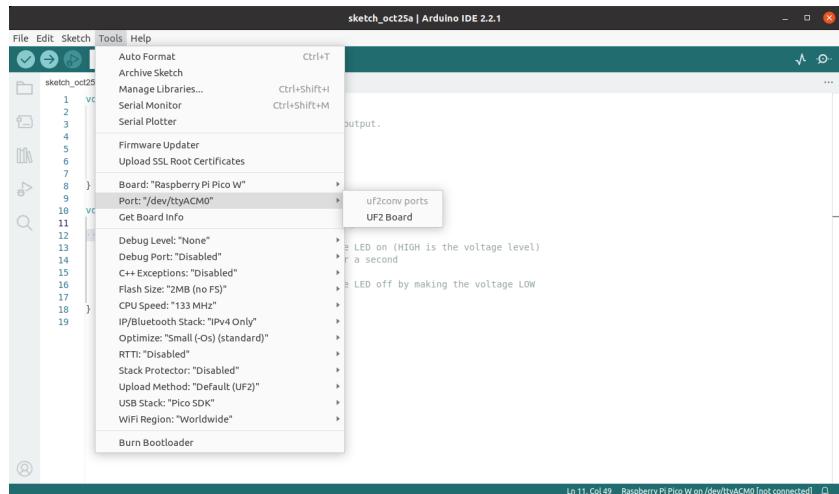
New Line 9600 baud

LED:OFF
LED:ON
LED:OFF
LED:ON
LED:OFF
LED:ON
LED:OFF
LED:ON
LED:OFF
LED:ON

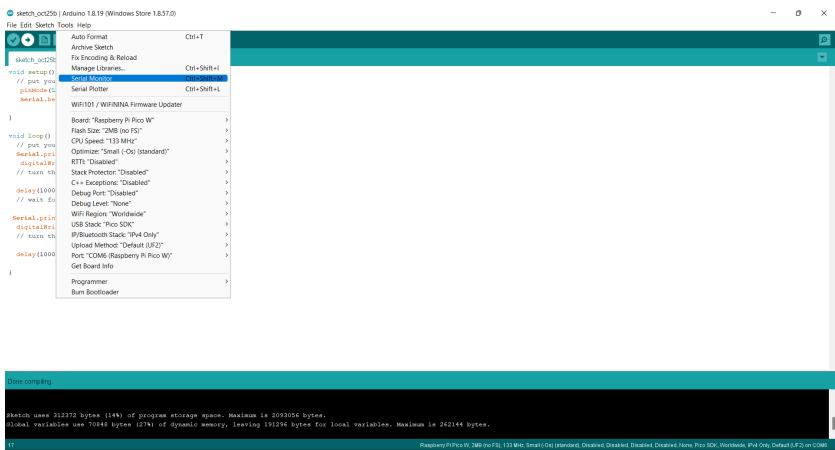
Line 4, Col 22 Raspberry Pi Pico W on COM0

• Ubuntu operating system

- If you get an error, close the Arduino IDE and re-open with device connected.
- When a port is assigned you must get something like this:



- Upload it again.
- Open the Serial Monitor Tools > Serial Monitor



- And you, probably, will get an error like this:



- Open a terminal and type: `sudo chmod a+rwx /dev/ttyACM0`

```

/home/pbs/.zshrc:131: unmatched "
→ ~ sudo chmod a+r /dev/ttyACM0
[sudo] password for pbs:
→ ~

```

- Close the serial monitor console and reopen it.

File Edit Sketch Tools Help

sketch_oct25a.ino

```

1 void setup() {
2     // put your setup code here, to run once:
3     // initialize digital pin LED_BUILTIN as an output.
4     pinMode(LED_BUILTIN, OUTPUT);
5
6     Serial.begin(9600);
7
8 }
9
10 void loop() {
11     // put your main code here, to run repeatedly:
12     Serial.println("LED: ON");
13     digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
14     delay(1000);                      // wait for a second
15     Serial.println("LED: OFF");
16     digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
17     delay(1000);                      // wait
18 }
19

```

Output Serial Monitor X

Message (Enter to send message to 'Raspberry Pi Pico W' on '/dev/ttyACM0')

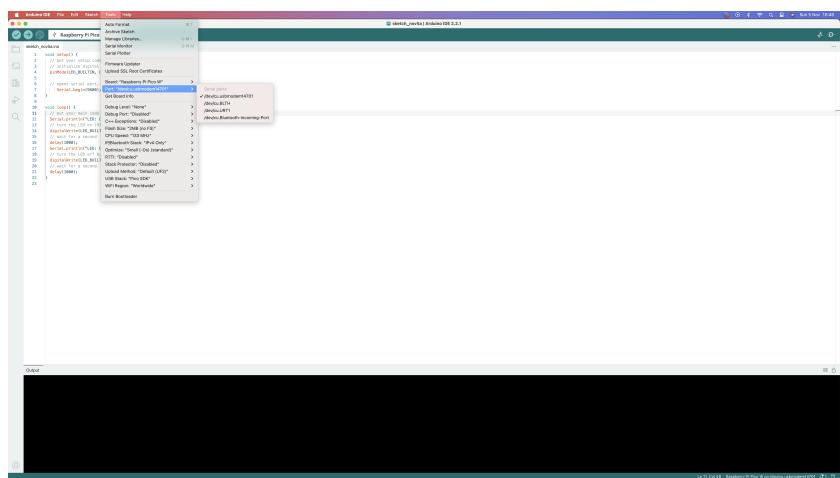
LED: ON
LED: OFF
LED: ON
LED: OFF
LED: ON
LED: OFF
LED: ON
LED: OFF

New Line 9600 baud

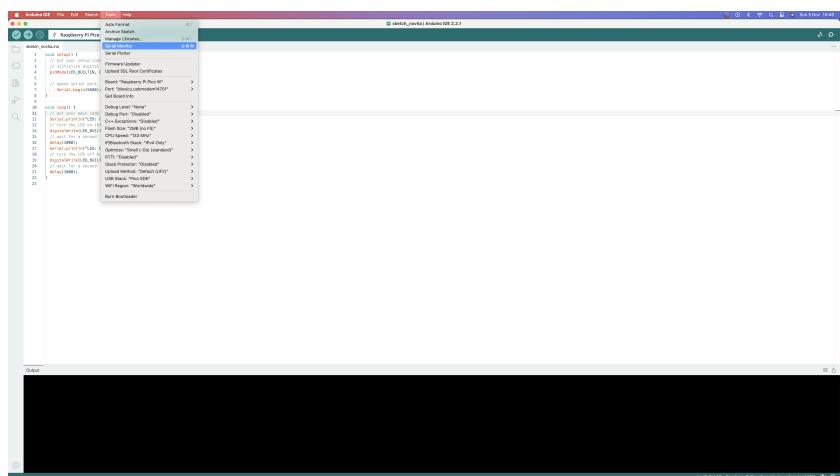
Ln 11, Col 49 Raspberry Pi Pico W on /dev/ttyACM0

- Mac OS

- If you get an error, close the Arduino IDE and reopen it with the Pico device connected
- When a port is assigned you must get something like this:



- Upload it again
- Open the Serial Monitor Tools > Serial Monitor

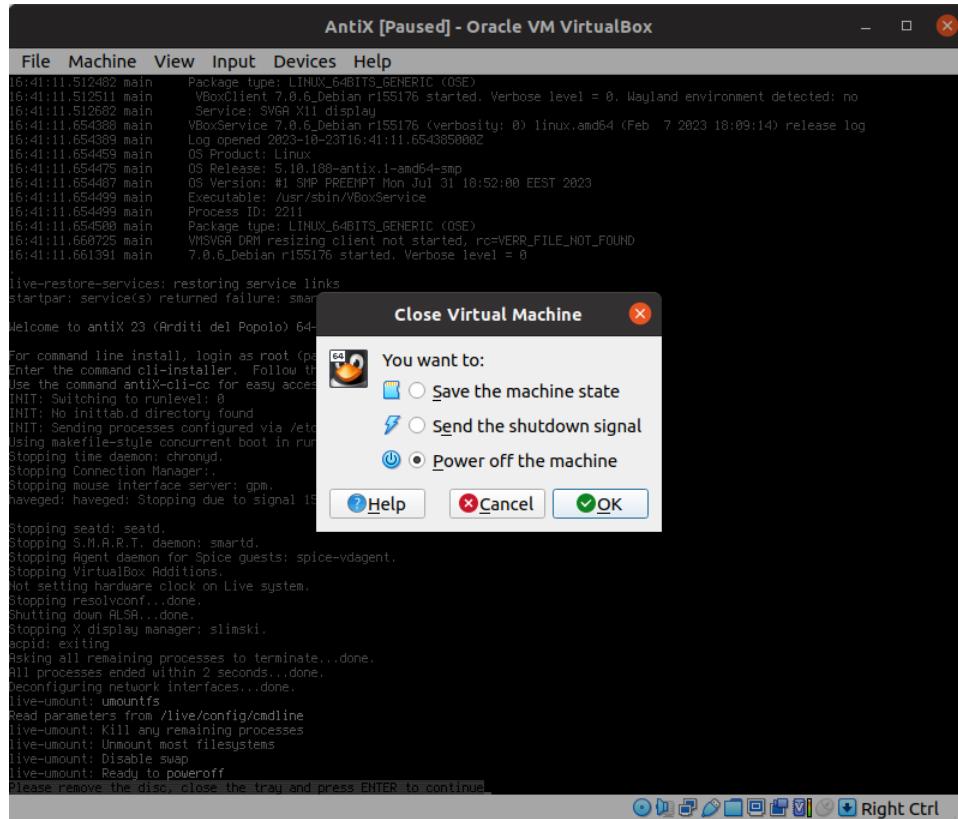


- And you can check messages on console

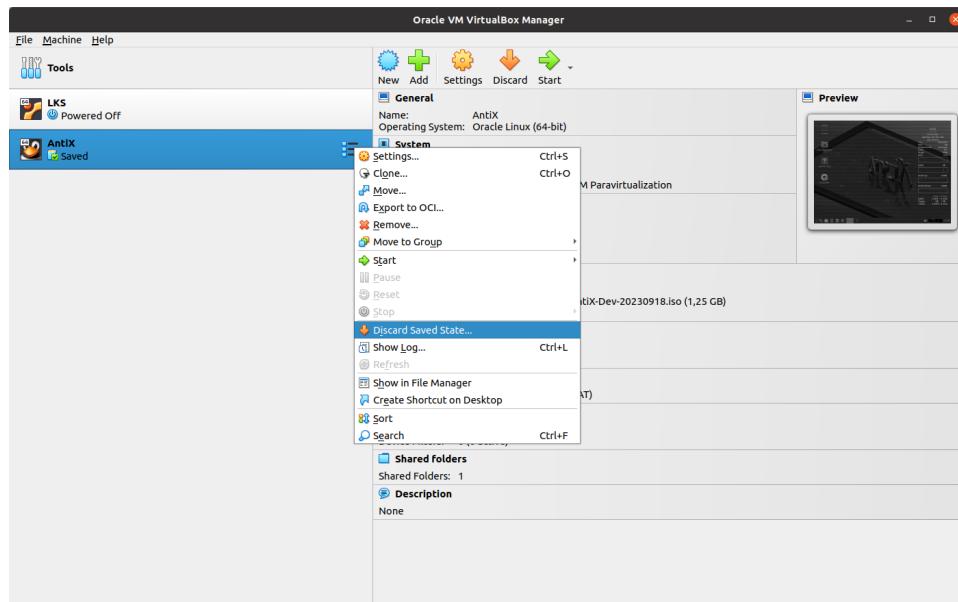
5 Connecting to AntiX (ARQCP Virtual Machine)

5.1 Windows operating system

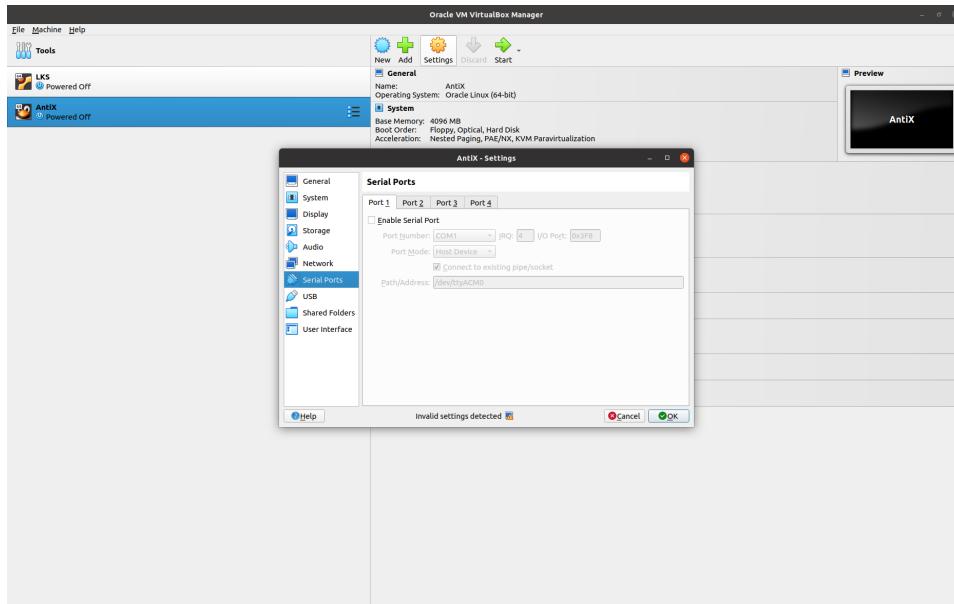
1. Power off the ARQCP Virtual machine (AntiX), if it is running



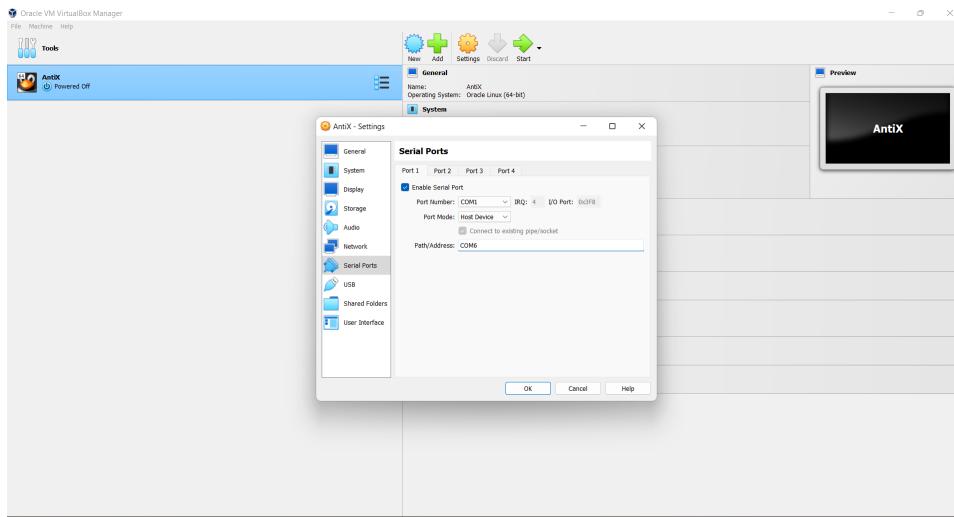
or in the Oracle VM Virtual Box Manager, right-click on AntiX, select Discard Saved State



2. In the Oracle VM Virtual Box Manager, select the AntiX, then click on Settings and select Serial Ports.



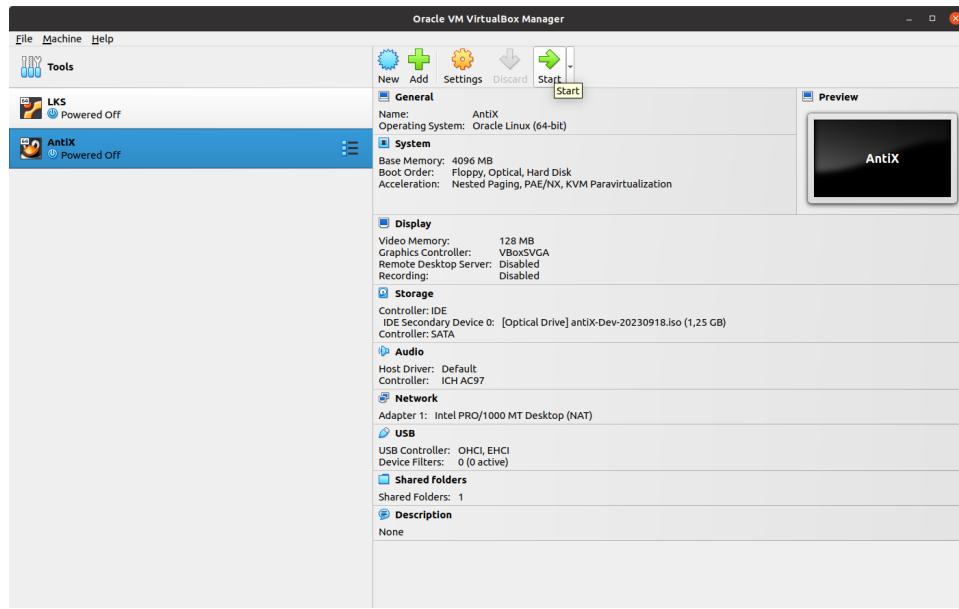
3. Enable Serial Port by checking it and fill in the form as follows:



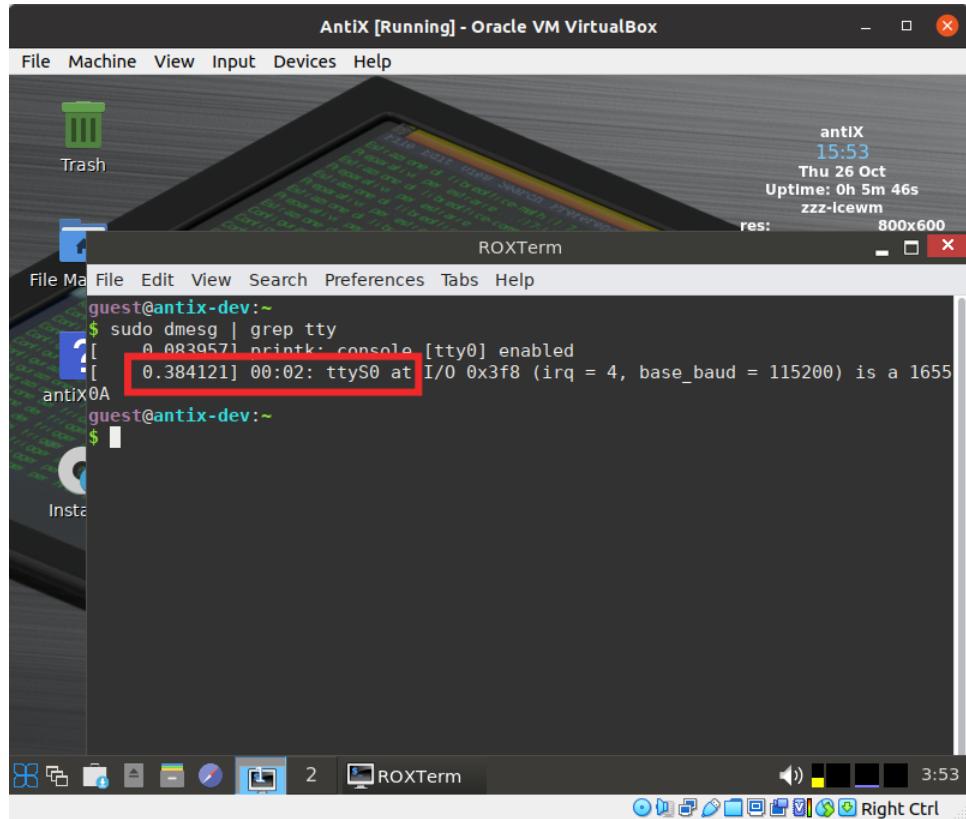
- **Note:** Here we are filling in the Path/Address with COM6, because it was the previously assigned port. Therefore, if your

system has assigned a different COM port you must fill in with such port.

4. Start the AntiX machine



5. Open a console (probably, you have to configure again the keyboard for Portuguese) and type sudo dmesg | grep tty (the password is: guest)



In this case it assigns the `ttyS0` port.

6. From the terminal type `cat /dev/ttyS0` and you get the messages from raspberry Pi Pico device.

AntiX [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

ROXTerm

File Edit View Search Preferences Tabs Help

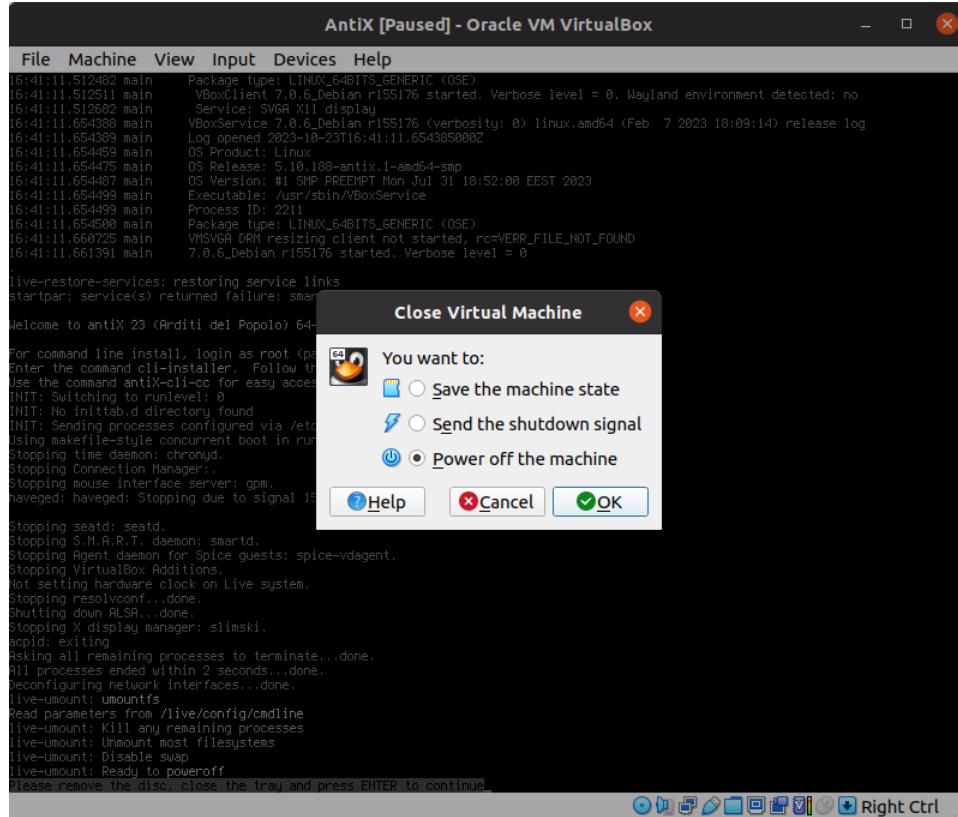
```
guest@antix-dev:~$ sudo dmesg | grep tty
[    0.083957] printk: console [tty0] enabled
[    0.384121] 00:02: ttyS0 at I/O 0x3f8 (irq = 4, base_baud = 115200) is a 1655
0A
guest@antix-dev:~$ cat /dev/ttyS0
LED: ON

LED: OFF

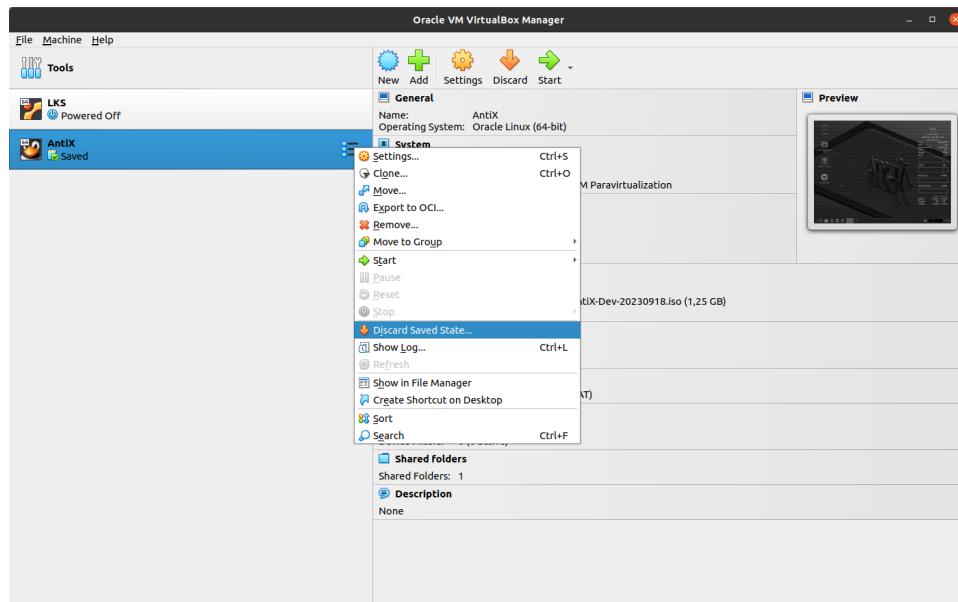
LED: ON
```

5.2 Ubuntu operating system

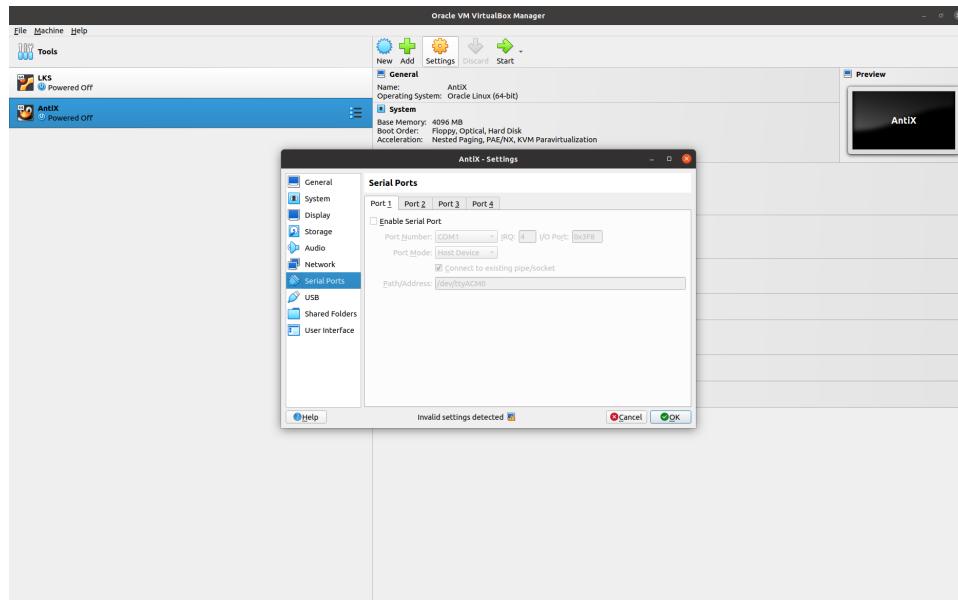
- ## 1. Power off the ARQCP Virtual machine (AntiX)



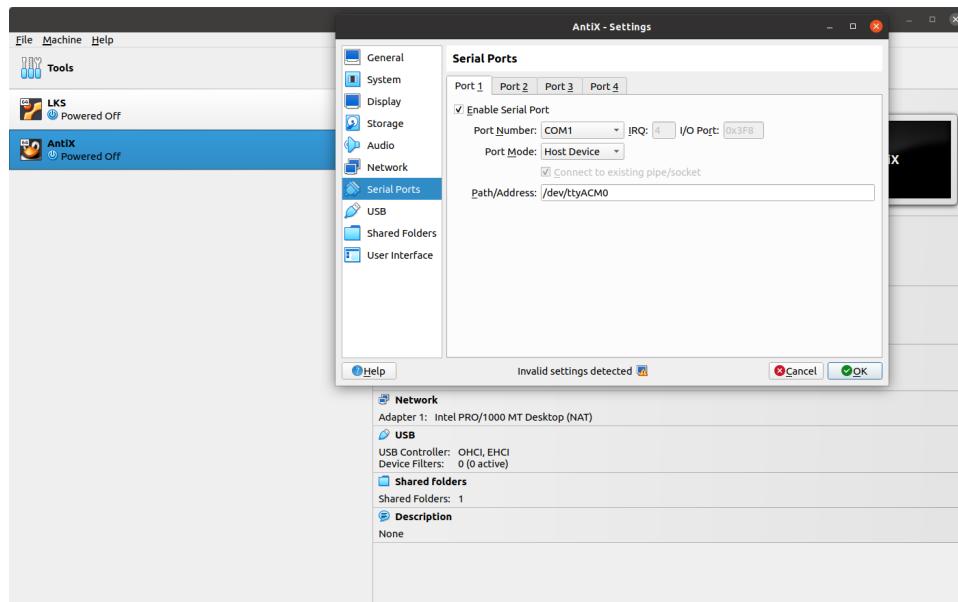
or in the Oracle VM Virtual Box Manager, right-click on AntiX, select Discard Saved State



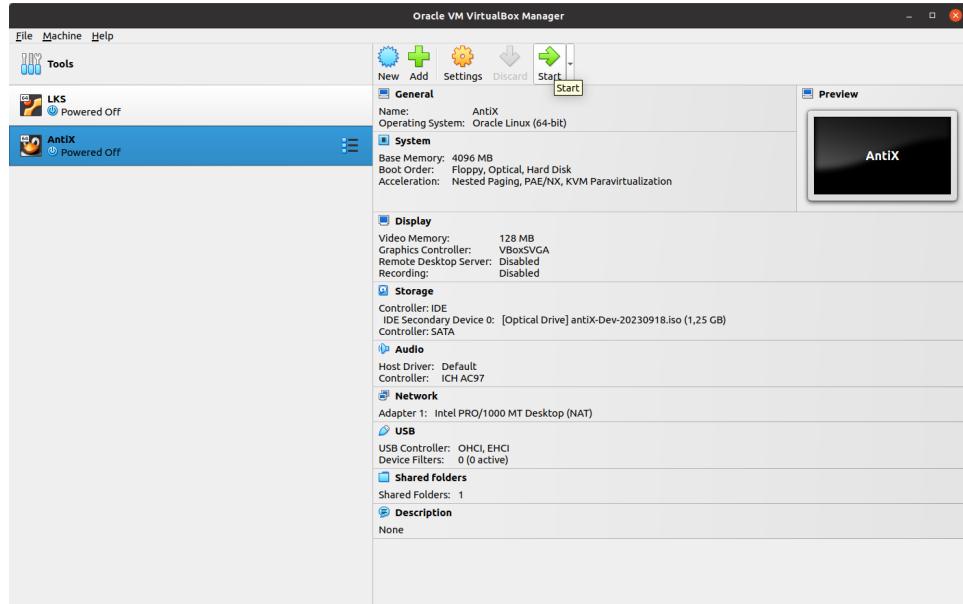
2. In the Oracle VM Virtual Box Manager, select the Antix, then click on Settings and select Serial Ports.



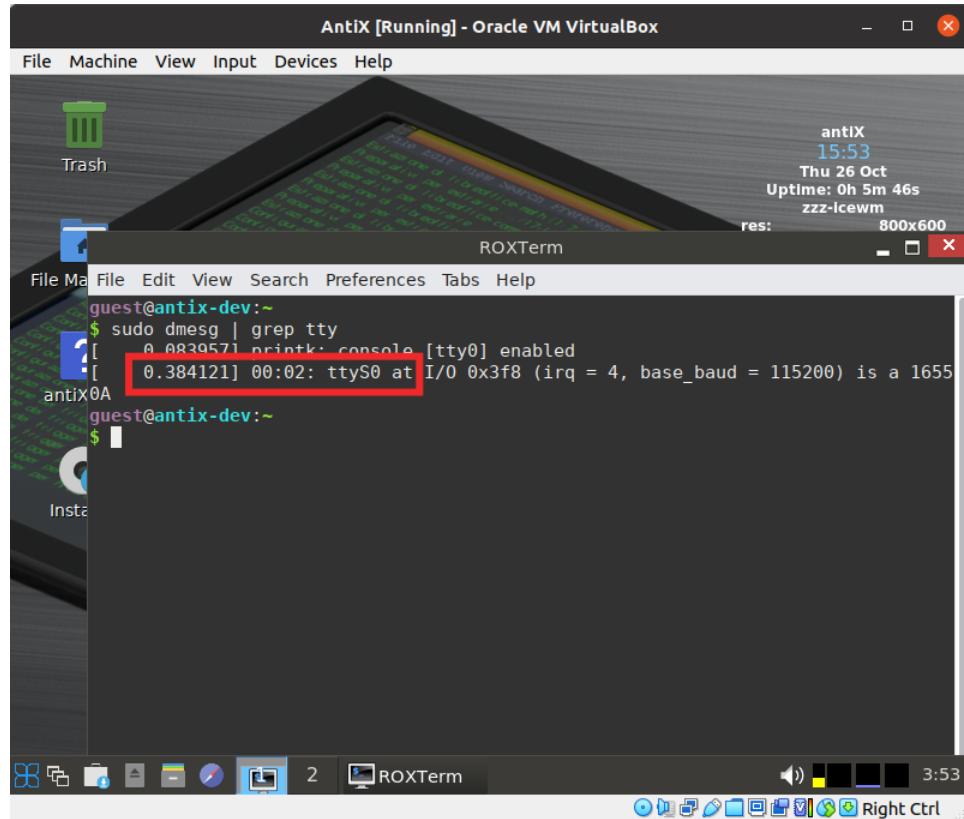
3. Enable Serial Port by checking it and fill in the form as follows:



4. Start the AntiX machine

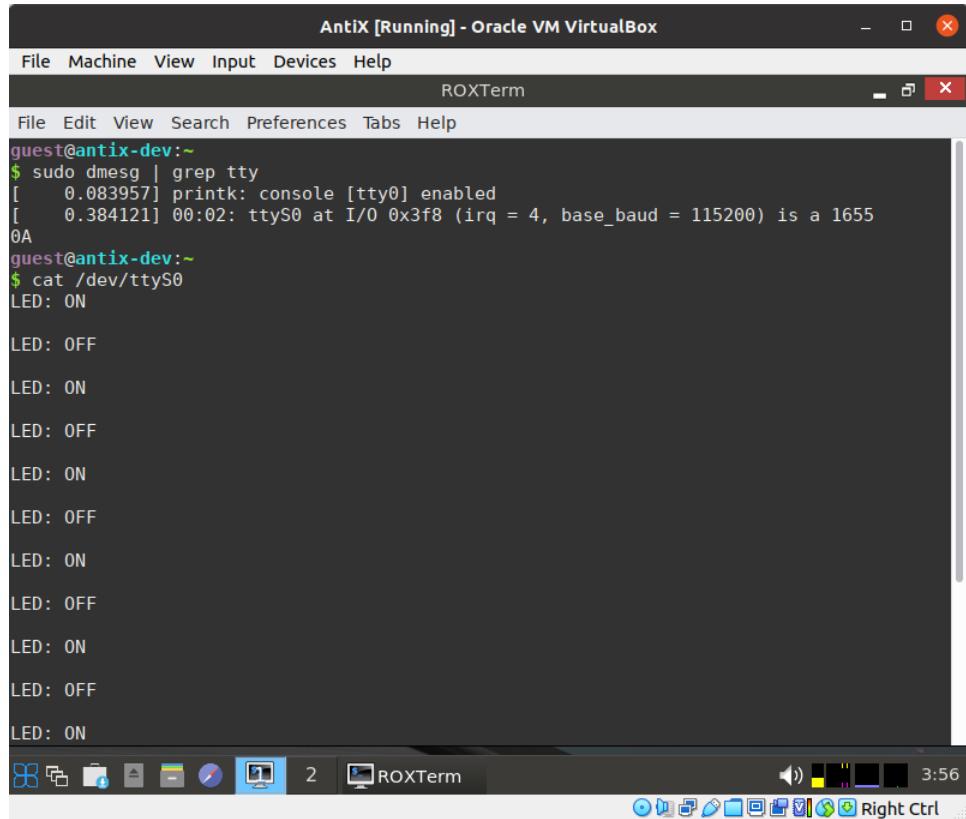


5. Open a console (probably, you have to configure gain the keyboard for Portuguese) and type sudo dmesg | grep tty (the password is: guest)



In this case it assigns the `ttyS0` port.

6. From the terminal type `cat /dev/ttyS0` and you get the messages from raspberry Pi Pico device.



The screenshot shows a terminal window titled "ROXTerm" running on an Antix Linux distribution. The window title bar says "AntiX [Running] - Oracle VM VirtualBox". The terminal window has its own menu bar with "File", "Edit", "View", "Search", "Preferences", "Tabs", and "Help". The main area of the terminal displays the following command-line session:

```
guest@antix-dev:~$ sudo dmesg | grep tty
[    0.083957] printk: console [tty0] enabled
[    0.384121] 00:02: ttyS0 at I/O 0x3f8 (irq = 4, base_baud = 115200) is a 1655
0A
guest@antix-dev:~$ cat /dev/ttyS0
LED: ON

LED: OFF

LED: ON

LED: OFF

LED: ON

LED: OFF

LED: ON

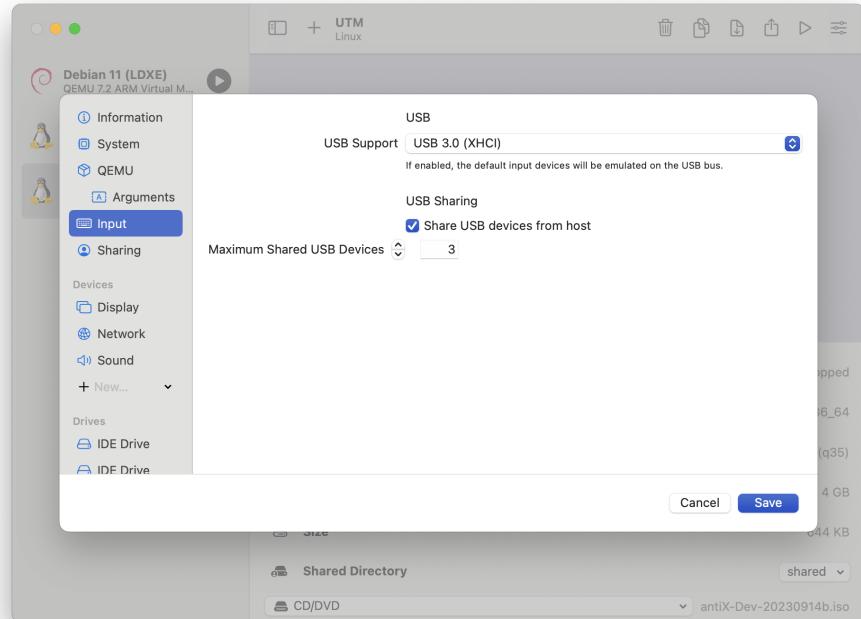
LED: OFF

LED: ON
```

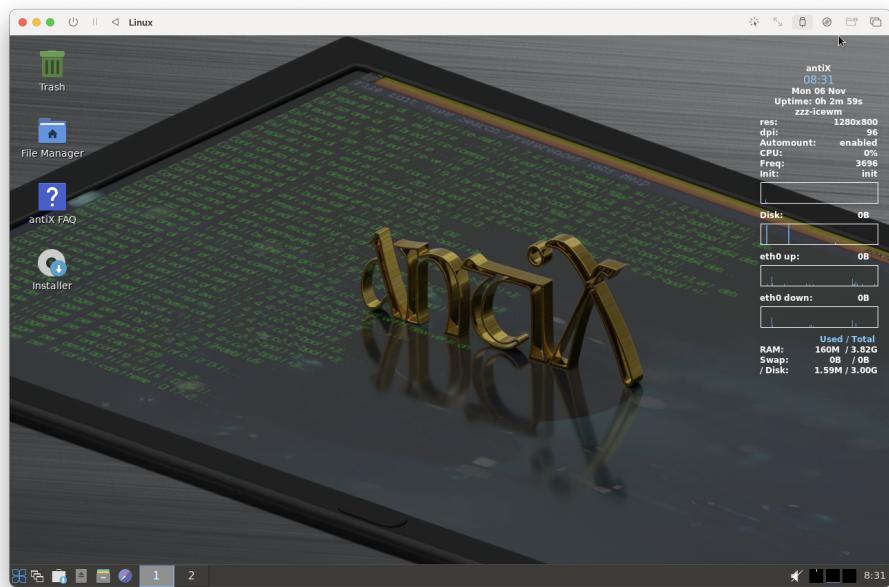
The terminal window is part of a desktop environment, as evidenced by the window manager interface at the bottom, which includes icons for file operations like "New", "Open", "Save", and "Print", a taskbar with two tabs ("ROXTerm" and another tab), and a system tray with icons for battery, signal strength, and system status.

5.3 UTM on Mac OS

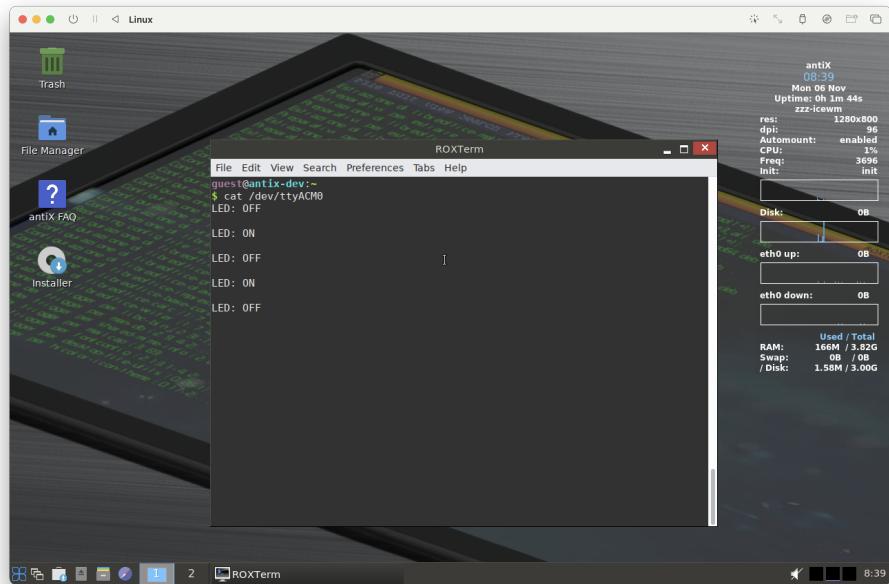
1. Power off the ARQCP Virtual machine (Antix)
2. In UTM, in the virtual machine configuration, make sure that “USB sharing” is enabled.



3. Once the virtual machine is started, plug in the device and you should see a prompt to attach the device
4. You can also press the USB button on the toolbar to see a list of devices which you can attach or detach



5. From the terminal type `cat /dev/ttyACM0` and you get the messages from the Raspberry Pi Pico device



6. If you have any error along these steps, try to quit UTM and reopen it