ASSURE REU 2024

Electronics Workshop

Thanh Tran
Chris Moeckel

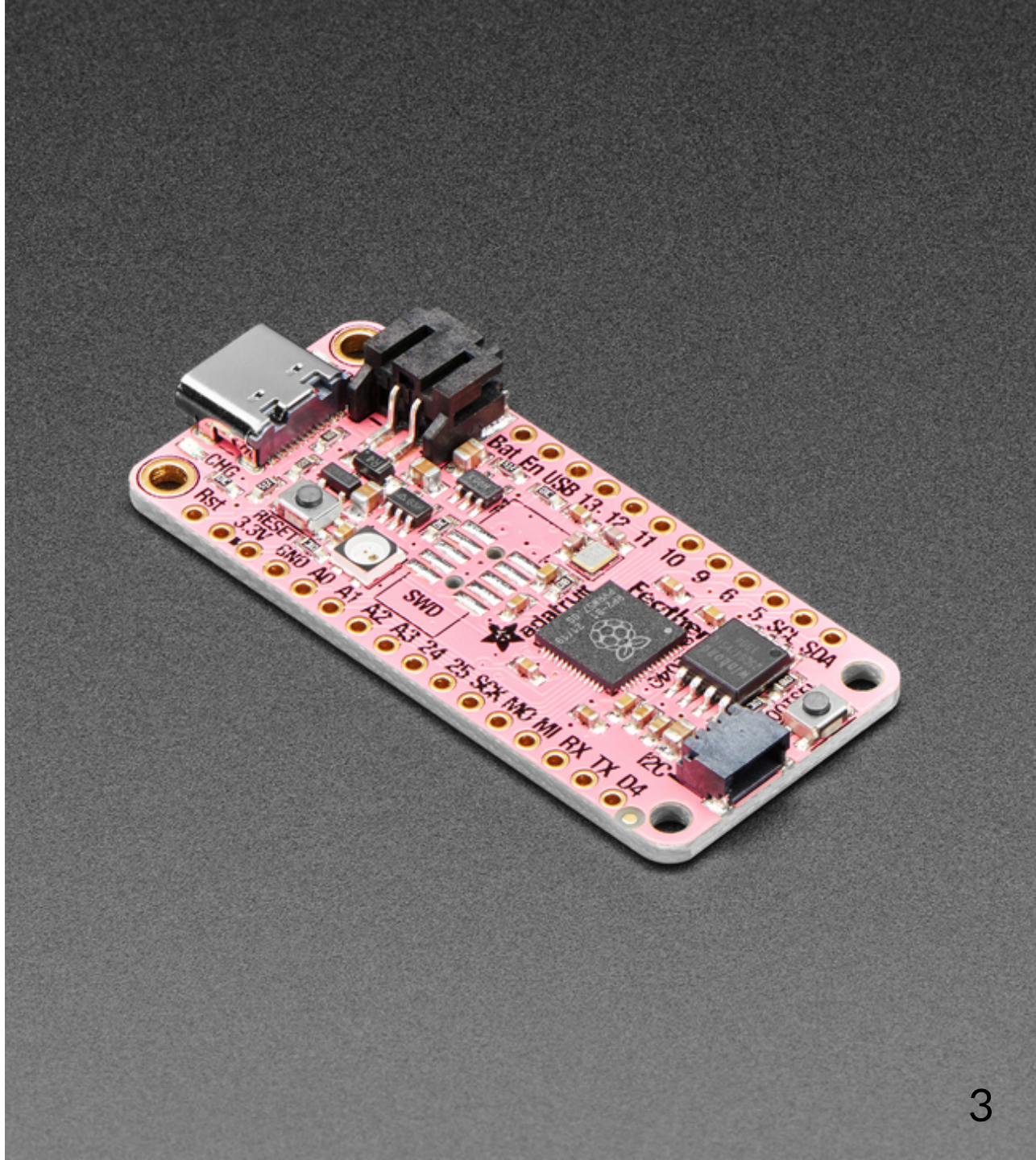
2024/06/05
Version 1

Overview

- Electronics for space science
- Balloon projects (COSI)
- What is a microcontroller
- How do we **debug** electronics
- Using microcontroller
- Making our own kite experiment

Microcontroller

- RP2040 microcontroller from Adafruit
- Programming it with CircuitPython (We will teach you)
- One task at a time looping forever
- Interacts with the world easily



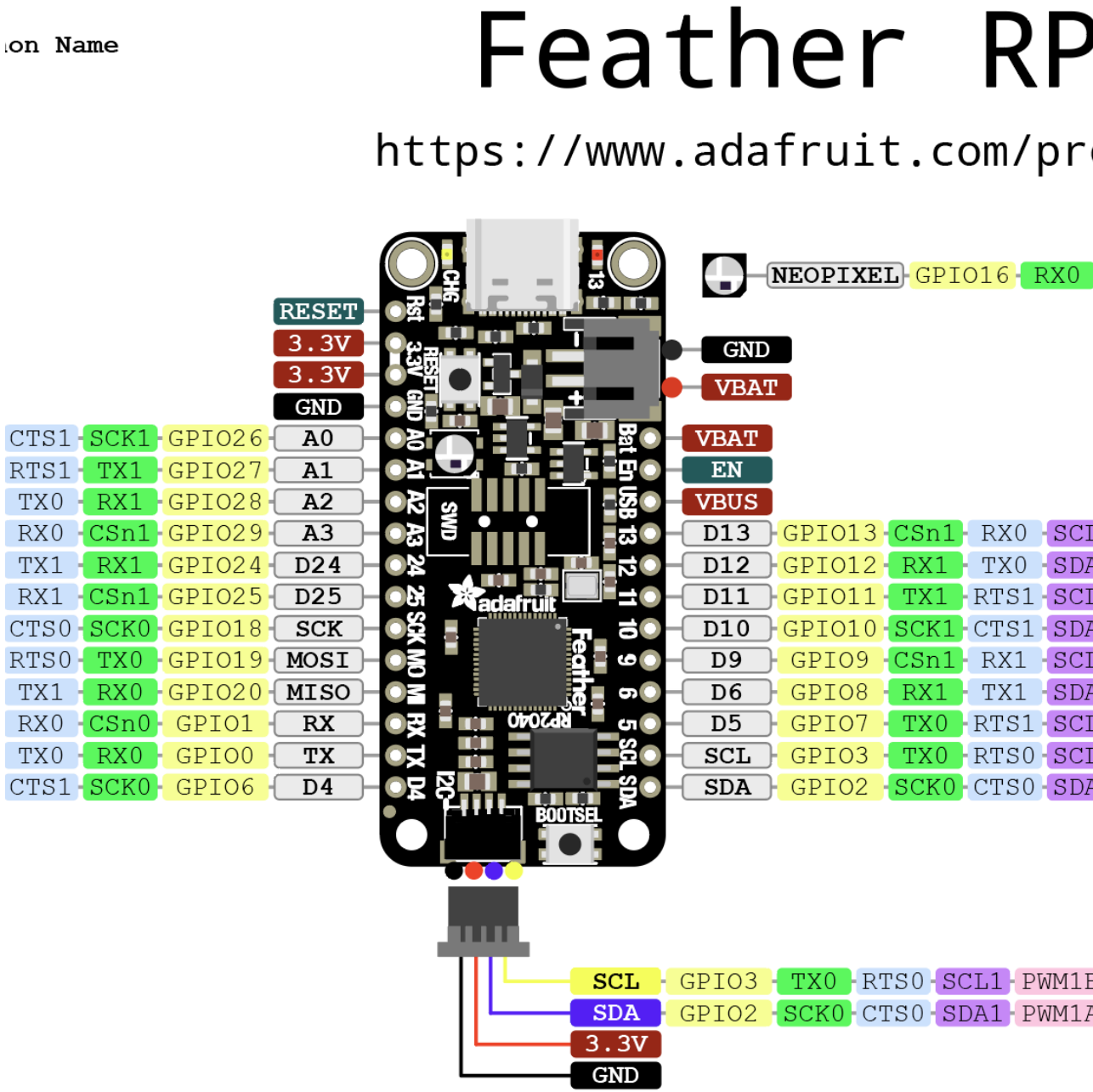
Tools for electronics debugging

How to tell what is happening at this moment? Digital multimeter!

- Black probe is ground (GND), reference point for voltage
- Red probe is measurement

Using digital multimeter

- Black probe always touches the GND pin on the top left
- Try touching red probe to 3.3V, VBAT/BAT, and VBUS/USB
- What voltages do you measure for each?



Programming the microcontroller

Using the wonderful tutorials from Adafruit:

- Install the Mu editor (Listening to the code)
- Write "Hello World" equivalent: Blinky
- Talk to the board with the serial console

Code Structure (Python)

Library

(Code that other people wrote for you)

```
# Write your code here :-)
import board
import digitalio
import time

led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT

while True:
    led.value = True
    time.sleep(2)
    led.value = False
    time.sleep(0.5)
```

Code Structure (Python)

Library

(Code that other people wrote for you)

Variables

(Assign values to words)

```
# Write your code here :-)
import board
import digitalio
import time

led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT

while True:
    led.value = True
    time.sleep(2)
    led.value = False
    time.sleep(0.5)
```

Code Structure (Python)

Library

(Code that other people wrote for you)

Variables

(Assign values to words)

Loop

(Do this as long as conditions is met)

```
# Write your code here :-)
import board
import digitalio
import time

led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT

while True:
    led.value = True
    time.sleep(2)
    led.value = False
    time.sleep(0.5)
```

Files for the Experiment

- `code.py` includes main loop, mostly editing this
- `boot.py` runs on startup
- `lib` contains libraries for talking to sensors

Computer: Operating system allows you to run several processes at the same time

Microcontroller: Execute whatever code it will find on its memory

.uf2-file: bootloader files (USB Flashing Format)
.py-file: Scripts to execute (Python)

Sensors

- BME680 is a temperature, pressure, humidity, and gas sensor
- LSM6DSO32 is an accelerometer and gyrometer

Plug in both in a chain, then copy code from repository to microcontroller to get measurements from them printed out on the screen

Preparing the microcontroller

- Download all the relevant files from github
- Flash the system ('nuke it') using the bootloader
- Install the latest bootloader for microcontroller
- Test the system with Blinky (Copy the code and hit reset)
- Test the system with the measurement code
- Use Mu editor (Mode: circuit python) and listen to the code using the Serial button

Setting up the kite experiment!

- Flash the system and reset the bootloader
- Copy over the code (boot.py, code.py) and the library files
- Plug in the battery, unplug the USB cable, hit reset button
- Wave the accelerometers around for 10 seconds
- Connect Pin GND and Pin A0 to allow computer access
- Plug USB cable back in, open and plot the data in the measurement.csv file

Setting up the kite experiment!

- Copy measurement logging code to microprocessor, including `boot.py`
- Press reset button to reload `boot.py` so that code can use the flash
- Take data for a few seconds, then open serial monitor and remove `boot.py` with `import os; os.remove("boot.py")`
- Press reset button to return control to the computer
- Open `measurements.csv` and check that you see various values
- Delete `measurements.csv` and copy `boot.py` back onto the microprocessor

Assembly of kite experiment

- Disconnect microcontroller from computer, leave battery attached
- Kapton tape microcontroller, sensors, and battery into a bundle ("payload") with a loop for attaching string
- Tie knot in string through payload loop
- Bring payload and kite outside

Running kite experiment

- Countdown!
- Right before launch, press reset button
- Slowly release string to allow kite to ascend
- Mark down important times in seconds from pressing reset
  - Try pulling string up and down sharply
  - Hover at various altitudes
- Return to Earth
- Bring payload and kite back inside for analysis
- Dismantle payload

Data analysis

- Connect microcontroller to computer, open serial monitor
- Confirm the experiment worked as intendet
- Press reset button to return control to computer
- Copy `measurements.csv` to a safe place on your computer
- Disconnect microcontroller from computer, put aside
- Open jupyter notebook using Anaconda
- Make graphs of time vs. our various measurements
  - Experiment with various types of graph or combined measurements on one graph