

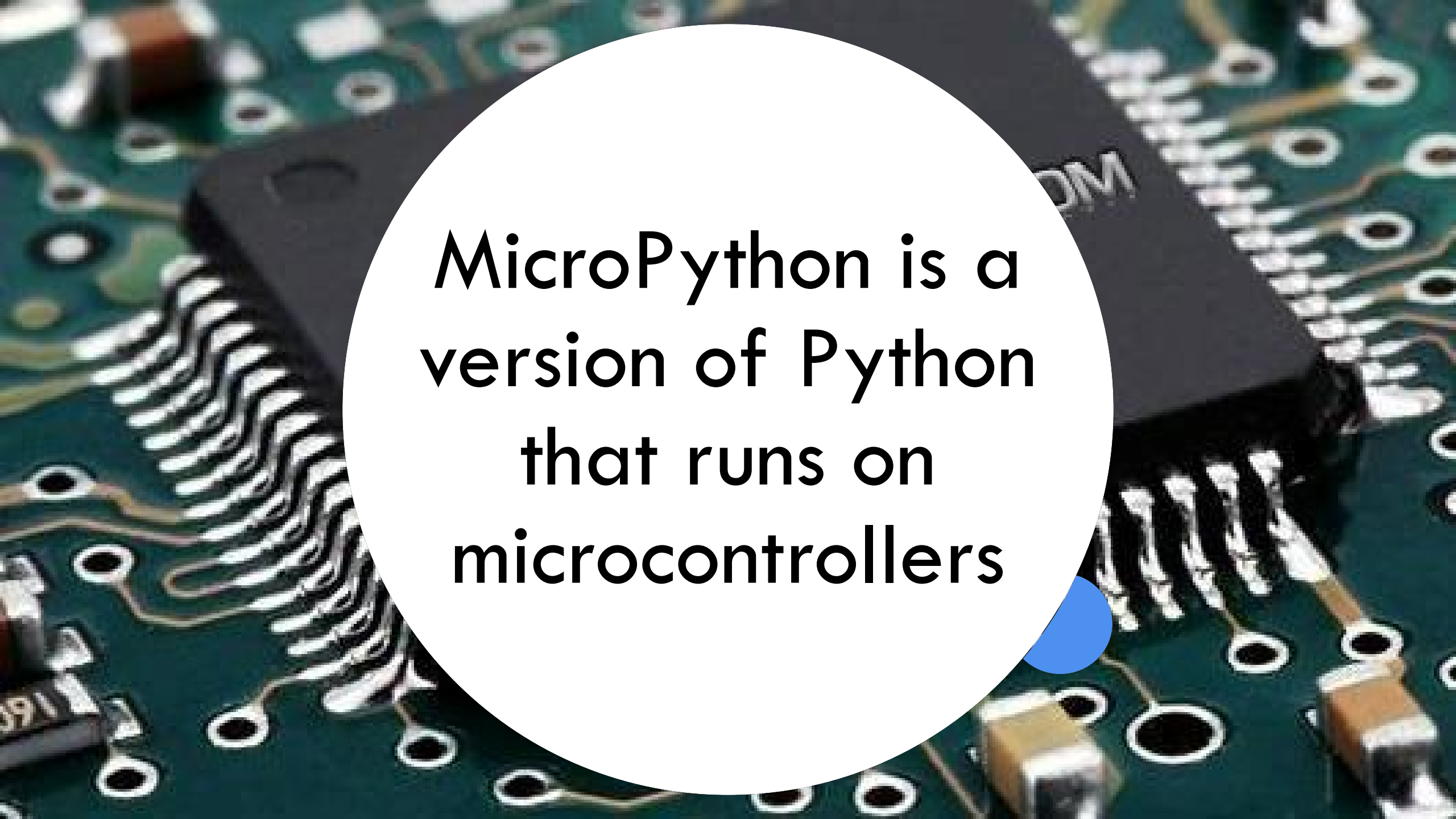


Integrating Computation
with MicroPython and
Sensors
John Liu
Saint Cloud State
University



Agenda

- What is MicroPython?
- Platforms running MicroPython
- Simple examples

A close-up photograph of a green printed circuit board (PCB) with various electronic components. A large white circle is centered over the image, containing the text 'MicroPython is a version of Python that runs on microcontrollers'. A small blue circle is partially visible at the bottom right edge of the white circle. The background shows intricate traces, solder joints, and components like a microcontroller chip and a connector.

**MicroPython is a
version of Python
that runs on
microcontrollers**

Why MicroPython?

- Same ease-of-use as PC Python, but runs directly on a microcontroller
- Interact directly with sensors
- Control motors, lights, displays etc.
- Construct a lab or demo that works without constant need for IT support
- Perfect for hands-on learning experience when integrating computation in labs

Thonny IDE

- Syntax highlighting
- File browser and transfer between PC and board
- Execute code on PC or board
- Shell for REPL
- Variables list for debugging
- More features

```
25 from machine import I2C, Pin
26 from math import atan
27 from time import sleep
28 from Driver_LiUDr_LIS3DH import LIS3DH
29 import sh1107
30
31 i2c=I2C(id=0,sda=Pin(12),scl=Pin(13),freq=400000)
32 sleep(0.01)
33 lis=LIS3DH(i2c)
34 display = sh1107.SH1107_I2C(128, 64, i2c, address=60, rotate=0)
35 display.poweron()
36 try:
37     while(True):
38         ret=lis.read_g() # ret is a tuple (ax,ay,az) so ret[0]=ax, ret[1]=ay, ar
39         if (ret[0]**2+ret[1]**2+ret[2]**2)**.5<0.25: # In case the set up is drc
40             display.fill(1) # Fill with color
41             display.large_text('Free!',0,0,3,0) # (text,x,y,font size, color)
42             display.large_text('!Fall!',0,32,3,0)
43             sleep(2) # Otherwise people can't see the printout!
44         else:
45             if ret[0] > 0.05:
46                 display.fill(0) # Clear display
```

MPY: soft reboot
SH1107: framebuffer is extended
ax= 0.05g ay= 0.00g az= 0.97g tilt= -3.2deg
ax= 0.06g ay= 0.01g az= 0.97g tilt= -3.5deg
ax= 0.06g ay= 0.01g az= 0.97g tilt= -3.3deg
ax= 0.06g ay= 0.00g az= 0.97g tilt= -3.5deg
ax= 0.05g ay= 0.00g az= 0.97g tilt= -3.2deg
ax= 0.06g ay= 0.00g az= 0.97g tilt= -3.4deg
ax= 0.05g ay= 0.00g az= 0.96g tilt= -3.2deg
ax= 0.06g ay=-0.01g az= 0.98g tilt= -3.3deg
ax= 0.05g ay= 0.01g az= 0.96g tilt= -3.2deg
Exiting loop...
MicroPython v1.23.0 on 2024-06-02; Adafruit Feather RP2040 with RP2040
Type "help()" for more information.
>>>



Hardware platforms running MicroPython

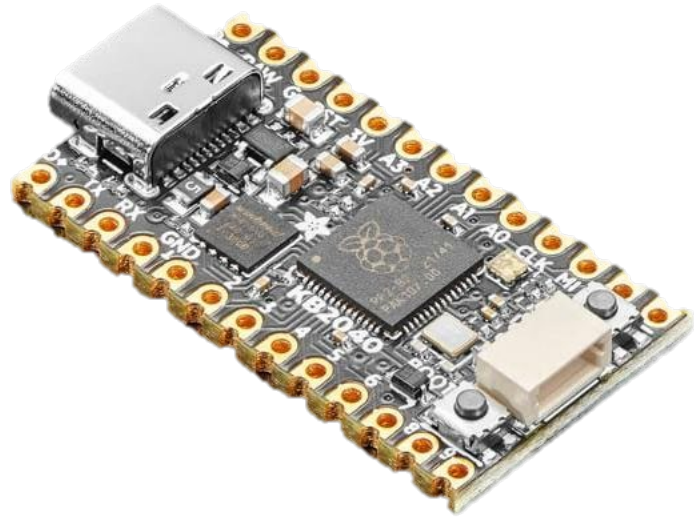
Many and more each year



Hardware

- Raspberry Pi Pico/Pico W/Pico 2-based boards
- ESP32/ESP32-S3/ESP32-Pico-based boards
- IMXRT-based boards
- Other boards or platforms such as Lego SPIKE, Vex IQ V2, based on nRF52840, ATSAM21, ATSAM51, STM32 etc.
- 100+MHz processor
- 256KB+ RAM (variables), 2MB+ FLASH (code)





RP2040

Dual 133MHz CPU

260KB RAM

2-8MB FLASH

- Adafruit KB2040 260KB/8MB
- Adafruit QT Pi 2040 260KB/8MB
- Adafruit Feather 2040 260KB/8MB
- Raspberry Pi Pico 260KB/2MB
- Raspberry Pi Pico W 260KB/2MB+WiFi

ESP32

Dual 240MHz CPU

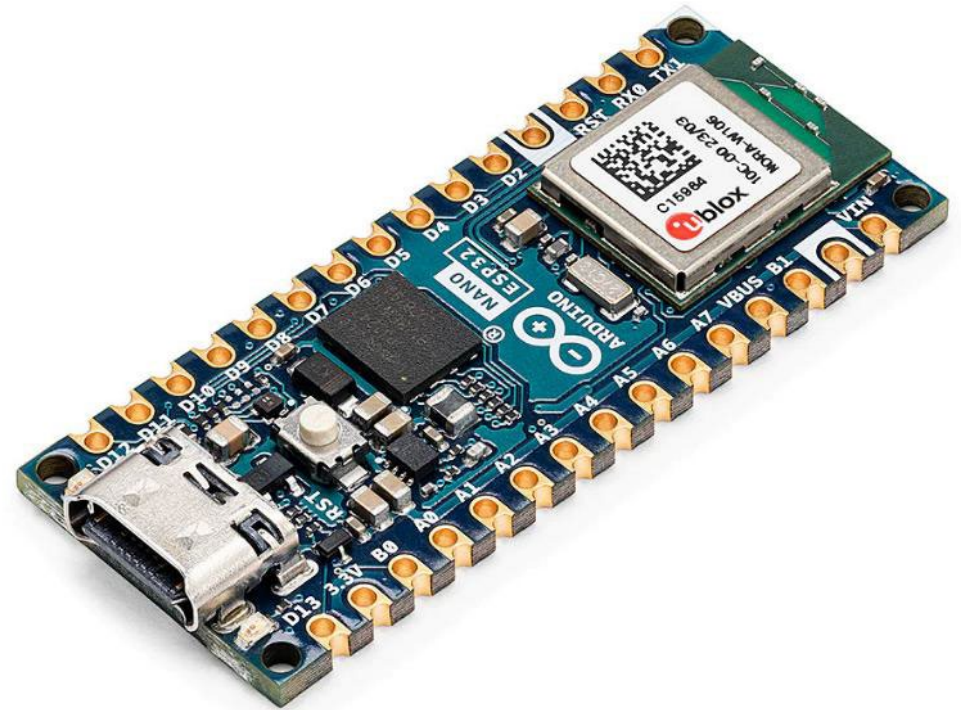
520KB RAM

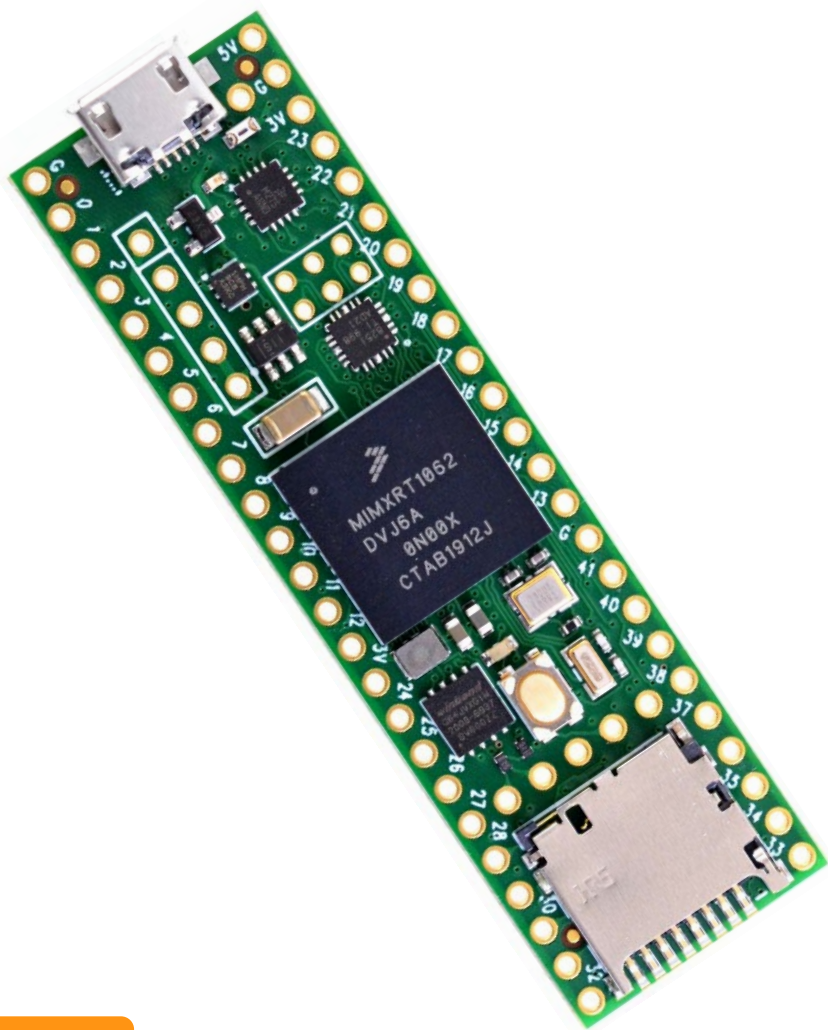
0-8MB PSRAM

4-16MB FLASH

WiFi/Bluetooth

- Adafruit Feather ESP32-V2 2MB/8MB
- Adafruit Qt Pi ESP32 Pico 2MB/8MB
- Arduino Nano ESP32 8MB/16MB
- Espressif ESP32 PICO-KIT-1 520KB/4MB
- TinyPICO 4MB/4MB





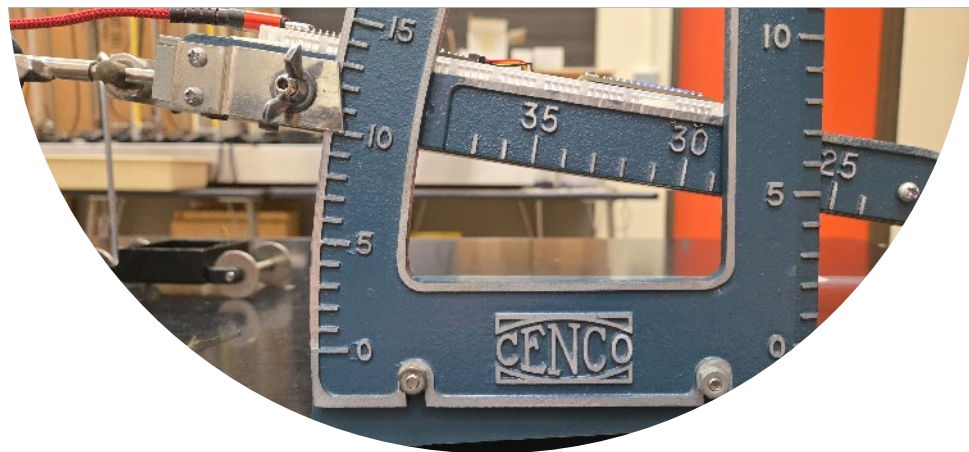
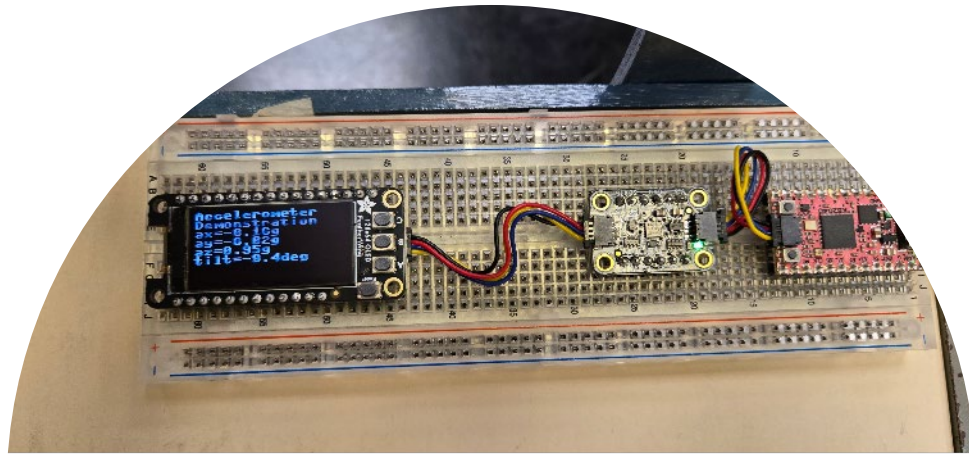
IMXRT

500-600MHz CPU

128KB-1MB RAM

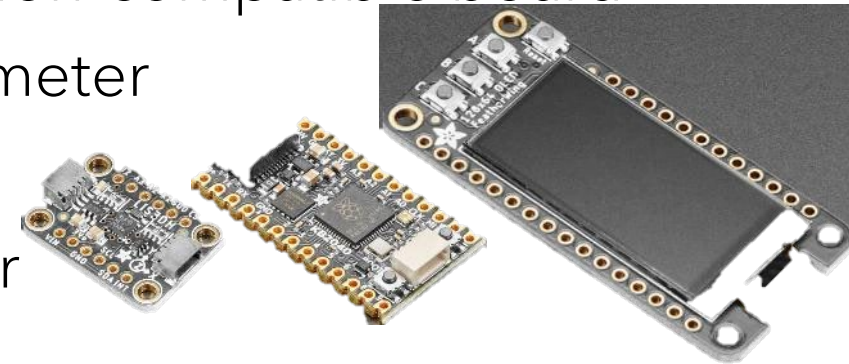
2-8MB FLASH

- Adafruit Metro M7 500MHz/28KB/8MB
- PJRC Teensy 4.0 600MHz/1MB/2MB
- PJRC Teensy 4.1 600MHz/1MB/8MB



Making a digital protractor with an accelerometer

- MicroPython-compatible board
- Accelerometer
- Display
- Computer



Code

```
from math import atan2
from time import sleep
from machine import I2C, Pin
from Driver_LiuDr_LIS3DH import LIS3DH
import sh1107
i2c=I2C(id=0,sda=Pin(12),scl=Pin(13),freq=400000)
sleep(0.01)
lis=LIS3DH(i2c)
display = sh1107.SH1107_I2C(128, 64, i2c, address=60, rotate=0)
display.poweron()
while(True):
    ret=lis.read_g() # ret is a tuple (ax,ay,az)
    tilt_deg=atan2(ret[0],ret[2])*57.3 # Calculate tilt angle
    outStr=f'ax={ret[0]:5.2f}g ay={ret[1]:5.2f}g az={ret[2]:5.2f}g tilt={-tilt_deg :6.1f}deg'
    print(outStr)
    display.fill(0)
    display.large_text('{:3d}'.format(-int(tilt_deg+0.5)),0,8,4,1)
    display.large_text('Deg',40,48,2)
    display.show()
    sleep(0.1)
```



Thank you

John Liu

St. Cloud State University

Dept. Mathematics and Physics

zliu@stcloudstate.edu

Code and resources:

<https://github.com/opensourcephysicslab/CompPhysMP-lib>