uPy$_A PDS9960 Library Documentation$

Release 1.0

Rune Langøy

Jan 26, 2022

Contents

Another APDS9960 / GY-9960LLC / APDS9900 micro python library optimized for ESP8266 / ESP12-E for:

- Light Sensing (Ambient Light and RGB Color Sensing)
- Proximity Sensing

CHAPTER 1

Documentation

Complete documentation is hosted on the “Read the Docs” page upy-apds9960.readthedocs.io

CHAPTER 2

Dependencies

This driver depends on:

- MicroPython

Tested on:

Sensor: GY-9960LLC
Sensor: APDS-9900
Devboard: Node MCU v1.0 & Raspberry PI Pico

CHAPTER 3

Installation

- Flash the device with MicroPython
- Copy the folder uPy_APDS9960 and content (apds9960LITE.py) to the root folder for APDS9960 circuits
- Copy the folder uPy_APDS9900 and content (apds9900LITE.py) to the root folder for APDS9900 circuits

The steps above is descsribed in the Thonny IDE tutorial.

CHAPTER 4

Examples

The examples in theis respository uses the NodeMCU devboard the devboard to use rpi pico please change the I2C inferface as show in the code below

```
#Change I2C interface from:
#  i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))
#to:
i2c =  machine.I2C(0,scl=machine.Pin(17), sda=machine.Pin(16))
```

Here is the NodeMCU Hookup.

CHAPTER 5

APDS9960 Example

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

#Init I2C Buss on RP2040
i2c =  machine.I2C(0,scl=machine.Pin(17), sda=machine.Pin(16))

apds9960=APDS9960LITE(i2c)       # Enable sensor
apds9960.prox.enableSensor()     # Enable Proximit sensing

while True:
        sleep_ms(25) # wait for readout to be ready
        print(apds9960.prox.proximityLevel)   #Print the proximity value
```

CHAPTER 6

APDS9900 Example

```
import machine
from time import sleep_ms
from uPy_APDS9900.apds9900LITE import APDS9900LITE

#Init I2C Buss on RP2040
i2c =  machine.I2C(0,scl=machine.Pin(17), sda=machine.Pin(16))

apds9900=APDS9900LITE(i2c)       # Enable sensor
apds9900.prox.enableSensor()     # Enable Proximit sensing

while True:
        sleep_ms(25) # wait for readout to be ready
        print(apds9900.prox.proximityLevel)   #Print the proximity value
```

6.1 Hardware Set-up

Connect Vin to 3.3 V or 5 V power source, GND to ground, SCL and SDA to the appropriate pins to the Raspberry PI Pico

APDS9960	Name	Remarks	RPI PICO	Function
1	VIN	+3.3V Power	36	3V3
2	GND	Ground	GND	GND
3	SCL	I2C clock	22	GP17 (SCL)
4	SDA	I2C Data	21	GP16 (SDA)
5	INT	Interrupt	26	GP20

6.2 Basics

Of course, you must import the device and library :)

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE
```

To set-up the device to gather data, initialize the I2C-device using SCL and SDA pins. Then initialize the library.

```
i2c =  machine.I2C(0,scl=machine.Pin(17), sda=machine.Pin(16))
apds9960=APDS9960LITE(i2c)          # Poweron APDS9960
```

6.2.1 Proximity

Proximity funxtionalites is accessed torough the apds9960.prox member `PROX`

```
apds9960.prox.enableSensor()          # Enable Proximity sensing
sleep_ms(25)                          # wait for readout to be ready
print(apds9960.prox.proximityLevel)   # Print the proximity value
```

6.2.2 Light Sensing

Proximity funxtionalites is accessed torough the apds9960.als member `ALS`

```
apds9960.als.enableSensor()           # Enable Light sensor
sleep_ms(25)                          # Wait for readout to be ready
print(apds9960.als.ambientLightLevel) # Print the ambient light value
```

6.3 Debug

If things does not work try to run the script below to verify that it i2c communication with the apds9960 is working as expected

```
import machine
i2c =  machine.I2C(0,scl=machine.Pin(17), sda=machine.Pin(16))

print('Scan i2c bus...')
devices = i2c.scan()

if len(devices) == 0:
  print("No i2c device !")
else:
  print('i2c devices found:',len(devices))

  for device in devices:
    print("Decimal address: ",device," | Hexa address: ",hex(device))

    if(device==0x39): # APDS9960 Address = 0x39
        deviceID=i2c.readfrom_mem(devices[0],0x92, 1) #Get deviceID
        deviceID=int.from_bytes(deviceID,'big')       #Conv byte to int
        if(deviceID==0x29):
           deviceID=9900
        elif(deviceID==0x20):
            deviceID=9901
```

(continues on next page)

(continued from previous page)

```
        else:
            deviceID=9960

        print("Found ADPS-",deviceID)
```

If successful the output should be:

```
Scan i2c bus...
i2c devices found: 1
Decimal address:  57  | Hexa address:  0x39
Found ADPS- 9960
```

Note: Be aware if the output shows:

```
"many i2c devices was listed"  check if the i2c pins are allocated correctly
"No i2c device"                check if the power is correctly connected
```

The Device id can be 0xa8, 0xab 0x9c or 0x55.)

CHAPTER 7

Sphinx documentation

Sphinx the Python Documentation Generator is used for this documentation, if you like to build a local copy of the documentation install Sphinx :

```
python -m pip install sphinx
```

Ceate html doc by

```
cd docs
make html
```

The html pages would be located at : docs/_build/html

CHAPTER 8

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 9

Table of Contents

9.1 Proximity & Light Sensing

9.1.1 Proximity Examples

Examples that demostrates the use of apds9960 as a proximity sensor

9.1.1.1 Simple Example

Basic proximity test program.

Listing 1: examples/prox/simple_proximity_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

#Init I2C Buss
i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

apds9960=APDS9960LITE(i2c)       # Enable sensor
apds9960.prox.enableSensor()     # Enable Proximit sensing

while True:
        sleep_ms(25) # wait for readout to be ready
        print(apds9960.prox.proximityLevel)   #Print the proximity value
```

9.1.1.2 Regular Example

Example exposing more functions

Listing 2: examples/prox/proximity_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

# Proximity Gain (PGAIN) values
APDS9960_PGAIN_1X = const(0)
APDS9960_PGAIN_2X = const(1)
APDS9960_PGAIN_4X = const(2)
APDS9960_PGAIN_8X = const(3)

# LED Drive values
APDS9960_LED_DRIVE_100MA  = const(0)
APDS9960_LED_DRIVE_50MA   = const(1)
APDS9960_LED_DRIVE_25MA   = const(2)
APDS9960_LED_DRIVE_12_5MA = const(3)

i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

apds9960=APDS9960LITE(i2c)
apds9960.prox.eLEDCurrent   =APDS9960_LED_DRIVE_100MA
apds9960.prox.eProximityGain=APDS9960_PGAIN_8X
apds9960.prox.enableSensor()

sleep_ms(50)

while True:
    sleep_ms(50)
    print("proximity:", apds9960.prox.proximityLevel )
```

9.1.1.3 Simple IRQ

Example showing use of a hardware IRQ raised at a given proximity value

Listing 3: examples/prox/simple_irq_proximity_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

apds9960=APDS9960LITE(i2c)
apds9960.prox.eLEDCurrent    =0 # LED_DRIVE_100MA
apds9960.prox.eProximityGain =3 # PGAIN_8X
apds9960.prox.enableSensor()

#IRQ Functionalities
apds9960.prox.setInterruptThreshold(high=10,low=0,persistance=7)
apds9960.prox.enableInterrupt()

ProxThPin=machine.Pin(0, machine.Pin.IN ,machine.Pin.PULL_UP)

sleep_ms(50)

```

(continues on next page)

(continued from previous page)

```
while True:
    sleep_ms(50)

    if(ProxThPin.value()==0):
        print("proximity:", apds9960.prox.proximityLevel )
        apds9960.prox.clearInterrupt()
```

9.1.1.4 IRQ Example

Example showing use of interrupts exposing more functionalities

Listing 4: examples/prox/irq_proximity_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

# Proximity Gain (PGAIN) values
APDS9960_PGAIN_1X = const(0)
APDS9960_PGAIN_2X = const(1)
APDS9960_PGAIN_4X = const(2)
APDS9960_PGAIN_8X = const(3)

# LED Drive values
APDS9960_LED_DRIVE_100MA  = const(0)
APDS9960_LED_DRIVE_50MA   = const(1)
APDS9960_LED_DRIVE_25MA   = const(2)
APDS9960_LED_DRIVE_12_5MA = const(3)

i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))
print("Lite APDS-9960 Proximity test ")

apds9960=APDS9960LITE(i2c)
apds9960.prox.eLEDCurrent   =APDS9960_LED_DRIVE_100MA
apds9960.prox.eProximityGain=APDS9960_PGAIN_8X

apds9960.prox.enableSensor()
apds9960.prox.setInterruptThreshold(high=10,low=0,persistance=7)
apds9960.prox.enableInterrupt()

ProxThPin=machine.Pin(0, machine.Pin.IN ,machine.Pin.PULL_UP)

sleep_ms(50)

while True:
    sleep_ms(50)

    if(ProxThPin.value()==0):
        print("proximity:", apds9960.prox.proximityLevel )
        apds9960.prox.clearInterrupt()
```

9.1.2 Light Sensor Examples

9.1.2.1 Light sensing example

Ambient Light and RGB Color Sensing test program.

Listing 5: examples/als/simple_light_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apdsS9960LITE import APDS9960LITE

#Init I2C Buss
i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

apds9960=APDS9960LITE(i2c)          # Enable sensor
print("Enable light Sensor")

apds9960.als.enableSensor()    # Enable Light sensor
apds9960.als.eLightGain=3           # x64 gain
#apds9960.prox.enableProximity()
sleep_ms(50)
print("Clear Light level: ", apds9960.als.ambientLightLevel)
print("Red   Light level: "  , apds9960.als.redLightLevel)
print("Green Light level: ", apds9960.als.greenLightLevel)
print("Blue  Light level: " , apds9960.als.blueLightLevel)

```

9.1.2.2 Ambient light IRQ

Ambient Light IRQ test program.

Listing 6: examples/als/simple_light_irq_apds9960.py

```
import machine
from time import sleep_ms
from uPy_APDS9960.apds9960LITE import APDS9960LITE

#Init I2C Buss
i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

apds9960=APDS9960LITE(i2c)          # Enable sensor
apds9960.als.enableSensor()    # Enable Light sensor
apds9960.als.eLightGain=3           # x64 gain
apds9960.als.setInterruptThreshold(high=100,low=0,persistance=7)
apds9960.als.enableInterrupt(True)      # Enable interrupt
apds9960.als.clearInterrupt()           # Clear interrupt
sleep_ms(50)

IrqThPin=machine.Pin(0, machine.Pin.IN ,machine.Pin.PULL_UP)
sleep_ms(50)

while True:
    sleep_ms(50)

    if(IrqThPin.value()==0):
```

(continues on next page)

(continued from previous page)

```
        print("Ambient light level:", apds9960.als.ambientLightLevel )
        apds9960.als.clearInterrupt()
```

9.2 Debug

9.2.1 I2C Debug Example

Test program for testing the I2C connection the the aps9960

Listing 7: examples/debug/i2c_test.py

```
import machine
i2c = machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))

print('Scan i2c bus...')
devices = i2c.scan()

if len(devices) == 0:
  print("No i2c device !")
else:
  print('i2c devices found:',len(devices))

  for device in devices:
    print("Decimal address: ",device," | Hexa address: ",hex(device))

    if(device==0x39): # APDS9960 Address = 0x39
        deviceID=i2c.readfrom_mem(devices[0],0x92, 1) #G et deviceID
        print("Found ADPS9960: Device ID: ",deviceID)
```

9.3 Thonny IDE Turorial

This is a short visual tutorial on how to use the Thonny IDE to flash the ESP8266 chip and upload and test the uPy_APDS9960 library

9.3.1 Download and install

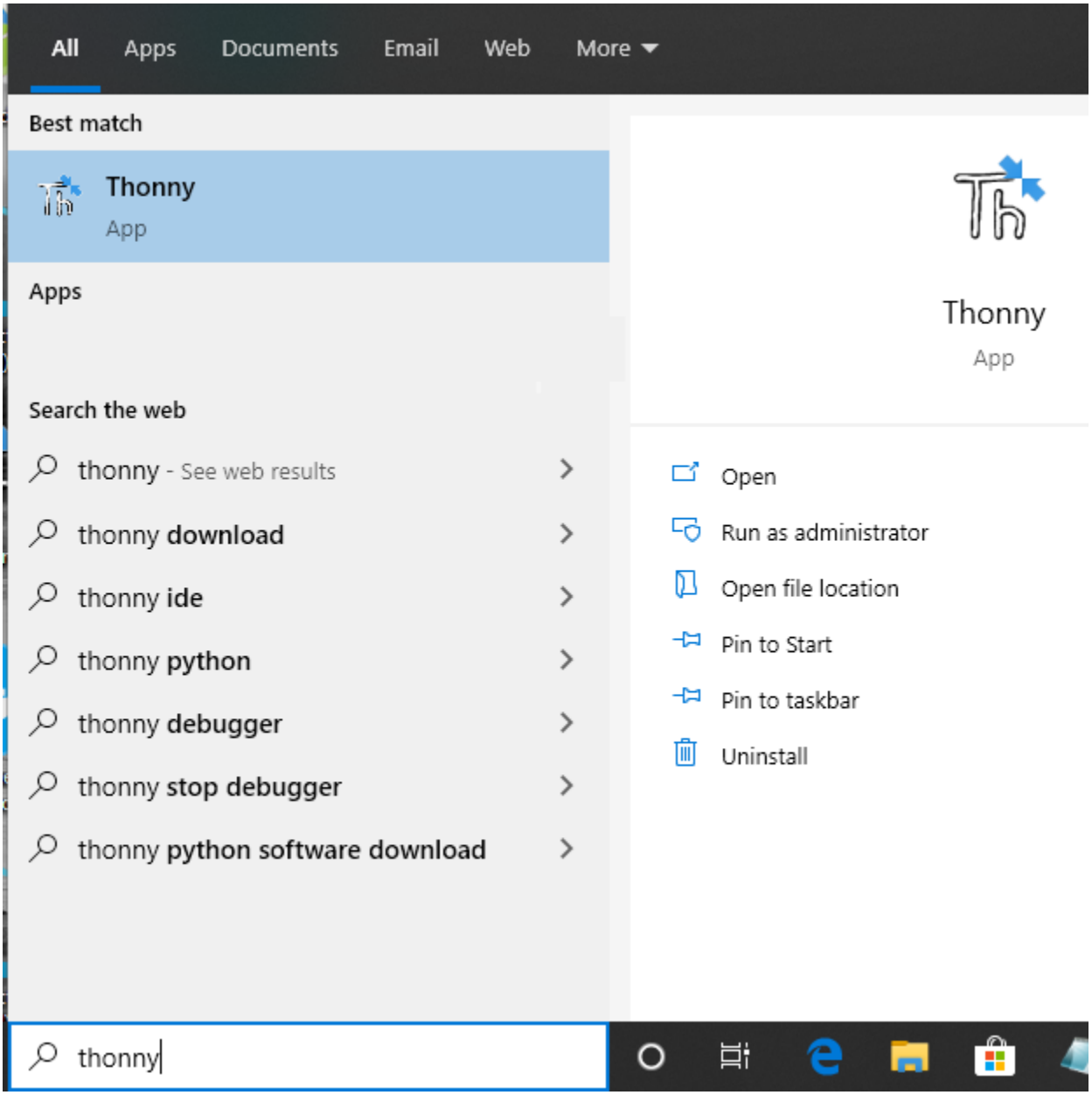
The latest version of tonny can be found at thonny.org

In this tutorial we uses the windows version and install Thonny on your computer

9.3.1.1 Download up Micropython (ESP8266) dev enviroment

Download latest MicroPython firmware for ESP8266

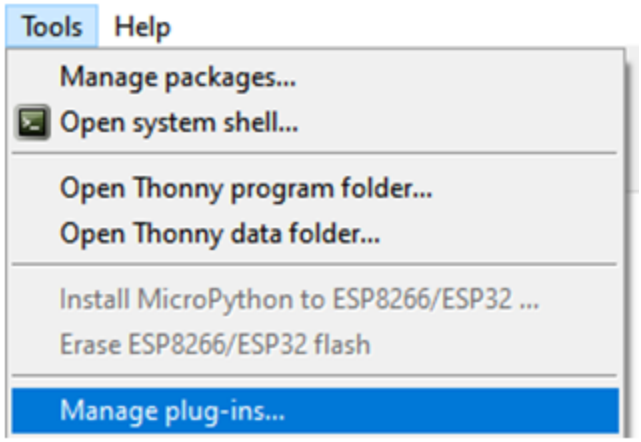
9.3.2 Start Thonny



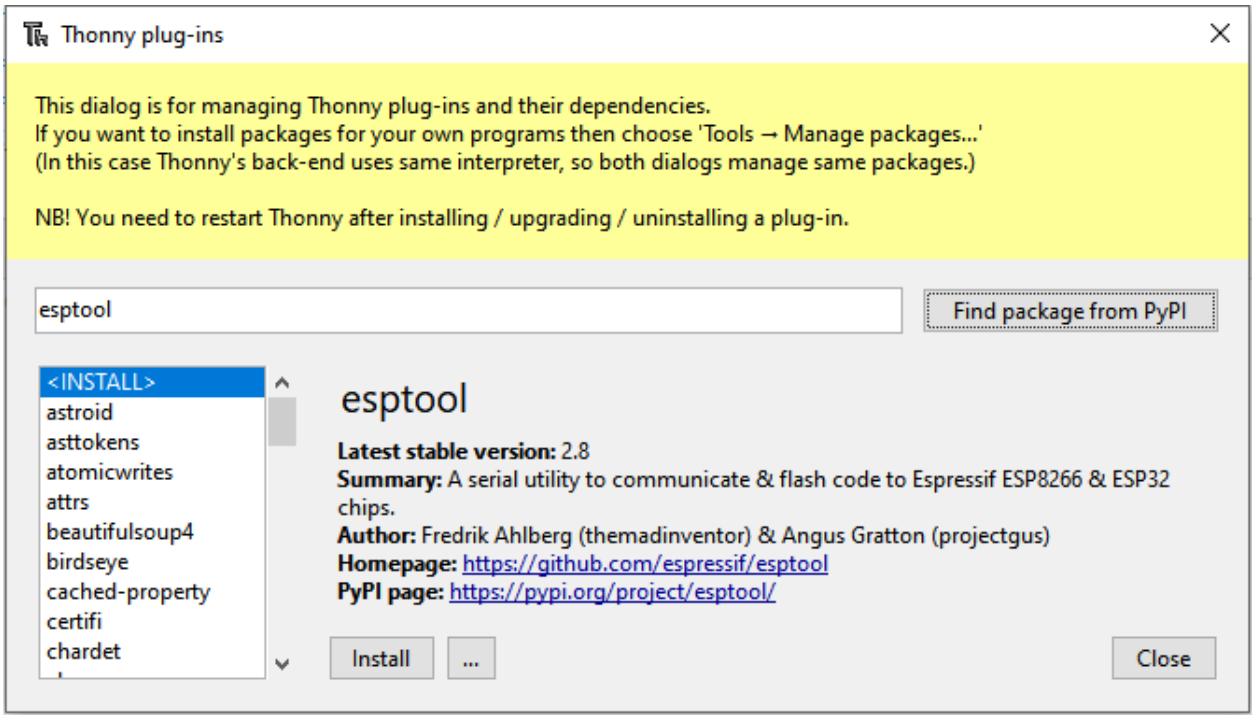
9.3.3 Flash new firmware

9.3.3.1 Installing esptool.py

From the menu “Tools” select “Manage Plut-ins. . . ”



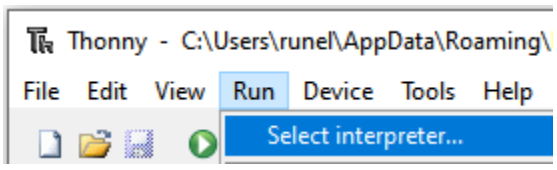
In the text field enter “esptool” and click the button “Find packages from PyPI”



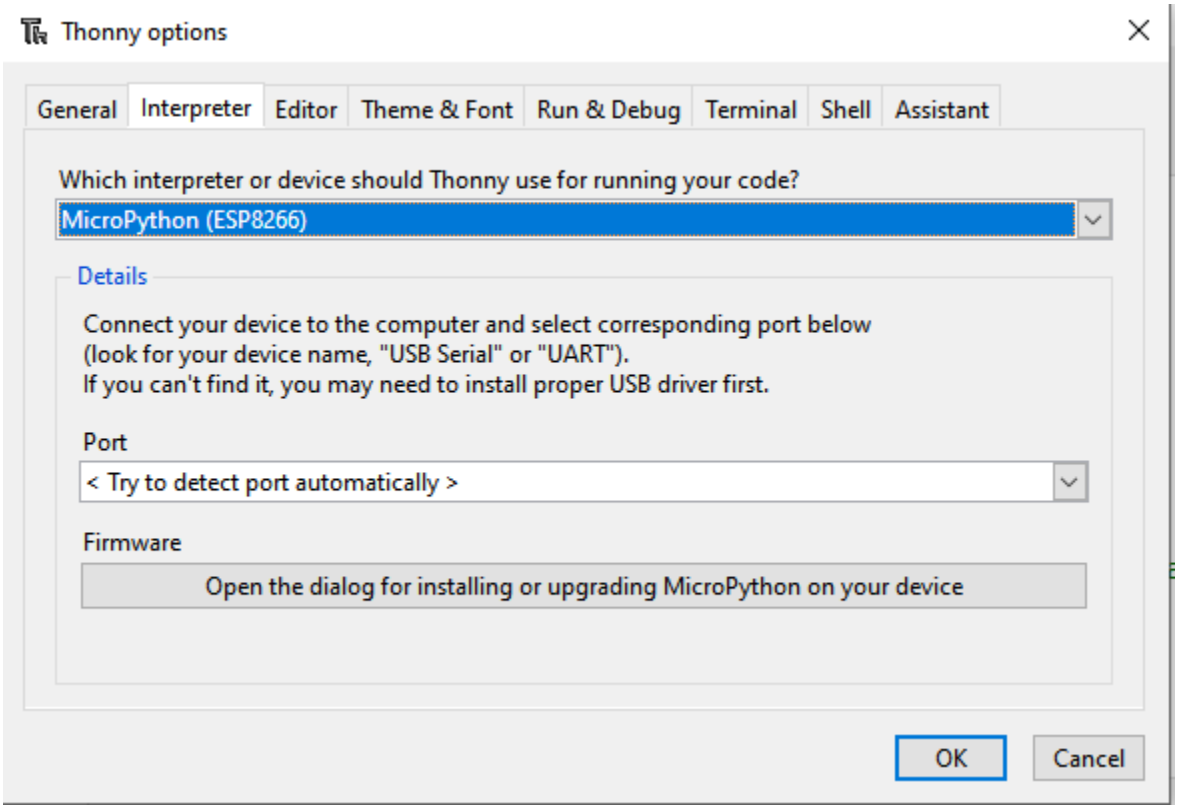
Click the “Install” button to finish the esptool installation

9.3.3.2 Seting up Micropython (ESP8266) dev enviroment

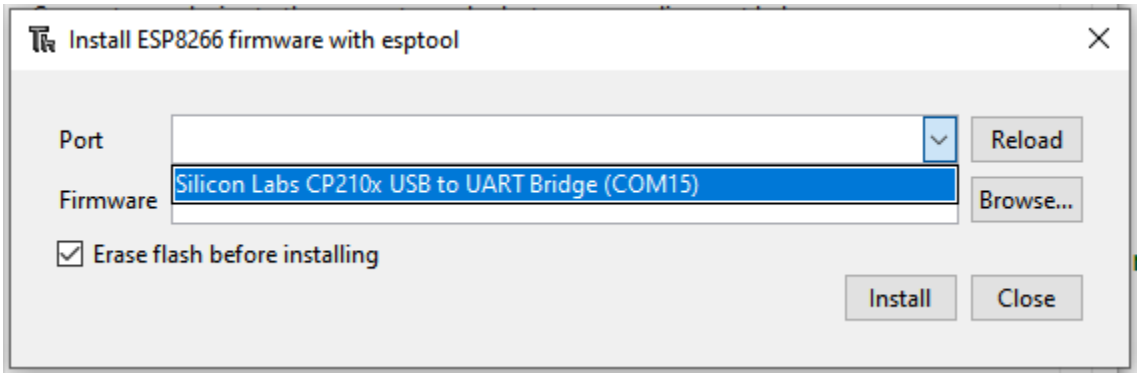
From the menu “Run” select “Select intepreter. . . ”



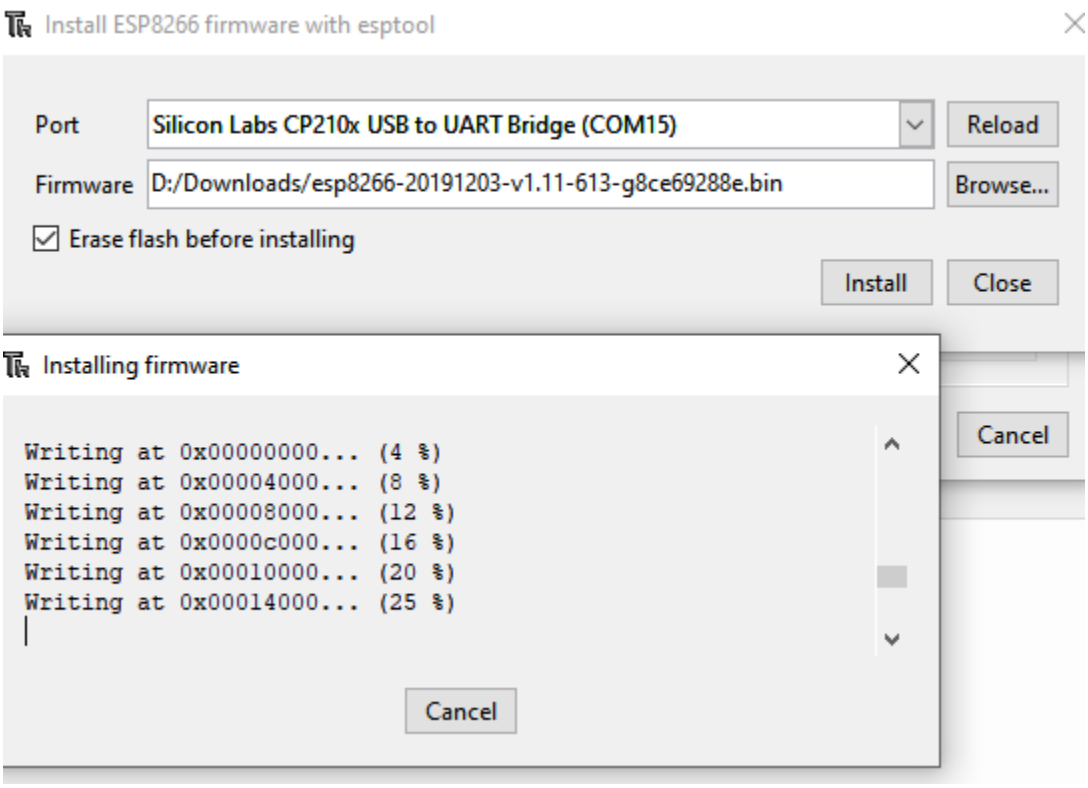
The ESP8266 firmware install/upgrade dialog is shown



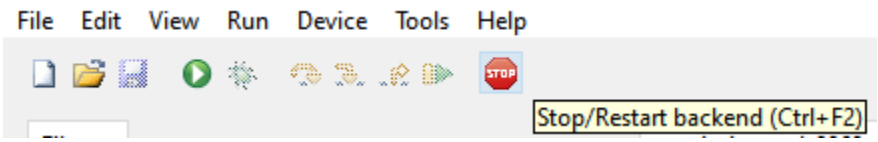
Make sure thet the ESP8266 development kit is connected. Select the "port dropdown" arrow to select the serial port for flashing the ESP8266.



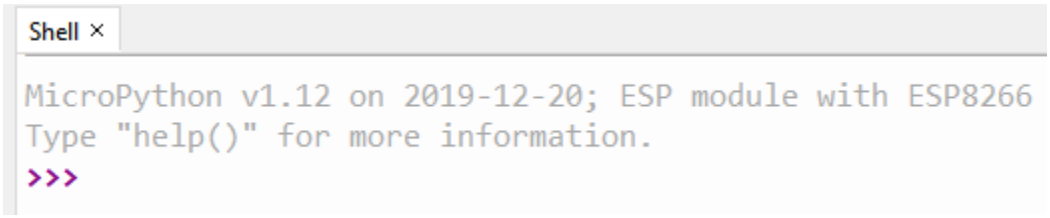
In the "Firmware" text field select the .bin file that was downloaded and click the install button



Click the stop icon to reset and connect to the ESP8266 board



Now you should be up and running as shown in the thonny shell windows

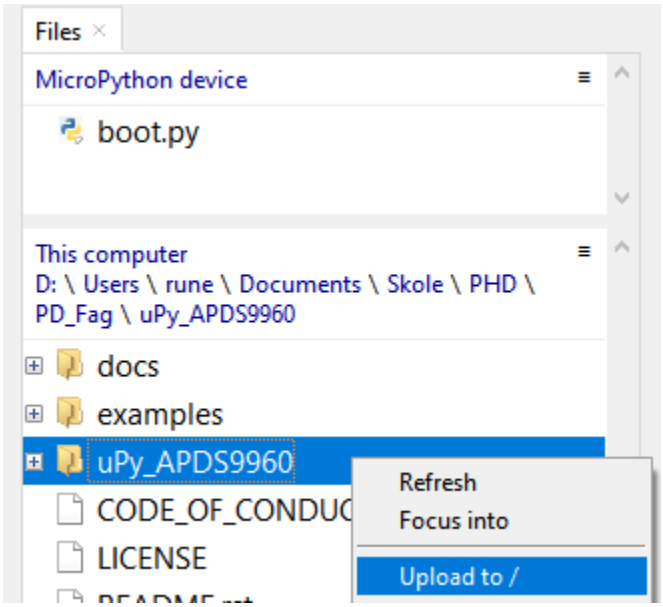


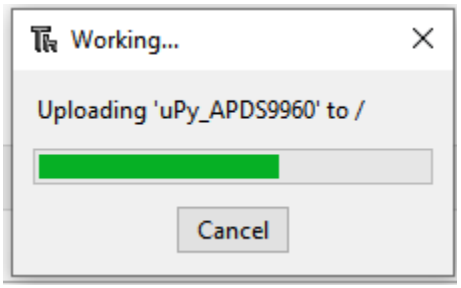
9.3.4 Running the examples

Start by uploadting the uPy_APDS9960 library.

9.3.4.1 Uploading uPy_APDS9960

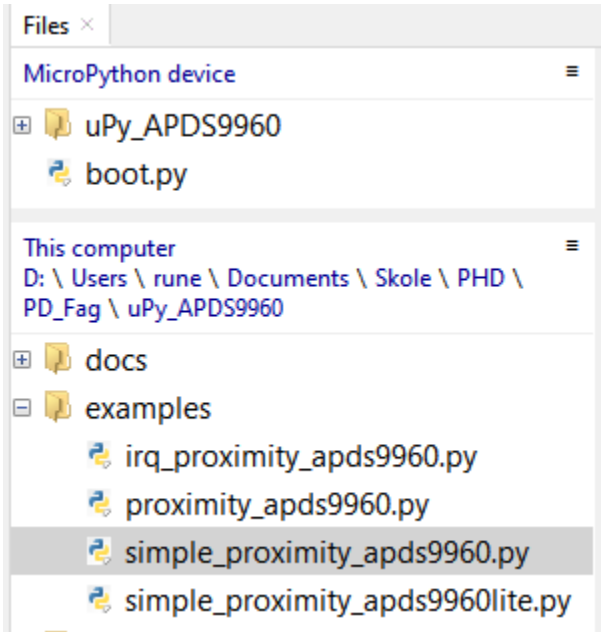
From the files windows under “This computer” right click on the folder “uPy_APDS9960” and from the dropdown menu select “Upload to /”





9.3.4.2 Running a example program

From the files windows under "This computer" click on the '+' sign infront of the folder "examples" to expand it.

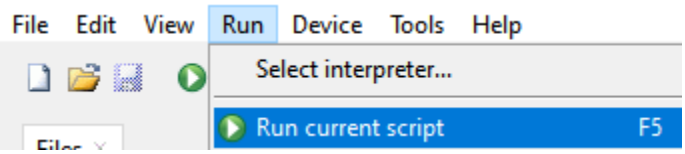


Double clikc on the file simple_proximity_apds9960.py and it wil be open the the editor

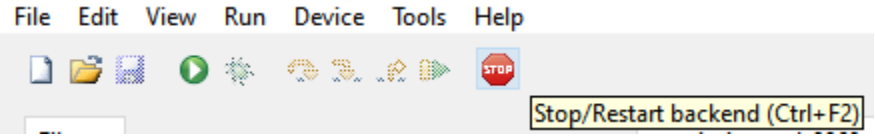
simple_proximity_apds9960.py

```
1 import machine
2 from time import sleep_ms
3 from uPy_APDS9960.APDS9960 import APDS9960
4
5 #Init I2C Buss
6 i2c =  machine.I2C(scl=machine.Pin(5), sda=machine.Pin(4))
```

You are now ready to run the program entring F5 or selecting the menu “Run” and “Run current script”

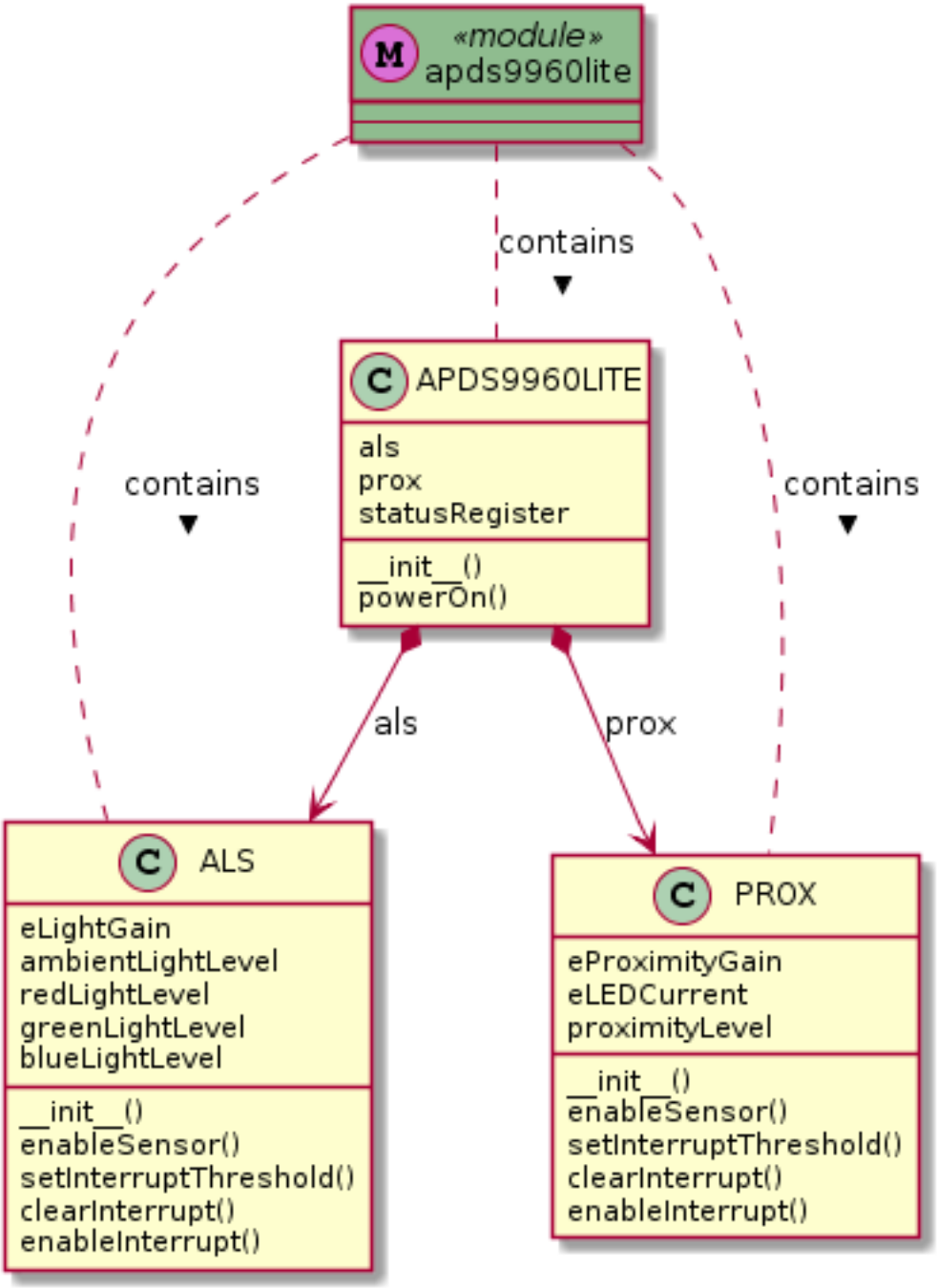


Click the stop icon to stop the program and return the command prompt



Have fun :)

9.4 uPy_APDS9960 Module



CHAPTER 10

Indices and tables

- genindex
- modindex
- search