tinyML® Foundation

Enabling Ultra-low Power Machine Learning at the Edge

tinyML Summit April 22 - 24, 2024

www.tinyML.org

EPFL

EMBEDDED SYSTEMS LABORATORY

BiomedBench

A benchmark suite of biomedical ML applications for ultra-low-power wearable devices

Dimitrios Samakovlis,
Stefano Albini, Ruben Rodriguez, Denisa Constantinescu, Davide Schiavone, Miguel Peon, David Atienza

EPFL - Embedded Systems Laboratory

dimitrios.samakovlis@epfl.ch

Outline

1. **Wearable domain** – Application + Device features

2. **BiomedBench** – Research gap filled

3. **Applications** – List of applications + characterization

4. **Architectural exploration** - Exploiting BiomedBench apps

5. **Conclusion** – Key points to remember

Application profile for wearables

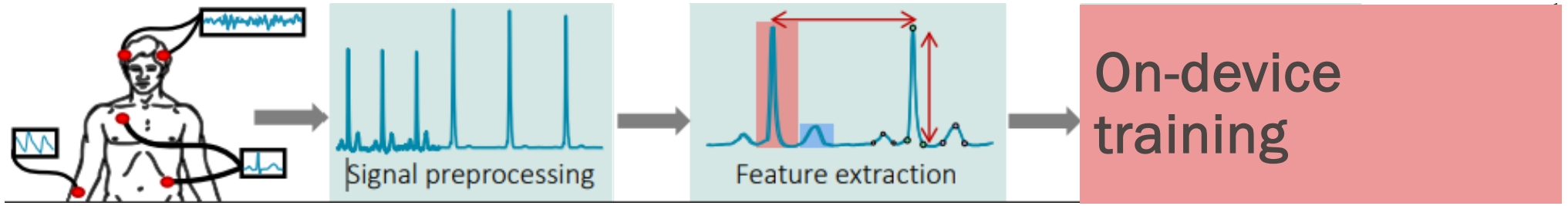


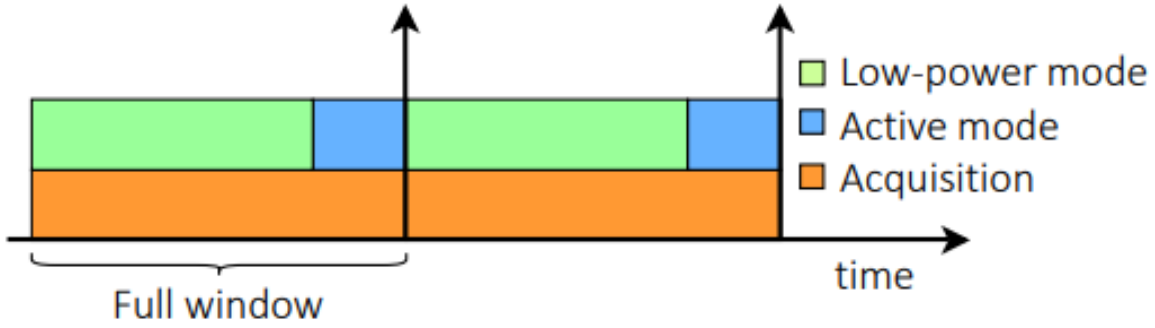
Figure from [1]

Physiological signal
i.e. ECG, EEG, PPG

Signal preprocessing
i.e. filtering, RMS

Feature extraction
i.e. FFT, R-peak detection

Model fine-tuning
i.e. personalization



Idle

Acquisition

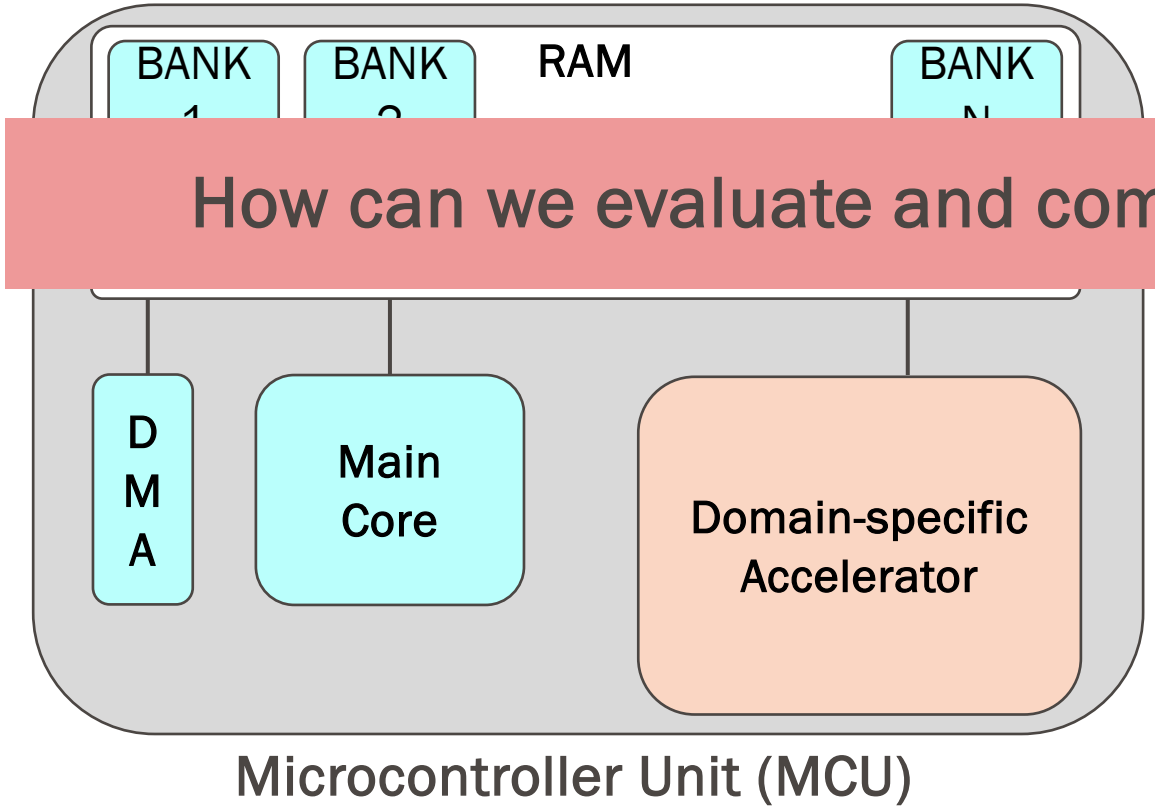
Processing

[1] E. De Giovanni et. al., *Modular Design and Optimization of Biomedical Applications for Ultra-Low Power Heterogeneous Platforms*, 2020

Wearable devices' characteristics

Low area and clock
• RAM < 1 MB • Clock < 200 MHz

Power management modes
• Deep sleep • Memory bank management



Microcontroller Unit (MCU)

How can we evaluate and compare the 2 design directions?

General purpose design
Example: CPU

Custom design
i.e. CGRA, in-memory computing Example: HEEPocrates

SW / HW Gap in wearable domain

How c **8 SoA end-to-end** health monitoring applications ands?

Characterization of application requirements

Comparative analysis on **SoA platforms**

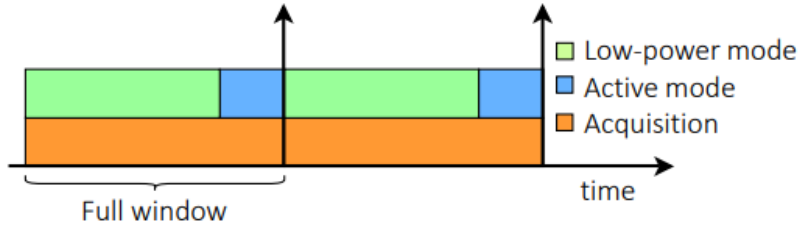
Open-source and open-ended

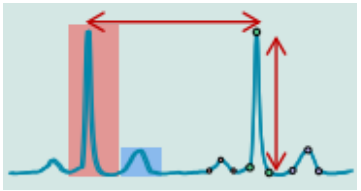
Need for new benchmark suite

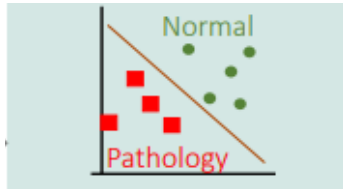
End-to-end applications
(Signal acquisition/idle)

Variety of physiological signals
(i.e. ECG, EEG, PPG)

Variety of kernels
(i.e. Preprocessing, Inference)







Multidisciplinary Embedded

- CoreMark
- EEMBC
- BenchIoT

Not representative of health monitoring applications

Biomedical

Limited signals and kernels

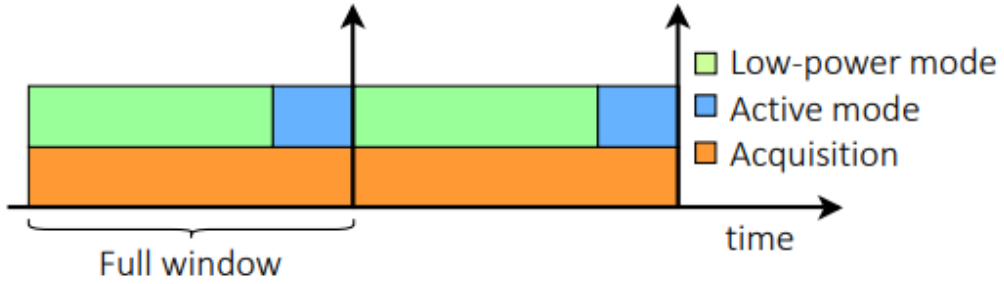
No end-to-end applications

BiomedBench applications

Application full name	Abbreviation	Input Signal	Main Kernels	Dominant Phase
Heartbeat Classifier [1]	HeartBeatClass	ECG	Morphological filtering	Preprocessing
Seizure Detection SVM [2]	SeizDetSVM	ECG	FFT, SVM	Feature extraction
Seizure Detection CNN [3]	SeizDetCNN	EEG	CNN	Inference
Cognitive Workload Monitoring [4]	CognWorkMon	EEG	FFT, Random forest	Feature extraction
Gesture Classifier [5]	GestureClass	EMG	ICA, MLP	Feature extraction
Cough Detection [6]	CoughDetect	Audio, IMU	MFCC, Random forest	Feature extraction
Emotion Classifier [7]	EmotionClass	PPG, GSR, ST	kNN	Inference
Biological-Backpropagation free [8]	Bio-BPfree	EEG	CNN gradients	Fine-tuning

[1] E. De Giovanni et. al., *Modular Design and Optimization of Biomedical Applications for Ultra-Low Power Heterogeneous Platforms*, 2020
[2] Farnaz Forooghifar et. al., *Self-Aware Wearable Systems in Epileptic Seizure Detection*, 2018
[3] Catalina Gómez et. al., *Automatic seizure detection based on imaged-EEG signals through fully convolutional networks*, 2020
[4] Renato Zanetti et. al., *Real-Time EEG-Based Cognitive Workload Monitoring on Wearable Devices*, 2022
[5] Mattia Orlandi et. al., *sEMG Neural Spikes Reconstruction for Gesture Recognition on a Low-Power Multicore Processor*, 2022
[6] L. Orlandic et. al., *A Multimodal Dataset for Automatic Edge-AI Cough Detection*, 2023
[7] Jose Miranda et. al., *Embedded Emotion Recognition within Cyber-Physical Systems using Physiological Signals*, 2018
[8] Saleh Baghersalimi et. al., *Layer-Wise Learning Framework for Efficient DNN Deployment in Biomedical Wearable Systems*, 2023

Application profile – Key metrics



Metrics	Importance
1. Full Window duration	Real-time constraints
2. Duty Cycle	Impact of idle period
3. Main Operations	Microarchitectural demands
4. Input Bandwidth	Impact of acquisition

BiomedBench – Application metrics

Application	Full Window duration (sec)	Duty Cycle	Main Operations	Input Bandwidth (B/sec)
HeartBeatClass	15.0	0.30 %	**FxP** branches (min/max search)	1.500
SeizureDetSVM	60.0	0.03 %	**32-bit FxP** multiplications	128
SeizureDetCNN	4.0	50.25 %	**16-bit FxP** MACs	11.500
CognWorkMon	56.0	2.10 %	**32-bit FxP** multiplications, shifts	4.000
GestureClass	0.2	95.50 %	**32-bit FP** MACs	187.500
CoughDet	0.3	29.00 %	**32-bit FP** multiplications	62.900
EmotionClass	10	0.15 %	**FP** branches (sorting)	803
Bio-BPfree	-	-	**32-bit FP** MACs	-

Distinct challenges through idle – acquisition - processing

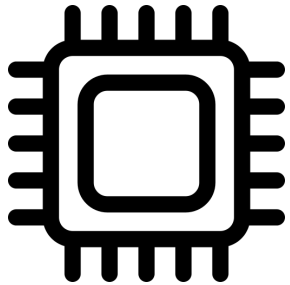
Portability of applications

Hardware Abstraction Layer

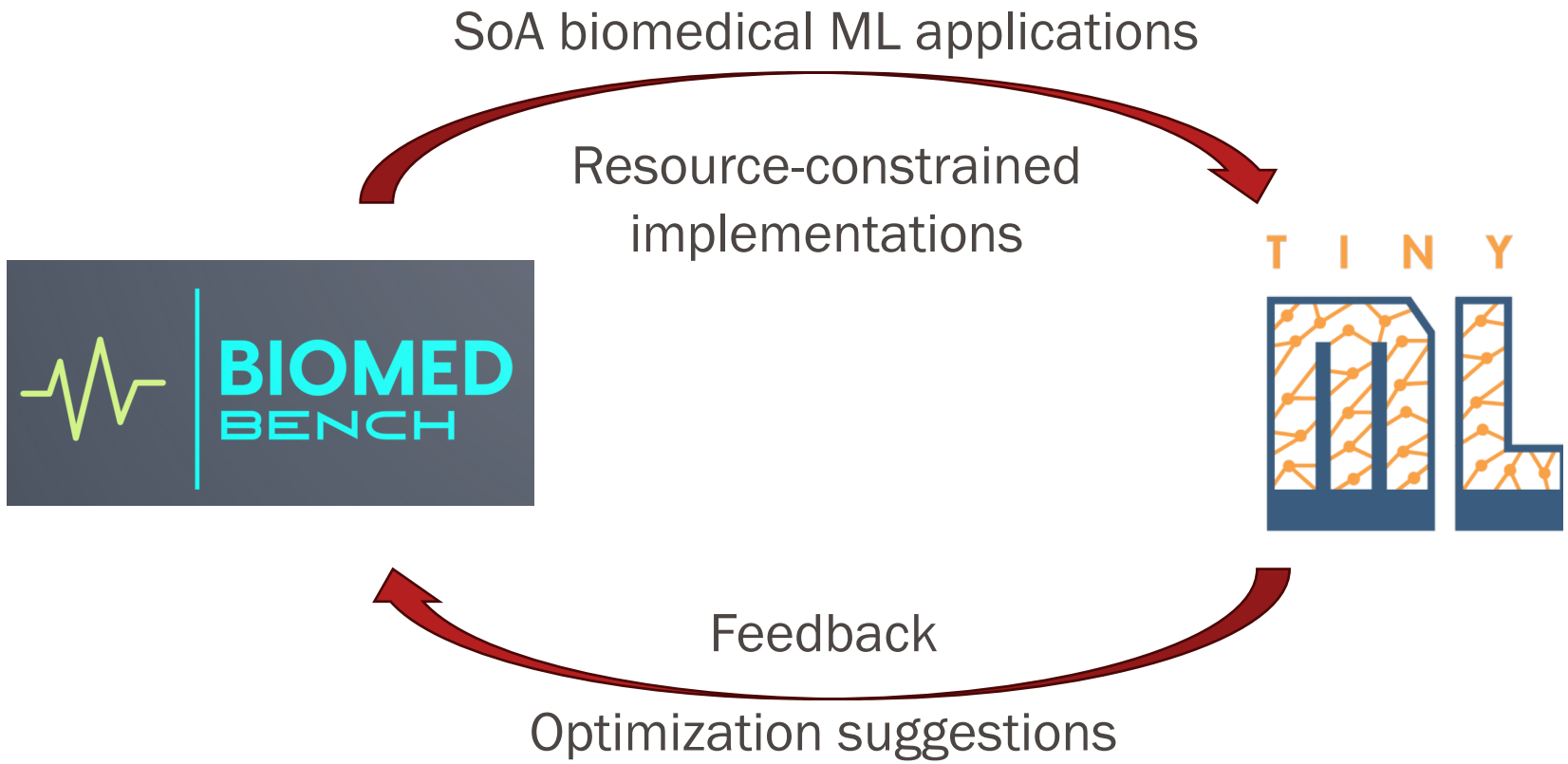
Need to extend FreeRTOS to achieve portable applications!

Portable

Not portable!



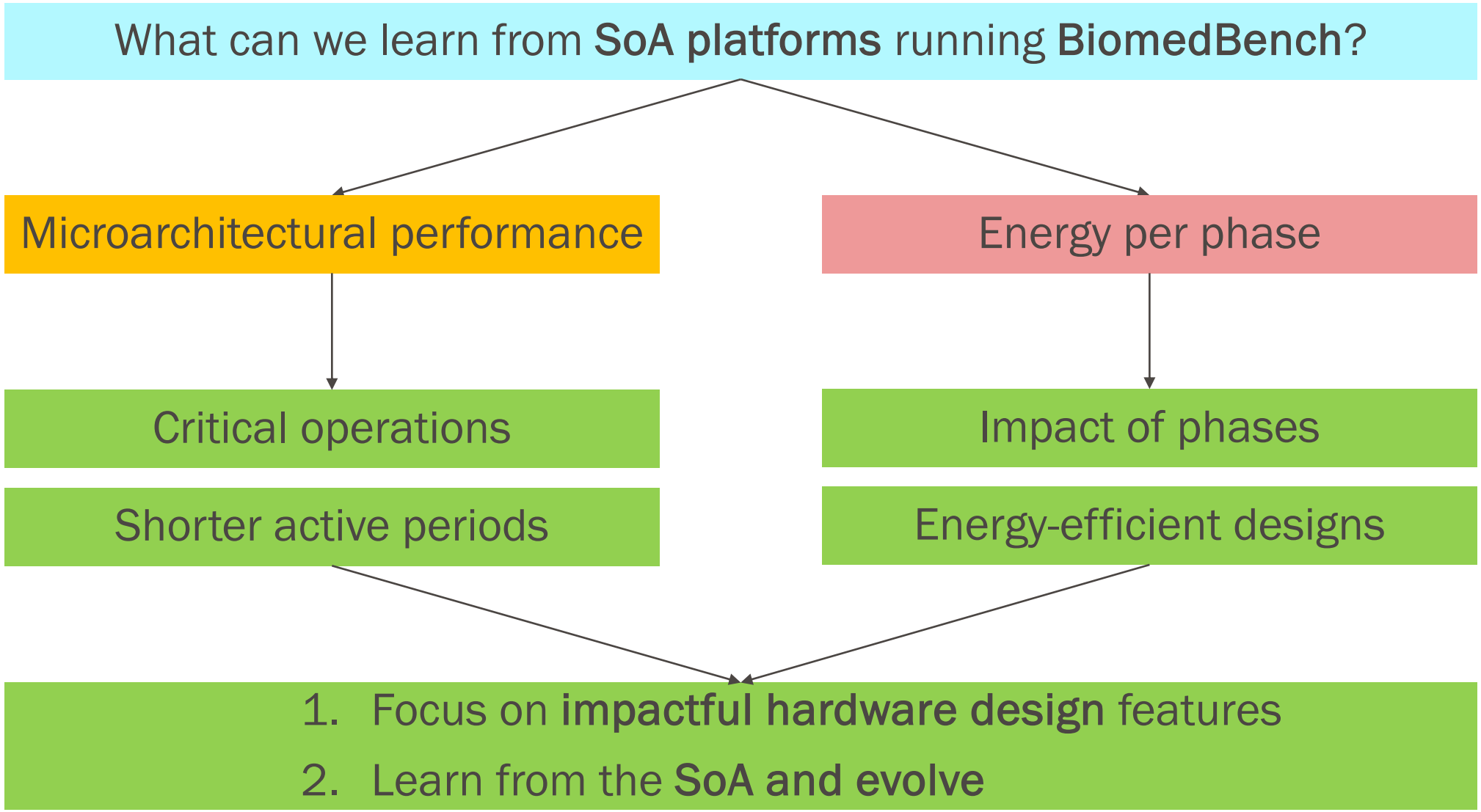
How BiomedBench relates with TinyML?



BiomedBench is **open-source** and **open-ended!**

- Evolves with the **SoA !**
- Improves on **user feedback!**

BiomedBench on SoA platforms



Evaluated platforms

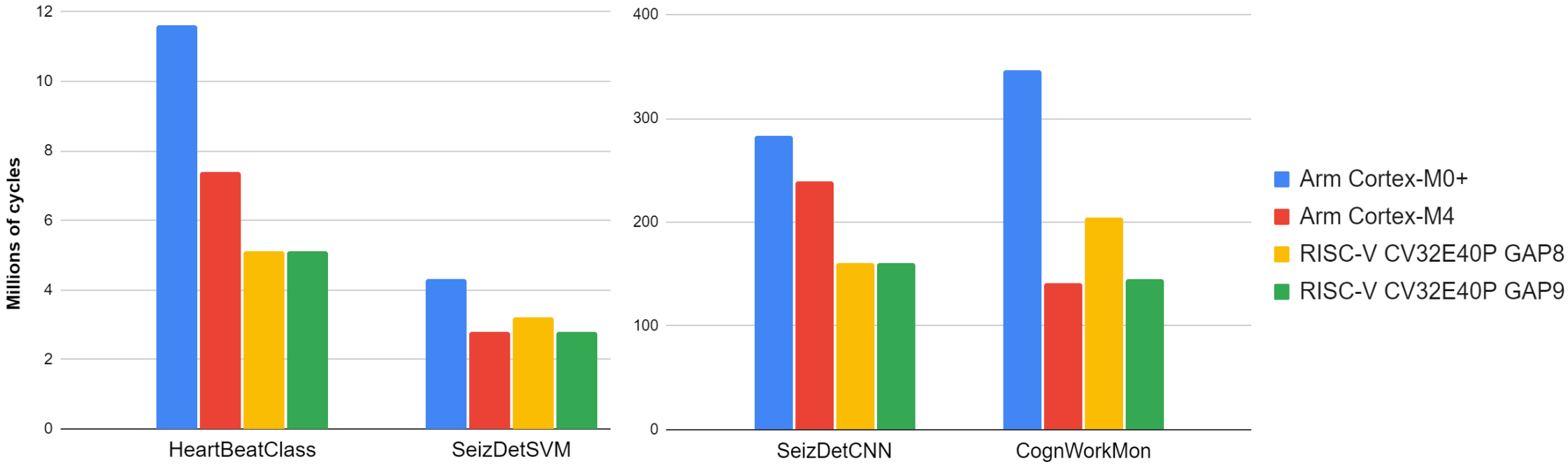
Ultra-low power (ULP) features	**Fine-grained sleep** modes
SPI / DMA support	**Microarchitectural variance**

Board	Manufacturer	MCU	Processor Family	Processor	RAM (KB) / FLASH (MB)
Nucleo L4R5ZI	STMicroelectronics	STML4R5ZI	Arm	**Arm Cortex-M4** @120MHz	640 / 2
Apollo 3	Ambiq	Apollo 3 Blue	Arm	**Arm Cortex-M4** @96MHz	384 / 1
GAPuino	GreenWaves Technologies	GAP8	RISC-V	**CV32E40P GAP8**@150 MHz (+8 identical cores)	512 / 2
GAP9_EVK	GreenWaves Technologies	GAP9	RISC-V	**CV32E40P GAP9**@240MHz (+9 identical cores)	1500 / 2
Raspberry Pi Pico	Raspberry	RP2040	Arm	**Arm Cortex-M0+** @133MHz (dual core)	264 / 2

Microarchitectural comparison

Millions of execution cycles per processor ~ Execution time

- No instruction-level parallelism
- Single cycle memory access



GAP9 is the fastest – Arm Cortex-M0+ is the slowest

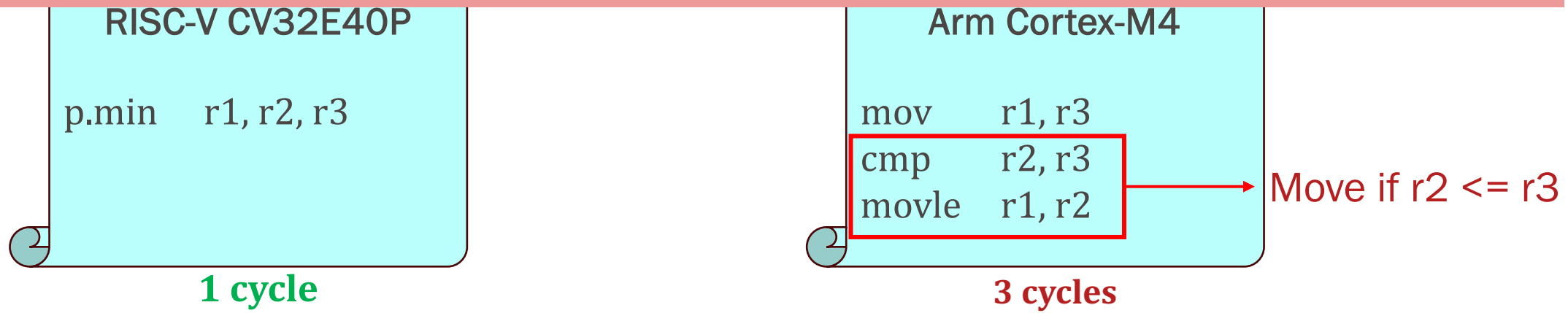
GAP8/Arm Cortex-M4 performances vary among applications

Heartbeat Classifier

- Morphological filtering (queue data movements + min/max search)
- **RISC-V processors need 31% fewer cycles than Arm Cortex-M4**

RISC-V	Arm Cortex-M4
1. 32 registers → no stack needed	1. 16 registers → heavy use of stack
2. 1-cycle min/max of 2 regs (microarchitectural ext. [1])	2. Conditional set

A small register file can significantly reduce performance

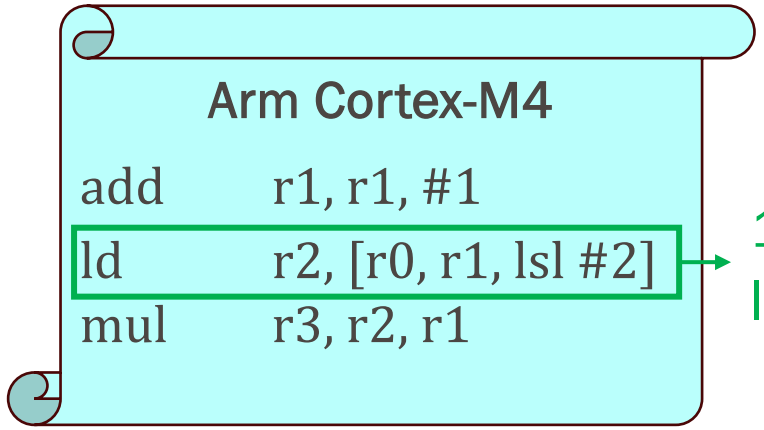


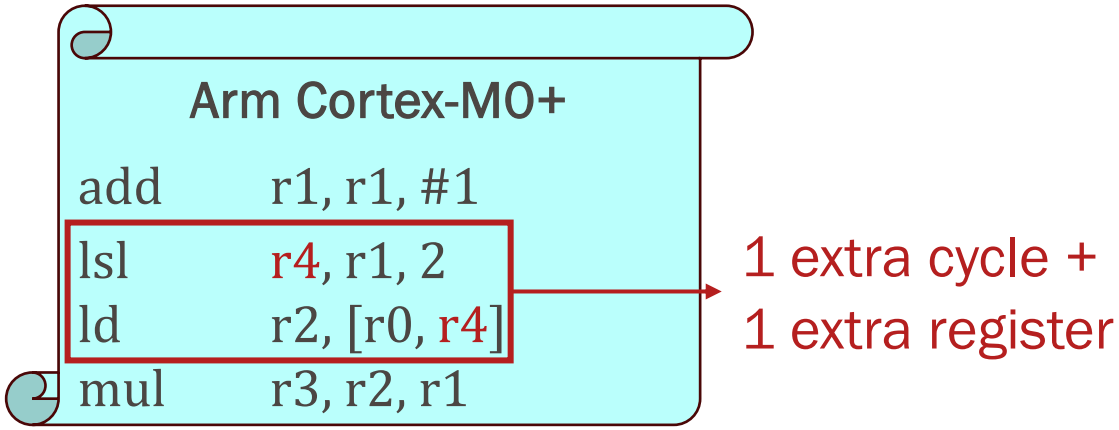
[1] Michael Gautschi et. al., *Near-Threshold RISC-V Core With DSP Extensions for Scalable IoT Endpoint Devices*, 2017

Heartbeat Classifier

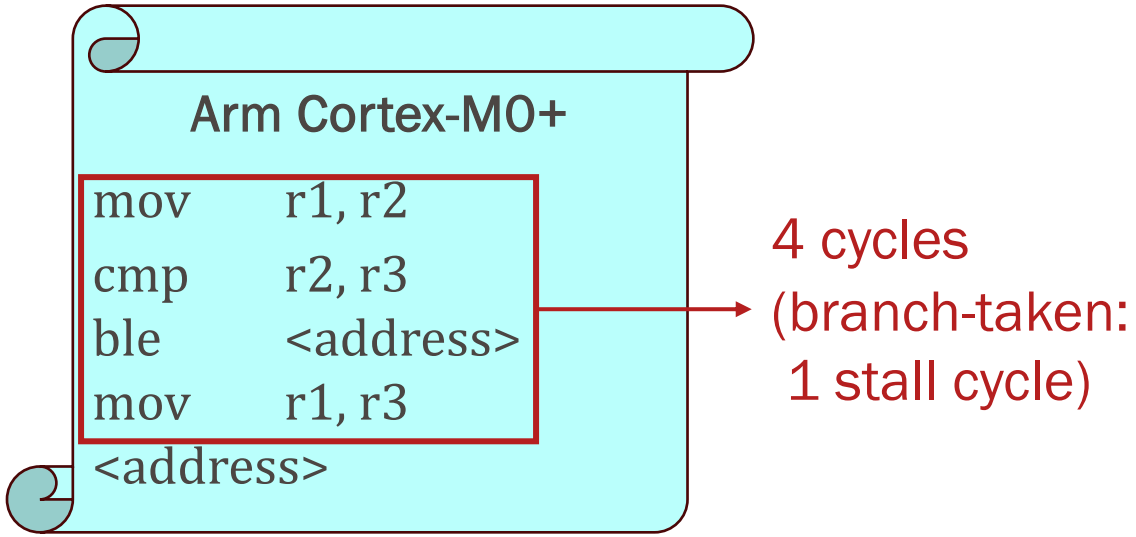
- Arm Cortex-M0+ needs **36% more cycles** than Arm Cortex-M4

1. AC-M0+ has a **weaker ISA** → more cycles + stack push/pop





2. AC-M0+ uses branches for min search



32-bit FxP applications

- **32-bit FxP multiplications:**
  - **64-bit intermediate result** followed by right shift
- RISC-V GAP9 / Arm Cortex-M4: **most efficient** (support for 64-bit results)
- RISC-V GAP8: **14%-45% more cycles** (SeizDetSVM – CognWorkMon)

Arm Cortex-M4

```
smull   r0, r1, r2, r3
orr.w   r3, r3, r1, lsl #20
```

3 cycles

RISC-V GAP9

```
p.muls.d   r0, r2, r3
addi.d     r0, r0, 1
srai.d     r0, r0, 0x1
```

4 cycles

.d → 64-bit extensions
adjacent 32-bit registers

RISC-V GAP8

```
mulh    r0, r1, r3
srli    r0, r0, 0xc
slli    r1, r1, 0x14
or      r0, r0, r1
```

9 cycles
(mulh → 4 cycles)

Support for 64-bit result of integer multiplications is crucial in 32-bit FxP applications

32-bit FxP applications (2)

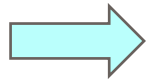
Arm Cortex-M0+: **54%-145% more cycles** than Arm Cortex-M4

Arm Cortex-M4

```
smull   r0, r1, r2, r3
lsrs    r3, r0, #12
orr.w   r3, r3, r1, lsl #20
```

Arm Cortex-M0+

```
bl      <lmul>
lsrs    r0, r0, #12
lsls    r1, r1, #20
orrs    r0, r1
```



```
<lmul>:
muls    ...
muls    ...
adds    ...
mov     ...
lsrs    ...
uxth    ...
muls    ...
lsrs    ...
        ..........
```

20 cycles

16-bit FxP application

- 16-bit FxP MACs (Seizure Detection CNN)
- **RISC-V processors need 33% fewer cycles than Arm Cortex-M4** (DSP extensions [1])

RISC-V CV32E40P

FxP MAC:

Loop:
lp.setupi r0, 100, **endL**
....
endl: sw 0(r12!), r4

1 cycle FxP MAC
0 cycles for loop

15% f

Arm Cortex-M4

FxP MAC:

Loop:
cmp r5, #100
bne <startL>

2 cycles FxP MAC
3 cycles for loop

ex-M0+

Arm Cortex-M0+

FxP MAC:

add r0, r0, r1

Loop:
cmp r5, #100
bne <startL>

3-cycle FxP MAC
3 cycles for loop

Well-designed DSP extensions yield excellent performance gains

[1] Michael Gautschi et. al., *Near-Threshold RISC-V Core With DSP Extensions for Scalable IoT Endpoint Devices*, 2017

Energy breakdown

Idle + Acquisition + Processing = Total Energy

Idle — **Deep-sleep Power** × Idle duration

Acquisition — **Acquisition Power** × Acquisition duration

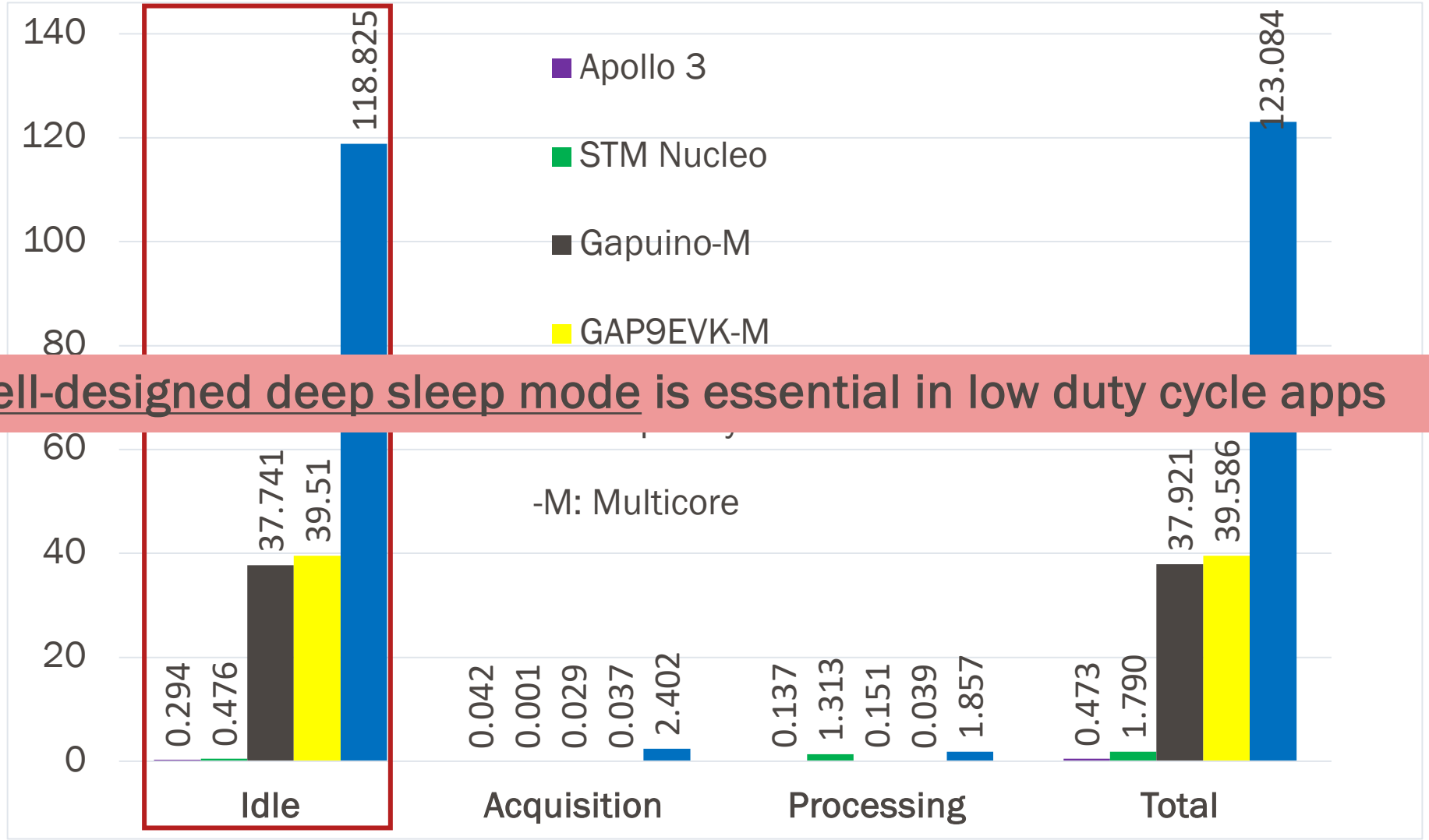
Processing — **Processing Power** × **Processing duration**

Microarchitecture performance

Which phase is more impactful on the total energy footprint?

SeizDetSVM – Idle energy impact

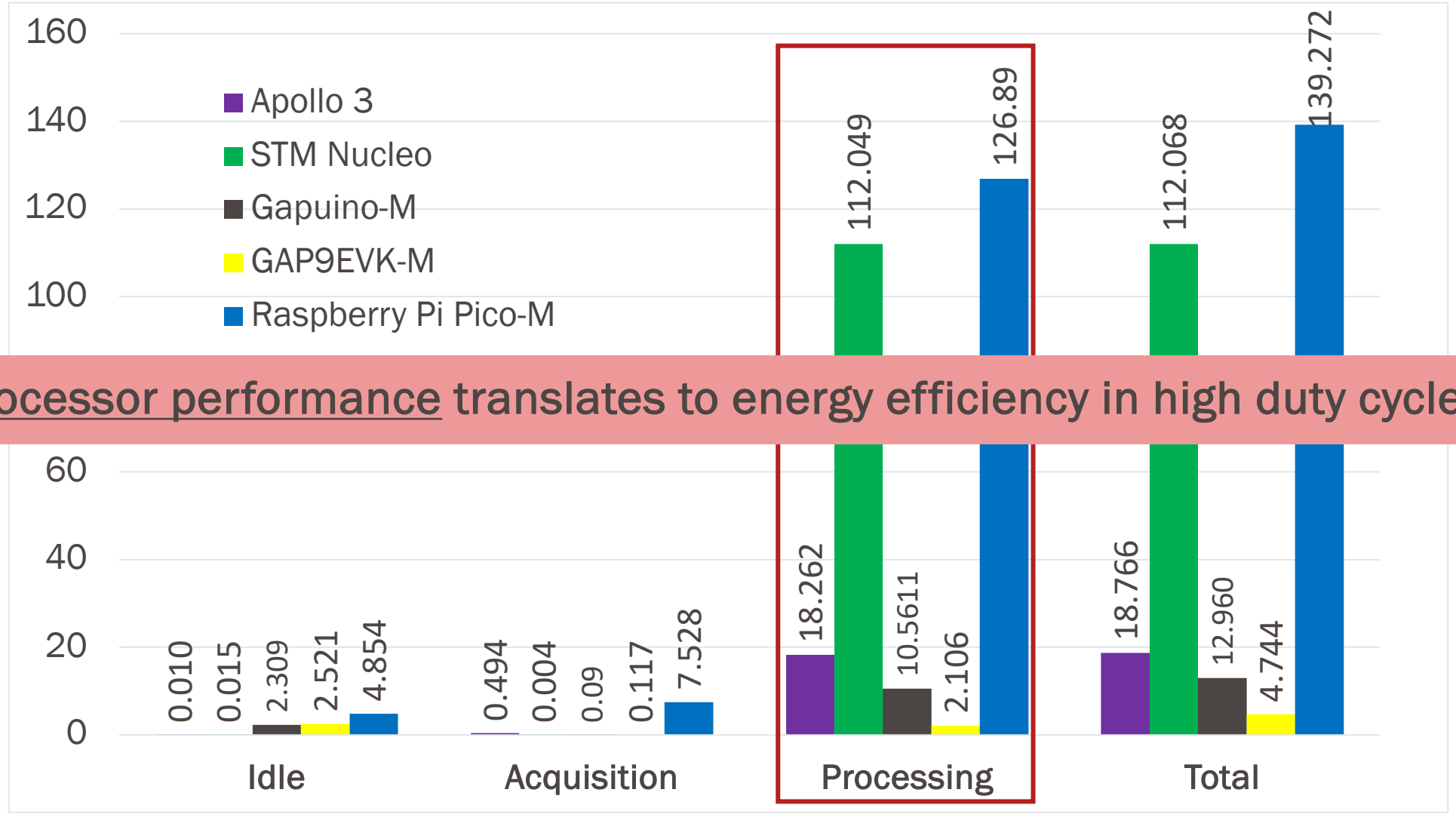
Very low duty cycle (< 0.1 %)



A well-designed deep sleep mode is essential in low duty cycle apps

SeizDetCNN – Processing impact

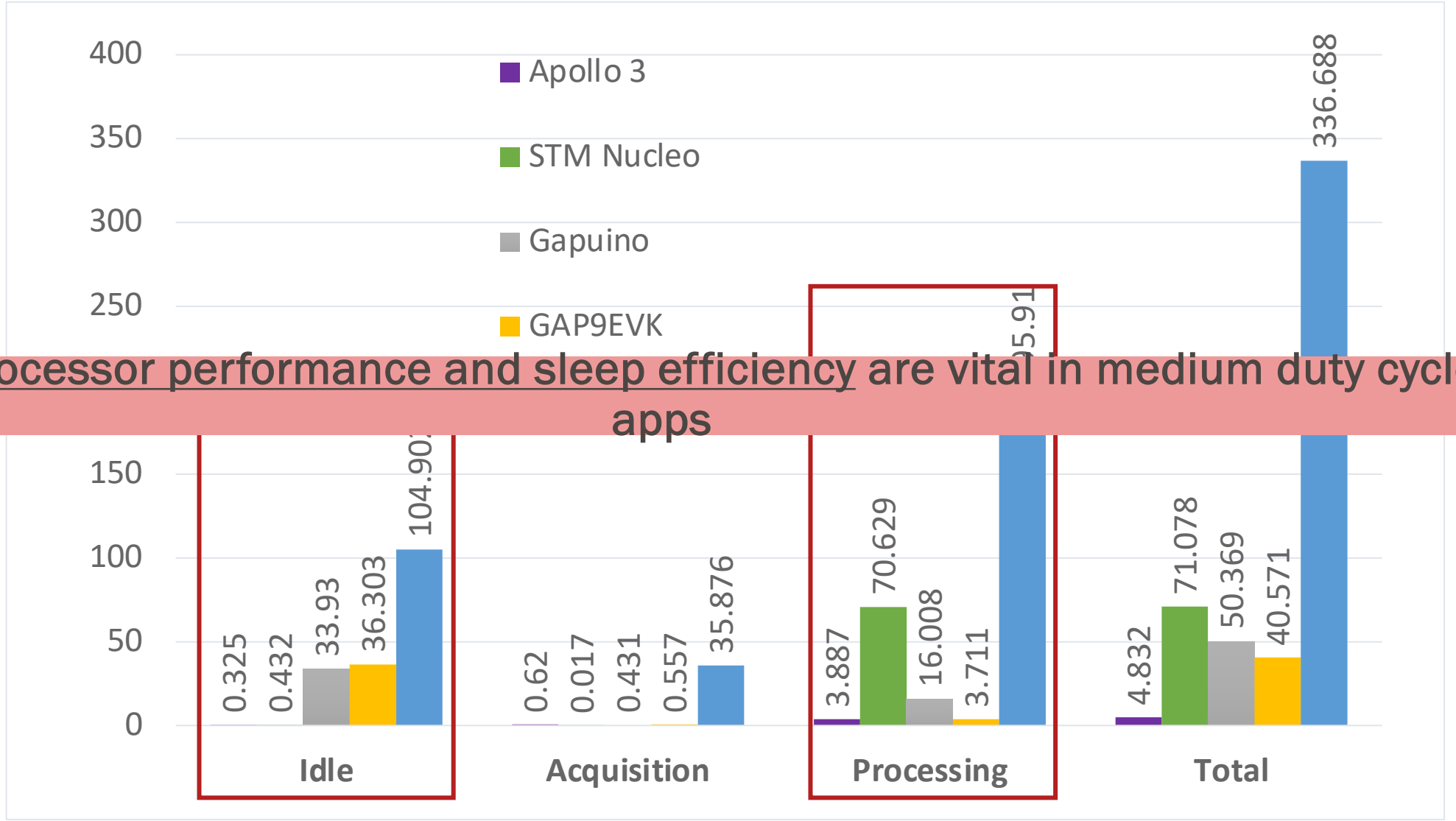
High duty cycle (~ 50 %)



Good processor performance translates to energy efficiency in high duty cycle apps

CognWorkMon – Balanced idle / processing

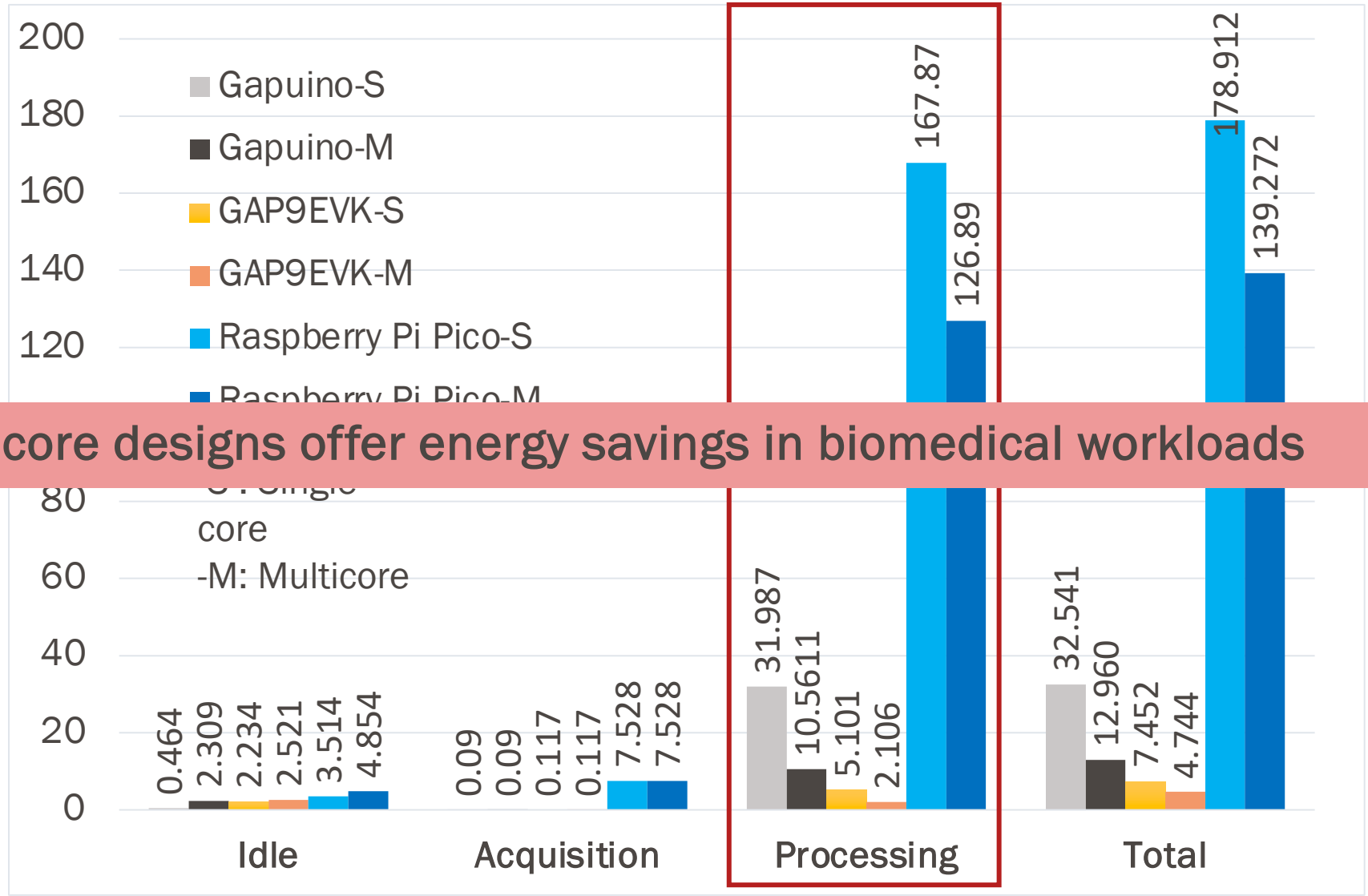
Medium duty cycle (~ 2 %)



Both processor performance and sleep efficiency are vital in medium duty cycle apps

SeizDetCNN – Multicore benefits

High degree of parallelism



Multicore designs offer energy savings in biomedical workloads

Key Conclusions

BiomedBench paves the way towards **application-driven hardware design!**

1. Unveils the key **application demands**
2. Enables **systematic SoA comparison**

BiomedBench is **open-ended and open-source!**

1. Evolves with the **SoA and feedback**
2. **TinyML** community can help us improve

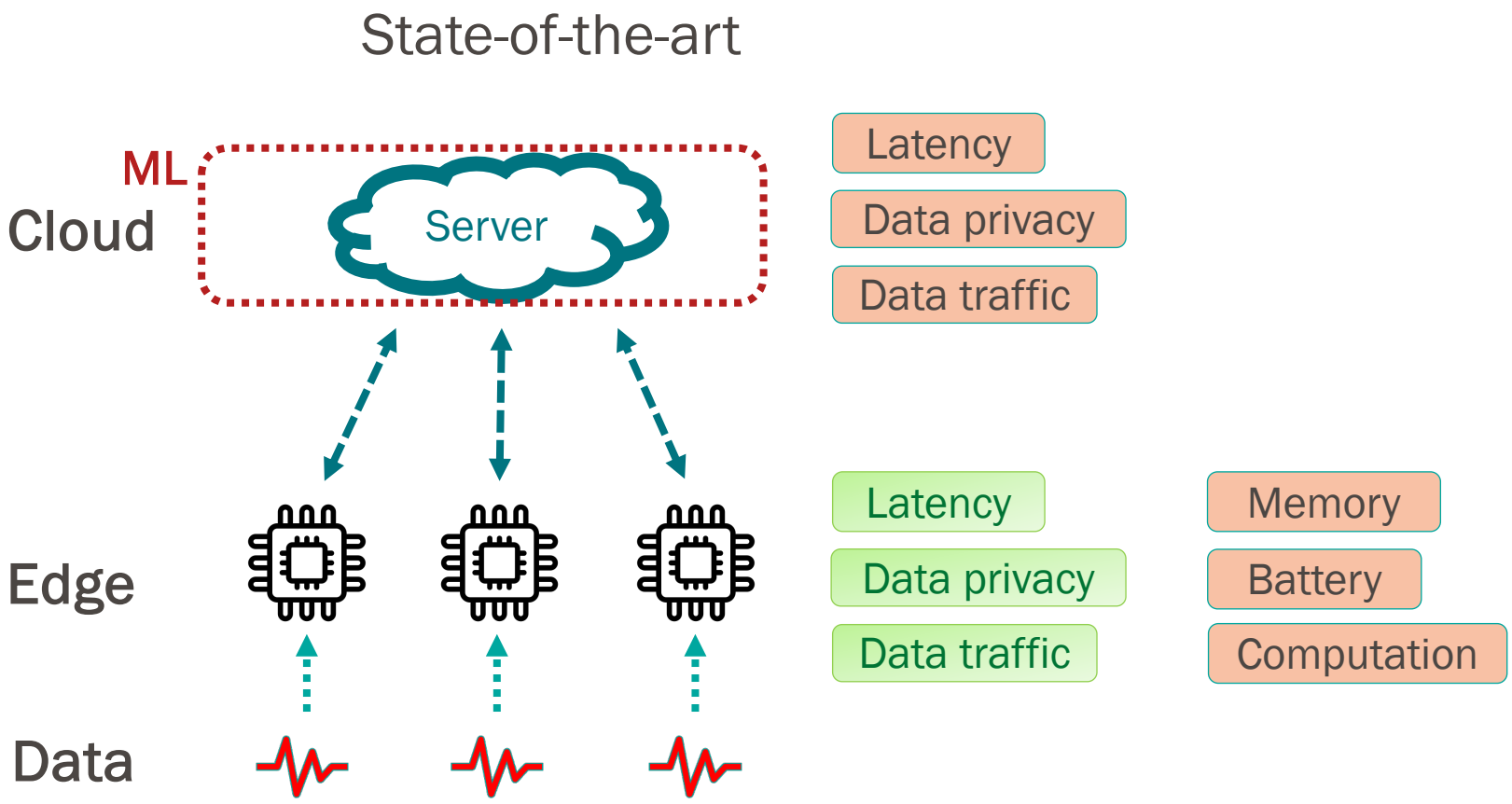
eslweb.epfl.ch/biomedbench
(Website under development!)

Thank you!

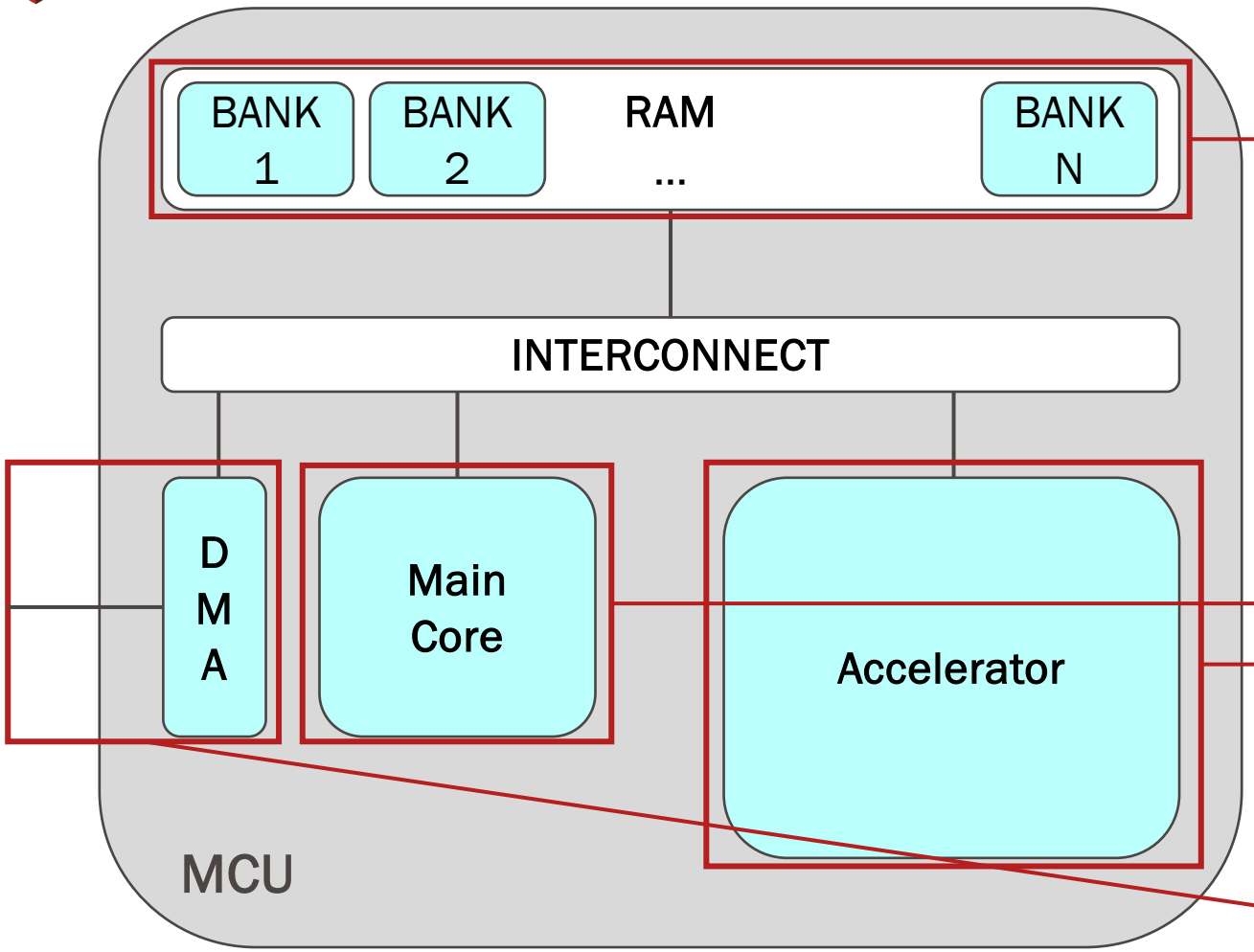
Dimitrios Samakovlis

EPFL - Embedded Systems Laboratory
dimitrios.samakovlis@epfl.ch

Machine Learning on Edge devices



SoA Hardware Optimizations



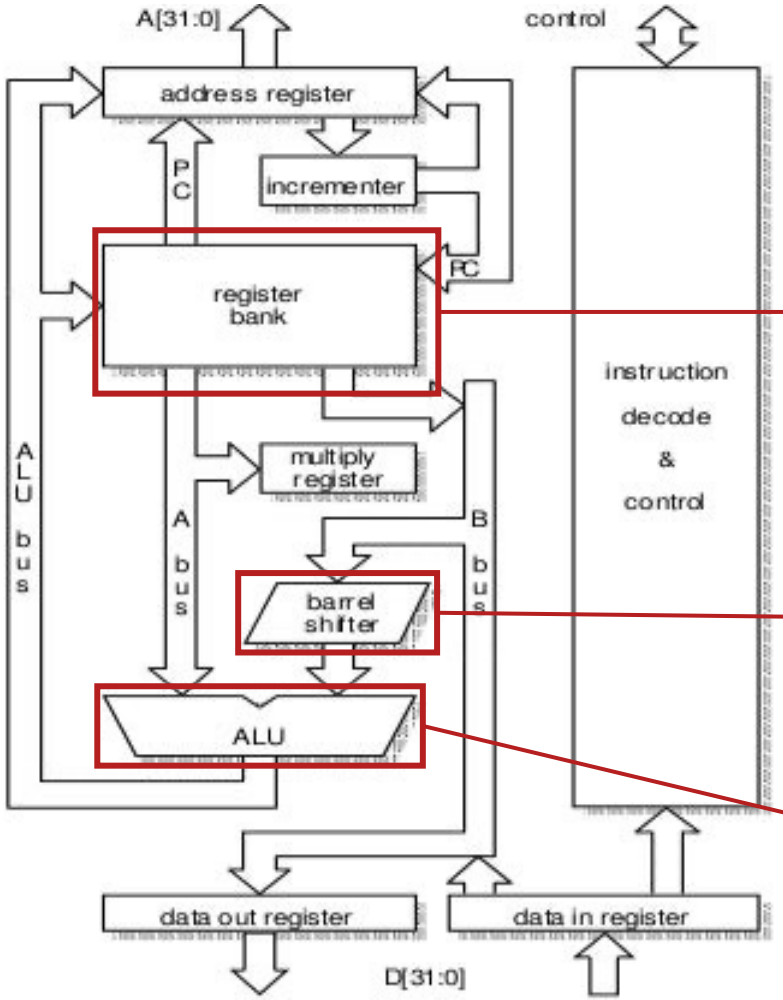
Independent bank power management [1]
- Active / Retention / OFF

Clock-gating / Power-gating
Specialized HW (i.e. CGRA) [1]
for long idle periods
DVFS and parallelization [1], [2], [3]

Fetch input signal from sensor to RAM while core is clock-gated [2], [3]

[1] E. De Giovanni et al., "Modular Design and Optimization of Biomedical Applications for Ultralow Power Heterogeneous Platforms“, Nov. 2020
[2] E. Flamand et al., “GAP-8: A RISC-V SoC for AI at the edge of the IoT,” Jul. 2018
[3] A. Pullini et al., “Mr.Wolf: An energy-precision scalable parallel ultra low power SoC for IoT edge processing,” Jul. 2019

Microarchitectural trade-offs



How complex should the microarchitecture be?

There is a tradeoff: faster processing vs area/energy?

How many registers?

Barrel shifter or simple shifter?

Divisions?

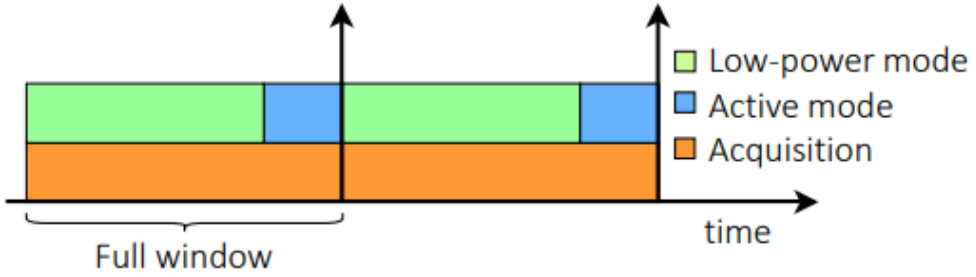
MACs?

32-bit integer multiplications with 64-bit results?

How can BiomedBench boost microarchitectural exploration?

[1] The ARM architecture, https://piembsystech.com/arm-microcontroller/, (Accessed 30.6.2023)

Why we need end-to-end applications?



Idle + Acquisition + Processing = Full application cycle

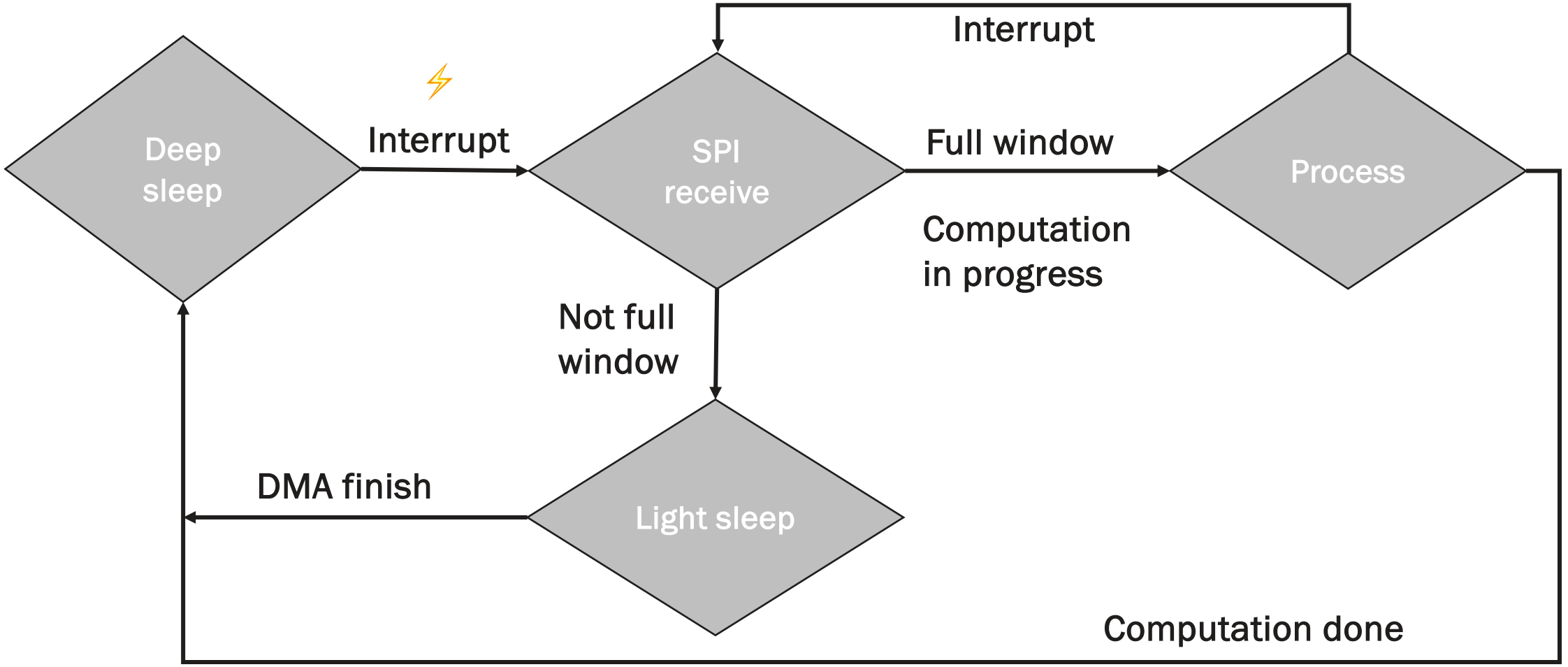
Total energy footprint is a result of interplay between the 3 phases!

Energy evaluation

- **MCU energy consumption** of all platforms executing BiomedBench
- We **emulate the sensor** using an external board sending data through **SPI**
  - ADC with 768 bytes buffer
- Measuring equipment:
  - Otii Arc [1] (> 100µA)
  - Fluke 8846A digital multimeter [2] (<100µA)

[1] Otii Arc Pro, QOITECH, https://www.qoitech.com/otii-arc-pro/ (Accessed: 28/6/2023)
[2] Fluke 8846A, FLUKE, https://assets.fluke.com/manuals/884xa___umeng0200.pdf (Accessed: 28/6/2023)

MCU runtime state diagram

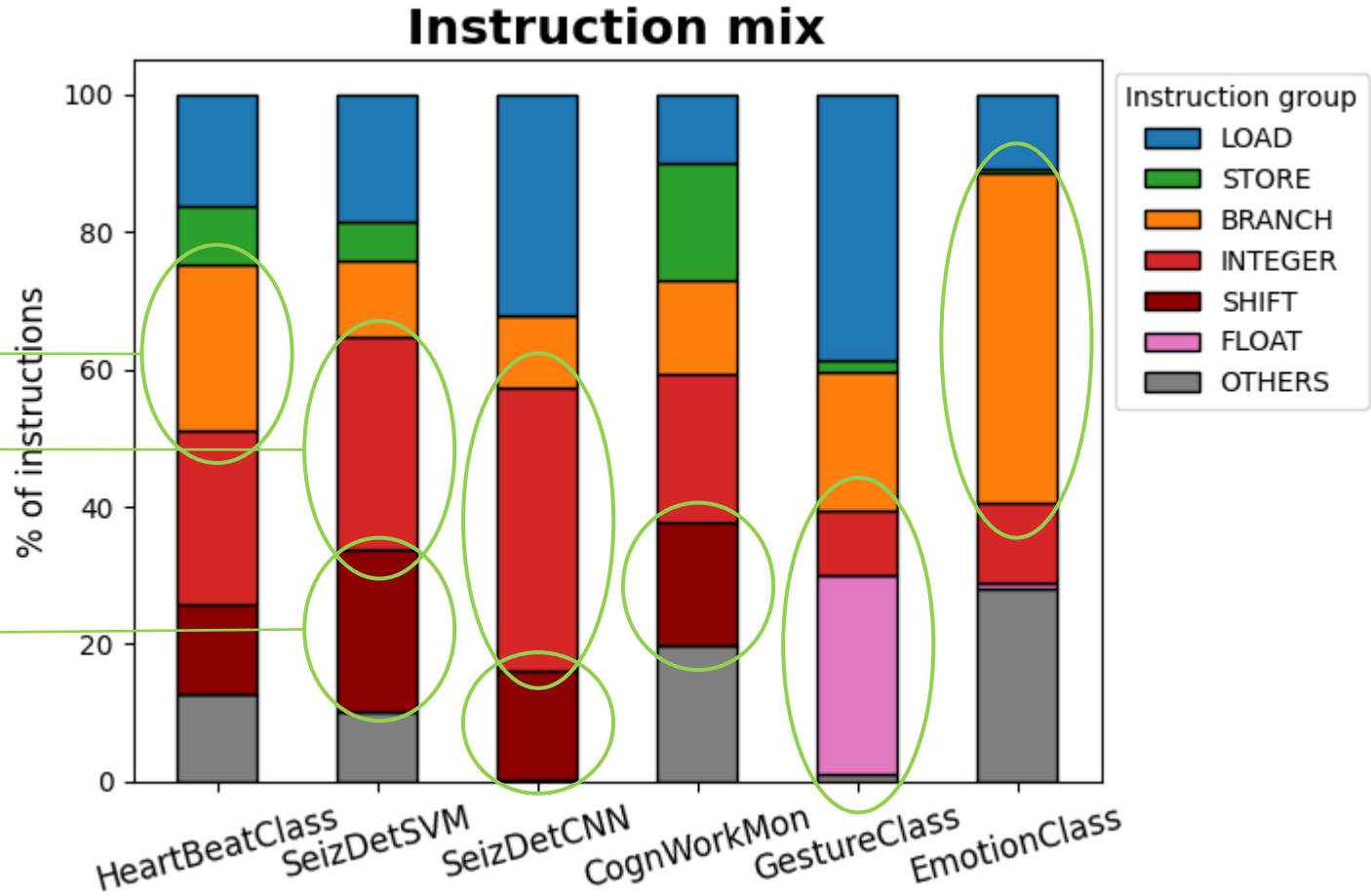


Instruction Mix

- GDB dump of instruction trace from Arm Cortex-M4
- Goal: Deeper insight into each application computational demands

Conditional execution (sorting, min/max search)
Compute-intensive

Fixed-point applications
Barrel shifter necessary!



GestureClass – CoughDetect - BioBPfree

- Main computations → **32-bit floating-point multiplications / MACs**

- RISC-V GAP9 / Arm-Cortex M4: **most efficient**
  - Only processors with a **floating-point unit (FPU)**

- RISC-V GAP9: **12-17% fewer cycles** than Arm-Cortex M4
  - **FPU MAC:** 1-cycle (RISCV) vs 3-cycle (AC-M4)
  - **FPU Load/Store:** 1-cycle (RISCV) vs 2-cycle (AC-M4)

- **Without FPU**
  - RISCV GAP8: **32x more cycles** than RISCV GAP9
  - Arm Cortex-M0+: **28x more cycles** than RISCV GAP9

EmotionClass

- kNN inference → Sorting in **32-bit floating-point**
- RISC-V GAP9 takes **36% fewer cycles** than Arm Cortex-M4

RISC-V CV32E40P GAP9

```
lp.setup    x1, a5, END
p.lw        a3, -4(a1!)
flt.s       a6, fa3, fa7
beqz        a6, ADD
mv          a7, a3
mv          a2, a4
ADD: addi   a4,a4,-1
END: nop
```

7 cycles

1 cycle load post-decrement

4 vs 6 cycles conditional set

1 vs 3 cycles loop

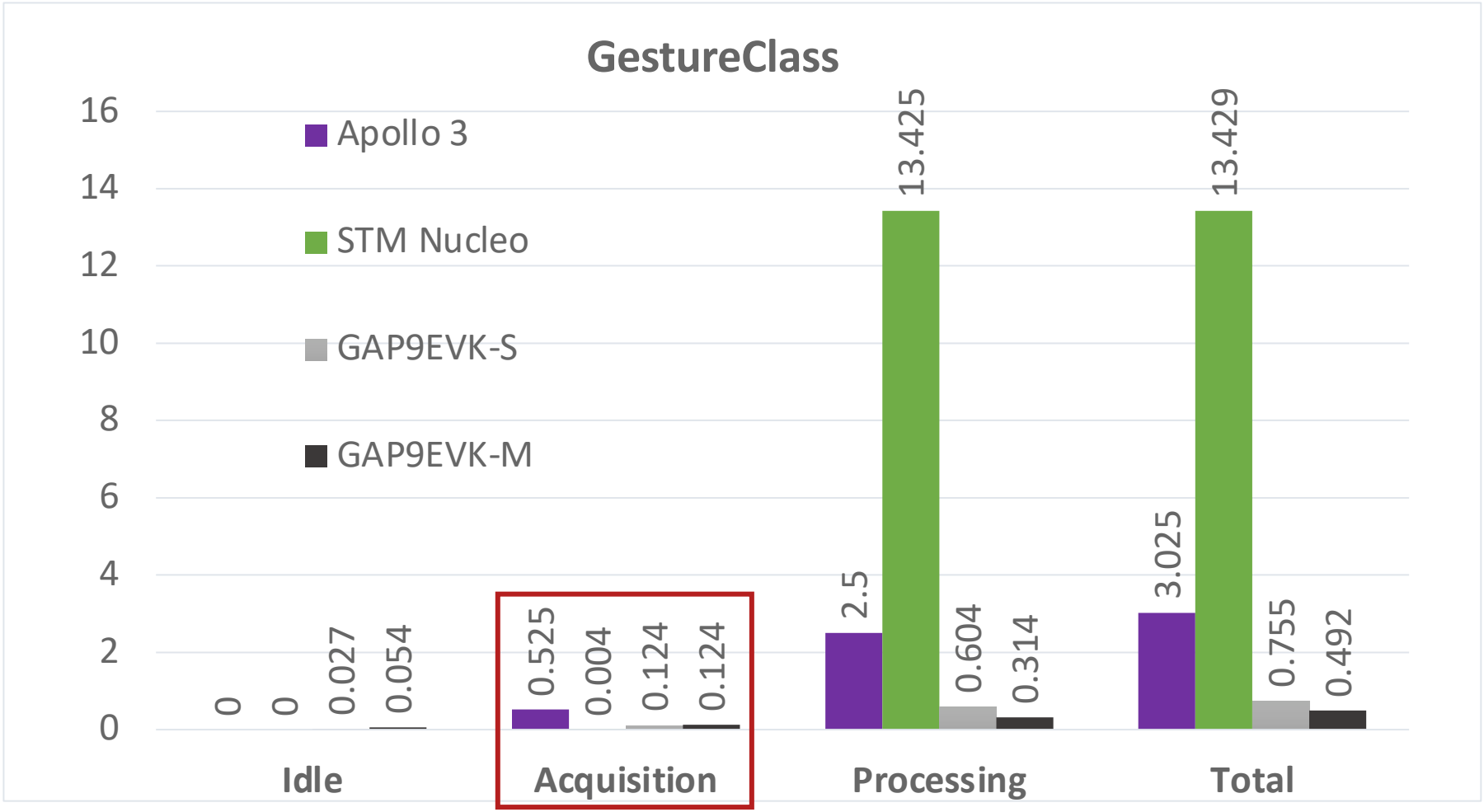
Arm Cortex-M4

```
vldmdb      r3!, {s15}
vcmpe.f32   s15, s14
vmrs        APSR_nzcv, fpscr
it          mi
movmi       r7, r4
it          mi
vmovmi.f32  s14, s15
add.w       r4, r4, -1
cmp         r4, r2
bne.n       START
```

11 cycles

GestureClass – Acquisition impact

- Very high duty cycle
- Data-intensive application



Acquisition footprint is not significant in modern platforms

Copyright Notice

www.tinyml.org