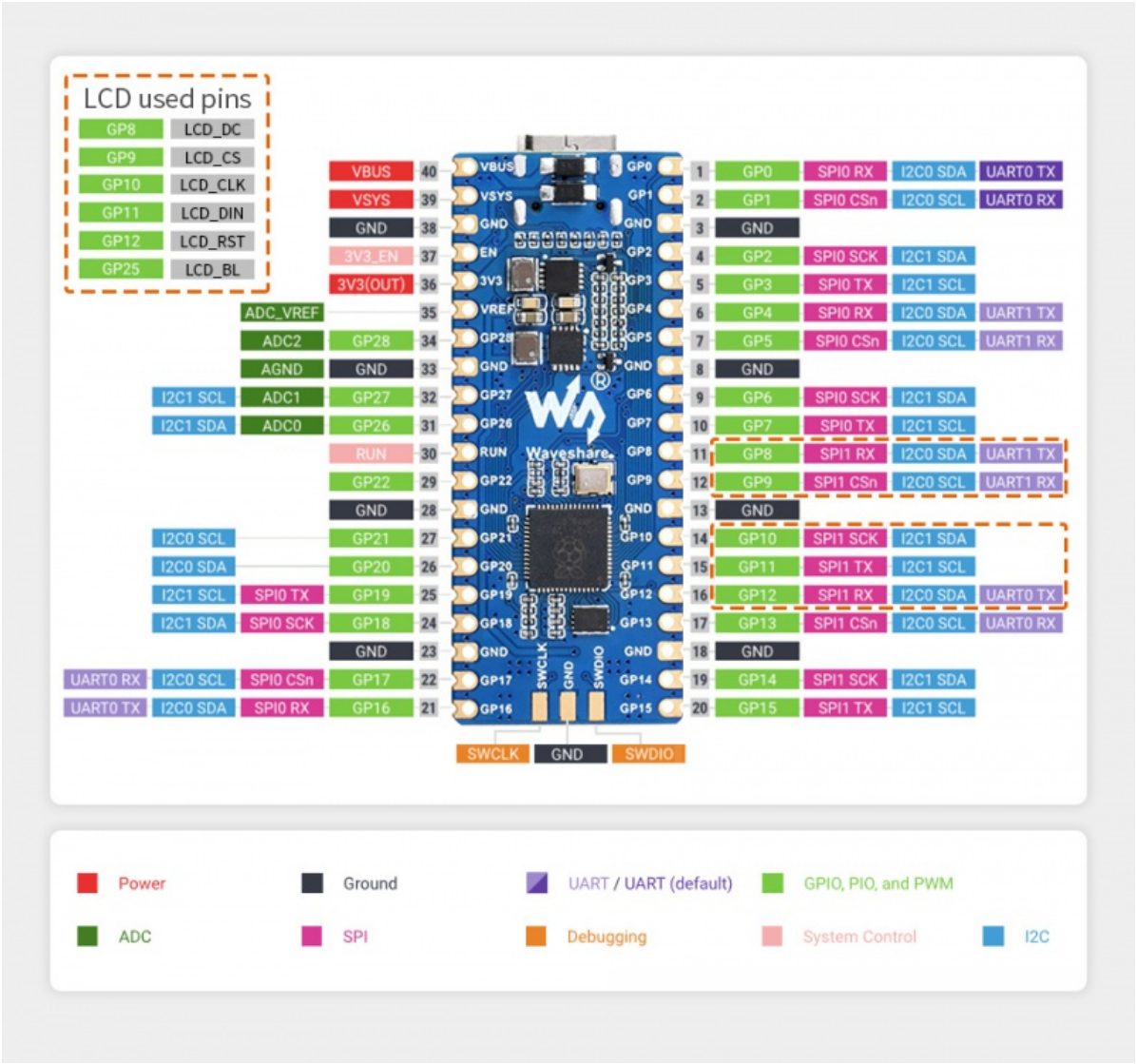
Overview

RP2040-LCD-0.96 is a low-cost, high-performance Pico-like MCU board with flexible digital interfaces. It incorporates Raspberry Pi's RP2040 microcontroller chip, as same as the one on Raspberry Pi Pico. For software development, either Raspberry Pi's C/C++ SDK, or the MicroPython is available, which makes it easy for you to get started, and integrate it into end products quickly.
In addition, there're also an onboard 0.96inch IPS display, Lithium battery recharge/discharge header, and high-efficiency DC-DC buck-boost chip.

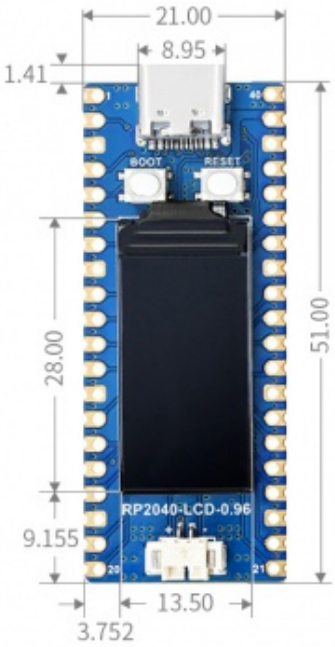
Features

- RP2040 microcontroller chip designed by Raspberry Pi in the United Kingdom.
- Dual-core Arm Cortex M0+ processor, a flexible clock running up to 133 MHz.
- 264KB of SRAM, and 2MB of onboard Flash memory.
- USB-C connector, keeps it up to date, easier to use.
- 0.96-inch 160×80 pixels 65K colorful IPS LCD display.
- Lithium battery recharge/discharge header, suitable for mobile devices.
- Onboard DC-DC chip TPS63000, high-efficiency DC-DC buck-boost chip, 1.8A current switch.
- Castellated module allows soldering directly to carrier boards (there should be a dedicated cut-out for embedding the bottom components).
- USB 1.1 with device and host support.
- Low-power sleep and dormant modes.
- Drag-and-drop programming using mass storage over USB.
- 26 × multi-function GPIO pins.
- 2 × SPI, 2 × I2C, 2 × UART, 3 × 12-bit ADC, 16 × controllable PWM channels.
- Accurate clock and timer on-chip.
- Temperature sensor.
- Accelerated floating-point libraries on-chip.
- 8 × Programmable I/O (PIO) state machines for custom peripheral support.

Pinout



Dimension



Notice

RP2040-LCD-0.96 uses the same RP2040 chip as the Raspberry Pi Pico, and it is compatible with the Raspberry Pi Pico, in this case, most of the accessories and codes can be used with the RP2040-LCD-0.96 as well.

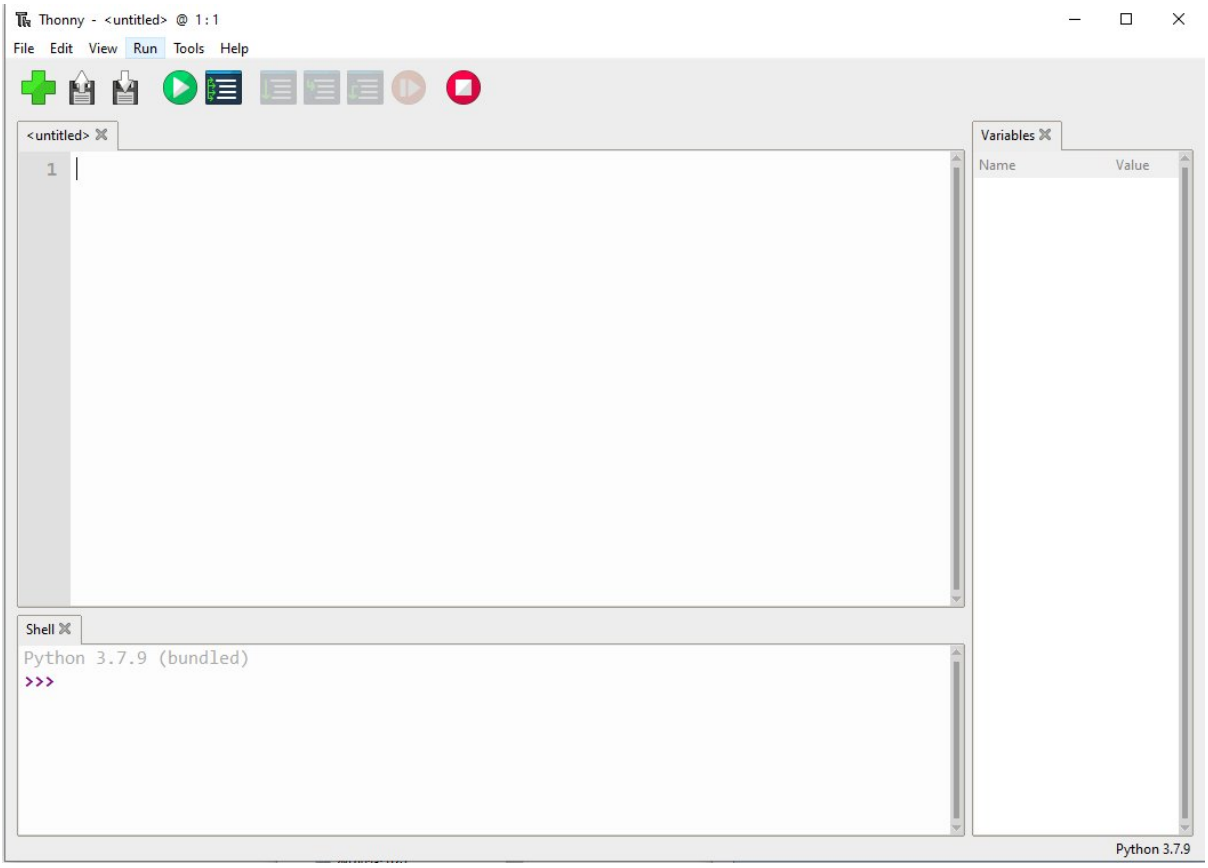
Software Setup

Please follow the guides of Raspberry Pi to install and set up Pico for the Pico.

- micro python guide of Pico
- C SDK guide of Pico

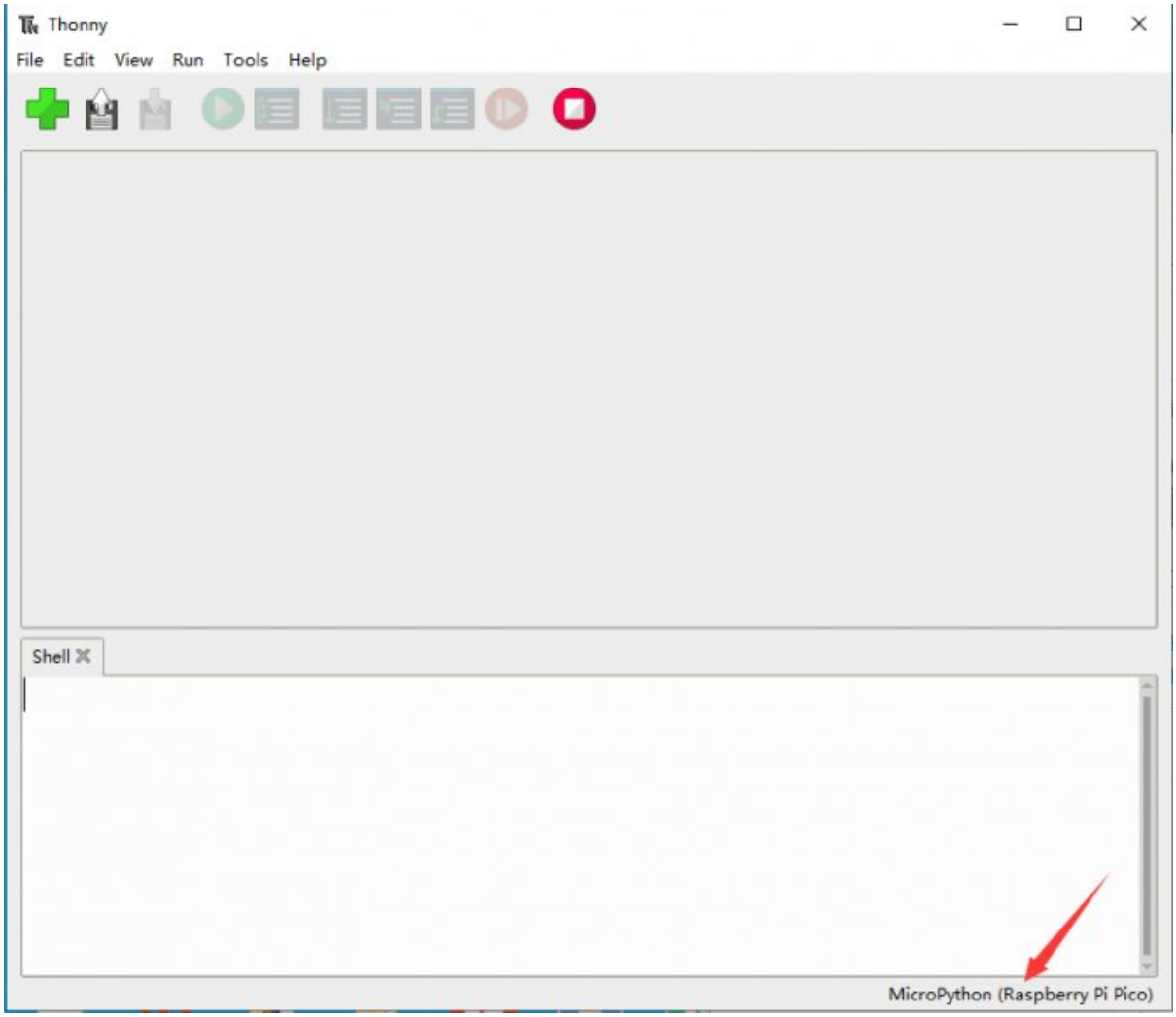
For easy use, we recommend you use the Thonny tool.

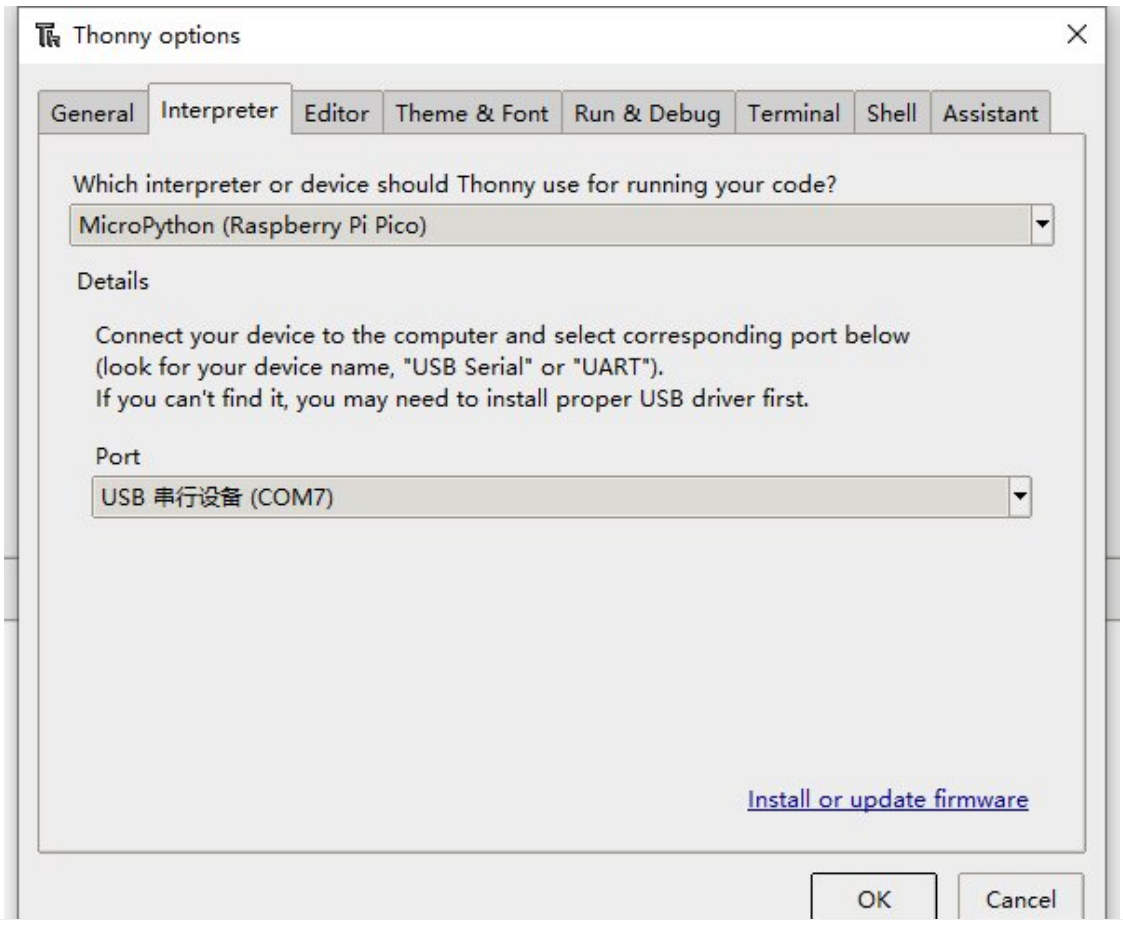
- Thonny website
- Please set the Thonny development environment to be RaspberryPi when setting.

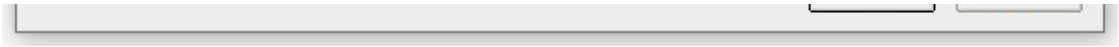


- Configure the Micrpython environment and select the Pico port.
    - First, connect the Raspberry Pi Pico to your computer, left click on the Configure environment option in the lower right corner of the Thonny --> Select Configure interpreter.
    - In the pop-up window, select MicroPython (Raspberry Pi Pico), and select

the corresponding port.

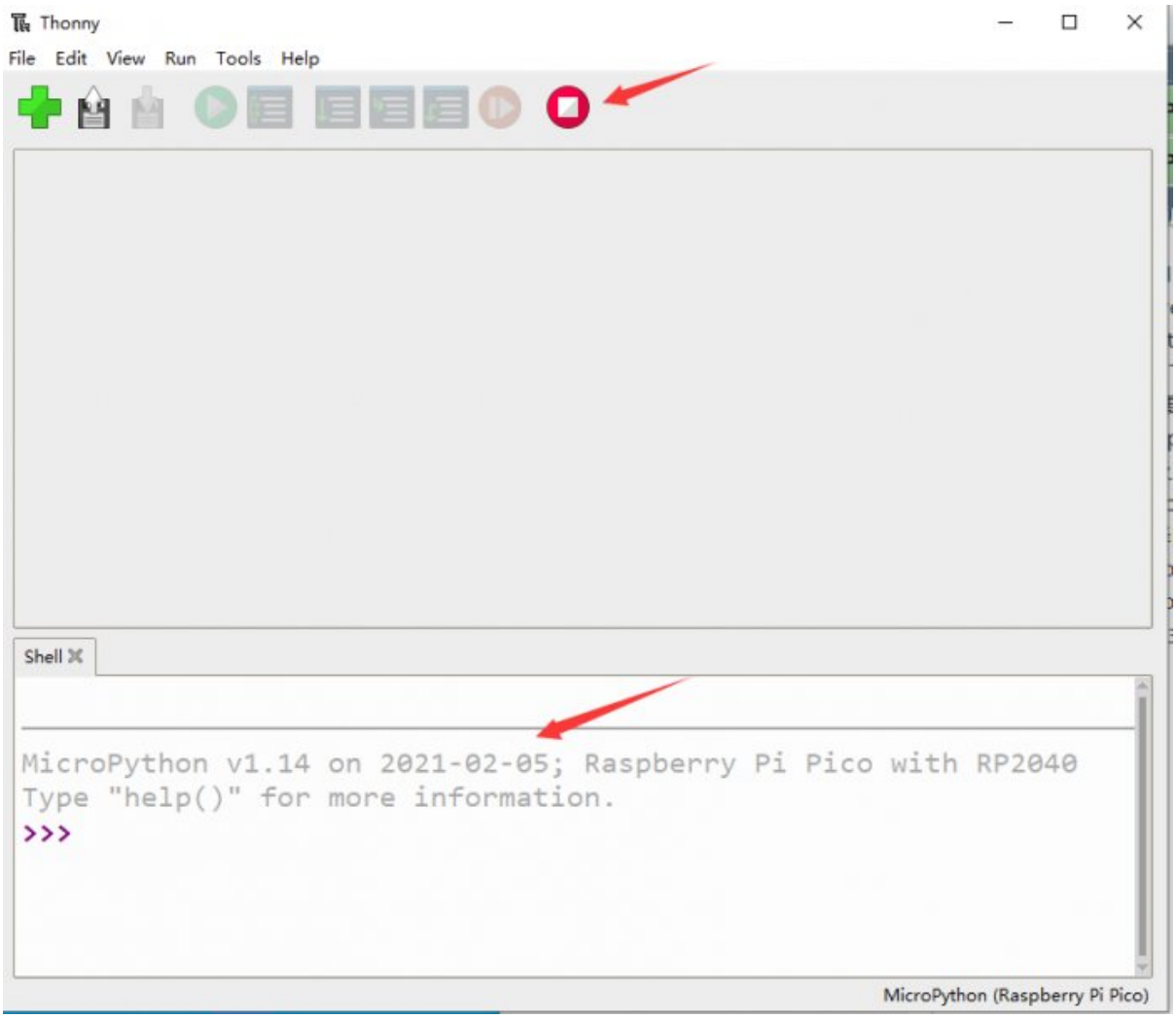






- Click OK and then back to Thonny, download the firmware library to the Pico. Then click Stop, and you can see the current environment in the Shell window.
- How to download the firmware library for Pico in Windows: Press and hold the BOOT key and connect to the computer, then release the BOOT key, a removable disk will appear on the computer, and copy the firmware library into it.
- How to download firmware library for RP2040 in Windows: After connecting to the computer, press the BOOT key and RESET key at the same time, release the RESET key, and then release the BOOT key, a removable disk will appear on the computer.

Copy the firmware library into it (you can also use the Pico method).



Examples

- Download Demo Codes to your Raspberry Pi and test.

External LED Example

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC.

Open the Lesson-5 External LED example with Thonny. Run the example, and you will find that the red LED is flashing.

- Codes:

```
led_external = machine.Pin(15, machine.Pin.OUT) #Set GP15 to output Mode
while True:
   led_external.toggle() #Toggle the LED every 5 seconds.
   utime.sleep(5)
```

Traffic Light System Examples

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC. Open the Lesson-9 Traffic-Light-System example by Thonny, run the codes and test the traffic light, the buzzer sounds when you press the button.

- Codes

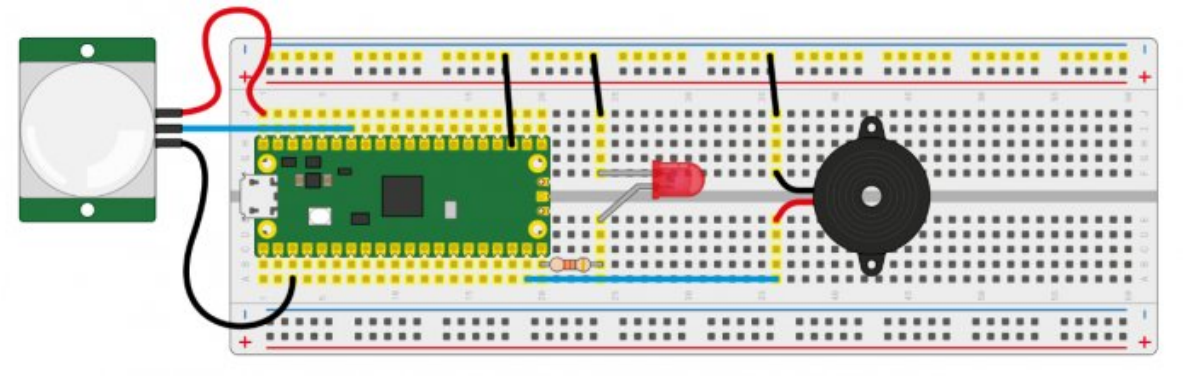
```
def button_reader_thread():  #Check if the button is pressed
   global button_pressed
   while True:
       if button.value() == 1:
           button_pressed = True

_thread.start_new_thread(button_reader_thread, ()) #Start a new thread to monito
r the stats of button
while True:
   if button_pressed == True: #If the button is pressed, turn on the LED and let
the buzzer work.
       led_red.value(1)
       for i in range(10):
           buzzer.value(1)
           utime.sleep(0.2)
           buzzer.value(0)
           utime.sleep(0.2)
       global button_pressed
       button_pressed = False
   led_red.value(1)
   utime.sleep(5)
   led_amber.value(1)
   utime.sleep(2)
   led_red.value(0)
   led_amber.value(0)
   led_green.value(1)
   utime.sleep(5)
```

```
    led_green.value(0)
    led_amber.value(1)
    utime.sleep(5)
    led_amber.value(0)
```

Burglar Alarm LED Buzzer Examples

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC. Open the Lesson-14 Burglar Alarm LED Buzzer examples by Thonny. The LED lights on if an object is moving around the Passive infrared sensor and the buzzer will indicate.

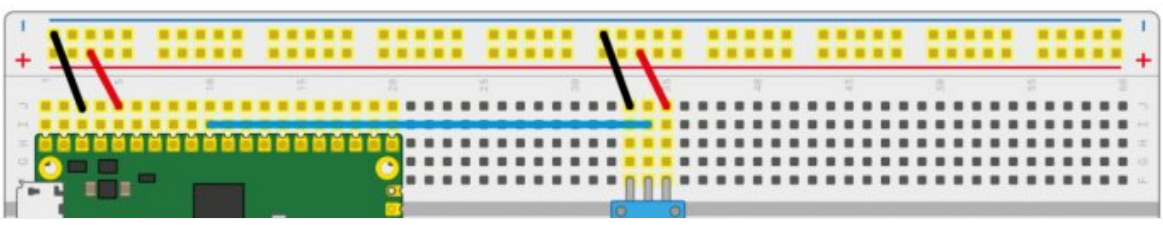


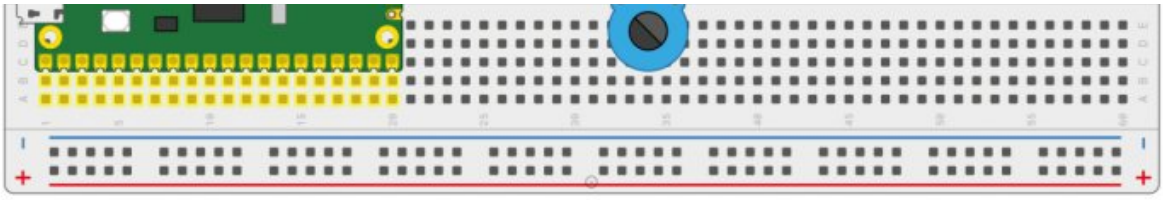
- Codes

```
def pir_handler(pin):  #Interrupt process function
   print("ALARM! Motion detected!")
   for i in range(50):
       led.toggle()
       buzzer.toggle()
       utime.sleep_ms(100)
sensor_pir.irq(trigger=machine.Pin.IRQ_RISING, handler=pir_handler)#Enable the I
nterrupt, the interrupt function is called when motions is detected.
while True:  #Toggle LED every 5s
   led.toggle()
   utime.sleep(5)
```

Potentiometer Example

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC. Open the Lesson-16 Potentiometer example by Thonny, you can adjust the potentiometer and check if the voltage printed to the Sheel window are changing as well.





- Codes

```
potentiometer = machine.ADC(26) #Set the GP26 pin as analog input
conversion_factor = 3.3 / (65535)
while True:
   voltage = potentiometer.read_u16() * conversion_factor #Convert the sampled d
ata to voltage value
   print(voltage) #Print the voltage data, it chanaged according to the sliding
rheostat.
   utime.sleep(2)
```

WS2812 Example

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC. Open the WS2812_RGB_LED.py file of Lesson-25 WS2812 example by Thonny, the LEDs light in Blue, Red, Green, and White.

Raspberry-Pi-Pico-Basic-Kit-WS2812.png

- Code

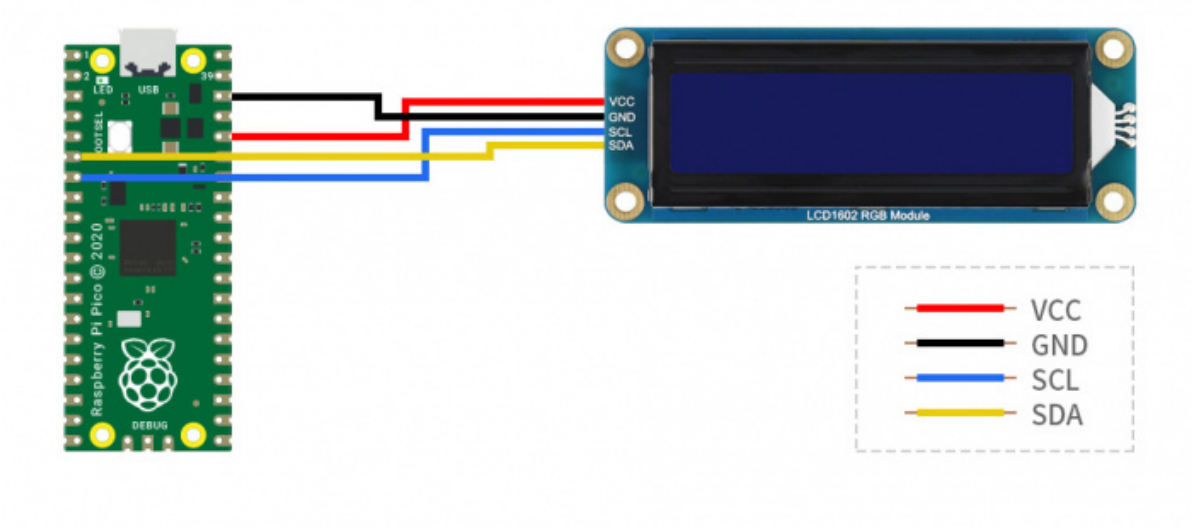
```
#This code uses the state machine mechanism. The following code is a decorator w
here we can initialize the hardware, set the pin level, etc.
#label("bitloop") We can define some tags in our code so that we can jump to the
m.
#jmp(not_x,"do_zero") If x=0, we jumpt to do_zero.
#nop() .set(0) [T2 - 1] The code jumpt to here if x = 0.
@asm_pio(sideset_init=PIO.OUT_LOW, out_shiftdir=PIO.SHIFT_LEFT, autopull=True, p
ull_thresh=24)
def ws2812():
   T1 = 2
   T2 = 5
   T3 = 1
   label("bitloop")
   out(x, 1)               .side(0)    [T3 - 1]
   jmp(not_x, "do_zero")   .side(1)    [T1 - 1]
   jmp("bitloop")          .side(1)    [T2 - 1]
   label("do_zero")
   nop()                   .side(0)    [T2 - 1]
```

```
# Create the StateMachine with the ws2812 program, outputting on Pin(22).
sm = StateMachine(0, ws2812, freq=8000000, sideset_base=Pin(0)) #Create the stat
```

```
s machine
# Start the StateMachine, it will wait for data on its FIFO.
sm.active(1) #Start the stats machine
# Display a pattern on the LEDs via an array of LED RGB values.
ar = array.array("I", [0 for _ in range(NUM_LEDS)])
print(ar)
print("blue")
for j in range(0, 255):
   for i in range(NUM_LEDS):
       ar[i] = j
   sm.put(ar,8)  #put() is put the data to output FIFO of the stats machine
   time.sleep_ms(5)
```

LCD1602 I2C Example

Connect the boards as in the picture below. Connect the Pico to Raspberry Pi or PC. Open the Lesson-21 LCD1602 I2C example by Thonny, you need to first save the RGB1602.py to Pico and then run the Choose_Color.py file. The LCD will change color every 5s. If you run the Discoloratio.py file, the LED displays RGB colors.



- Codes

Choose_Color.py

```
#Define colors
rgb9 = (0,255,0) #green
lcd.setCursor(0, 0) #Set the position of cursor
# print the number of seconds since reset:
lcd.printout("Waveshare") #Print the string
lcd.setCursor(0, 1) #Move the cursor to second row.
lcd.printout("Hello,World!")#Print the string
lcd.setRGB(rgb1[0],rgb1[1],rgb1[2]); #Set the back light
```

Discoloration.py

```
t=0
while True:

    r = int((abs(math.sin(3.14*t/180)))*255);  #RGB changes as time goes
    g = int((abs(math.sin(3.14*(t+60)/180)))*255);
    b = int((abs(math.sin(3.14*(t+120)/180)))*255);
    t = t + 3;
    lcd.setRGB(r,g,b);#Set the RGB data again.
# set the cursor to column 0, line 1
    lcd.setCursor(0, 0) #Set the curson to the first row.
# print the number of seconds since reset:
    lcd.printout("Waveshare")#Print the string
    lcd.setCursor(0, 1) #Set the cursor to second row
    lcd.printout("Hello,World!")#Print the string
    time.sleep(0.3)
```

Documents

- RP2040-LCD-0.96 Schematic
- ST7735S Datasheet

Demo Codes

- Demo codes (0.96inch LCD)

Pico Quick Start

Download Firmware

- MicroPython Firmware Download
- C_Blink Firmware Download [Expand]

Video Tutorial [Expand]

- Pico Tutorial I - Basic Introduction
- Pico Tutorial II - GPIO [Expand]
- Pico Tutorial III - PWM [Expand]
- Pico Tutorial IV - ADC [Expand]
- Pico Tutorial V - UART [Expand]
- Pico Tutorial VI - To be continued... [Expand]

MicroPython Series

- 【MicroPython】 machine.Pin Function
- 【MicroPython】 machine.PWM Function
- 【MicroPython】 machine.ADC Function
- 【MicroPython】 machine.UART Function
- 【MicroPython】 machine.I2C Function
- 【MicroPython】 machine.SPI Function
- 【MicroPython】 rp2.StateMachine

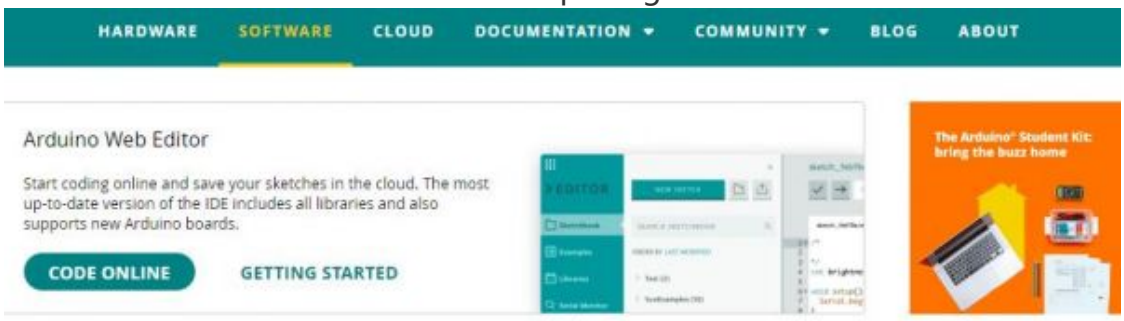
C/C++ Series

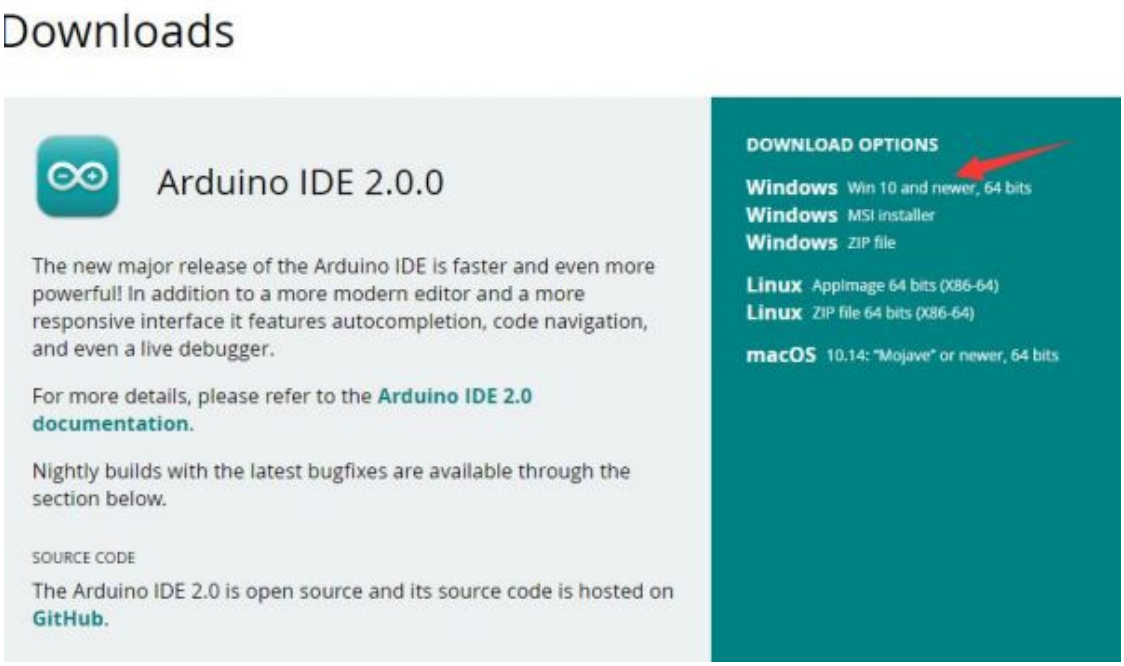
- 【C/C++】 Windows Tutorial 1 - Environment Setting
- 【C/C++】 Windows Tutorial 1 - Create New Project

Arduino IDE Series

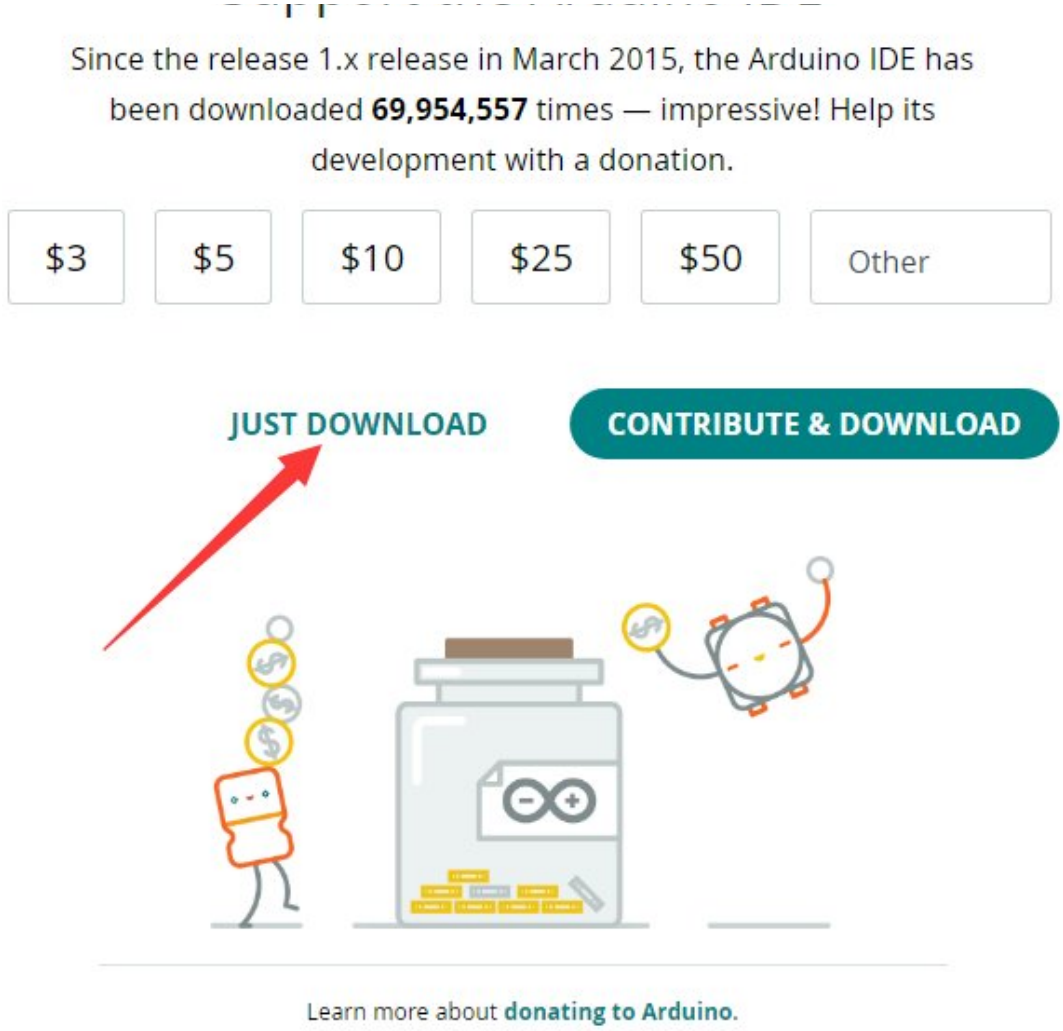
Install Arduino IDE

1. Download the Arduino IDE installation package from Arduino website.





2. Just click on "JUST DOWNLOAD".

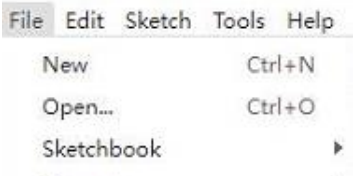


3. Click to install after downloading.

4. **Note: You will be prompted to install the driver during the installation process, we can click Install.**

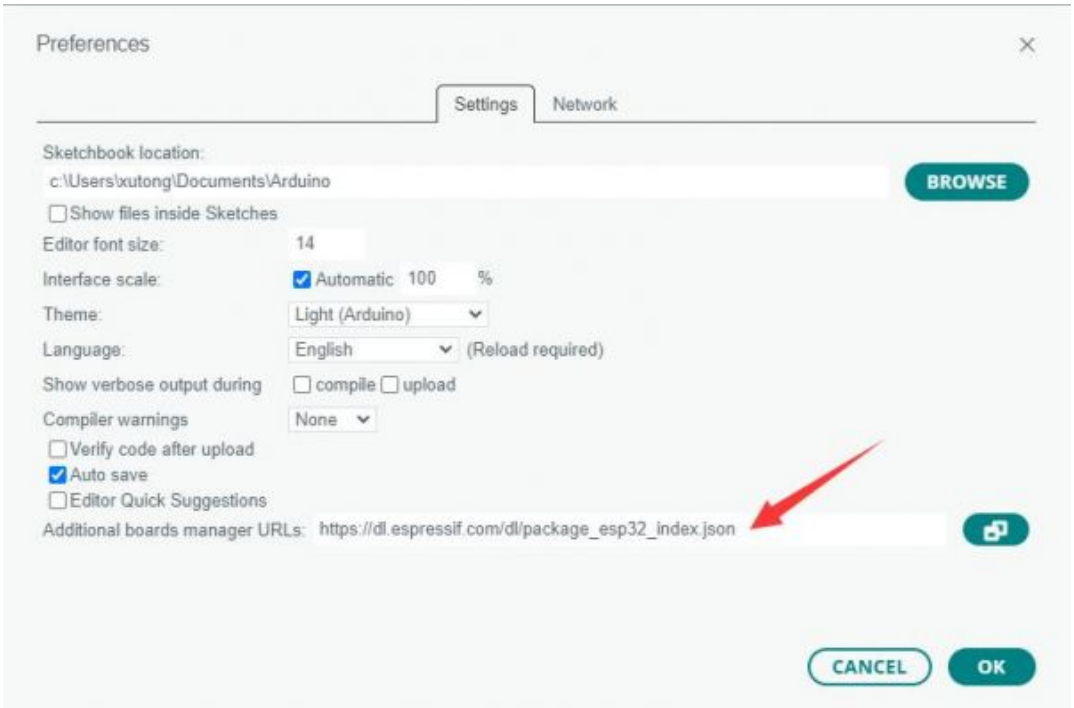
Install Arduino-Pico Core on Arduino IDE

1. Open Arduino IDE, click the File on the left corner and choose "Preferences".



2. Add the following link in the additional development board manager URL, then click OK.

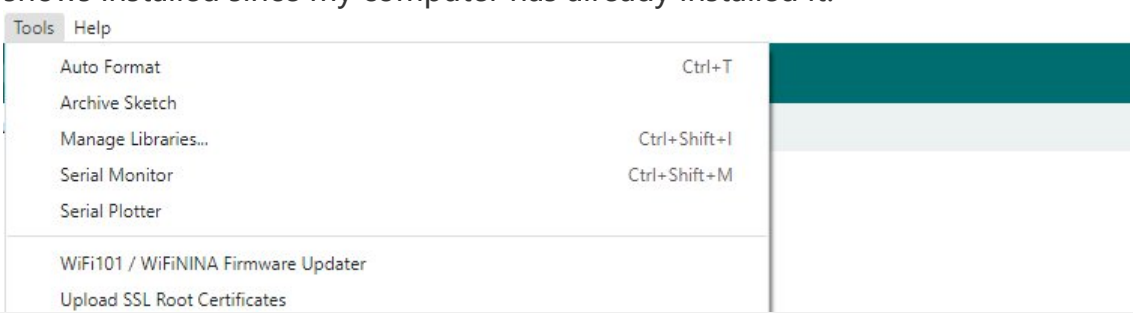
```
https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json
```

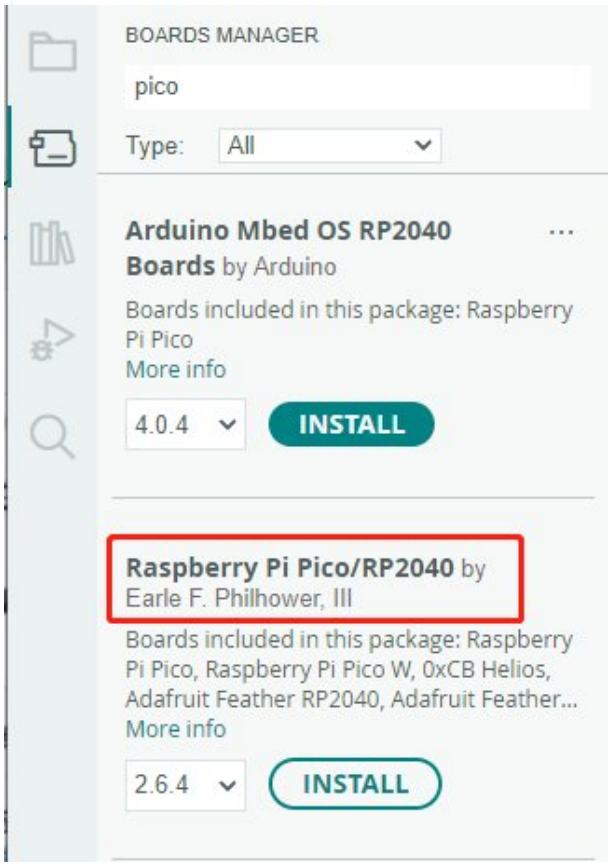


Note: If you already have the ESP8266 board URL, you can separate the URLs with commas like this:

```
https://dl.espressif.com/dl/package_esp32_index.json,https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json
```

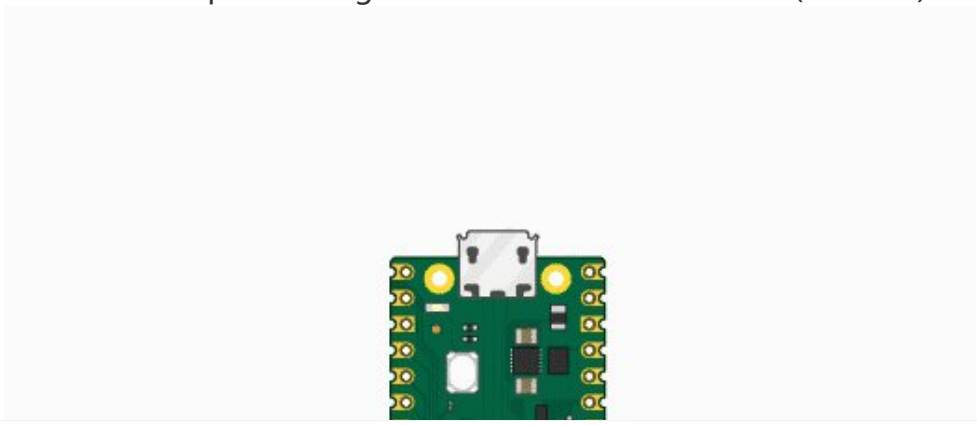
3. Click on Tools -> Dev Board -> Dev Board Manager -> Search for pico, it shows installed since my computer has already installed it.

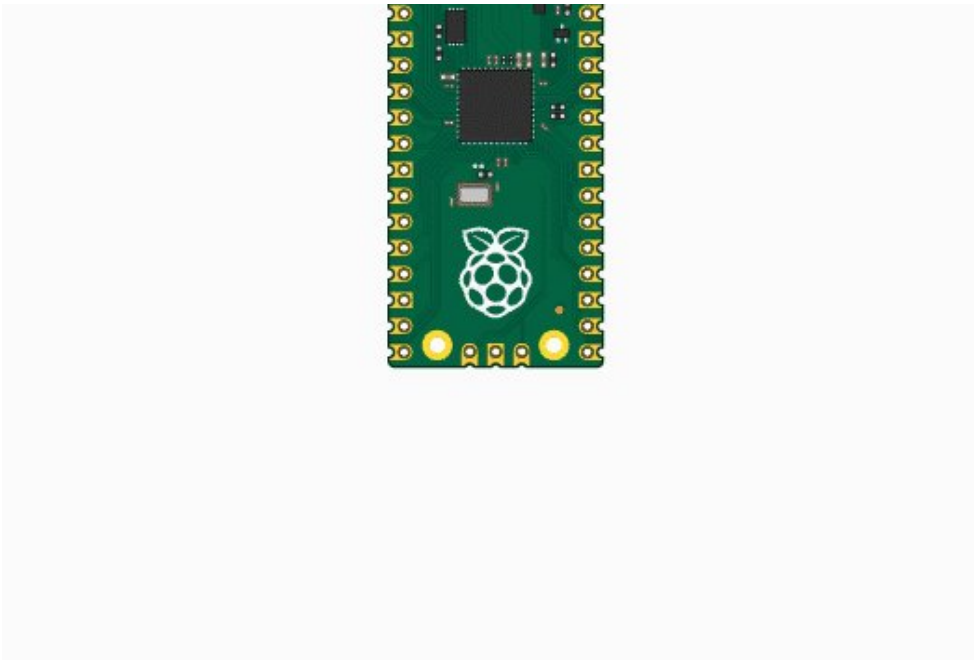




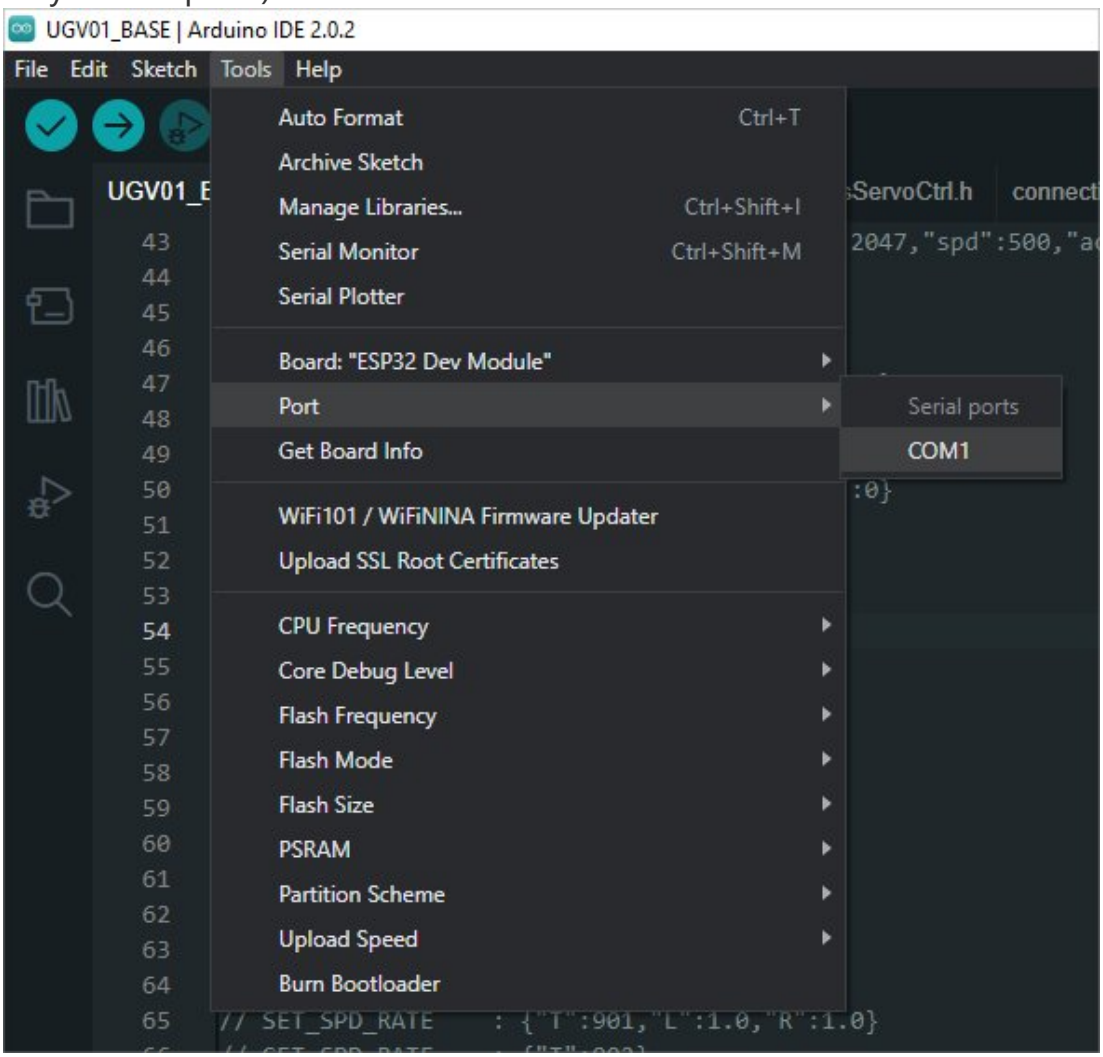
Upload Demo At the First Time

1. Press and hold the BOOTSET button on the Pico board, connect the Pico to the USB port of the computer via the Micro USB cable, and release the button when the computer recognizes a removable hard drive (RPI-RP2).

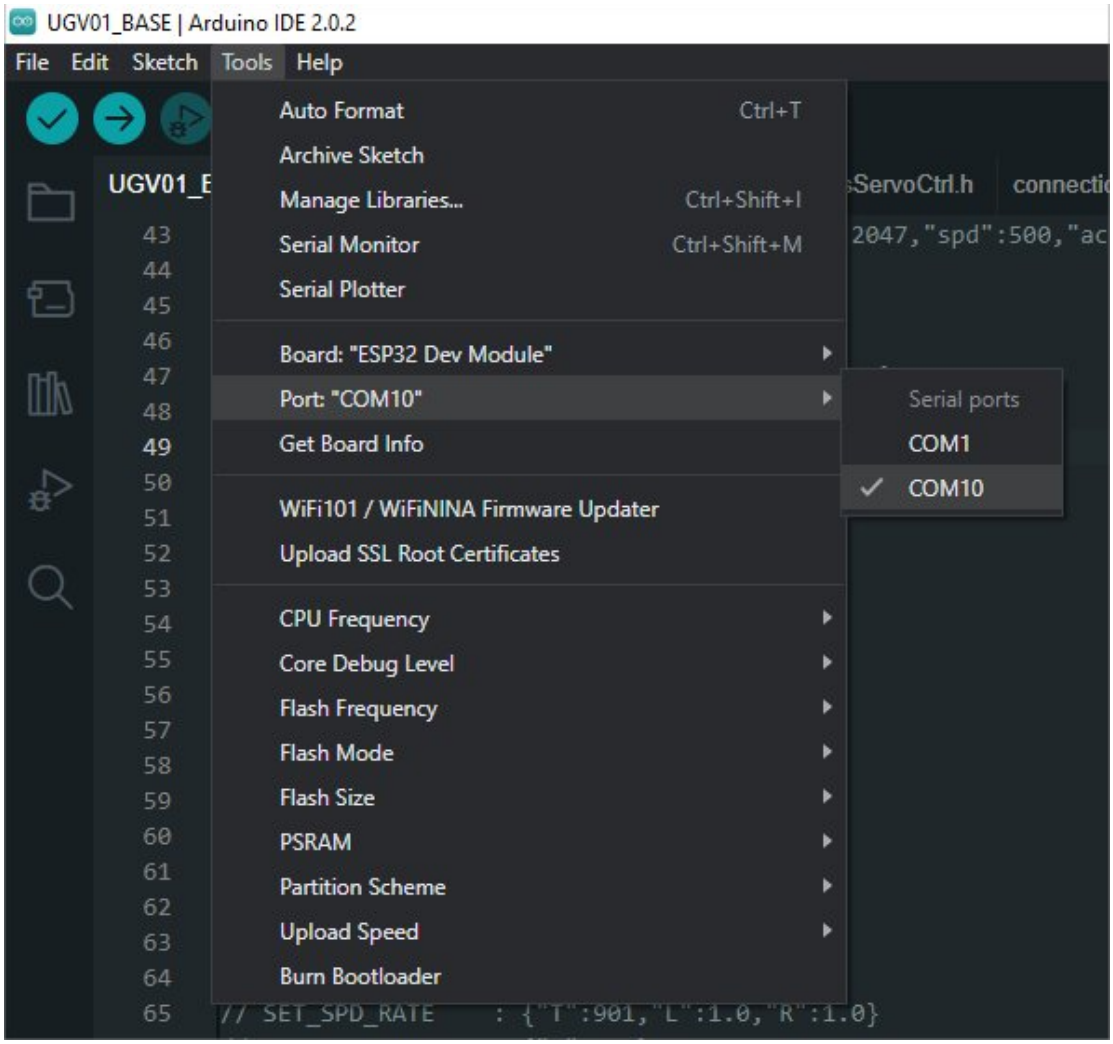




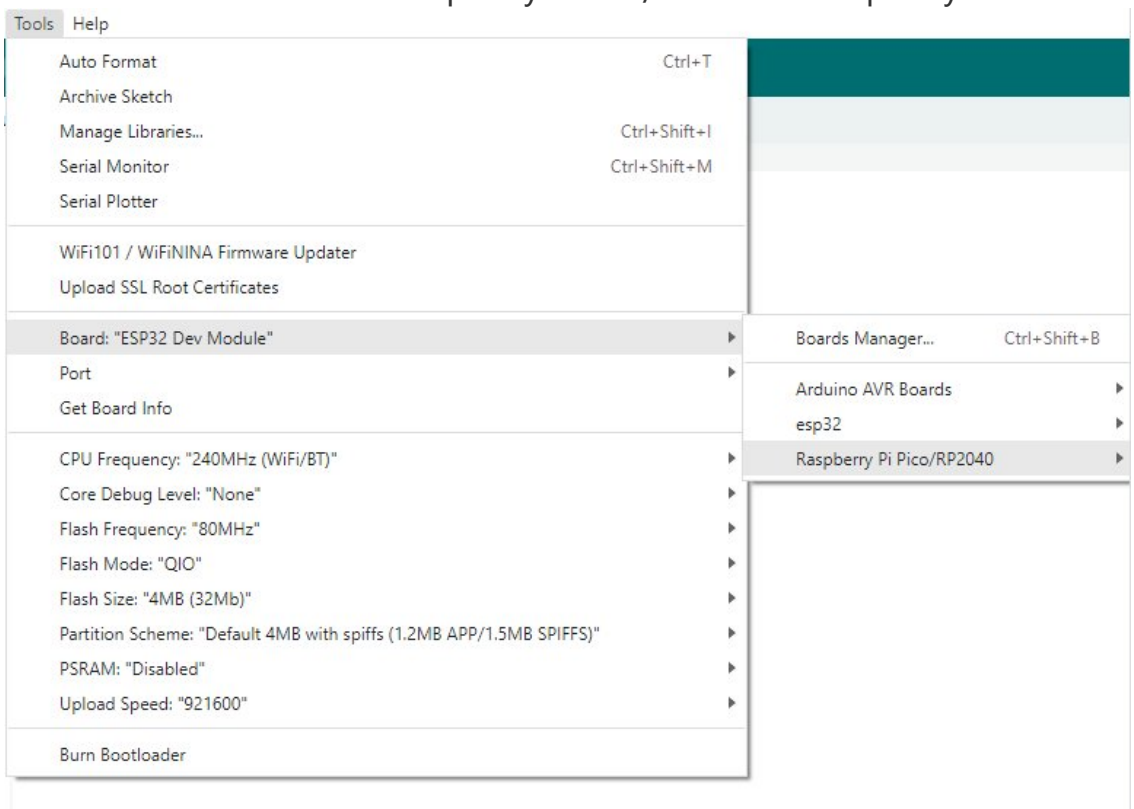
2. Download the demo, open arduino\PWM\D1-LED path under the D1-LED.ino.
3. Click Tools -> Port, remember the existing COM, do not need to click this COM (different computers show different COM, remember the existing COM on your computer).



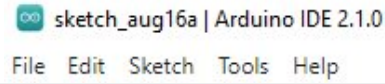
4. Connect the driver board to the computer with a USB cable, then click Tools -> Ports, select uf2 Board for the first connection, and after the upload is complete, connecting again will result in an additional COM port.



5. Click Tool -> Dev Board -> Raspberry Pi Pico/RP2040 -> Raspberry Pi Pico.



6. After setting, click the right arrow to upload.



- If you encounter problems during the period, you need to reinstall or replace the Arduino IDE version, uninstall the Arduino IDE needs to be uninstalled cleanly, after uninstalling the software you need to manually delete all the contents of the folder C:\Users\[name]\AppData\Local\Arduino15 (you need to show the hidden files in order to see it) and then reinstall.

Pico-W Series Tutorial (To be continued...)

Open Source Demo

- MicroPython Demo (GitHub)
- MicroPython Firmware/Blink Demo (C)
- Official Raspberry Pi C/C++ Demo
- Official Raspberry Pi MicroPython Demo
- Arduino Official C/C++ Demo

Resource

Example Demo

- Raspberry-Pi-Pico-Basic-Kit Example Demo
- Raspberry-Pi-Pico-Sensor-Kit Example Demo

Official Document

Pico W

- Pico W Datasheet
- Pico W Network Connection
- Pico W step (3D file)

Firmware

- Pico W MicroPython Firmware

Pico

User Manual

- Getting started with pico

- Pico c sdk
- Pico python sdk
- Raspberry-pi-pico-faq

Schematic & Datasheet

- Raspberry Pi Pico Schematic
- Pico Pinout
- Pico Datasheet
- Rp2040 Datasheet
- Hardware Design Manual

Related Books

- Raspberry Pi Pico Get Started MicroPython
- Related Raspberry Pi Books Download

Raspberry Pi Open-source Demo

- Official Raspberry Pi C?C++ Example Demo
- Official Raspberry Pi MicroPython Example Demo

Development Software

- Thonny Python IDE (Windows V3.3.3)
- Zimo221.7z
- Image2Lcd.7z

Demo Codes

- Demo codes for Raspberry-Pi-Pico-Basic-Kit
- Demo codes for Raspberry Pi Pico Sensor Kit

Support

Technical Support

If you need technical support or have any feedback/review,

please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.

Working Time: 9 AM - 6 AM GMT+8 (Monday to Friday)

Submit Now