Embedded System Design Techniques™

Designing IoT Sensor Nodes using the ESP8266

Session 5: Device Management and the Automated Universe

July 14th, 2017

Jacob Beningo

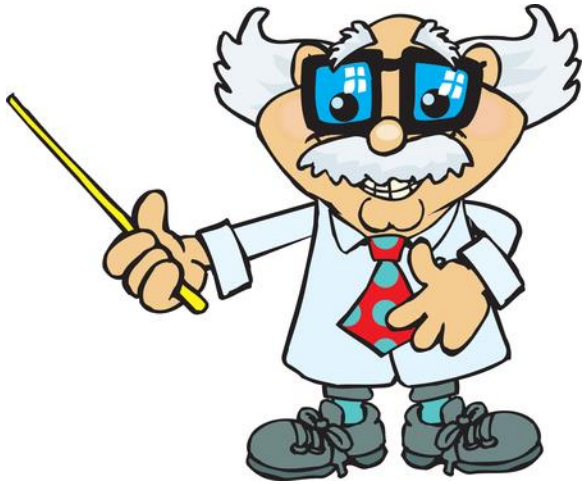
Course Overview

Topics:

- The IoT Architecture
- Getting Started with the ESP8266
- Interfacing Sensors to the ESP8266
- Connecting the ESP8266 to the internet
- **Device Management and the Automated Universe**

Session Overview

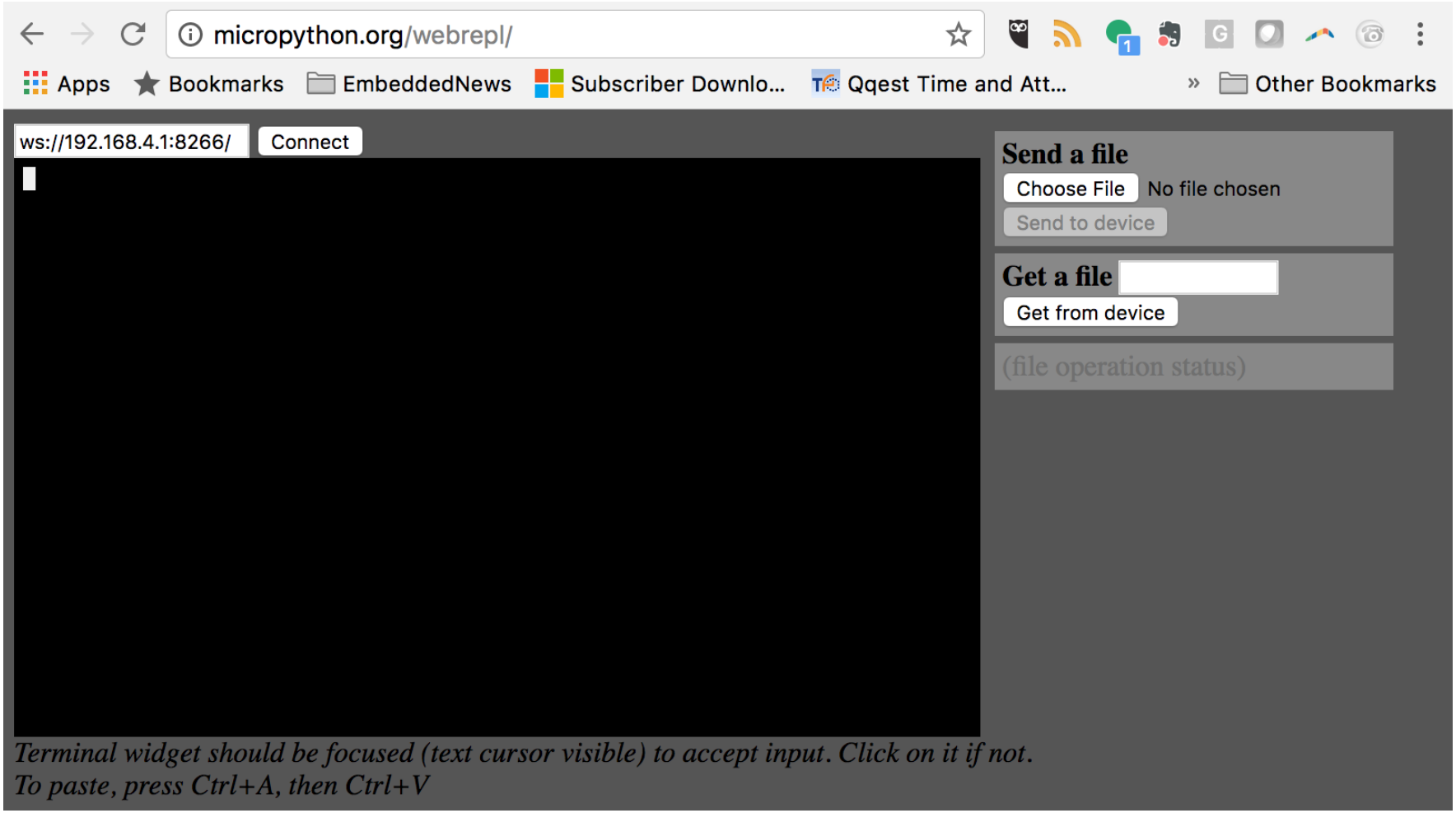
- Device Management
- Firmware Updates
- More Micro Python



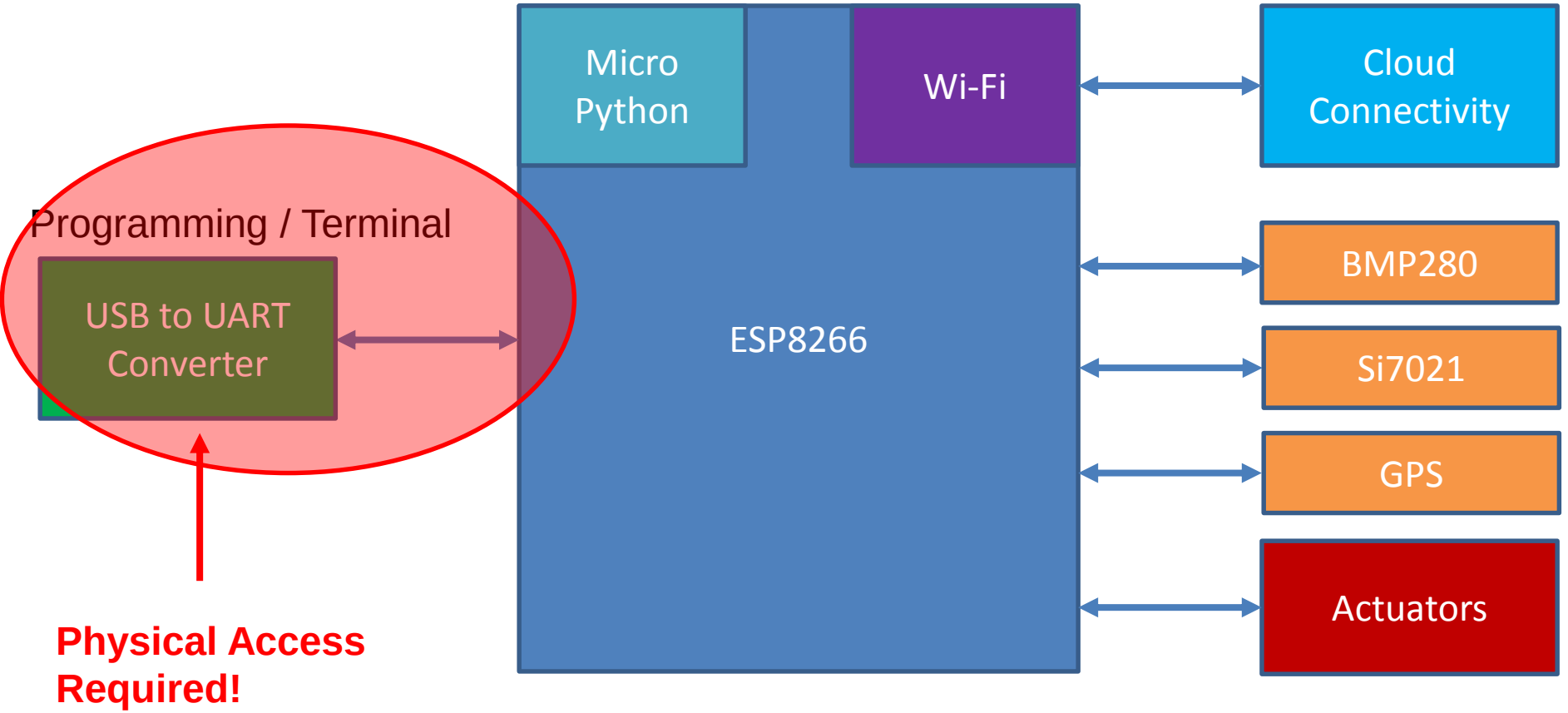
Device Management & Configuration

- Firmware Updates
- Kernel Updates
- Device Configuration
  - Security
  - Application Settings
  - Discovery

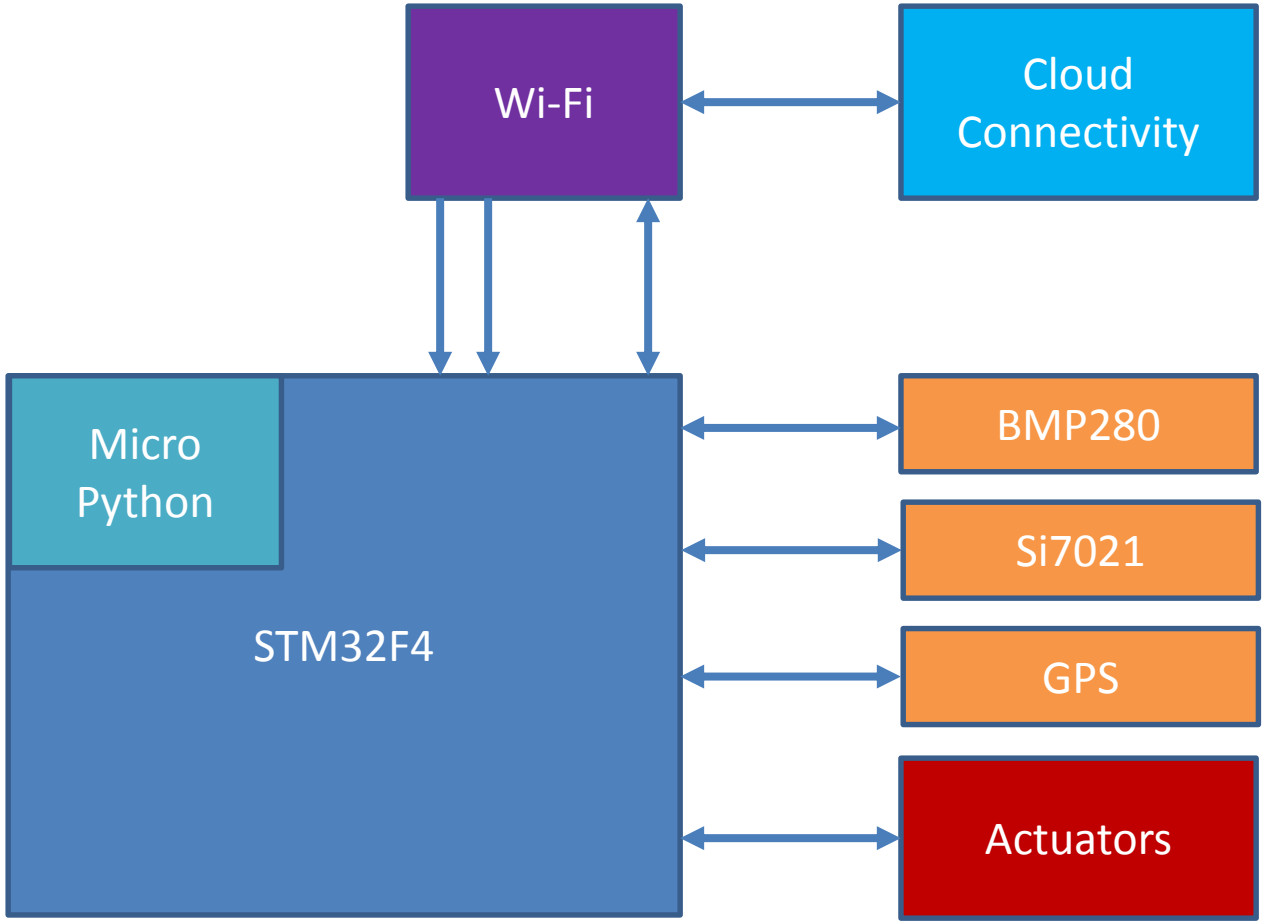
Updating the ESP8266 Remotely



Kernel Updates

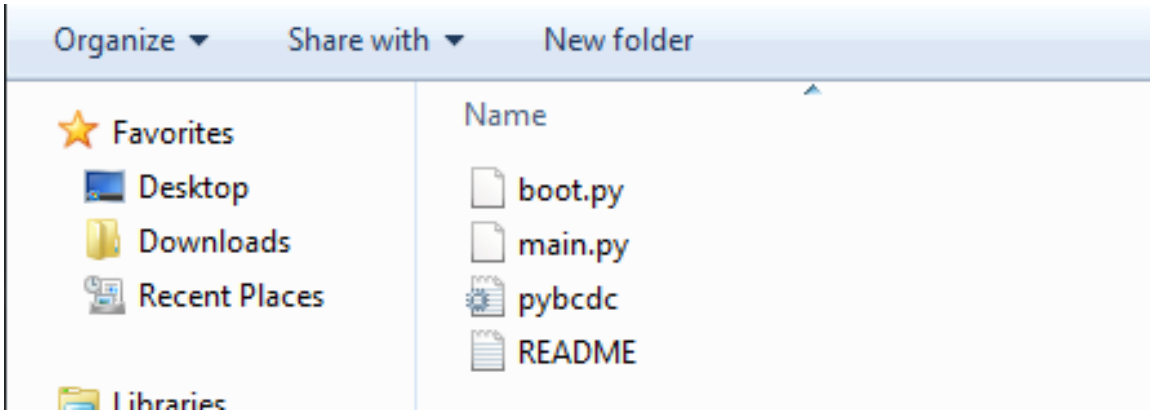


Kernel Updates - Alternative



Firmware Updates

- Two different methods
  - Copy and paste updated scripts
  - Use rshell to copy



Security

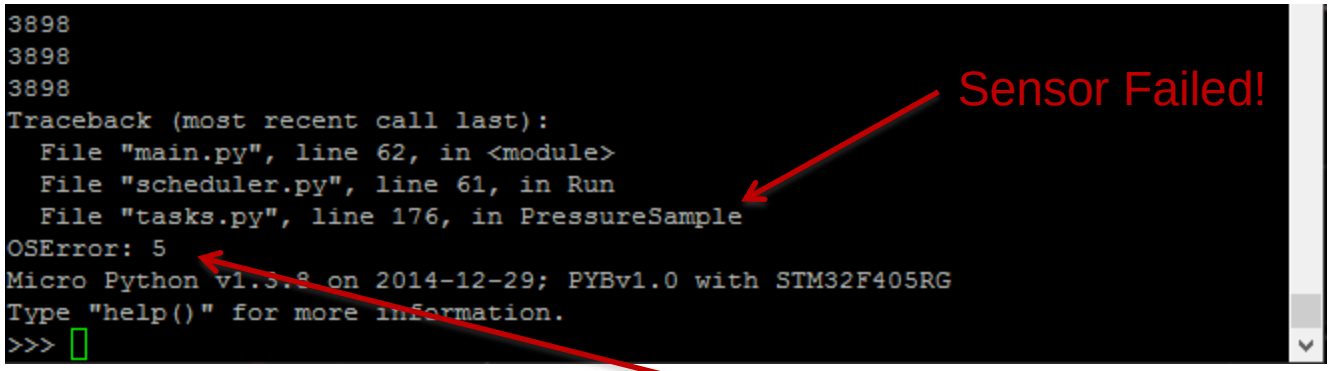
More Micro Python – Managing Time

```
import time

time.sleep(1)              # sleep for 1 second
time.sleep_ms(500)         # sleep for 500 milliseconds
time.sleep_us(10)          # sleep for 10 microseconds

start = time.ticks_ms()  # get millisecond counter
delta = time.ticks_diff(time.ticks_ms(), start)
                           # compute time difference
```

More Micro Python – Managing Time

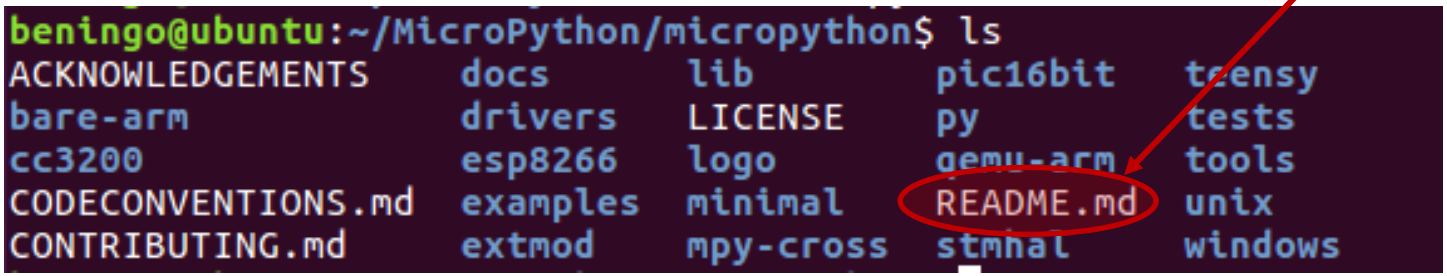
```
from machine import Timer

tim = Timer(-1)
tim.init(period=5000,
         mode=Timer.ONE_SHOT,
         callback=lambda t:print(1))
tim.init(period=2000,
         mode=Timer.PERIODIC,
         callback=lambda t:print(2))
```

More Micro Python - Debugging




```
# tasks.py
def PressureSample():
    # Read the previous conversion
    RxData = bytearray(4)

    try:
        # Read the last conversion from the sensor
        RxData = I2C2.mem_read(2, int(BMP180_Address[0]), 0xF6)

        # Start next conversion
        I2C2.mem_write(0xE0, int(BMP180_Address[0]),0xF4)
    except OSError as er:
        print("Received Exception OSError: " + str(er))
```

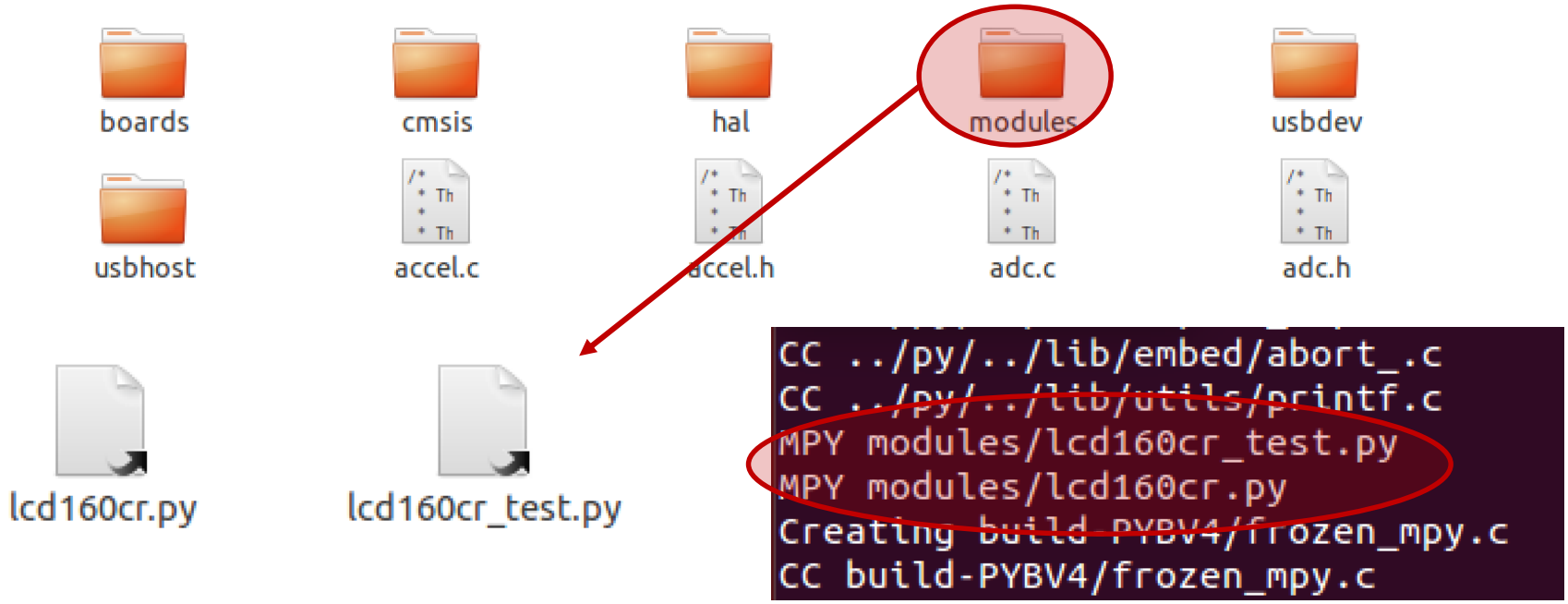
Customize the Kernel



Folder	Purpose
Bare-arm	Minimal version for ARM MCU's
Teensy	uP version for Teensy 3.1
Pic16bit	uP for 16 bit Microchip parts
Cc3200	uP for CC3200 from TI
Esp8266	uP for esp8266 wifi module
Py	Core Python implementation, compiler, runtime, etc
Stmhal	uP for STM32F405RG using St's HAL

Frozen Modules

- Frozen modules are pre-compiled Python libraries that can be built into the kernel



Frozen Modules

```
>>> import lcd160cr
>>> lcd160cr.myfunction()
```

- Any extra functionality can be built into the kernel using Frozen modules

Conclusions

- ESP8266 is an interesting option for creating an IoT device
- The module with Micro Python is marginally stable and operates inconsistently
- Firmware update mechanism has many potential security vulnerabilities
- Great for rapid prototyping and proof of concept

Additional Resources

- Download Course Material for
  - Python Doxygen Templates
  - Example source code
  - Blog
  - YouTube Videos
- Embedded Bytes Newsletter
  - http://bit.ly/1BAHYXm

From www.beningo.com under

- Blog > CEC – Designing IoT Sensor Nodes using the ESP8266

The Lecturer – Jacob Beningo

Jacob Beningo

Principal Consultant

Social Media / Contact

- E : jacob@beningo.com
- T : 810-844-1522
- : Jacob_Beningo
- : Beningo Engineering
- : JacobBeningo
- EDN : Embedded Basics
- ARM Connected Community

Consulting

- Advising
- Coaching
- Content
- Consulting
- Training

BENINGO EMBEDDED GROUP

www.beningo.com