

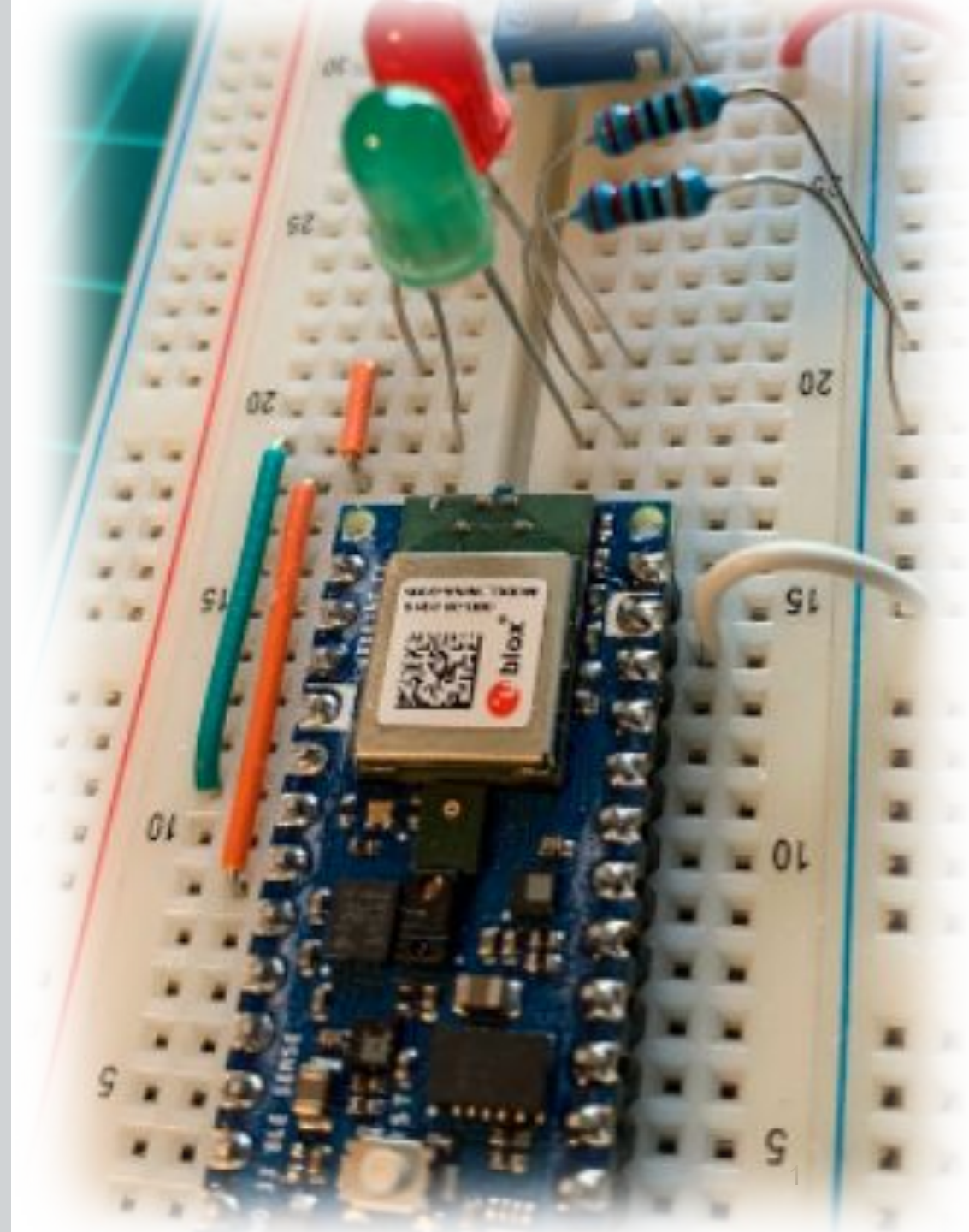
# IESTI01 – TinyML

## Embedded Machine Learning

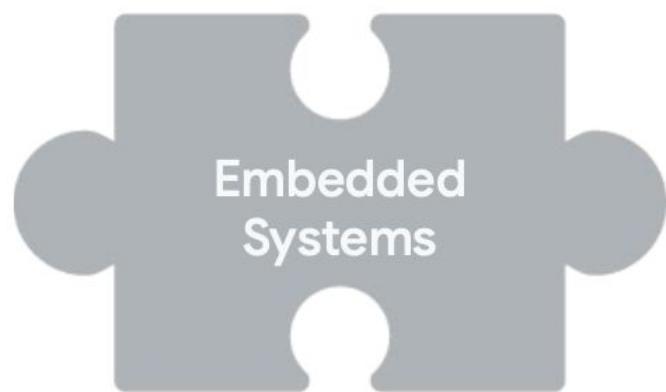
### 3. TinyML Challenges: - Embedded Systems



Prof. Marcelo Rovai  
UNIFEI



# What are the Challenges for TinyML?



**TinyML**

## **Software**

**Applications**

**Libraries**

**Operating System**

**Hardware**

## Software

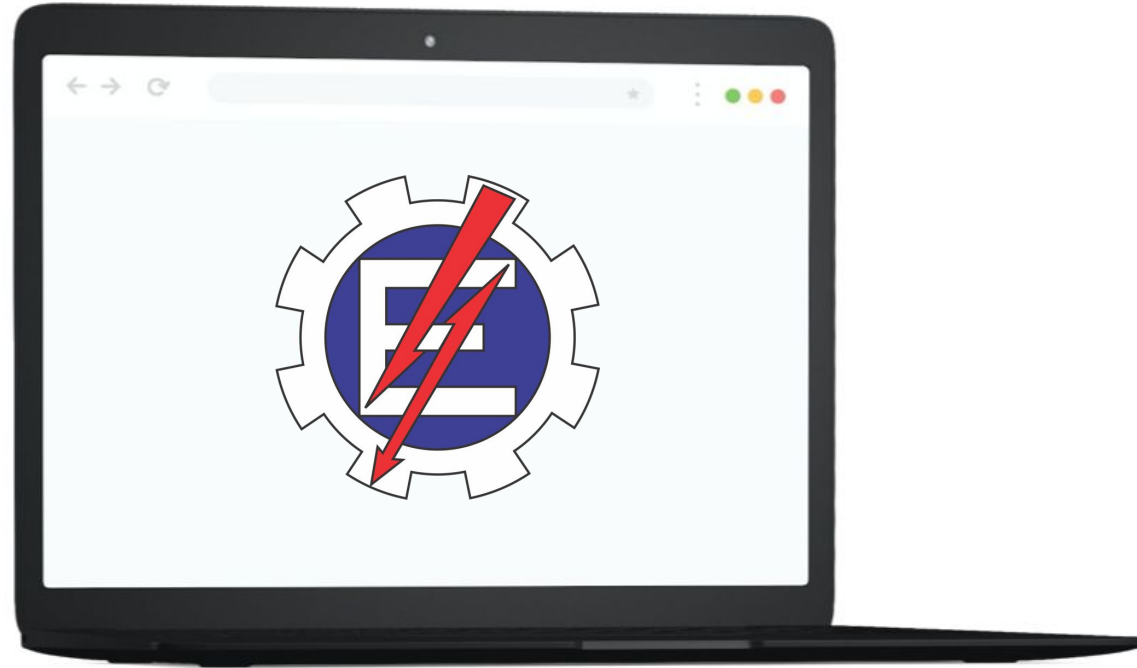
Applications

Libraries

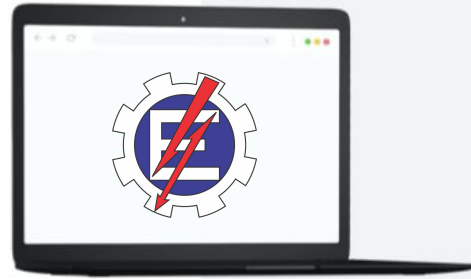
Operating System

Hardware

# Building Blocks of Computing Hardware



# Hardware



# Software

# Compute



# Memory



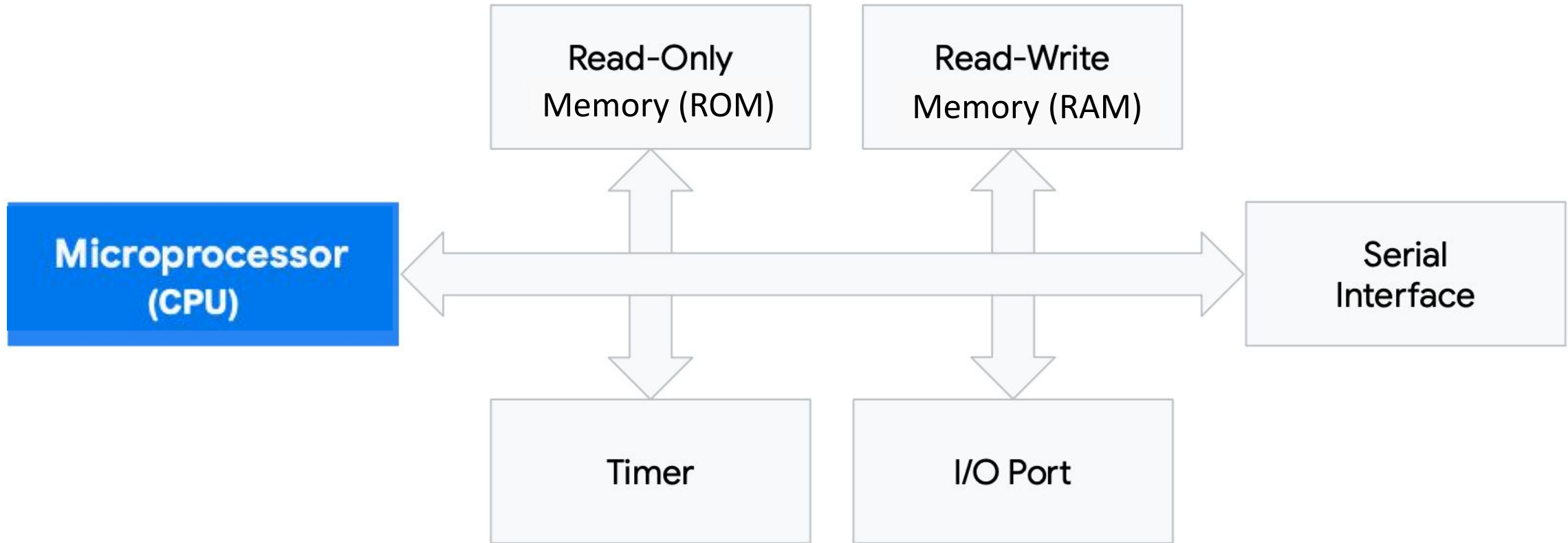
# Storage





# Microprocessor v. Microcontroller

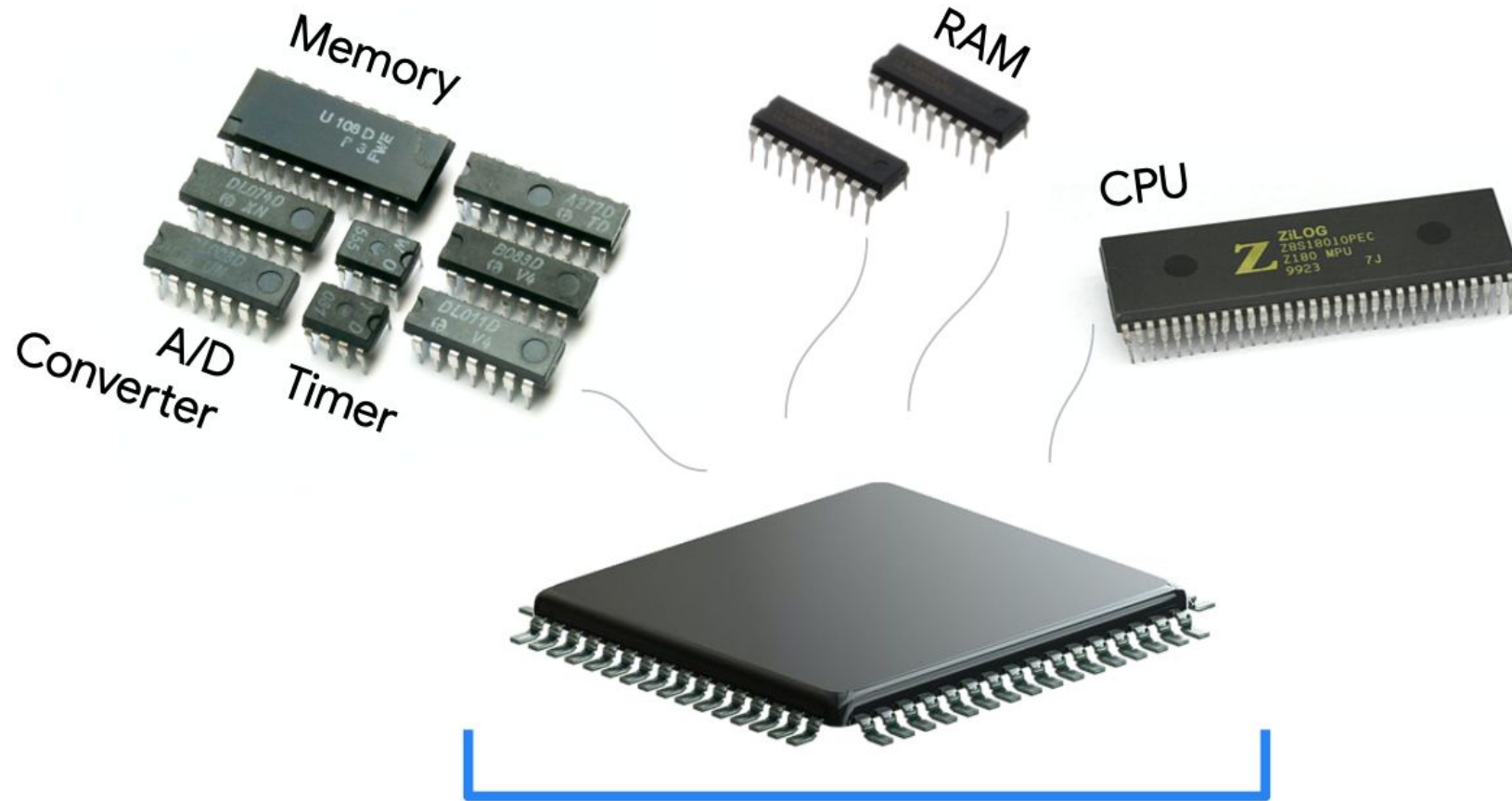
# Microprocessor: only **one part** of the puzzle



# Microcontroller

CPU	Read-Only Memory (ROM)	Read-Write Memory
Timer	I/O Port	Serial Interface

# Microcontroller: a **complete package**





# Microprocessor (CPU)

- Heart of a **computer system**
- Just the processor, memory and storage are **external**
- Mainly used in **general purpose systems** like laptops, desktops and servers
- **Offers flexibility** in design
- System size is **big**

# Microcontroller

- Heart of an **embedded system**
- Memory and storage are all **internal** to the system
- Mainly used in **specialized, fixed function systems** like phones, MP3 players, etc.
- **Limited flexibility** in design
- System size is **tiny**

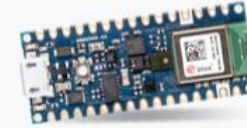
# Orders of Magnitude Difference

	Microprocessor	>	Microcontroller	
Platform				Nano
Compute	1GHz–4GHz	~10X	1MHz–400MHz	64MHz
Memory	512MB–64GB	~10000X	2KB–512KB	256KB
Storage	64GB–4TB	~100000X	32KB–2MB	1MB
Power	30W–100W	~1000X	150μW–23.5mW	

# Implications

- How complicated is the running task?
- How much memory does it need to have?
- How long does the job have to perform?

## Microcontroller



1MHz-400MHz

2KB - 512KB

32KB - 2MB

150 $\mu$ W-23.5mW



# Hardware



Anomaly Detection  
Sensor Classification  
20 KB

KeyWord Spotting  
Audio Classification  
50 KB

Image  
Classification  
250 KB+



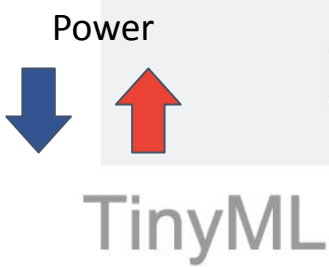
Rpi-Pico  
(Cortex-M0+)



Arduino Nano  
(Cortex-M4)



Arduino Pro  
(Cortex-M7)



EdgeML

Object Detection  
Complex Voice  
Processing  
1 MB+

Video  
Classification  
2 MB+



RaspberryPi  
(Cortex-A)



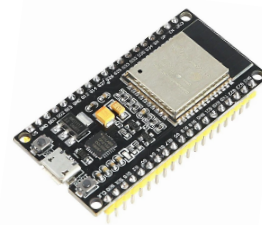
SmartPhone  
(Cortex-A)



Jetson Nano  
(Cortex-A + GPU)

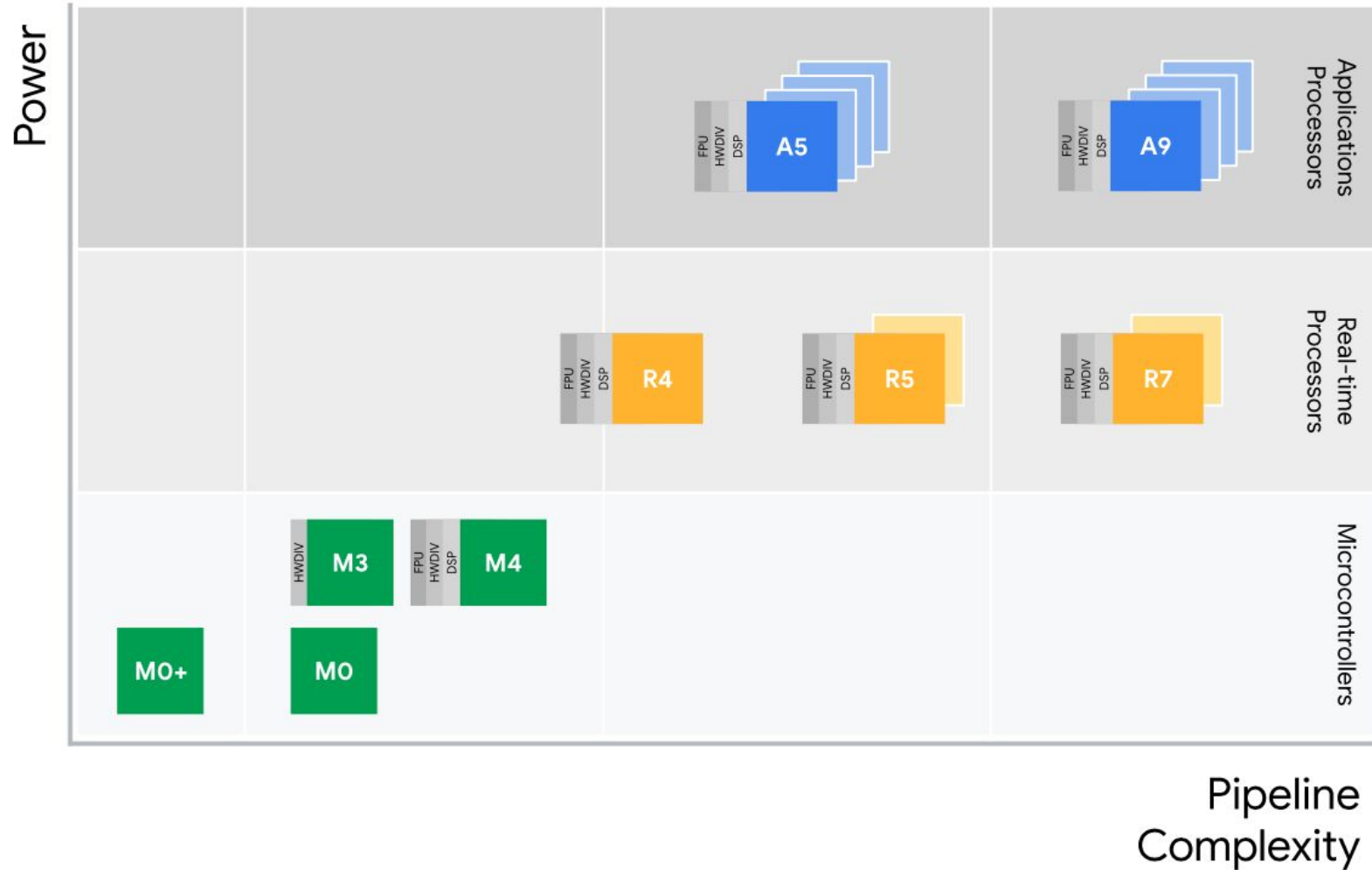


# Computing Hardware

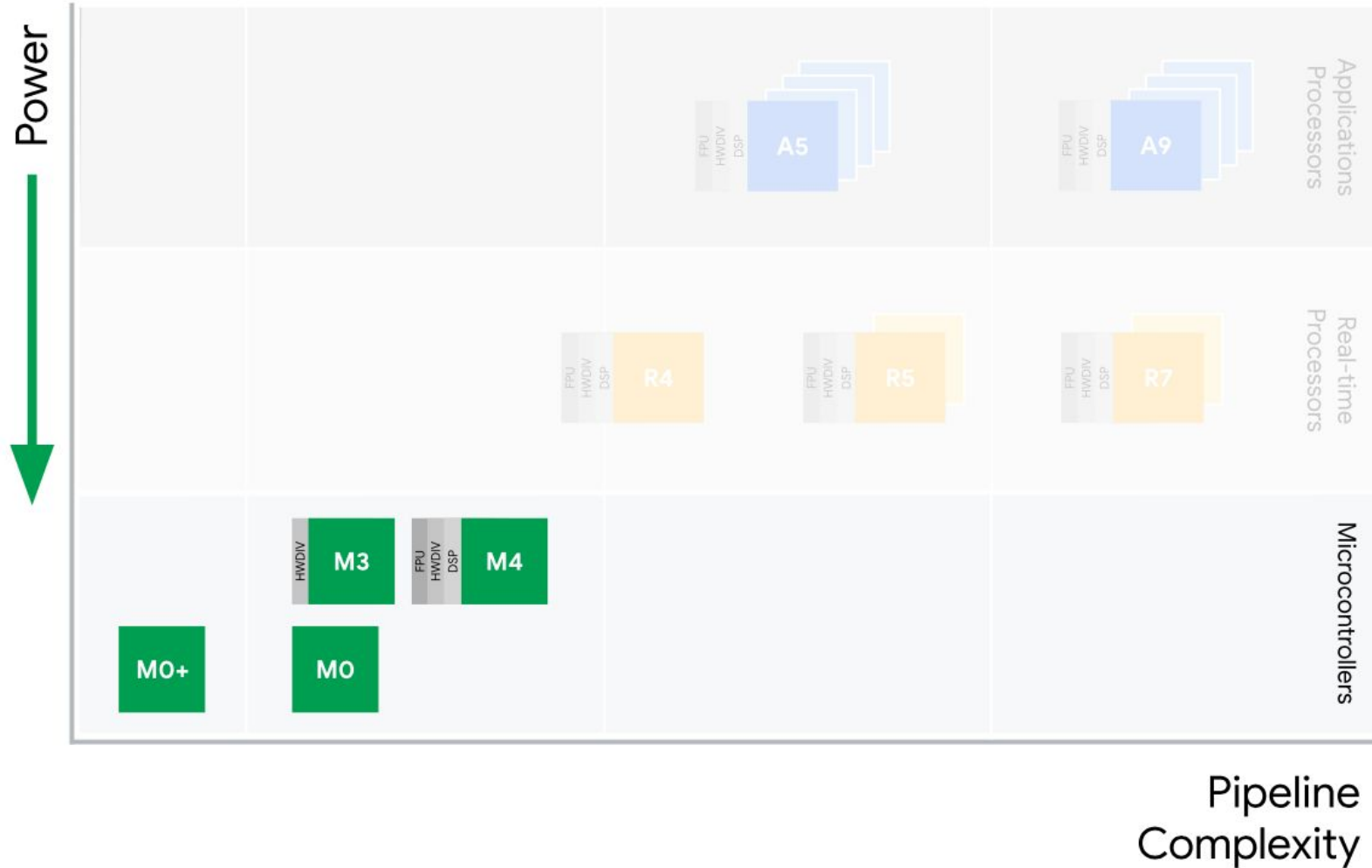


	Raspberry Pico	Arduino Nano Sense	ESP 32	Seeed Wio Terminal	Arduino Portenta
32Bits CPU	Dual-core Arm Cortex-M0+	Arm Cortex-M4F	Xtensa LX6 Dual Core	Arm Cortex-M4F	Dual Core Arm Cortex M7/M4
CLOCK	133MHz	64MHz	240MHz	120MHz	480/240MHz
RAM	264KB	256KB	520KB	192KB	1MB
ROM	2MB	1MB	2MB	4MB	2MB
Radio	---	BLE	BLE/WiFi	BLE/WiFi	BLE/WiFi
Sensors	No	Yes	No	Yes	No

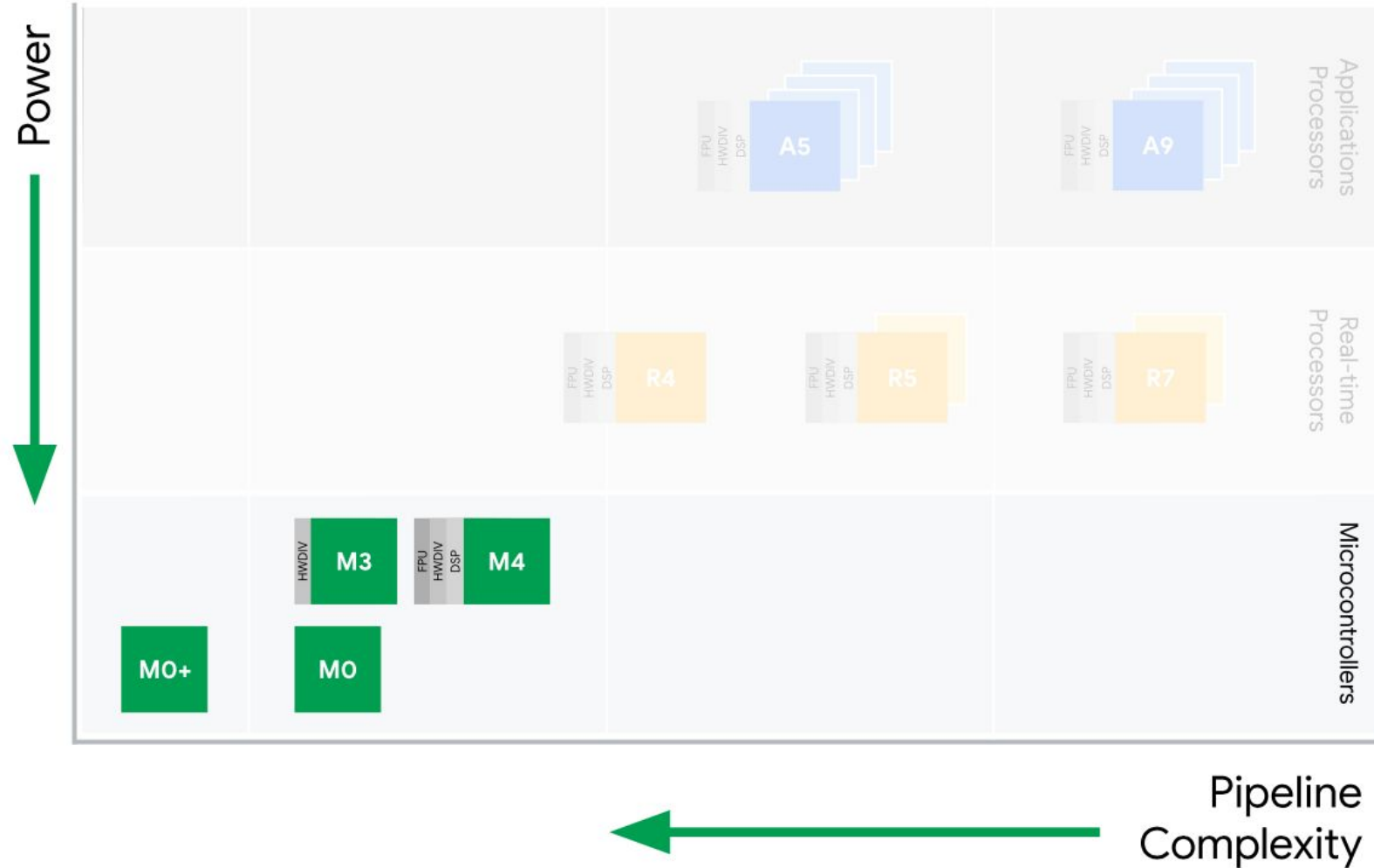
# ARM Cortex Processor Profiles



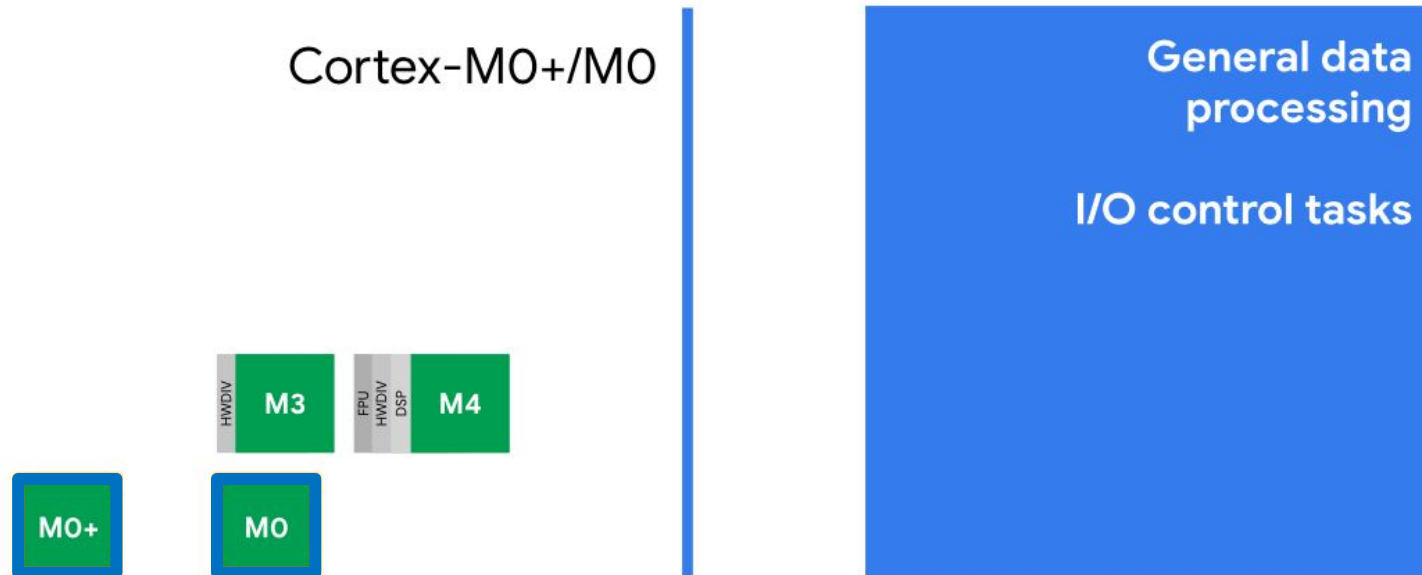
# ARM Cortex Profiles



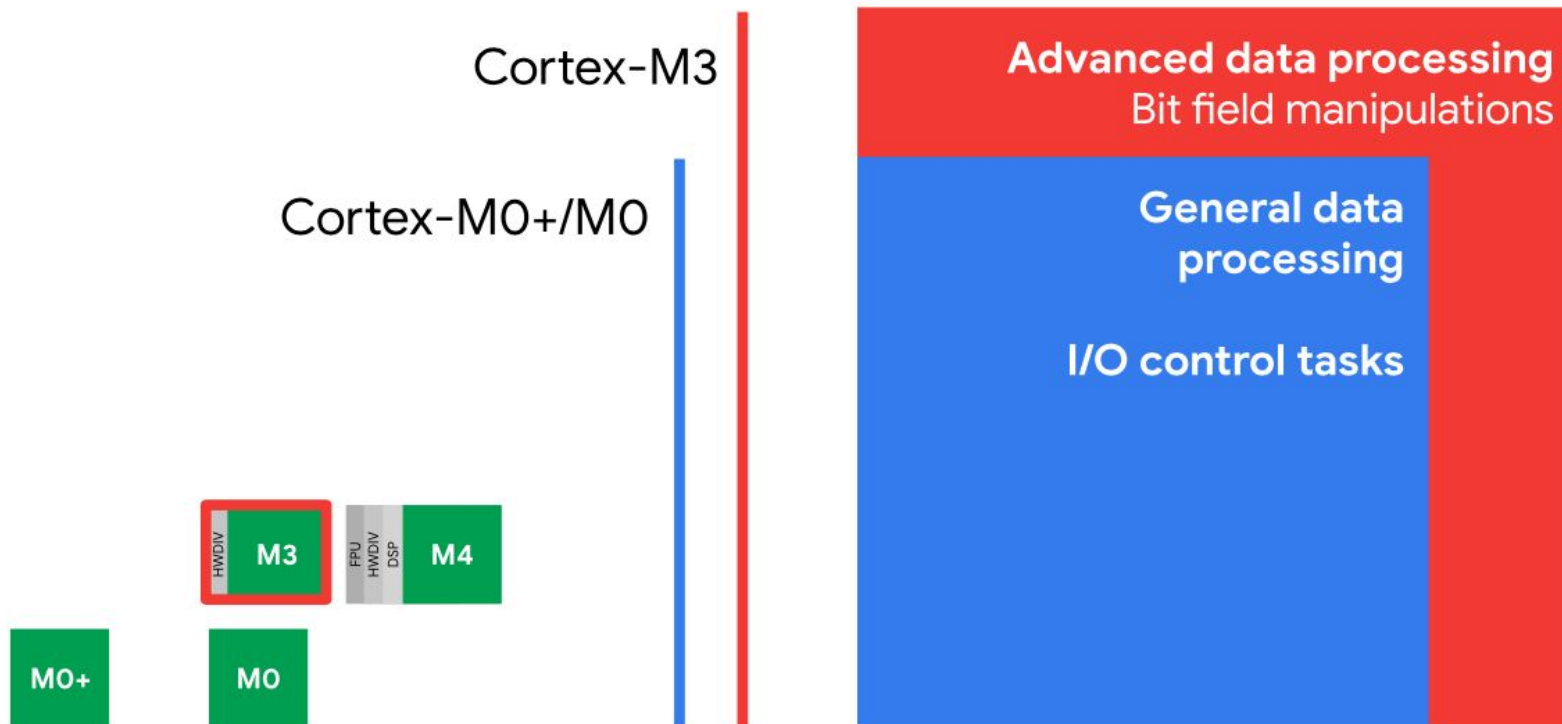
# ARM Cortex Profiles



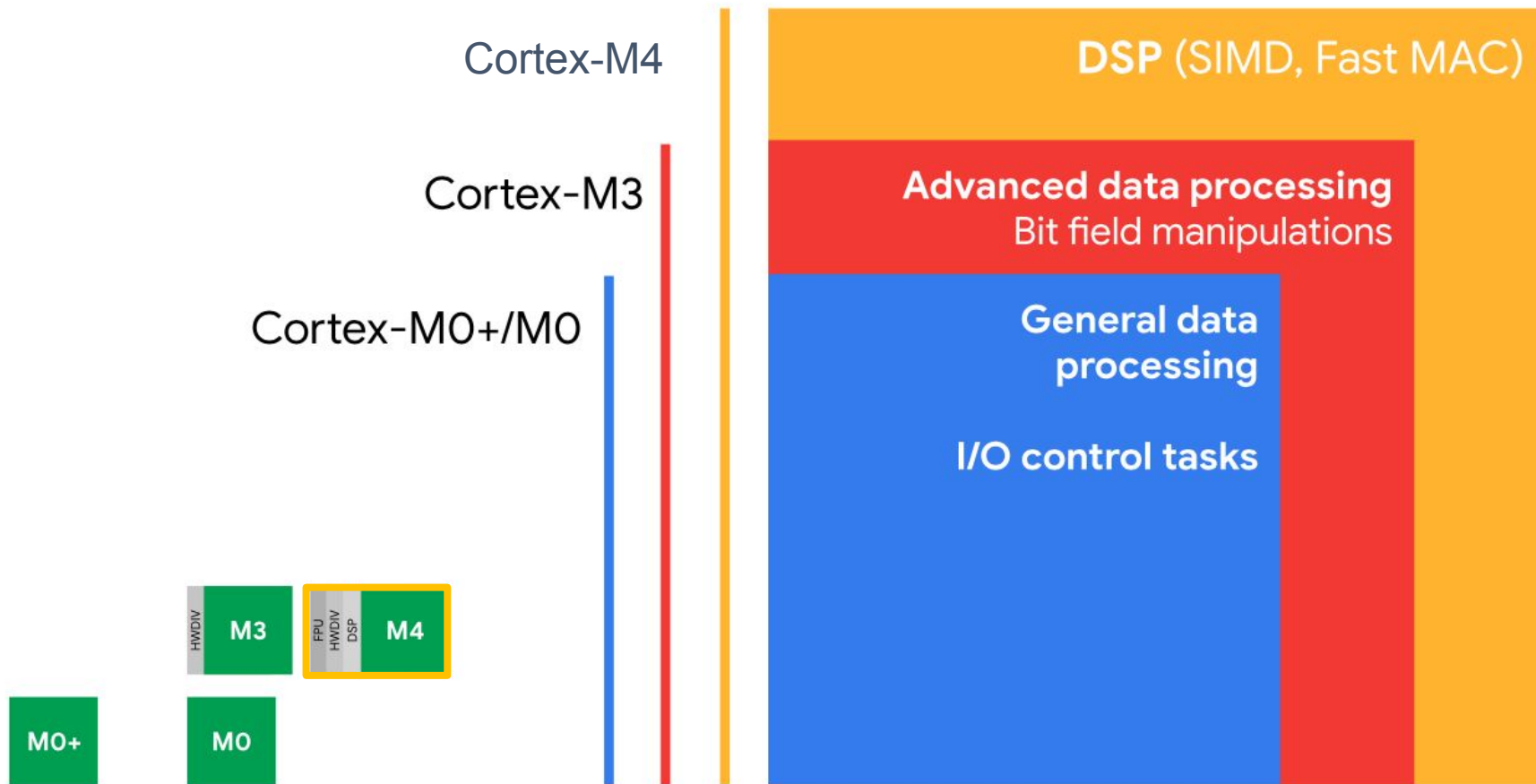
# ARM Cortex-M ISA



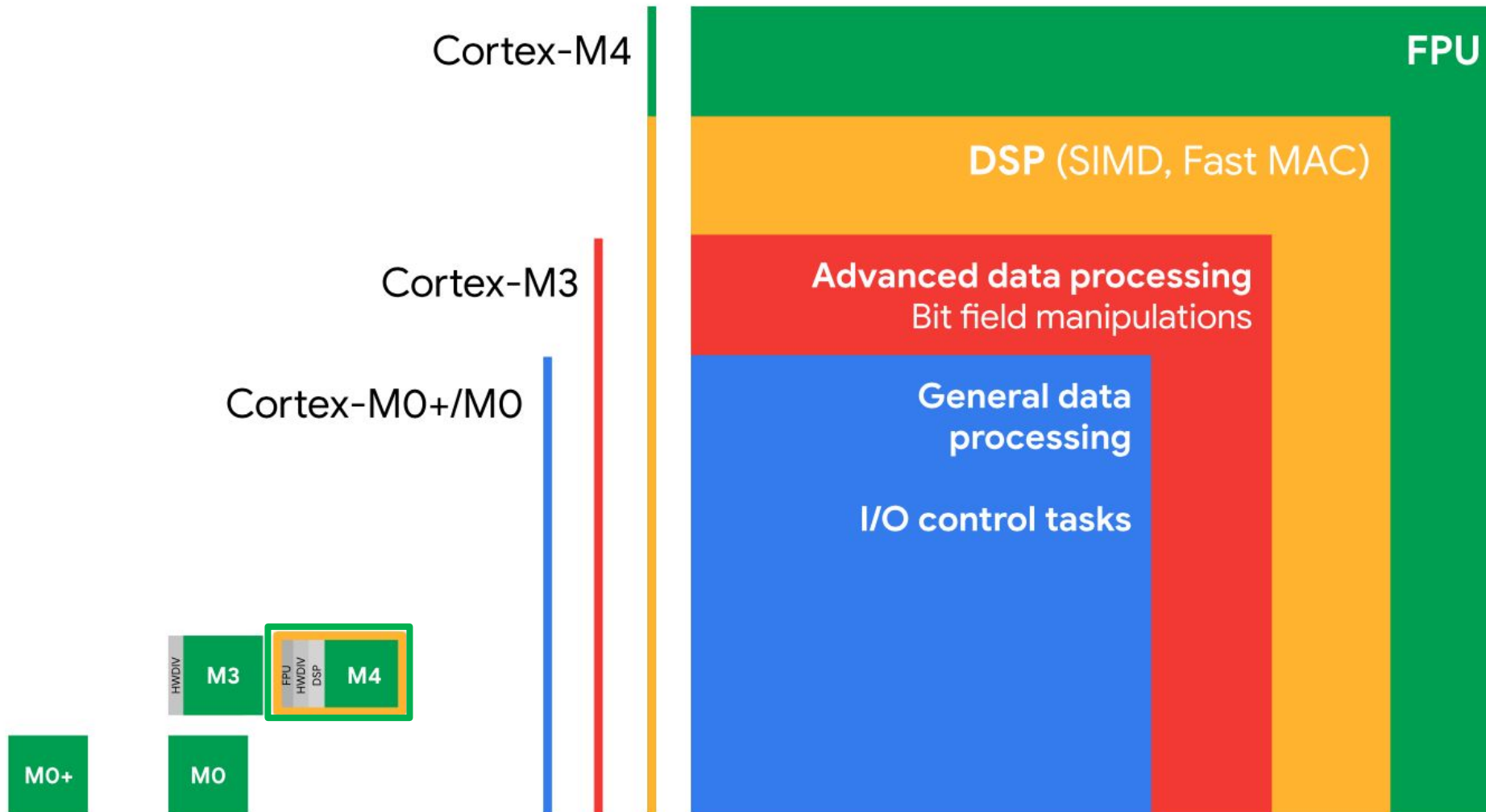
# ARM Cortex-M ISA



# ARM Cortex-M ISA



# ARM Cortex-M ISA





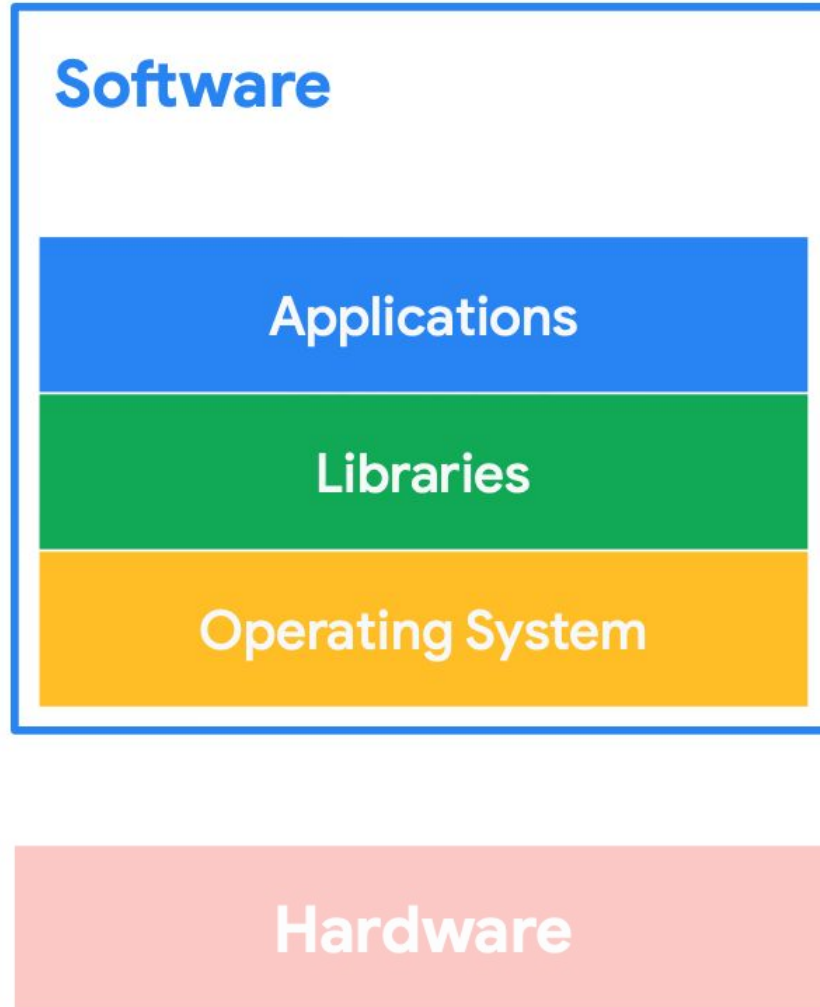
## Software

Applications

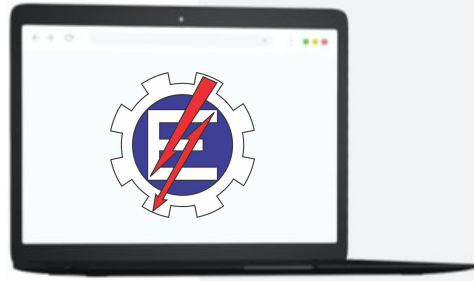
Libraries

Operating System

Hardware



Hardware



**Software**

## Software

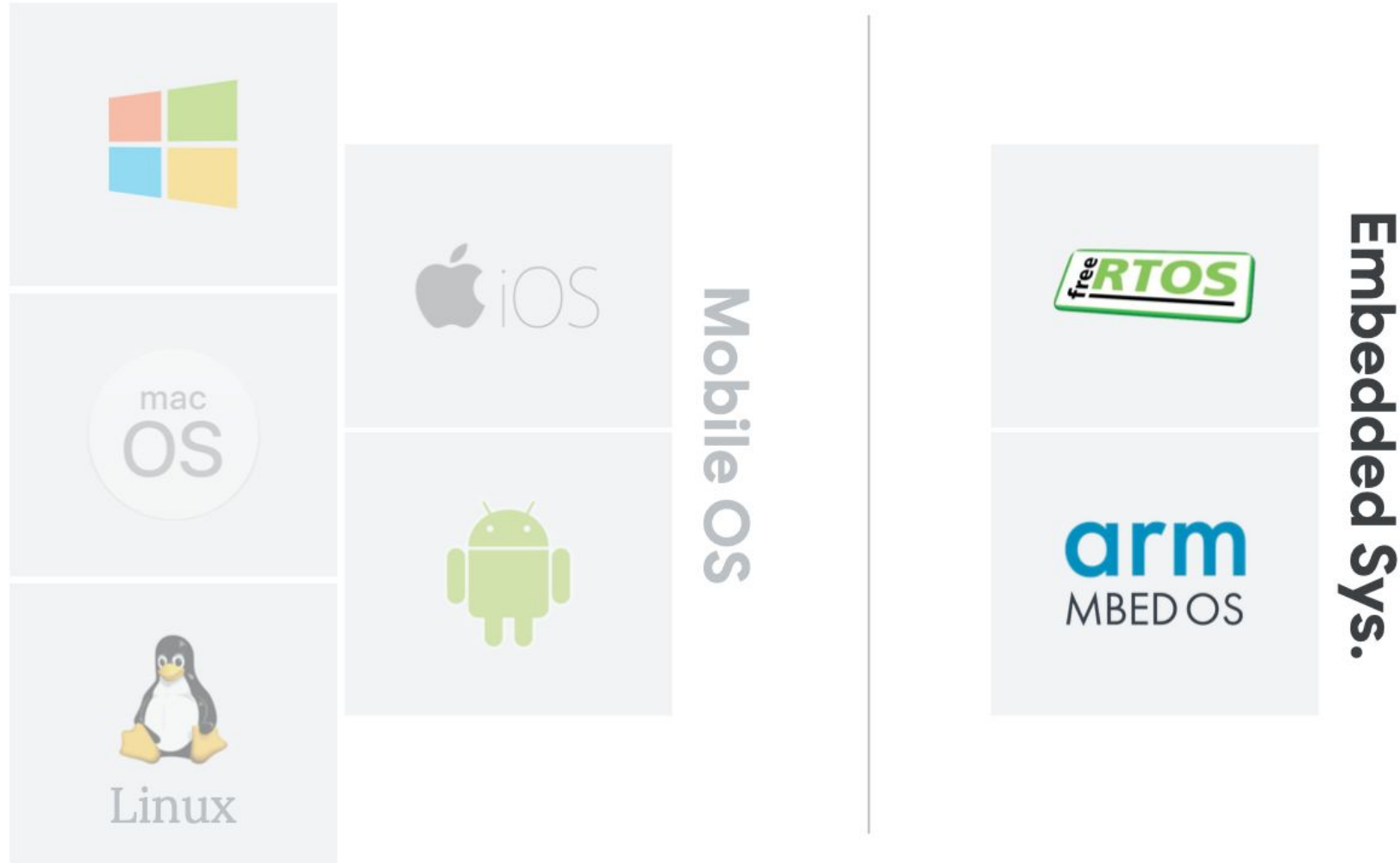
Applications

Libraries

Operating System

Hardware

# Widely Used Operating Systems



## Software

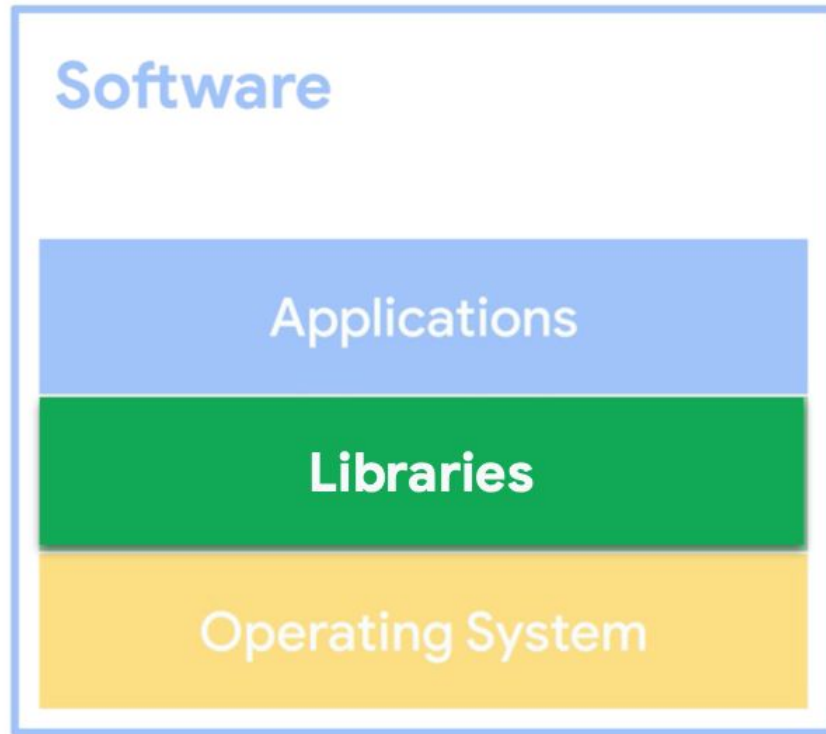
```
graph TD; subgraph Software; Applications; Libraries; OS[Operating System]; end; Hardware;
```

Applications

Libraries

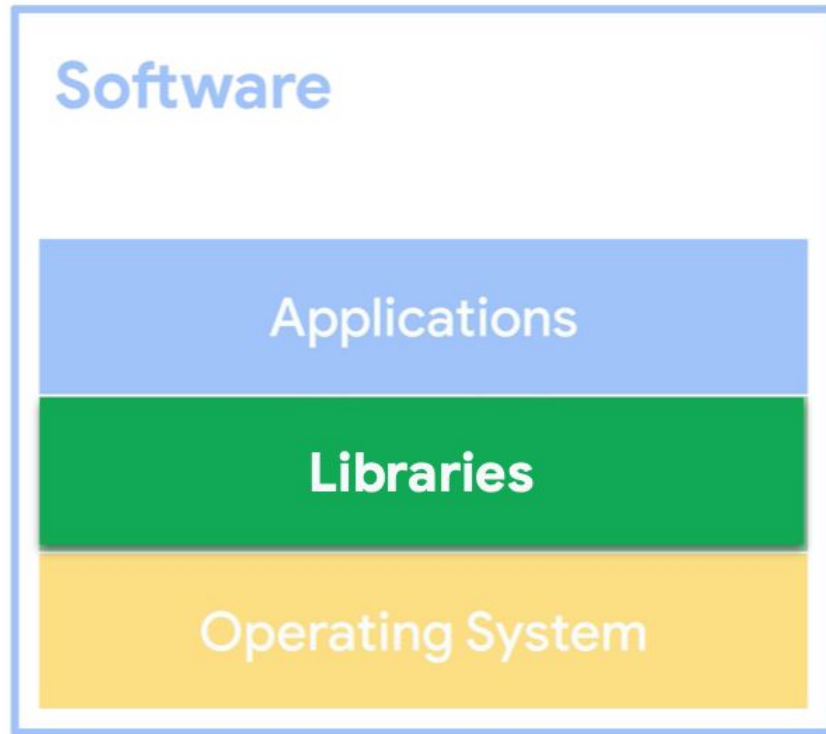
Operating System

Hardware



```
import numpy as np
```

```
for x in range(10):  
    np.SaveTheWorld()
```



# Portability Opportunity

Able to execute the same code on different microprocessor hardware and architectures.

# Portability Trade-offs

Option 1	Universal Code Portability/Compatibility		✓
	Cost (\$)		✗
	Power Consumption (W)		✗
	Engineering Effort		✗
Option 2	Lower Code Portability		✗
	Cost (\$)	✓	
	Power (W)	✓	
	Eng. Effort	✓	

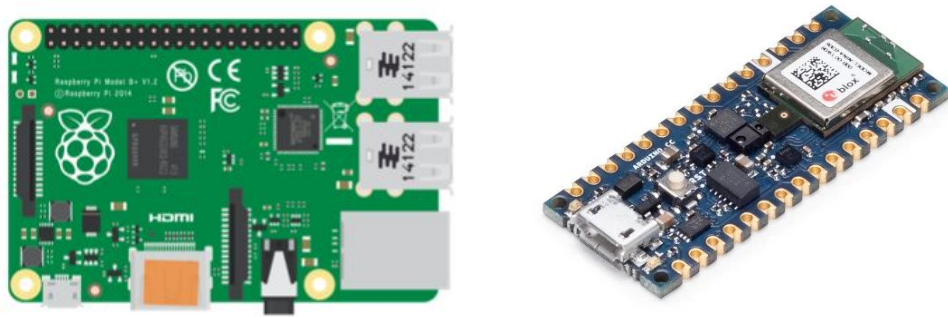


# Portability Trade-offs

Option 1	Universal Code Portability/Compatibility		✓
	Cost (\$)		✗
	Power Consumption (W)		✗
	Engineering Effort		✗
Option 2	Lower Code Portability		✗
	Cost (\$)	✓	
	Power (W)	✓	
	Eng. Effort	✓	

# Portability Trade-offs

Sacrifice portability across systems for efficiency in system performance and power efficiency



Specific HW Implementation of a Library

Option 2

Lower Code Portability



Cost (\$)



Power (W)



Eng. Effort



# Summary

- **Embedded hardware** is extremely limited in performance, power consumption and storage
- **Embedded software** is not as portable and flexible as mainstream computing

# Reading Material

# Main references

- [Harvard School of Engineering and Applied Sciences - CS249r: Tiny Machine Learning](#)
- [Professional Certificate in Tiny Machine Learning \(TinyML\) – edX/Harvard](#)
- [Introduction to Embedded Machine Learning - Coursera/Edge Impulse](#)
- [Computer Vision with Embedded Machine Learning - Coursera/Edge Impulse](#)
- Fundamentals textbook: [“Deep Learning with Python” by François Chollet](#)
- Applications & Deploy textbook: [“TinyML” by Pete Warden, Daniel Situnayake](#)
- Deploy textbook [“TinyML Cookbook” by Gian Marco Iodice](#)

I want to thank **Shawn Hymel** and Edge Impulse, **Pete Warden** and **Laurence Moroney** from Google, Professor **Vijay Janapa Reddi** and **Brian Plancher** from Harvard, and the rest of the **TinyMLedu** team for preparing the excellent material on TinyML that is the basis of this course at UNIFEI.

The IESTI01 course is part of the **TinyML4D**, an initiative to make TinyML education available to everyone globally.

Thanks



**UNIFEI**