Embedded Systems Programming

Spring 2021

Jack Leightcap[1,2]

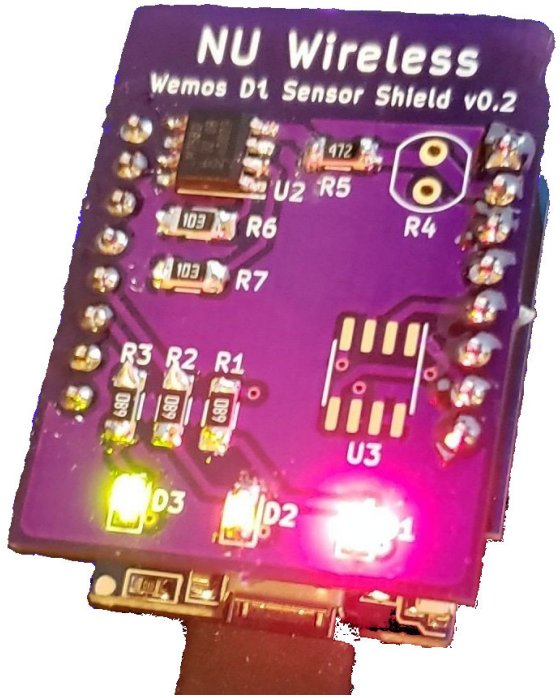
[1]IEEE – nuieeeofficers@gmail.com

[2]Wireless Club – nuwirelessclub@gmail.com

April 12, 2021

Roadmap

- Designed a PCB to connect to Wemos D1 Mini
- Soldered it
- Now to program with it!



Connecting To Board: Windows

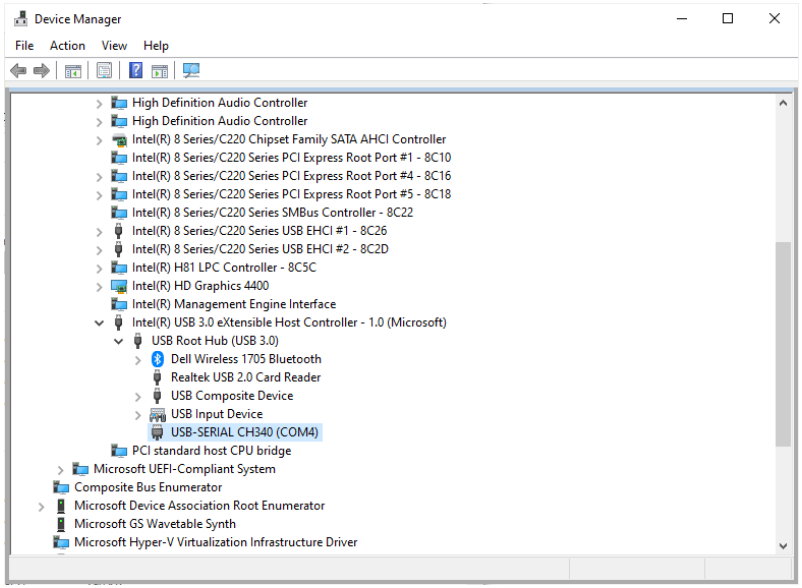


Figure: Find COM Port

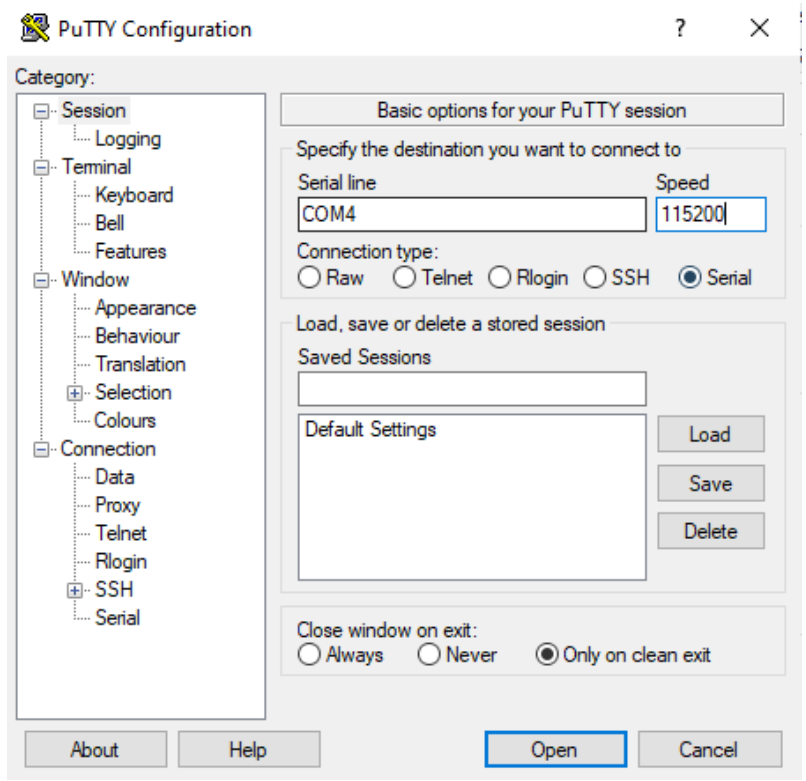


Figure: Connect with PuTTY

Connecting To Board: MacOS, Linux

In a terminal window,

```
$ screen /dev/ttyUSB0 115200
```

Figure: Connect with GNU Screen

MicroPython

Where are we now?

```
# This '>>>' is a prompt for a Python REPL
>>>
# can do some crazy math
>>> 2 + 2
4
# some more python
>>> list(map(lambda x: chr(x), [72, 105, 33]))
['H', 'i', '!']
# list what files we have around
>>> import os
>>> os.listdir()
['boot.py', 'inet.py', 'led.py', 'light.py', 'main.py', 'temp.py']
```

Figure: Python REPL

Drivers: LEDs

- Want to be able to control the board's LEDs
- The embedded programming equivalent of "Hello, World!"

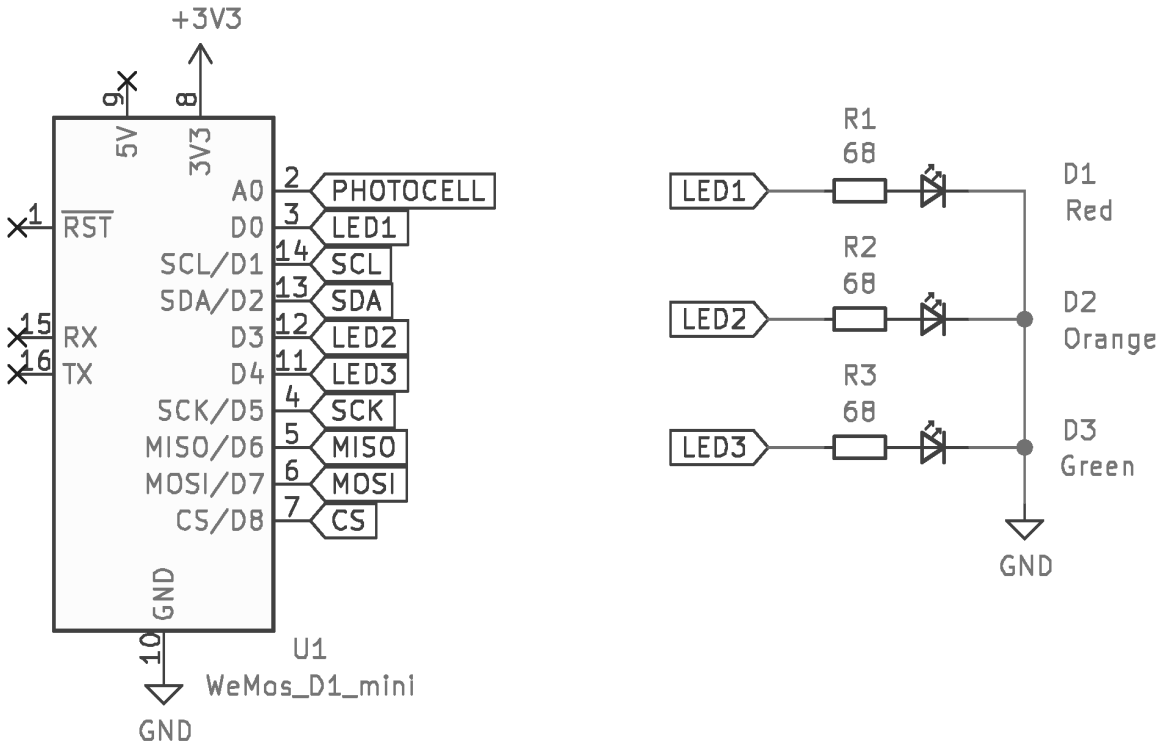


Figure: Relevant Schematic Pieces

Driver: LEDs – First Attempt

```
>>> import machine
>>> red = machine.Pin(0, machine.Pin.OUT) # D0
>>> orange = machine.Pin(3, machine.Pin.OUT) # D3
>>> green = machine.Pin(4, machine.Pin.OUT) # D4
>>> green.on() # ???
```

Drivers: LEDs – Physical Versus Mapped Pins

MicroPython uses *physical pins* of the ESP8266, not the *mapped pins* (`D0`, etc.) of the Wemos specifically.

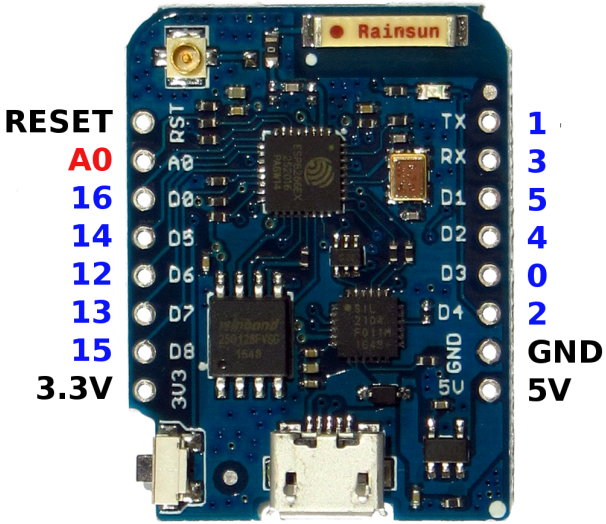


Figure: ESP8266 Pin Map

```
>>> import machine
>>> red = machine.Pin(16, machine.Pin.OUT)
>>> orange = machine.Pin(0, machine.Pin.OUT)
>>> green = machine.Pin(2, machine.Pin.OUT)
>>> green.on() # :^D
>>> green.off()
>>> # encapsulated in a driver:
>>> import led
>>> leds = led.LED()
>>> leds.set("GREEN", False)
```

Drivers: Light Sensor

ADC – Analog-Digital Converter

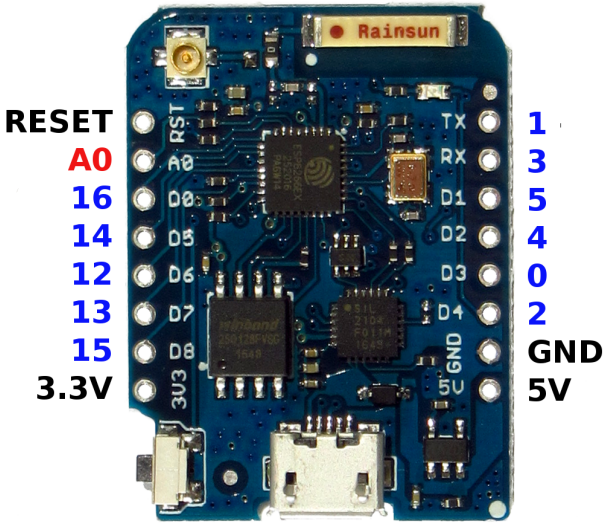


Figure: ESP8266 Pin Map

```
>>> import machine
>>> light = machine.ADC(0)
>>> light.read()
123
>>> light.read() # put finger over photocell
34
>>> # encapsulated in a driver:
>>> import light
>>> lightsensor = light.LightSensor()
>>> lightsensor.read()
...
```

Drivers: Temperature Sensor

- Uses `I2C` (*Inter-Integrated Circuit*)
  - *Serial* – Send one bit at a time over one wire, spread over time
  - *Parallel* – Send multiple bits at a time at once, over multiple wires
- Read a lots of documentation to get bespoke numbers...

```
>>> import machine
>>> i2c = machine.I2C(freq=400000, # frequency
                      scl=machine.Pin(5, machine.Pin.OUT), # clock
                      sda=machine.Pin(4)) # data
>>> i2c.writeto(72, b'00000000')
>>> output = i2c.readfrom(72, 2)
>>> output = int.from_bytes(output, "big")
>>> (output >> 5) / 8
32.5
>>> # encapsulated in a driver:
>>> import temp
>>> tempsensor = temp.TempSensor()
>>> tempsensor.read() # turns out this sensor isn't very accurate
32.5
```

Drivers: Internet

- Put it all together!
- (Not that exciting to implement...)

```
>>> import inet
>>> internet = inet.WebDriver("SSID", "pword")
Connection successful
('192.168.4.1', '255.255.255.0', ...)
>>> internet.serve(
  ["The temperature is:",
   "The light is:"],
  [tempsensor.read,
   lightsensor.read]
)
# connect your phone to that SSID...
# go to that IP address...
```

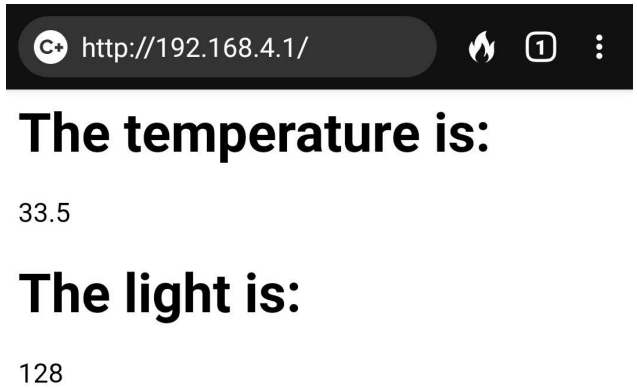


Figure: Readings on Phone