Design and Evaluation of a Task Characterization Model for Performance Control of Embedded Devices at Runtime

Colloquium of Bachelorthesis

Tobias Westphal

Bachelor of Science Informatik Technischer Systeme
Department Computer Science
Faculty of Computer Science and Engineering
Hamburg University of Applied Sciences

15.12.2022

HAW HAMBURG

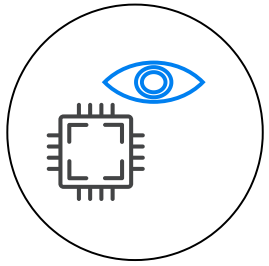
Motivation

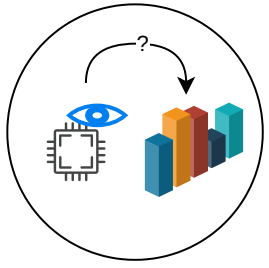
- **Goal** of embedded devices: Maximize energy efficiency
- **Problem**: static clock configuration & changing performance utilization
- **Improvement**: Adapt configuration with Dynamic Voltage and Frequency Scaling (DVFS) at runtime
- **RIOT**: ScaleClock → Boards based on ARM Cortex-M4 processor → Debug and Trace Features

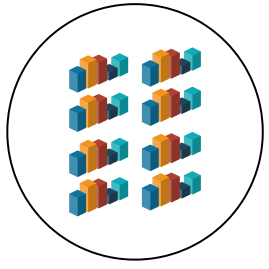
→ Can debug components be used a resource to trace task properties at runtime?

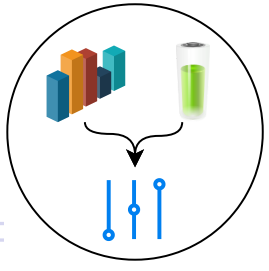
→ To which extent do traced task properties indicate tasks which improve their energy efficiency by lowering the CPU frequency?

Table of Contents







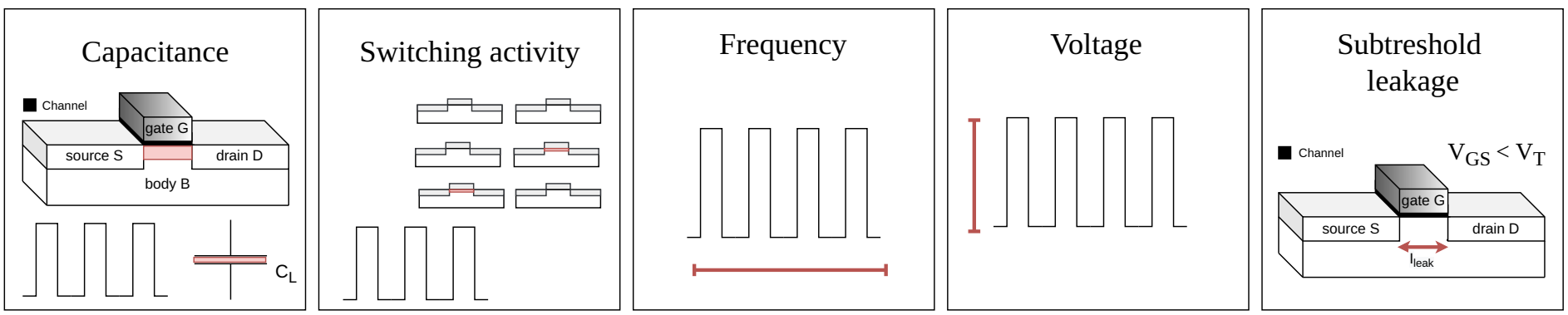


Outline

Energy Consumption

$$E_{total} = (C \cdot \alpha \cdot f \cdot V^2 + V \cdot I_{leak} \cdot N \cdot k_{design}) \cdot t$$

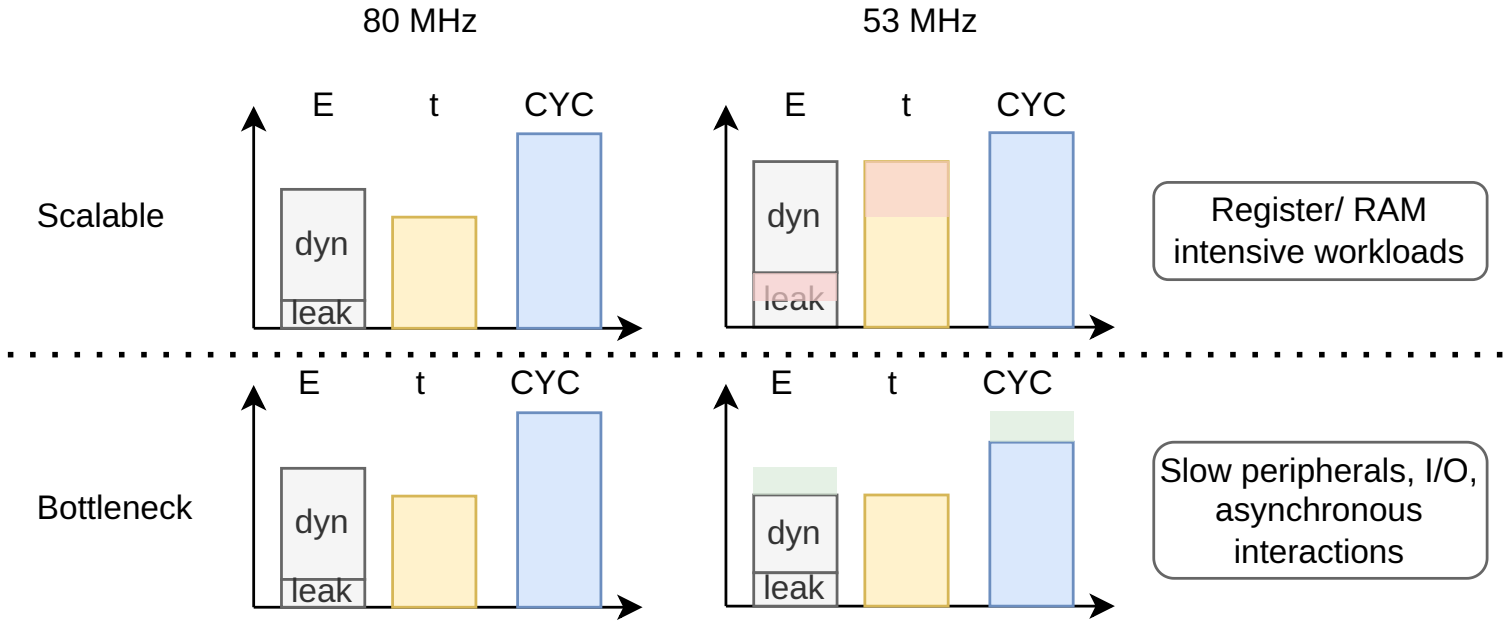
P_{dyn} P_{leak}



Hardware-specific | Task-specific | Adjustable | Adjustable | Hardware-specific

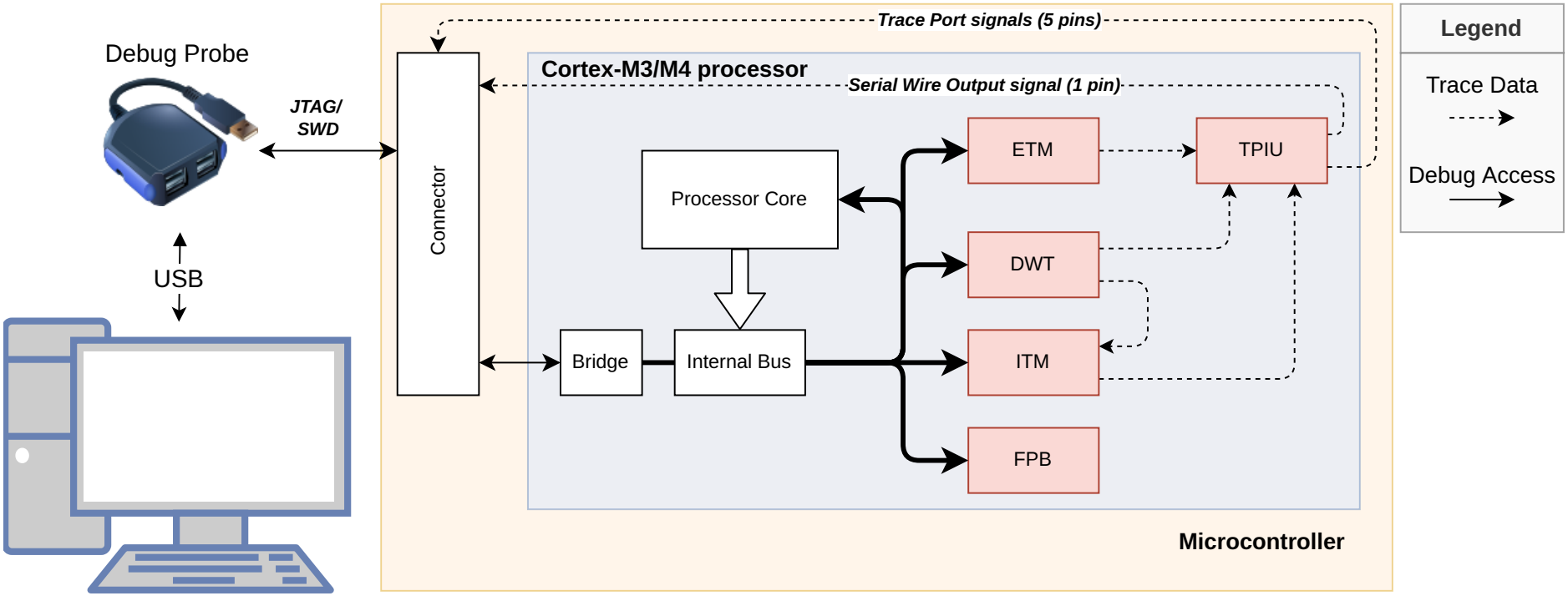
Energy Efficiency

$$t = \frac{CYC}{f} \Rightarrow E_{total} = \overbrace{\alpha \cdot C \cdot CYC \cdot V^2}^{P_{dyn}} + \overbrace{V \cdot I_{leak} \cdot N \cdot k_{design} \cdot \frac{CYC}{f}}^{P_{leak}}$$

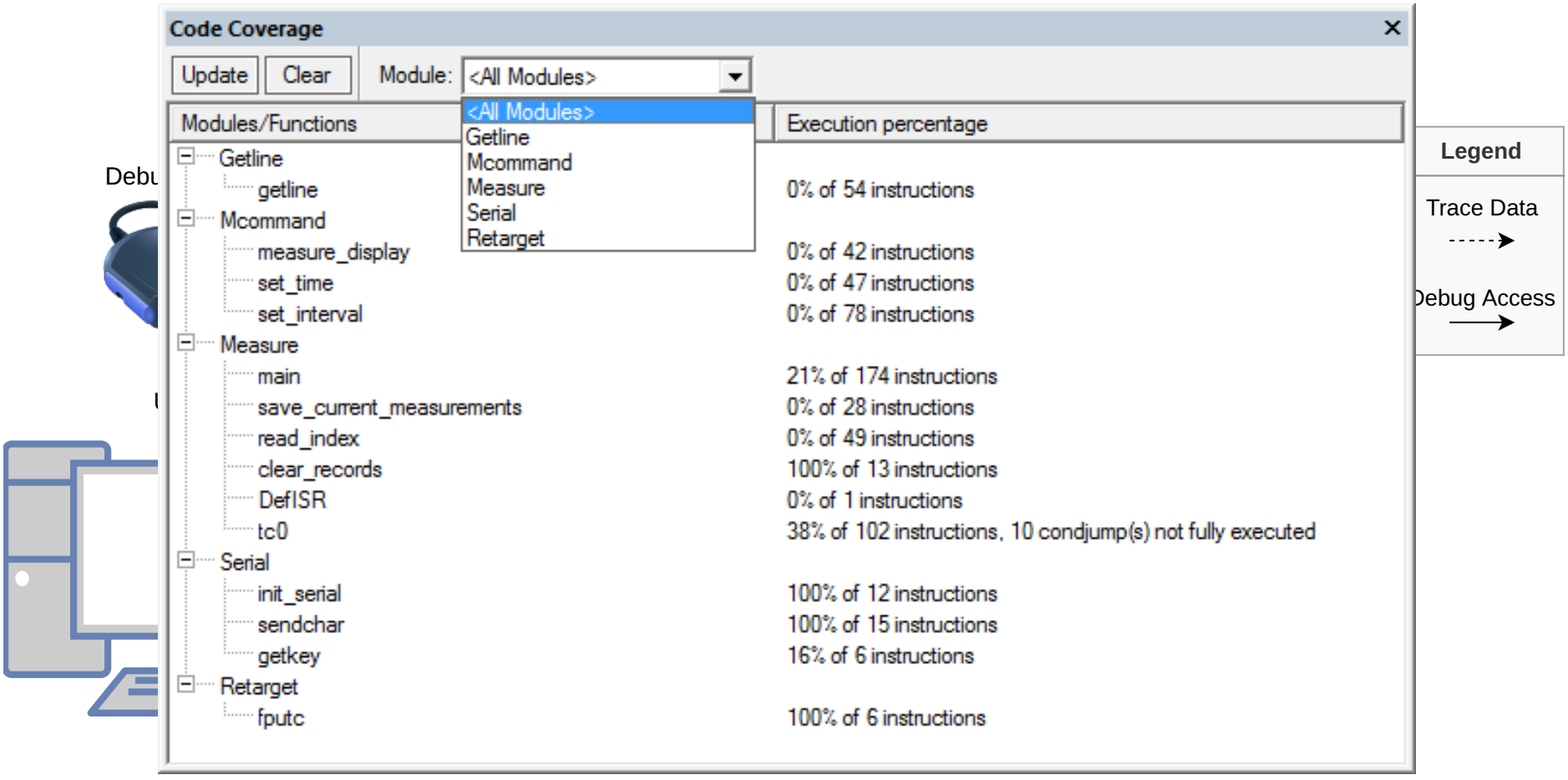


Outline

Debug and Trace Features - Common Use



Debug and Trace Features - Common Use

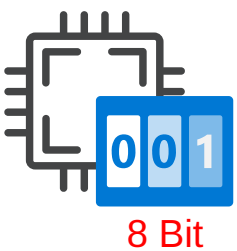


Requirements for Selecting Trace Features

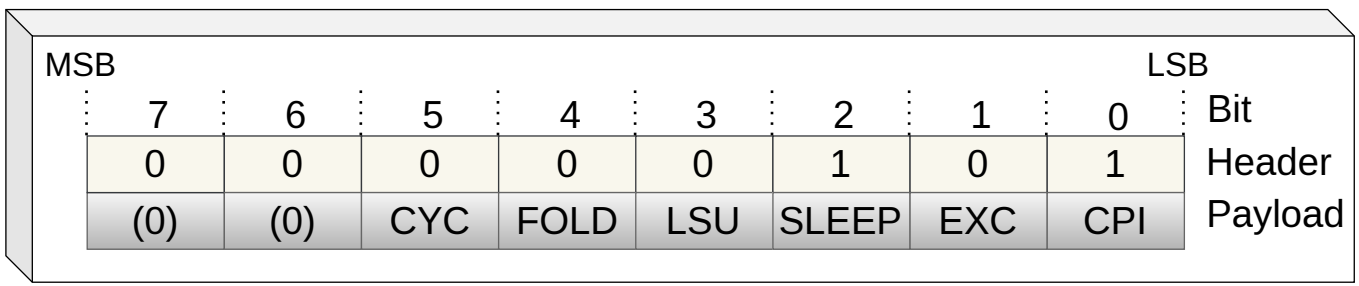
1. Potential indicator of performance utilization
2. Not invasive in terms of CPU load or program execution
3. No task customization or prior task knowledge

DWT Profiling Counter

Counter Type	Increments on ...
LSU	**additional cycles** required to execute **load or store instructions**.
CPI	**additional cycles** required to execute **multi-cycle instructions** (not LSU cycles)
CYC	**each processor clock cycle.**



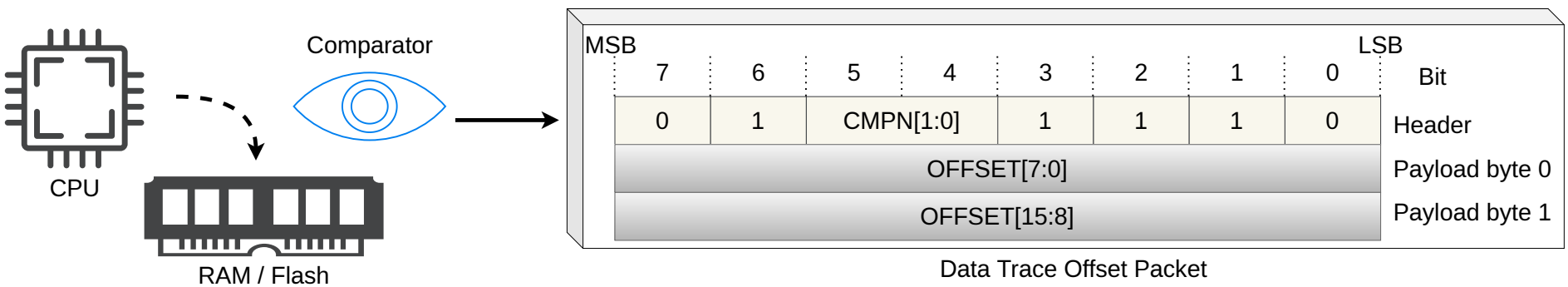
overflow



Event Counter Packet

DWT Comparator

- listens for CPU access to memory addresses (DWT_COMP)
- Data Access → Data Trace Packet (DWT_FUNCTION)
- Instruction Access → CMPMATCH Event (DWT_FUNCTION)
- Range of addresses (DWT_MASK)
- COMP Amount and Mask → observable range
- Indicator: detect slow memory (Flash)

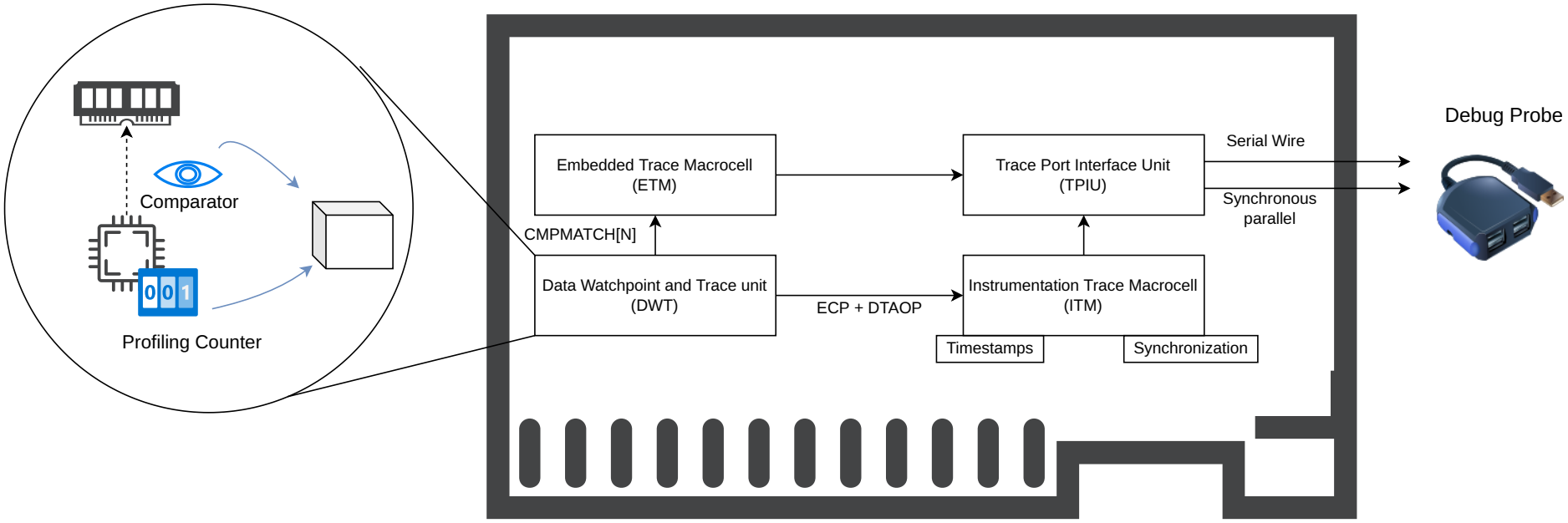


Data Trace Offset Packet

Accessibility

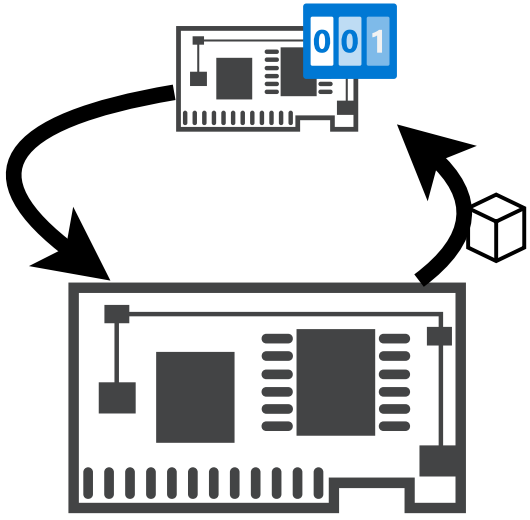
- Profiling Counter → 8 Bit Registers / Packets (SWO / Trace Port)
- Comparators → Packets (SWO) / ETM (Trace Port)

→ Problem: not directly accessible via software at runtime

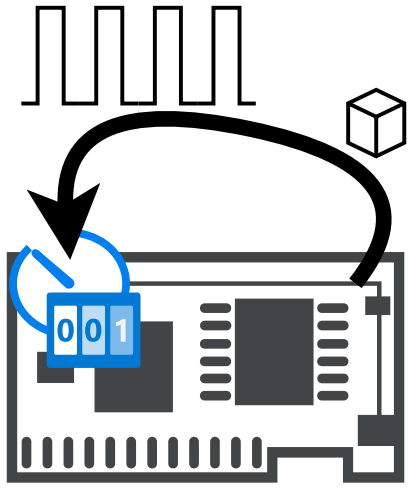


Outline

Discussion: Access to Trace Features



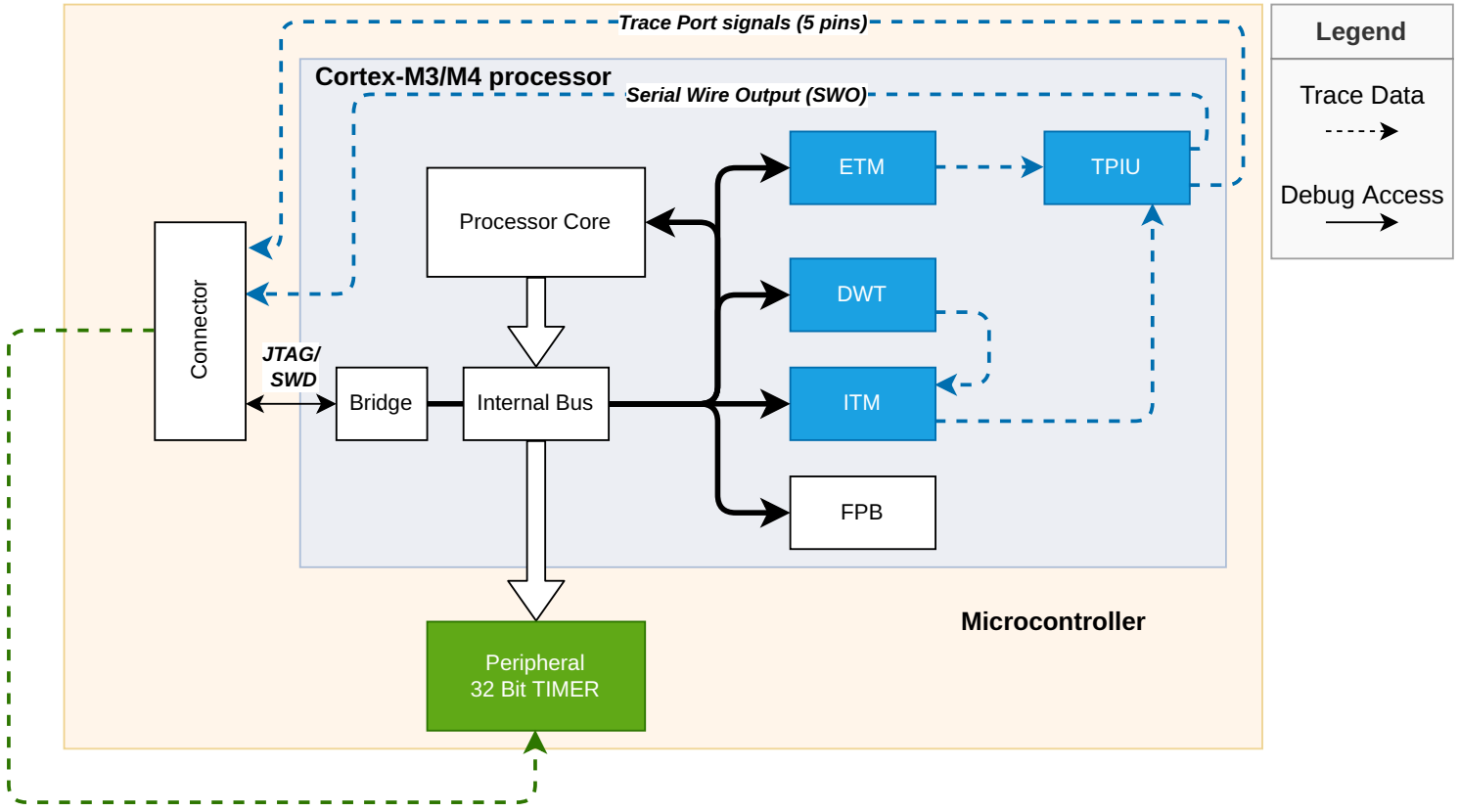
Deserialization Probe

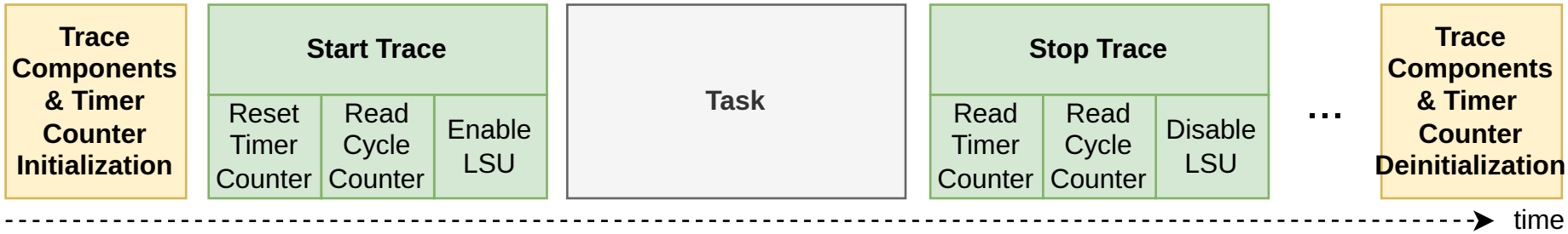


Packet Flank Counter

+ preserves non-invasiveness	− packets not deserialized
+ simultaneous feature tracing	+ preserves non-invasiveness
− high effort	+ timer counter is common [4]
− user needs special hardware	

Concept of Feedback Mechanism & Software Trace



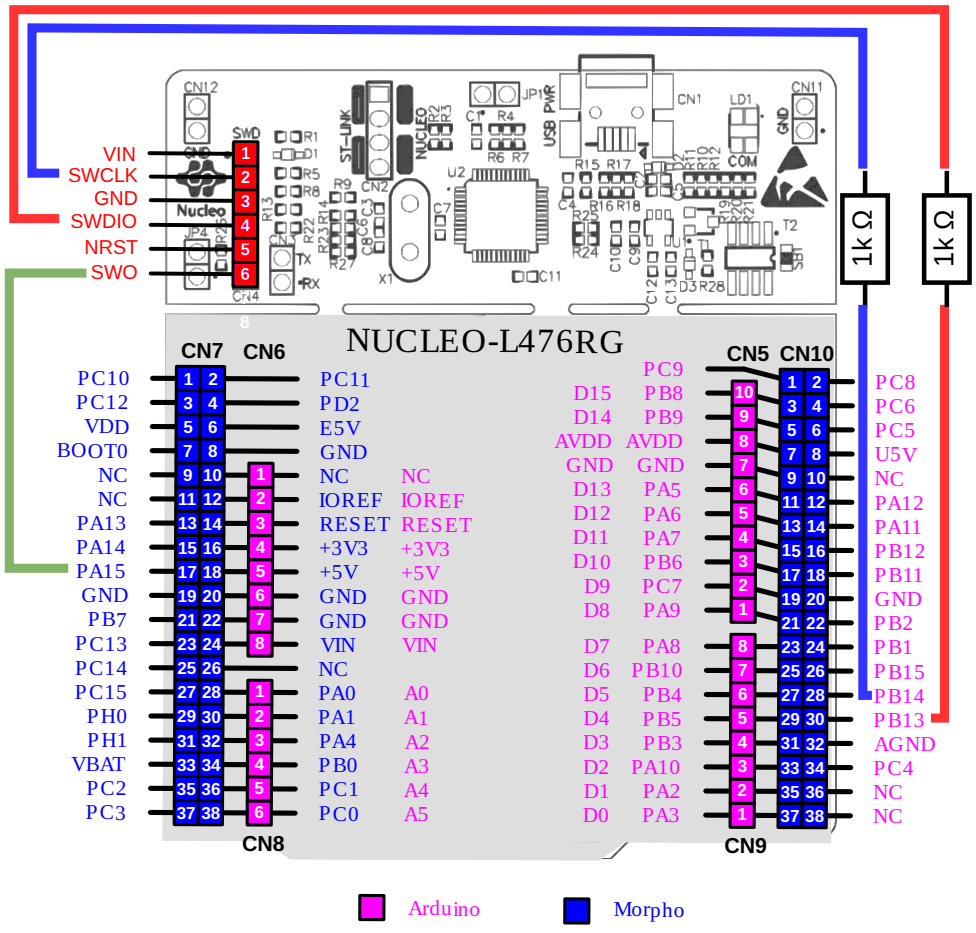


Wiring on Nucleo-L476RG

Connection:

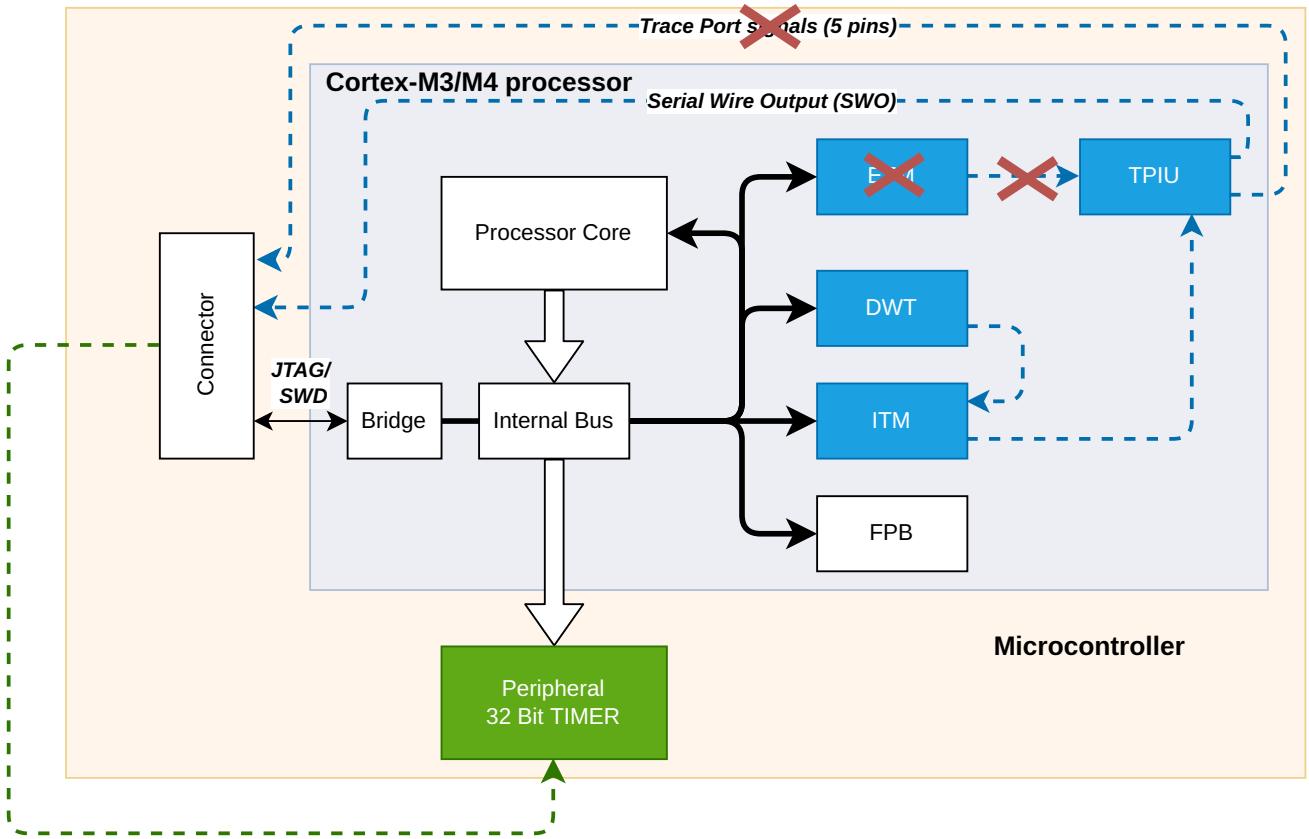
- Trace data line (short = resistance reduction)
- SWDP switch (hack, resistors = short-circuit Protection)

→ low hardware overhead



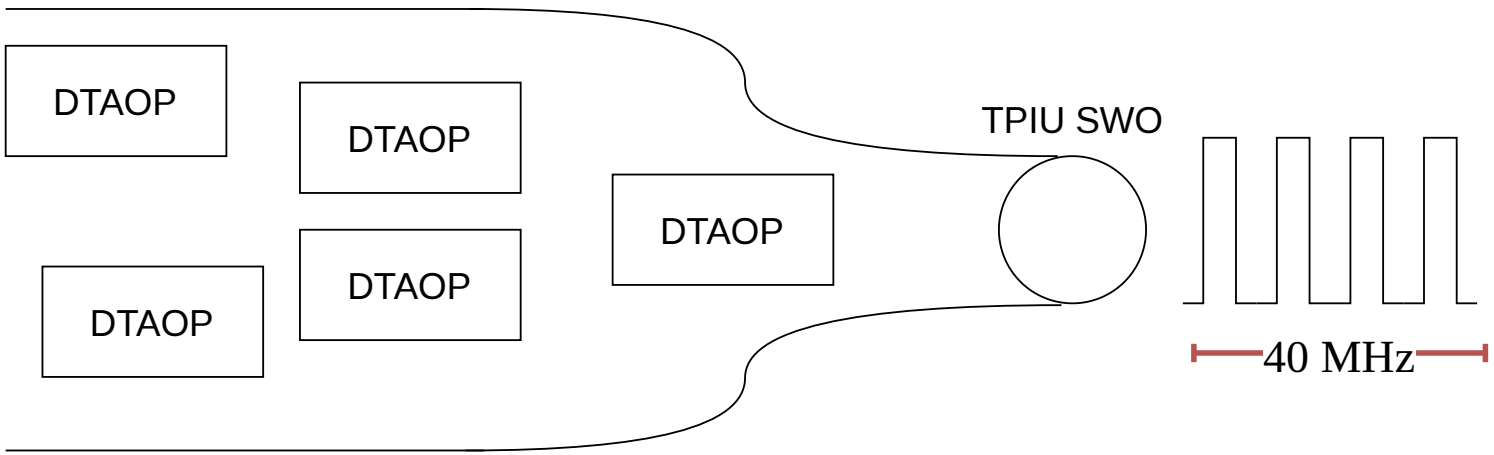
Implementation Constraints

- higher bandwidth Trace Port pins not accessible on MCU of Nucleo-L476RG (Board Choice) → no Instruction Address Matching



Implementation Constraints

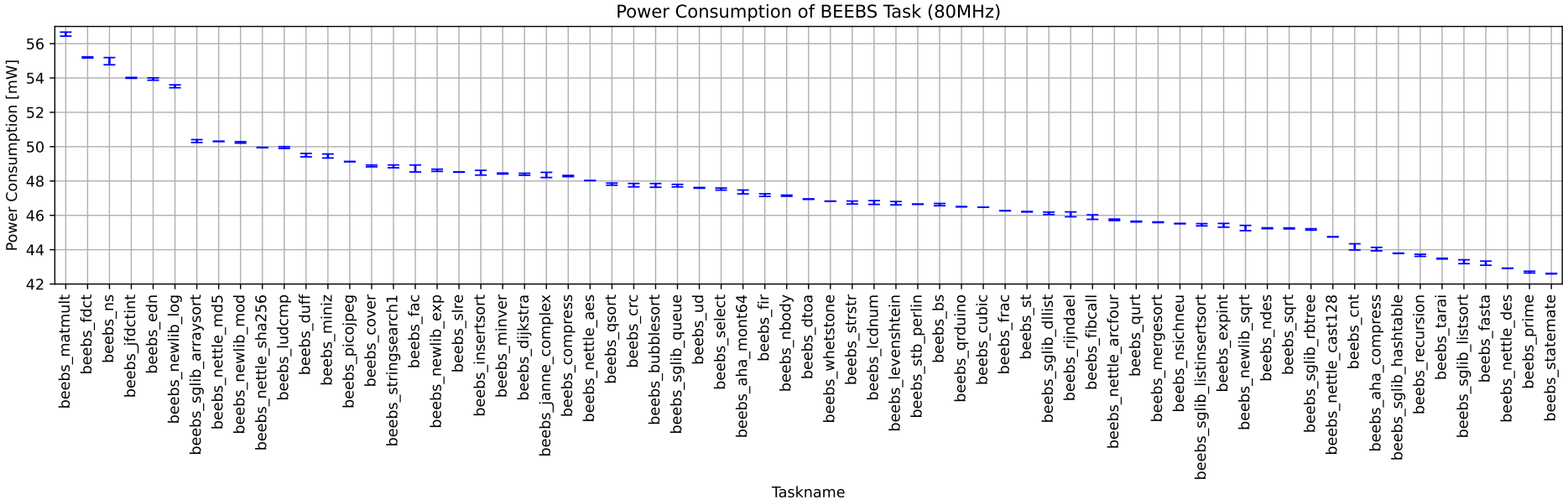
- higher bandwidth Trace Port pins not accessible on MCU of Nucleo-L476RG (Board Choice) → no Instruction Address Matching
- TPIU SWO (40 MHz): ECP ($\frac{80MHz}{256cycles} \cdot 16Bit = 5MHz$)
  DTAOP($\frac{80MHz}{?cycles} \cdot 24Bit = 40MHz \rightarrow 48cycles$)
  → DTAOP Packet congestion (No Trace Port)



Outline

Selection of Benchmark Suite: BEEBS

- target: evaluate energy consumption of embedded processors
- open-source
- 69 different processing tasks
- Areas: Automotive, Consumer, Network, Telecommunication and Security
- I/O tasks are excluded



Trace Data Normalization

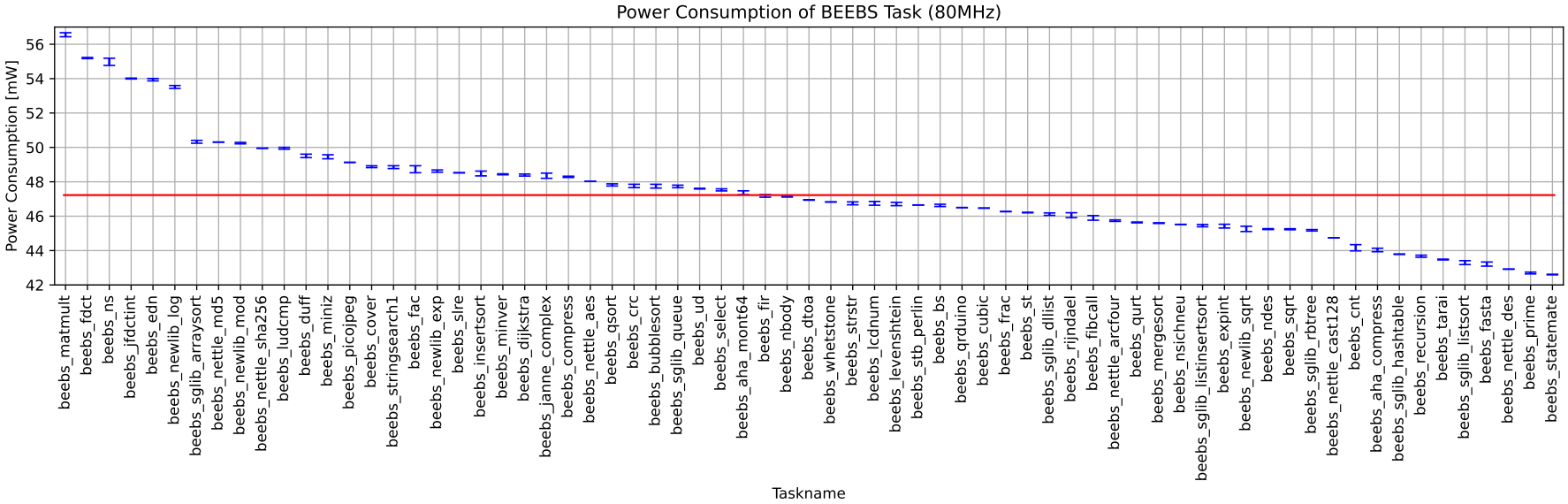
Trace Method	**Unit per Event**	**Measured Flanks per Packet**	**Scaling Factor**
Profiling Counters	256 Cycles	5	256/5
Comparator Address Matching	1 Data Access	8.25	1/8.25

$$P_{normalized} = \frac{CNT \cdot SF}{CYC} \tag{1}$$

Outline

Energy Overhead of Tracing

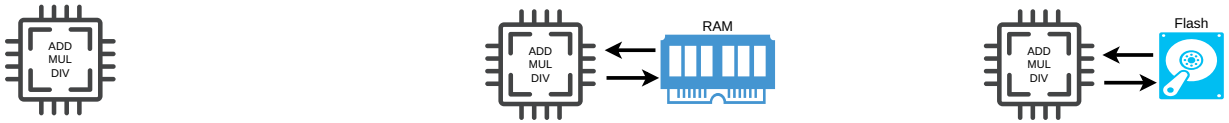
- Flash ADD (46mW), Reg Div (33mW)

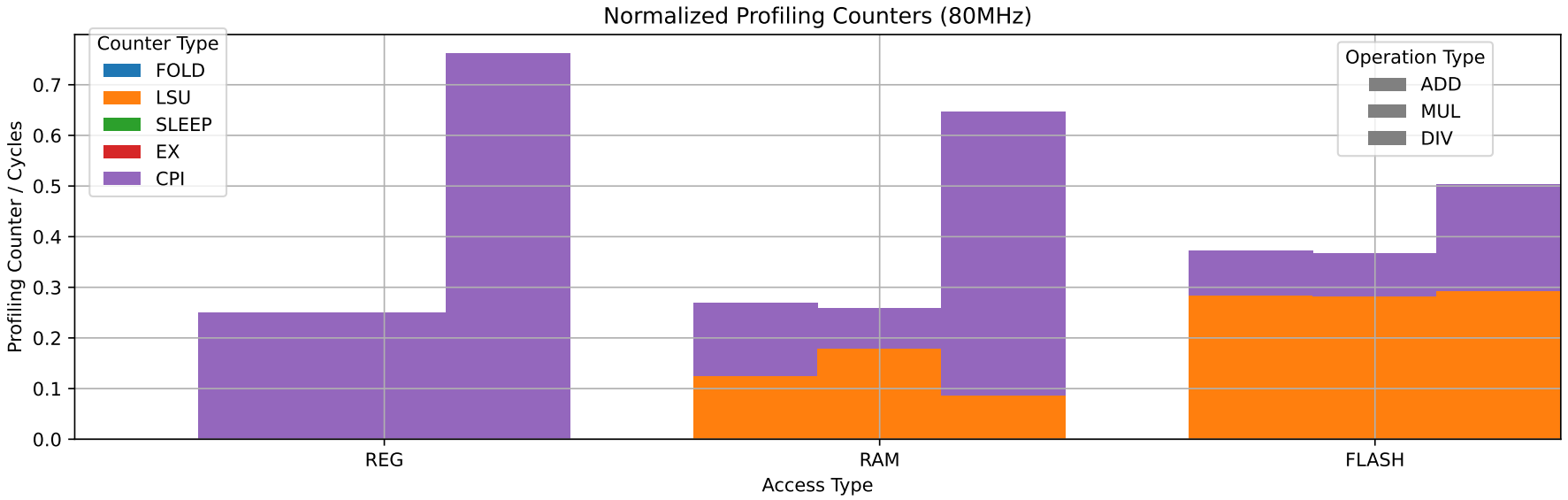


Trace Method	**no generated packets**	**generated packets**
Profiling Counters	6.32% ($\frac{3mW}{47.423mW}$)	8.01% ($\frac{3.8mW}{47.423mW}$)
Comparators	6.5% ($\frac{3.1mW}{47.423mW}$)	16.44% ($\frac{7.8mW}{47.423mW}$)

Outline

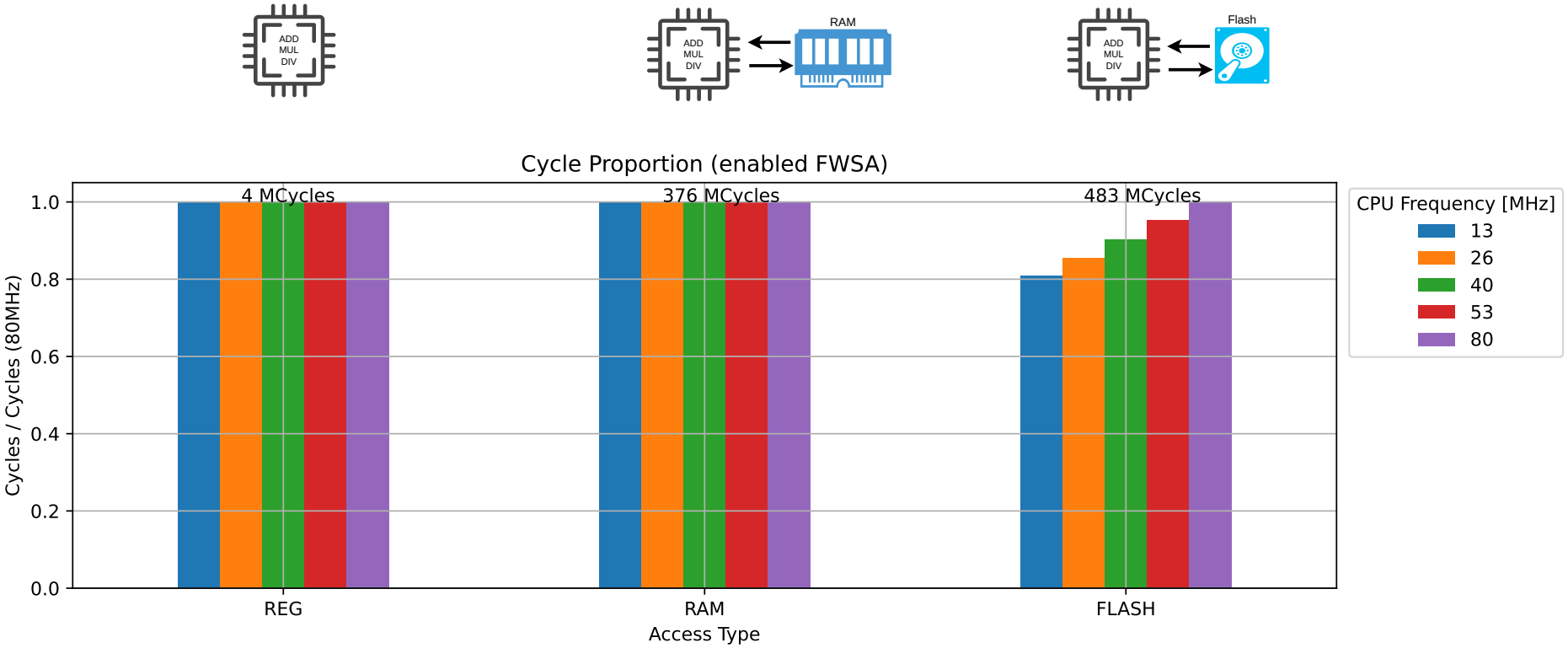
Register, RAM or Flash intensive Workloads





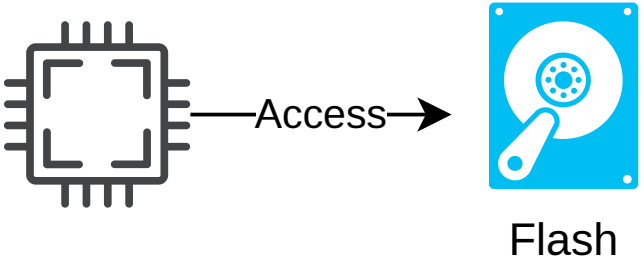
- LSU reacts to RAM and flash memory access
- LSU bigger with flash access - more cycles needed for flash access
- CPI correlates with cost of operations (divide: 2 to 12 cycles, add/multiply: 1 cycle)

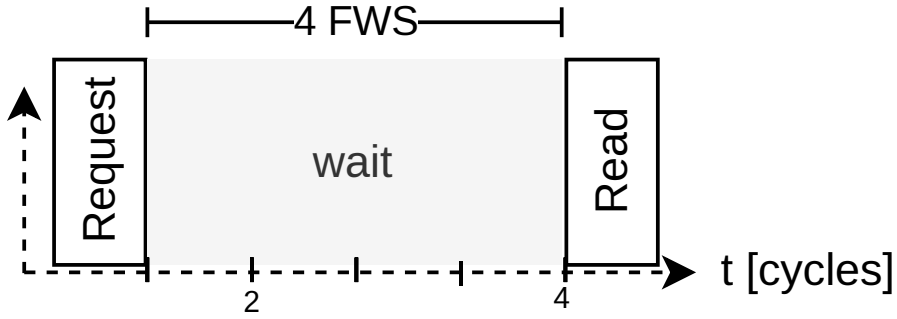
Register, RAM or Flash intensive Workloads

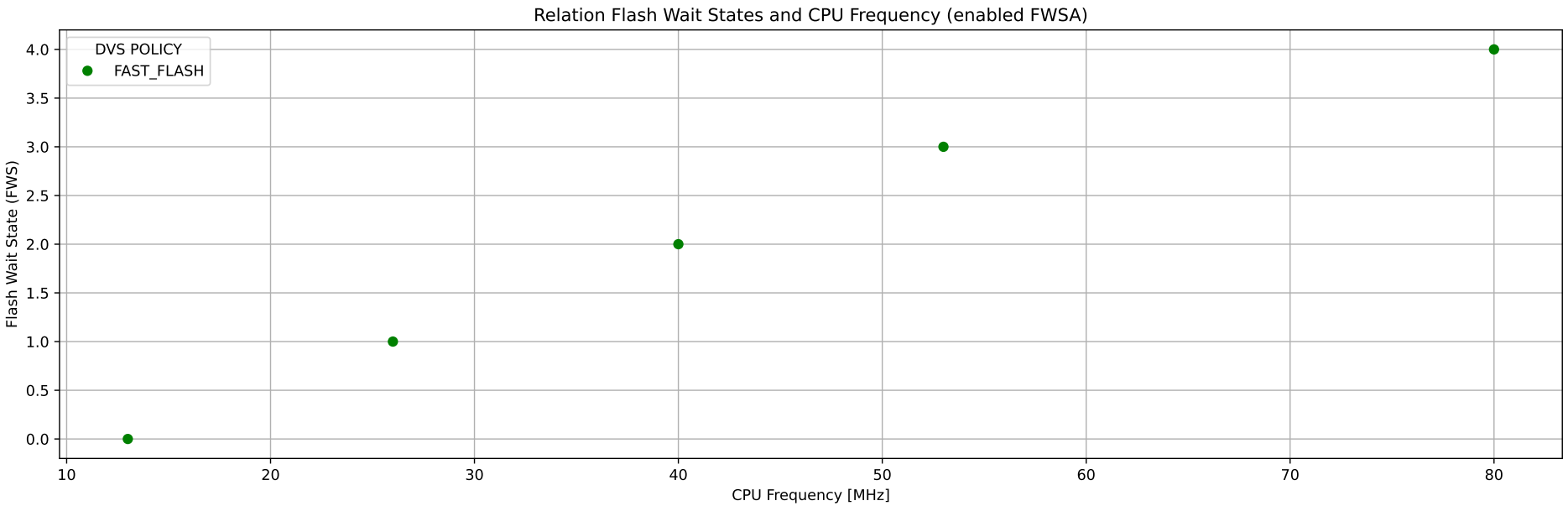


- Cycle savings only with flash access and enabled Flash Wait State Adaption (FWSA)

Register, RAM or Flash intensive Workloads

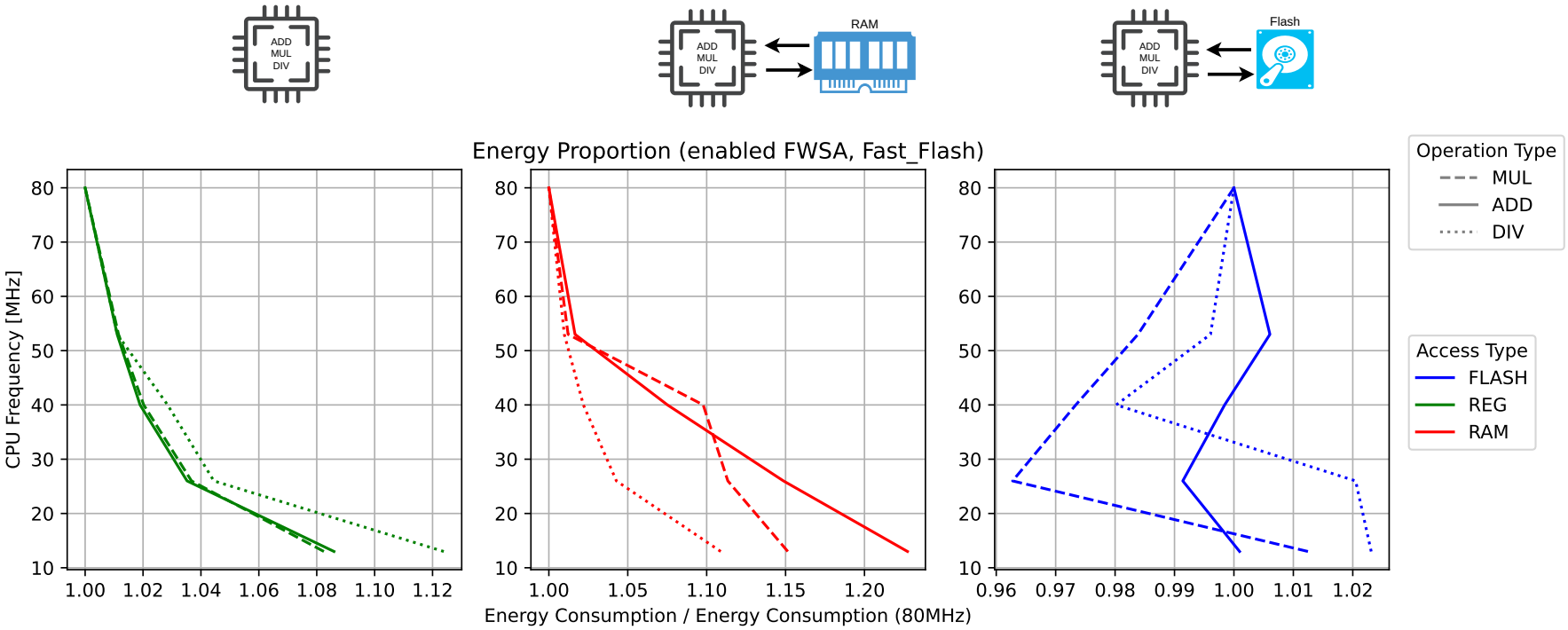






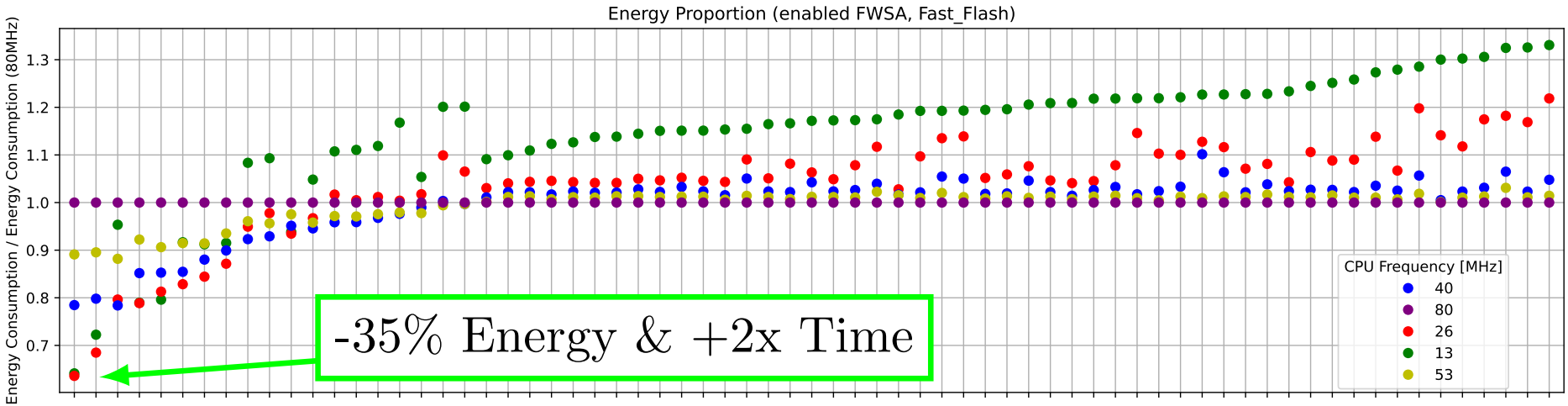
- Every frequency step reduces one FWS

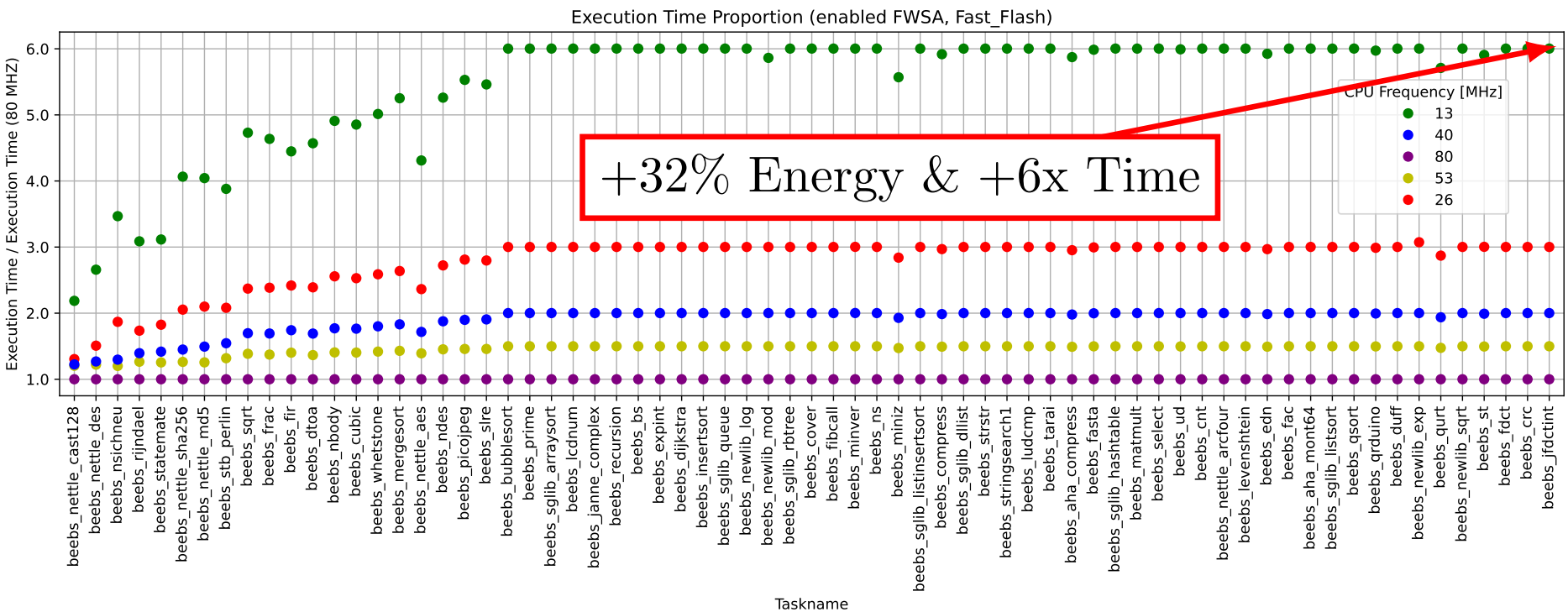
Register, RAM or Flash intensive Workloads



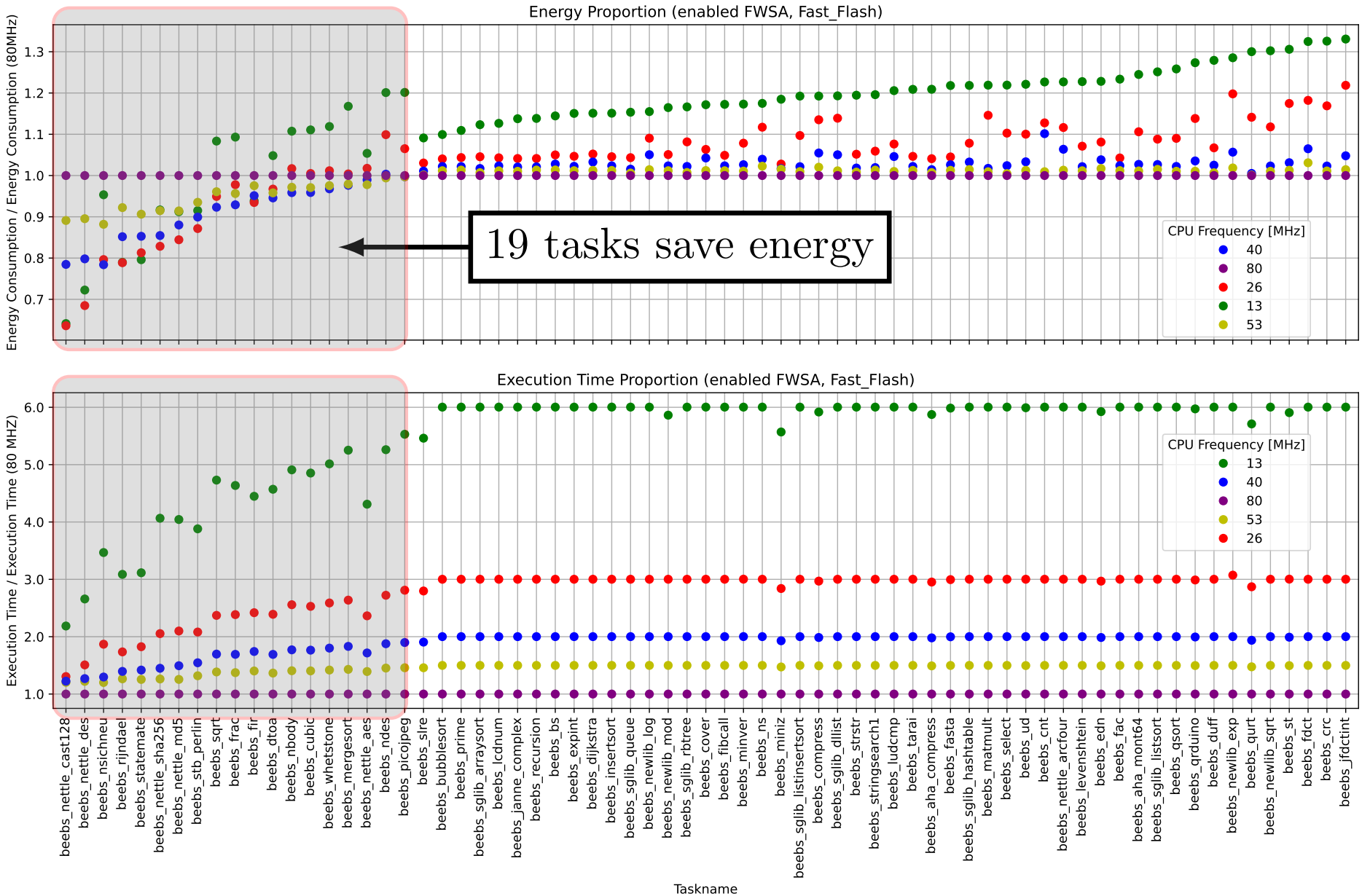
- RAM and Register Workload most energy efficient frequency (MEEF) at 80 MHz
- Flash Workload has lower MEEF (Flash Access, Cycle Reduction)
- Flash: More CPU intensive Operation (divide) raises MEEF

BEEBS Energy Consumption & Execution Time

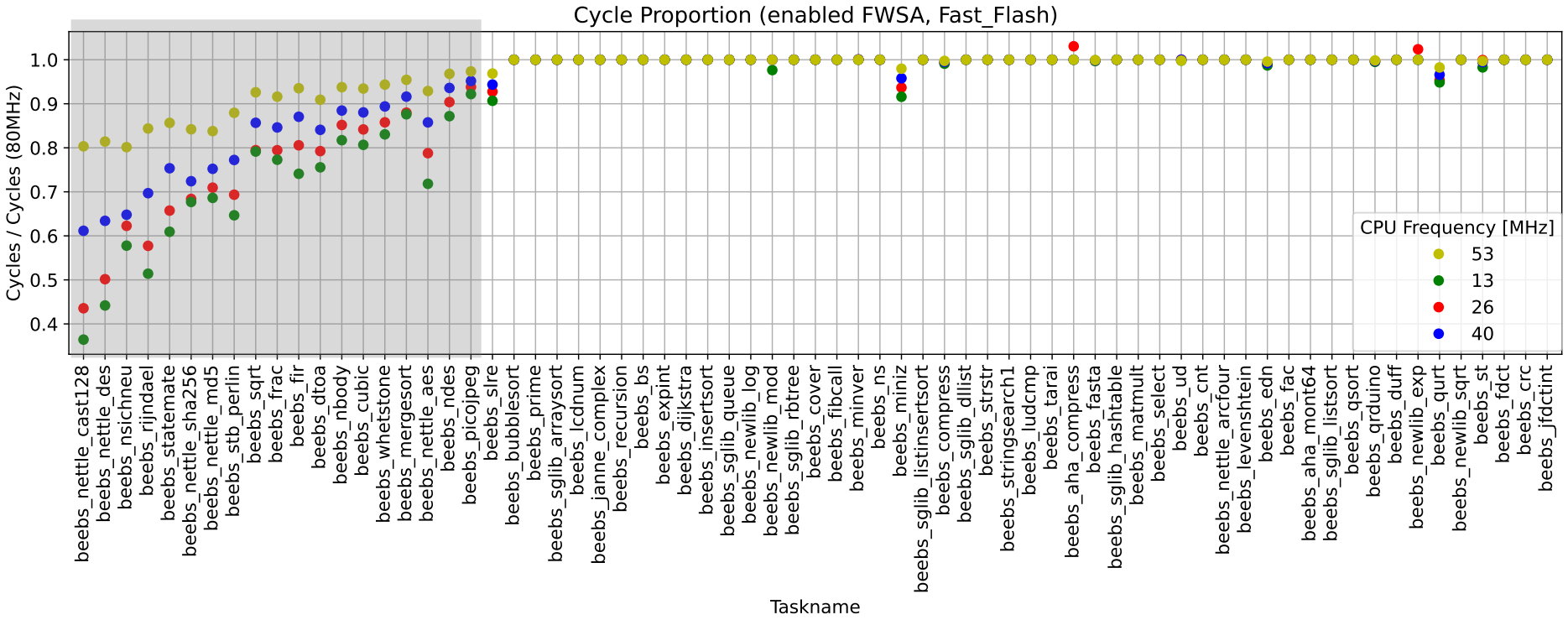




BEEBS Energy Consumption & Execution Time



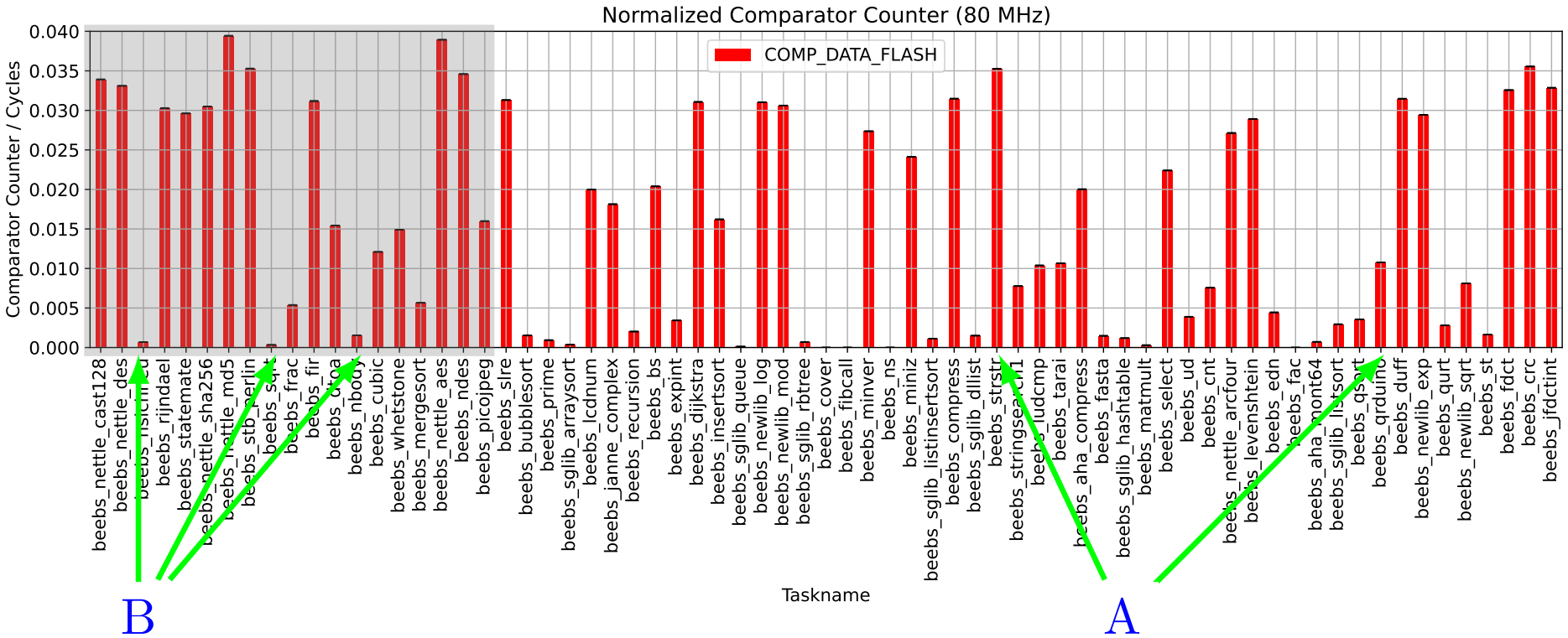
BEEBS Tracing Cycles



- Cycle saving correlates with energy saving $\rightarrow E_{total}$
- Cycle and energy reduction due to lower FWS
- Invasively measuring cycles (2 Frequencies):
  Technique: Cycle Reduction $\rightarrow$ Scale to 40 MHz

$\rightarrow$ 87% ($\frac{43+17}{69}$) tasks most / more energy efficient frequency setting

BEEBS Tracing Flash Access

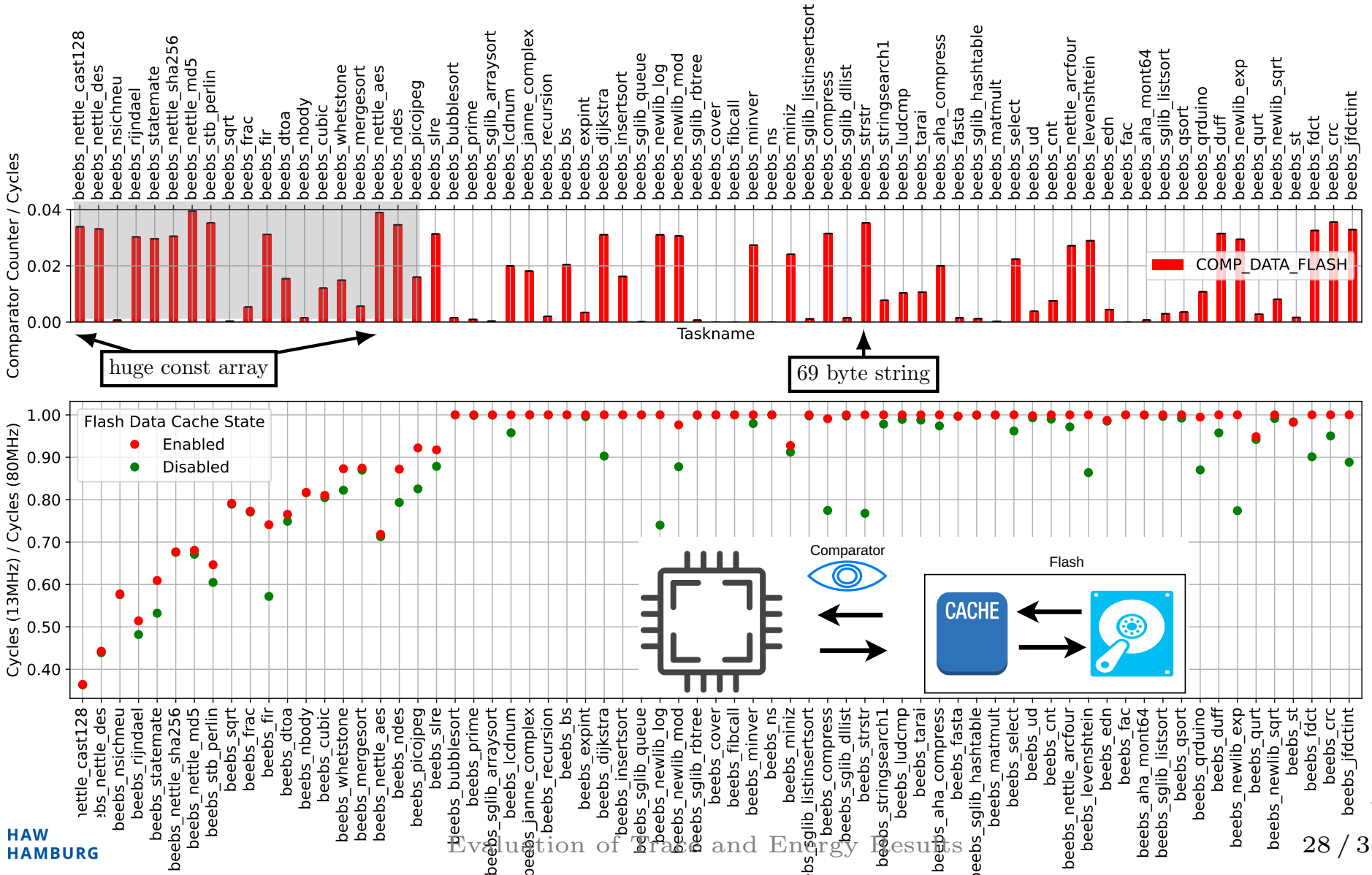


- Energy Savings at lower frequencies → Cycle Saving → FWSA

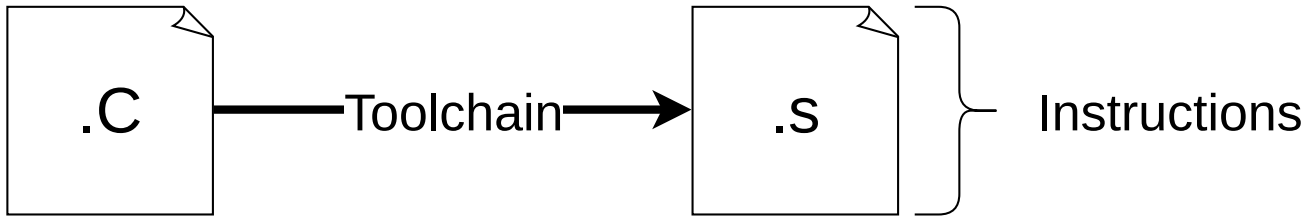
A Why do some tasks show flash usage but no saved cycles?

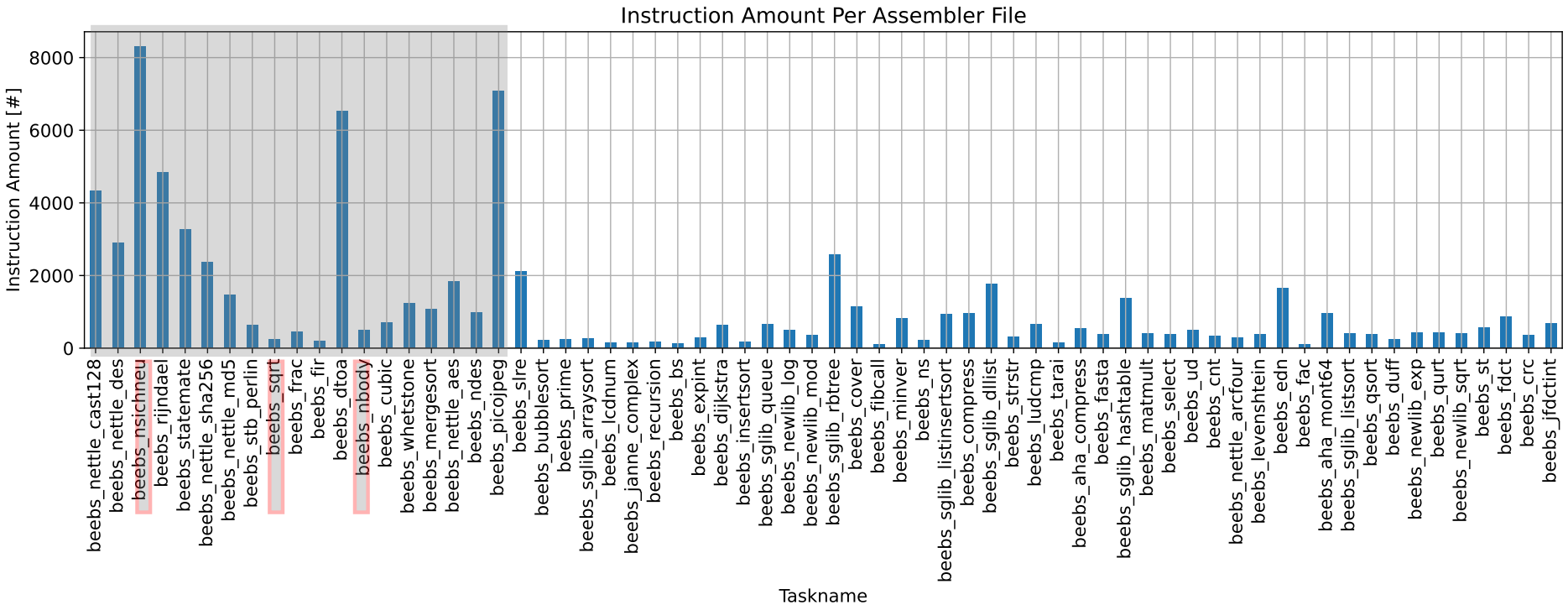
B Why do some tasks show saved cycles but no flash usage?

A Why do some tasks show flash usage but no saved cycles?

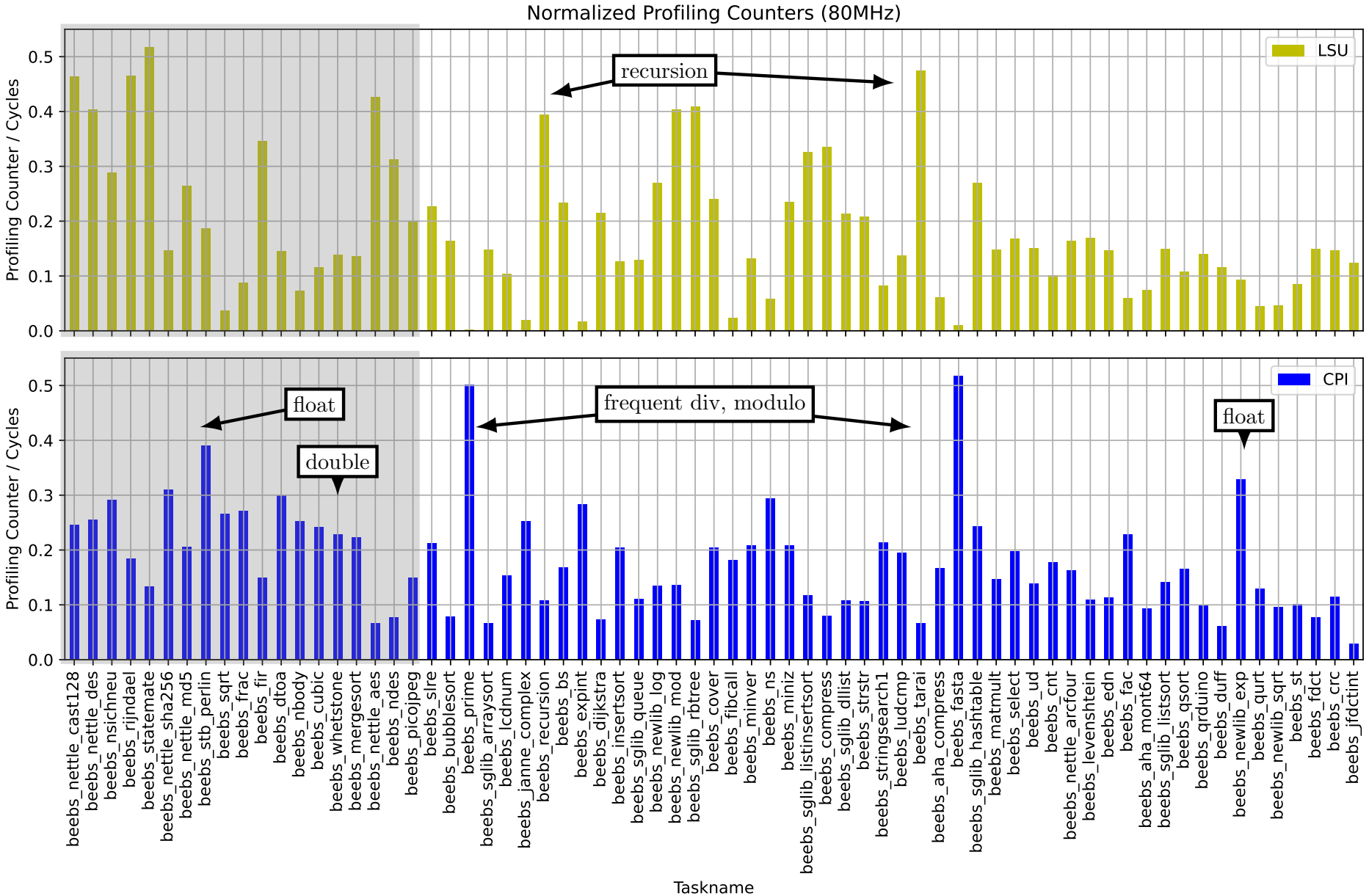


B Why do some tasks show saved cycles but no flash usage?



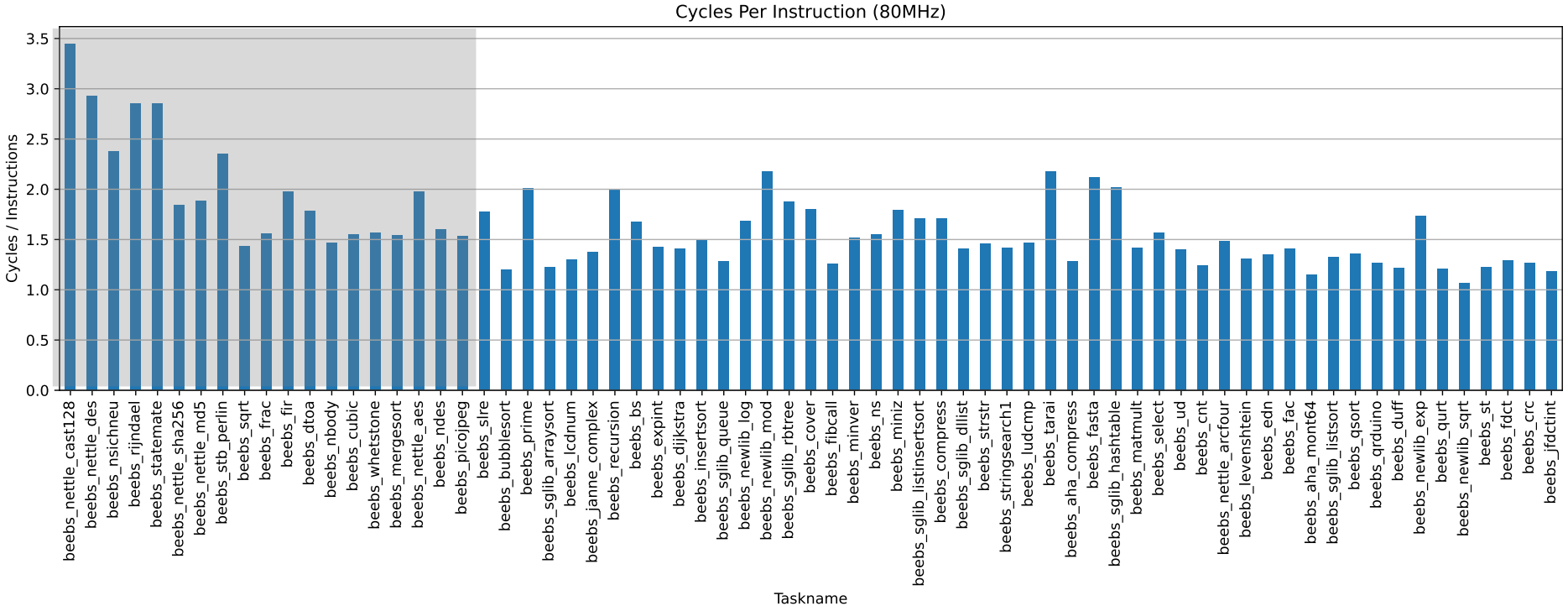


BEEBS Tracing with Profiling Counters

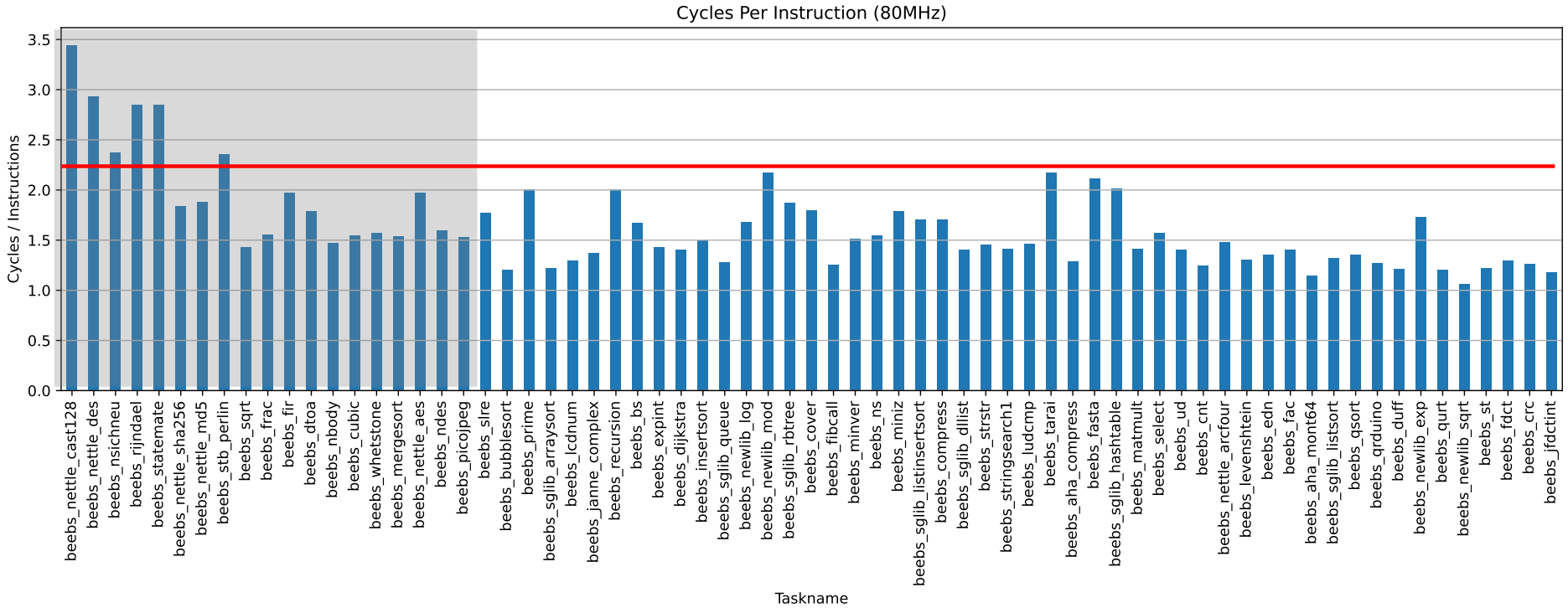


BEEBS Cycles Per Instruction

$$INSTR_{executed} = CYC_{CNT} + FOLD_{CNT} - CPI_{CNT} - EX_{CNT} - SLEEP_{CNT} - LSU_{CNT}$$



BEEBS Cycles Per Instruction



threshold technique:

- $>$ 2,35 cycles per instruction $\rightarrow$ scale to 26 MHz

$\rightarrow$ $\frac{69-13}{69} = 78.7\%$ tasks more/most energy efficient frequency setting

$\rightarrow$ traceable at a single frequency, but multiple tracings steps

Outline

Conclusion I

Traced task properties from debug features?

- task characterization model detects at runtime:
  - flash usage
  - multi-cycle, load/store instructions
  - how many instructions are executed
- feedback mechanism produces low hardware overhead (2 resistors, 3 cables, timer counter) + 6.3% to 16.44% power overhead

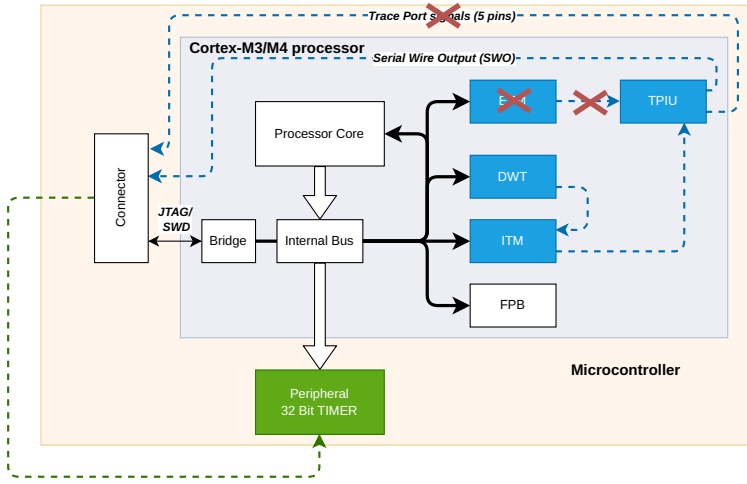
Task properties indicate energy saving tasks?

- invasive cycle saving $\rightarrow$ 19/19 energy saving tasks + 7/50
- cycle per instructions $\rightarrow$ 6/19 energy saving

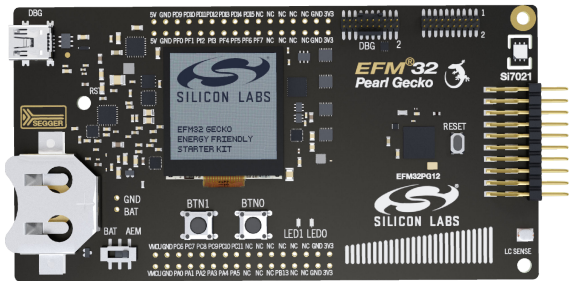
Conclusion II

Problems

- Data flash access $\not\rightarrow$ cycle saving tasks (packet congestion, flash cache)
- Instruction flash access $\not\rightarrow$ cycle saving tasks (No Trace Port)



Conclusion III



Future Work

- Evaluation with SLSTK3402A Board (Parallel Trace Port, ETM Module Accessibility)
- Cycles Per Instruction valuable → Hardware Shield

[5]

Question & Discussion

QUESTIONS?

References I

[1] A. Keil, "Code coverage," https://www.keil.com/support/man/docs/uv4cl/uv4cl_db_dbg_codecoverage.htm, retrieved 2022-11-25.

[2] ——, "µvision user's guide - execution profiler," https://developer.arm.com/documentation/101407/0537/Debugging/Debug-Windows-and-Dialogs/Execution-Profiler, retrieved 2022-11-25.

[3] ——, "Performance analyzer," https://www.keil.com/support/man/docs/uv4cl/uv4cl_db_dbg_perfanalyzer.htm, retrieved 2022-11-25.

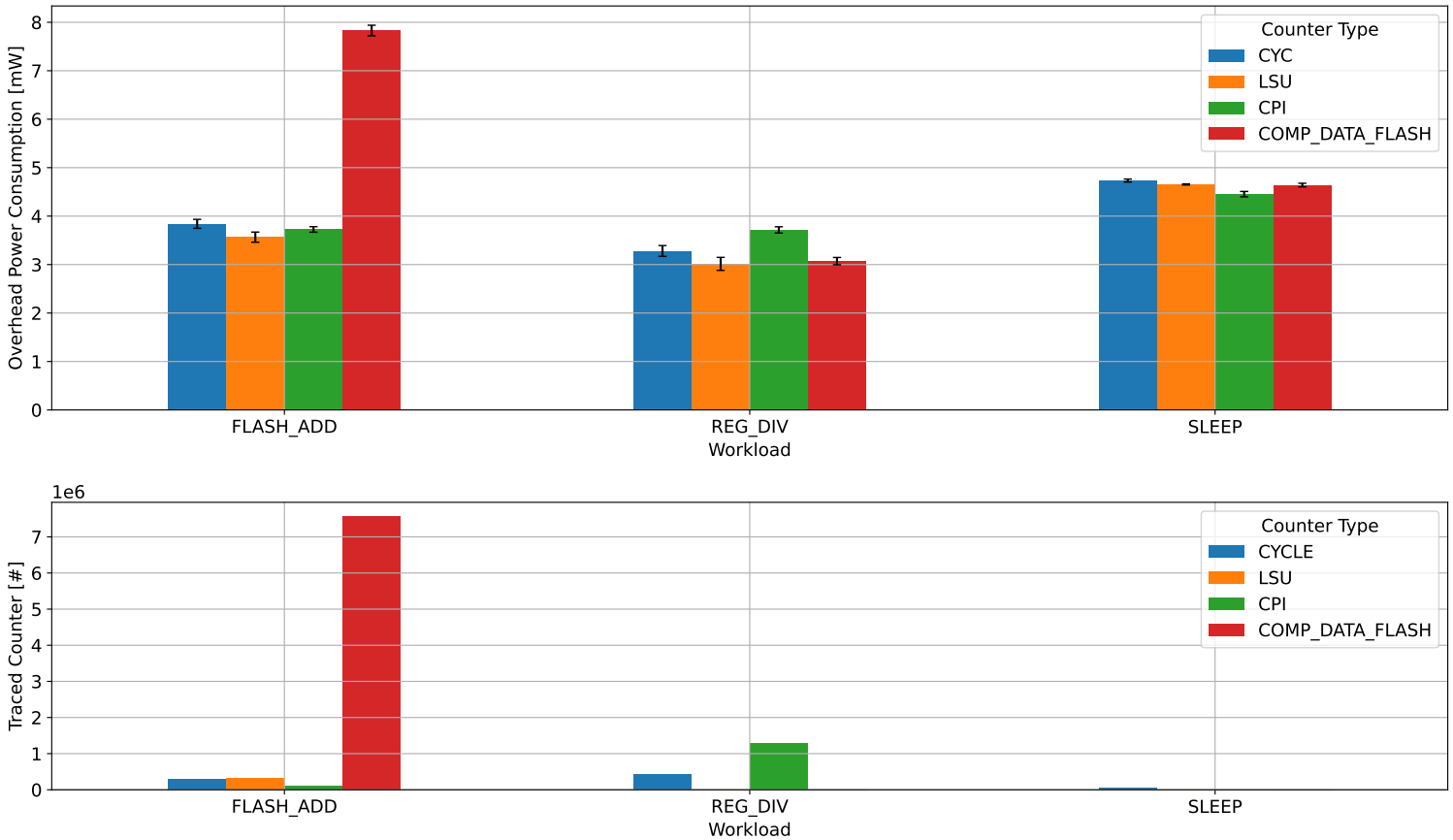
[4] N. Gandraß, M. Rottleuthner, and T. C. Schmidt, "Work-in-Progress: Large-scale Timer Hardware Analysis for a Flexible Low-level Timer-API Design," in *Proceedings of EMSOFT 2021*. New York, NY, USA: ACM, October 2021, pp. 35–36. [Online]. Available: https://doi.org/10.1145/3477244.3477617

References II

[5] *UG257: EFM32 Pearl Gecko PG12 Starter Kit User's Guide*, Silicon Labs, January 2017, Rev. 1.0. [Online]. Available: https://www.silabs.com/documents/public/user-guides/ug257-stk3402-usersguide.pdf

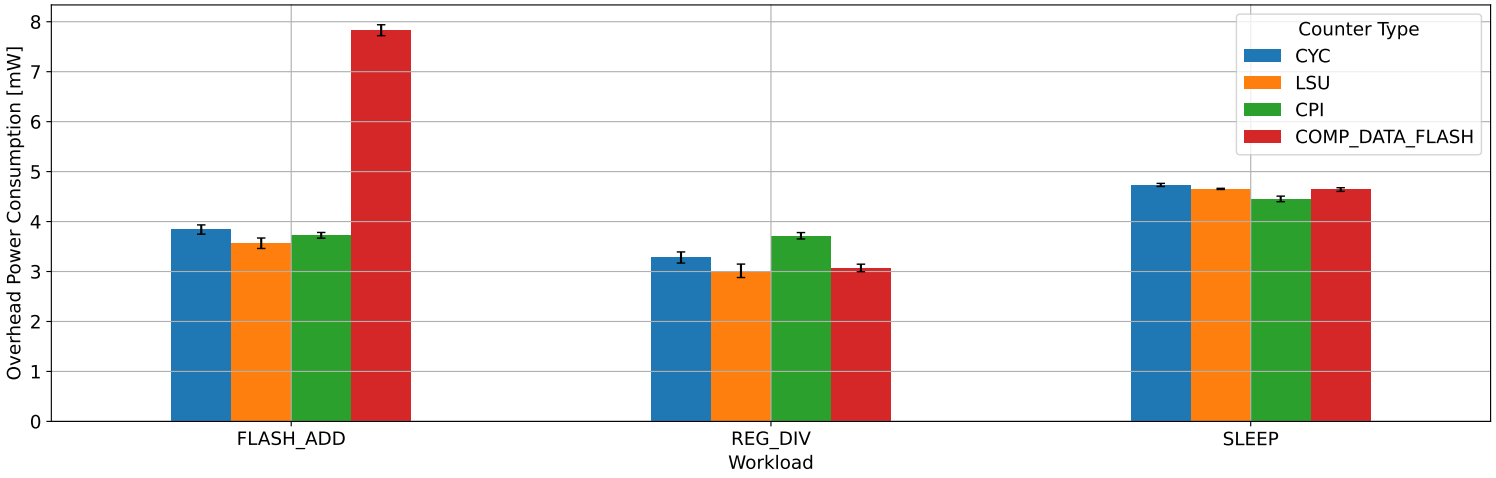
[6] Tektronix, "Dmm7510," https://www.tek.com/de/products/keithley/digital-multimeter/dmm7510, retrieved 2022-12-8.

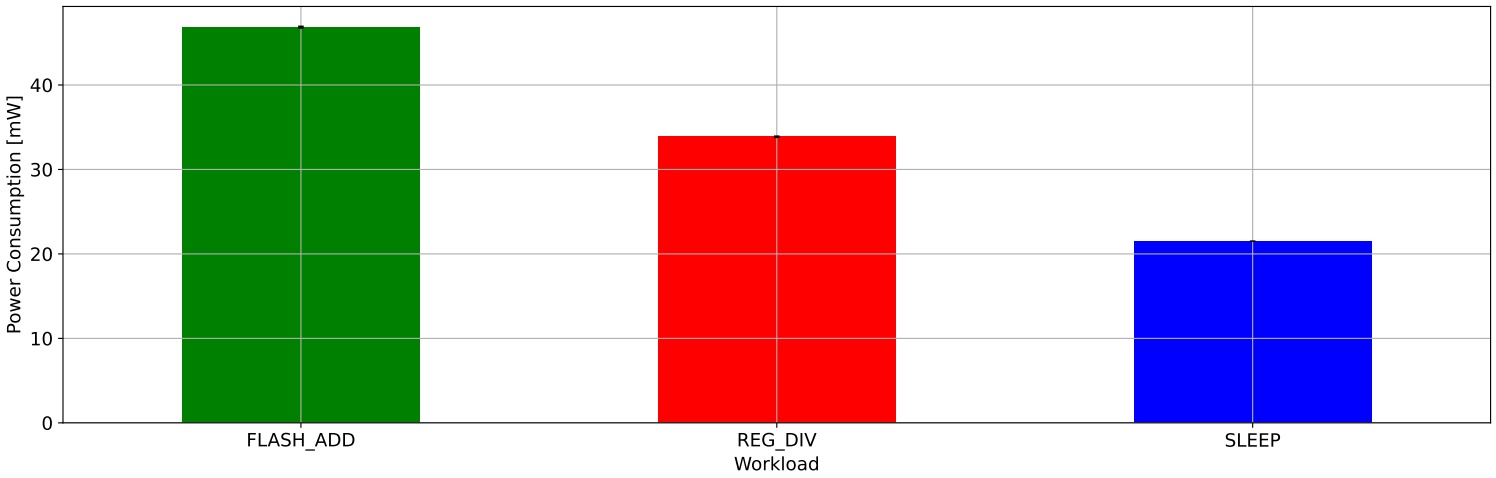
Energy Overhead I



- static overhead power consumption - enabled modules
- dynamic overhead power consumption - higher for comparators

Energy Overhead II



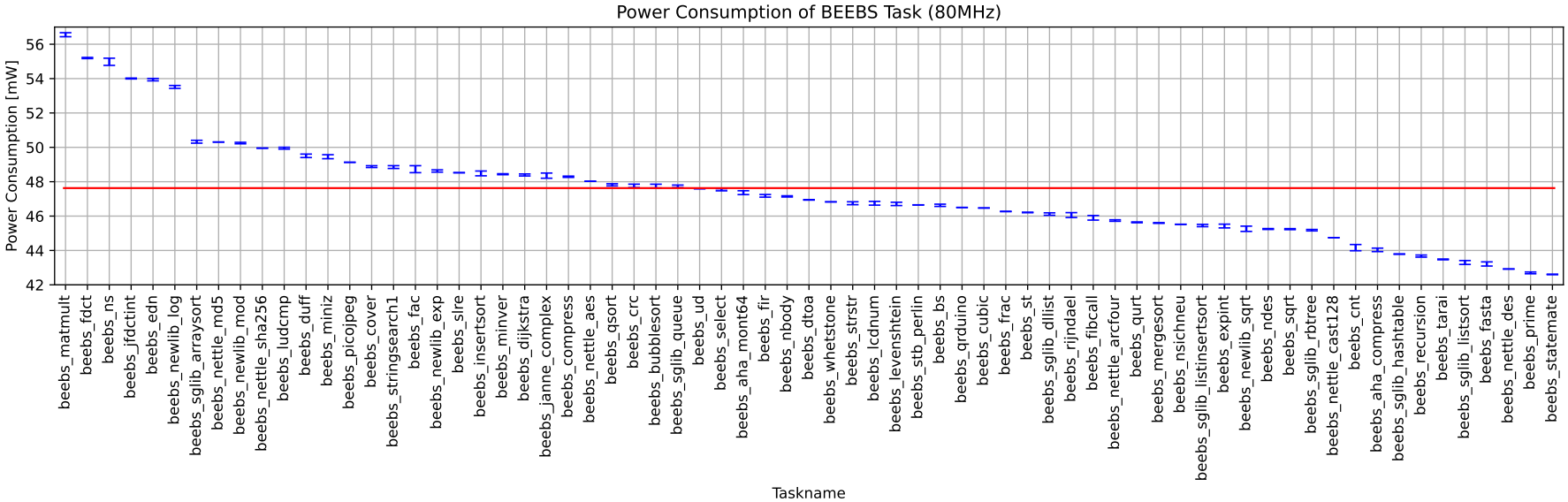


Processing Tasks

- Profiling Counters: 3 to 3.8 mW
- Comparators: 3.1 to 7.8mW

Energy Overhead III

- Flash ADD (46mW), Reg Div (33mW)



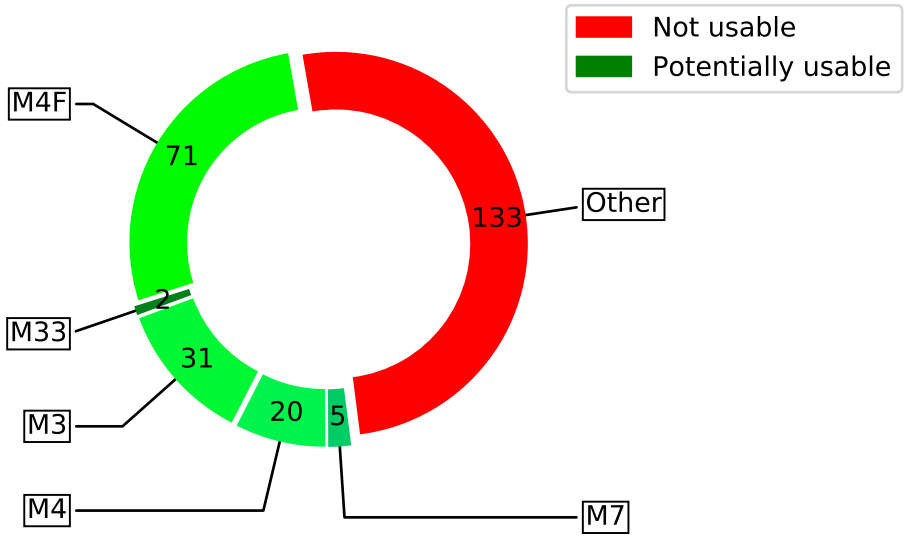
Trace Method	**no generated packets**	**generated packets**
Profiling Counters	6.32% $(\frac{3mW}{47.423mW})$	8.01% $(\frac{3.8mW}{47.423mW})$
Comparators	6.5% $(\frac{3.1mW}{47.423mW})$	16.44% $(\frac{7.8mW}{47.423mW})$

Spread of potential Model use with RIOT

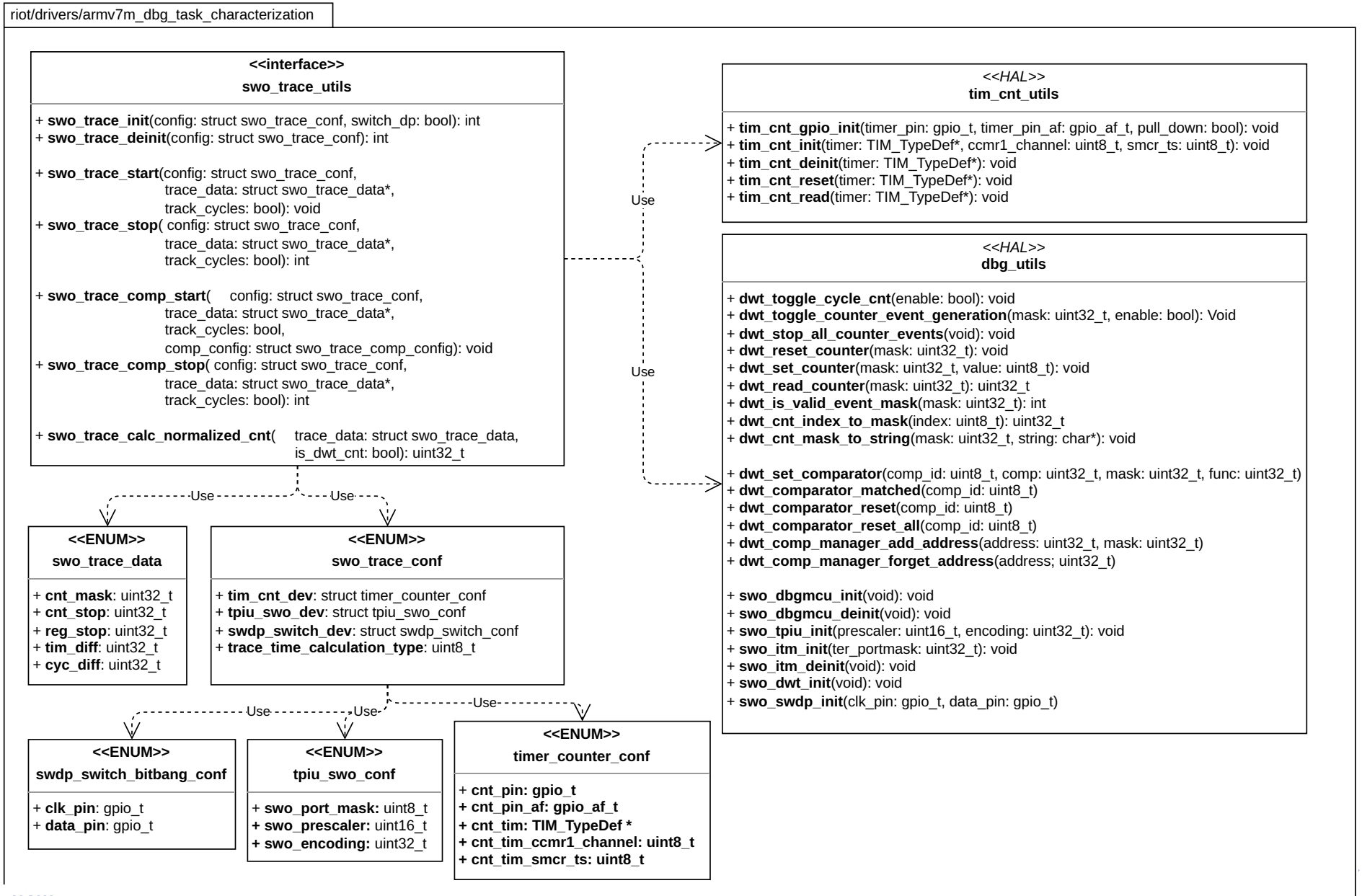
Potential:

- Cortex-M processor (M4, M33, M3, M7)
- optionally having DWT, ETM, TPIU (SWO / Trace Port)

→ 49.23% of RIOT Boards usable

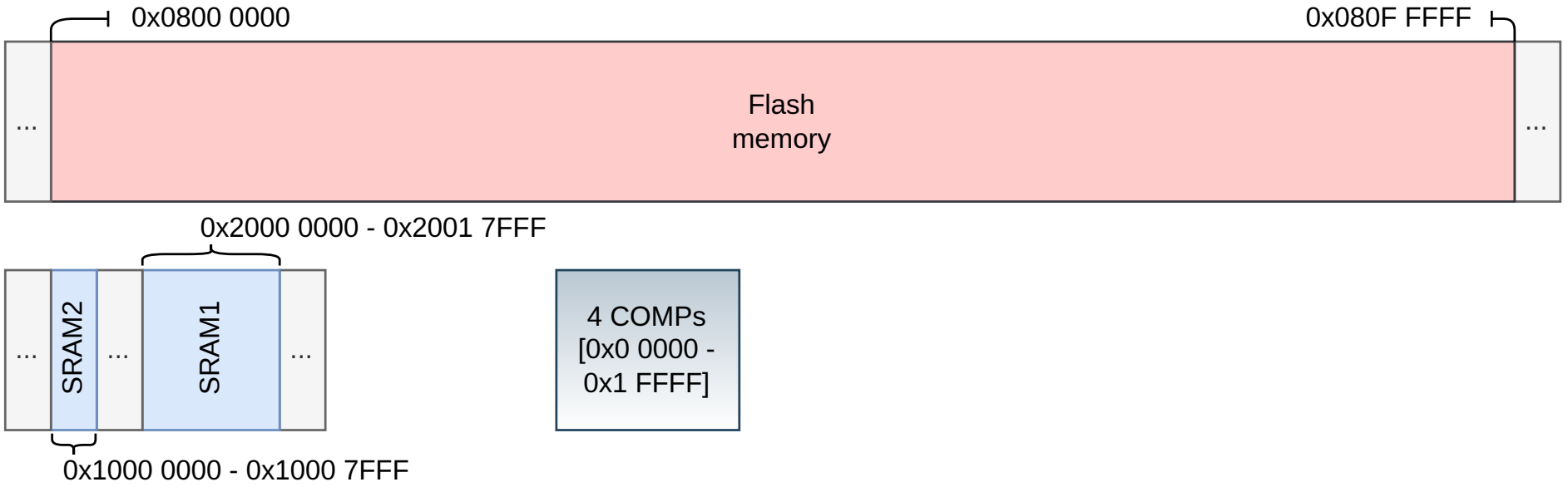


Software Trace - Class Diagram



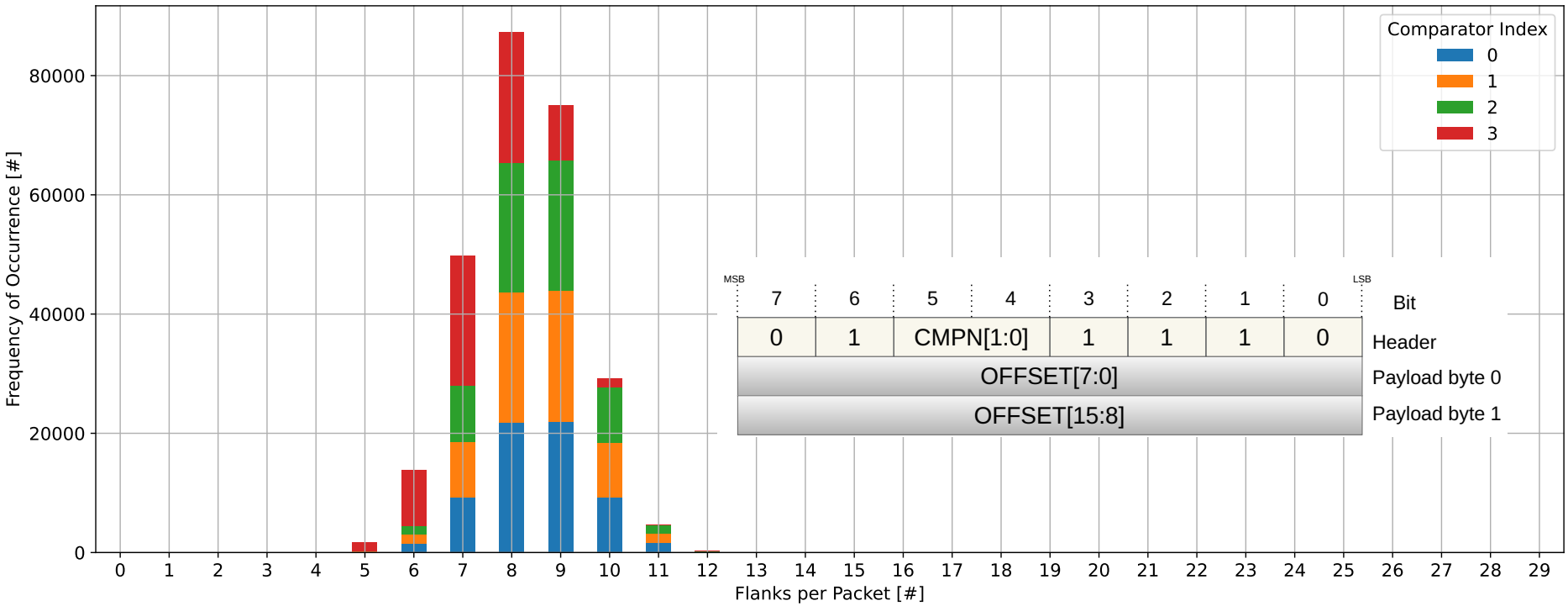
Implementation Constraints I

- 0x20000 Addresses per Trace observable (Board Choice) → complete Flash in 8 iterations traceable



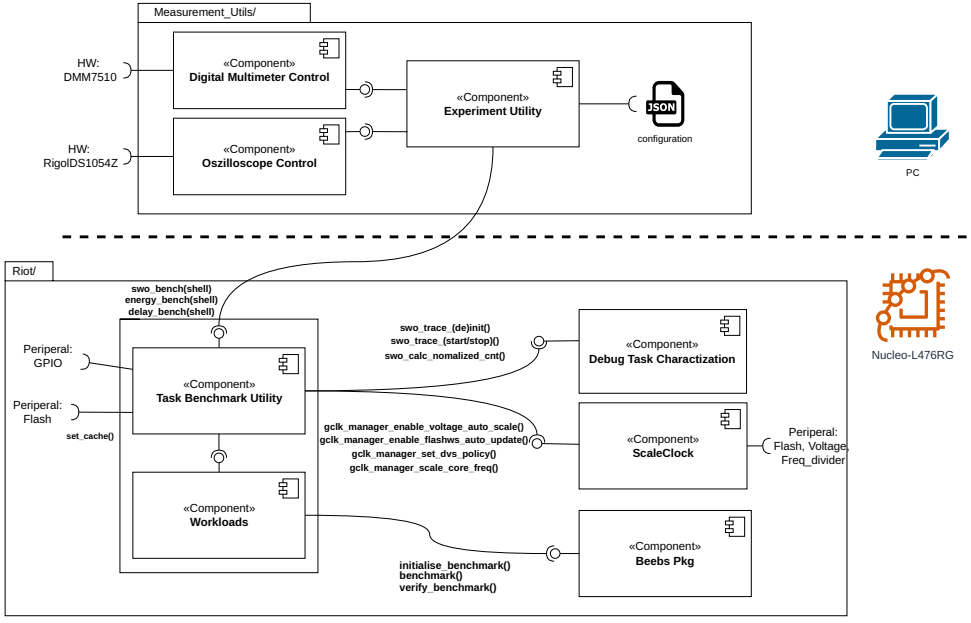
Implementation Constraints II

- Packet flanks: (ECP fixed (5 Flanks), DTAOP varies (8.254 avg))
  → inaccuracy with comparator tracing



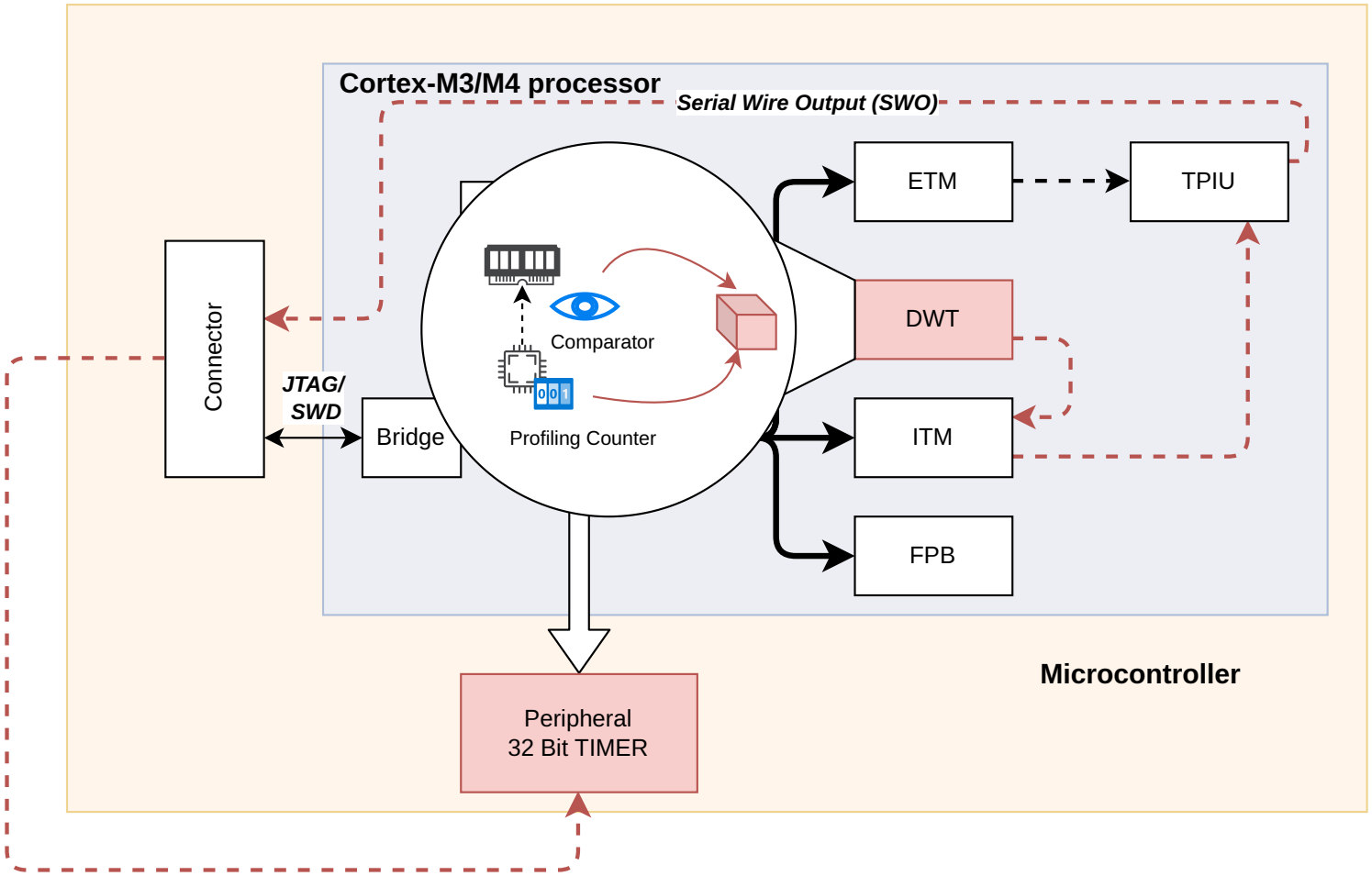
Measurement Setup

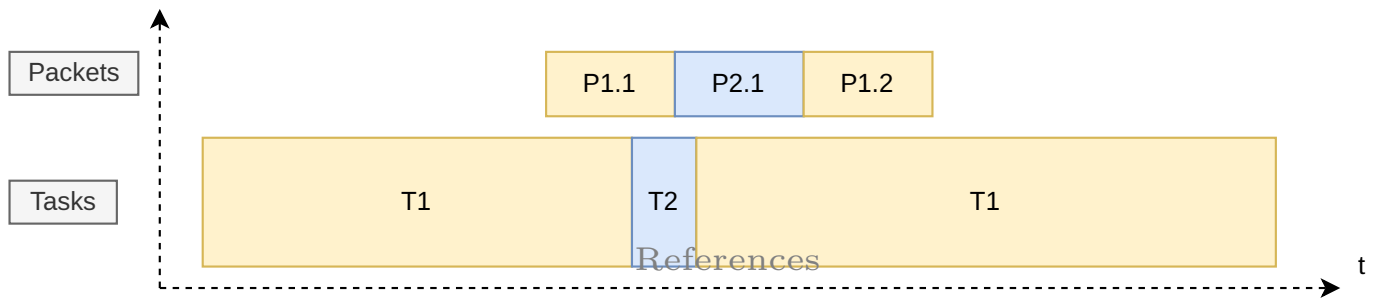
Keithley DMM7510 [6]



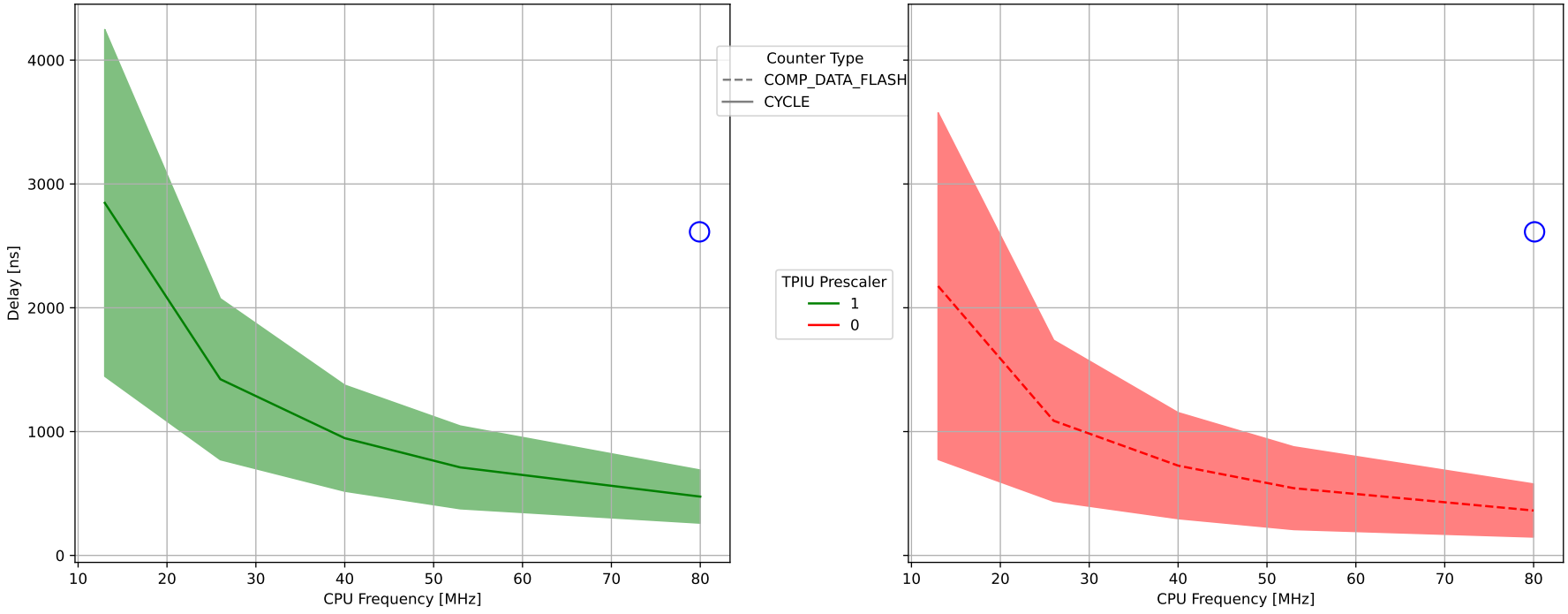
- Reproducibility (reboot)
- Soundness (1s and 10 repetitions, 10000 Current Samples per Second)
- Automation (Benchmark setup)
- Verifiability (Benchmark suite)

Packet Delay



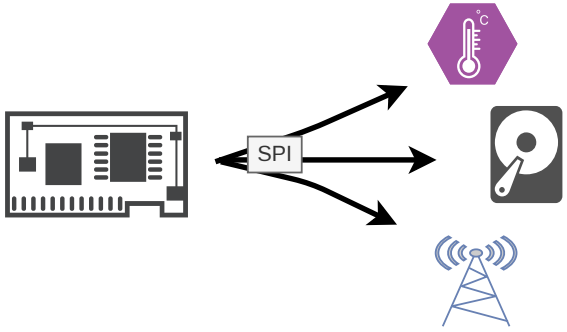


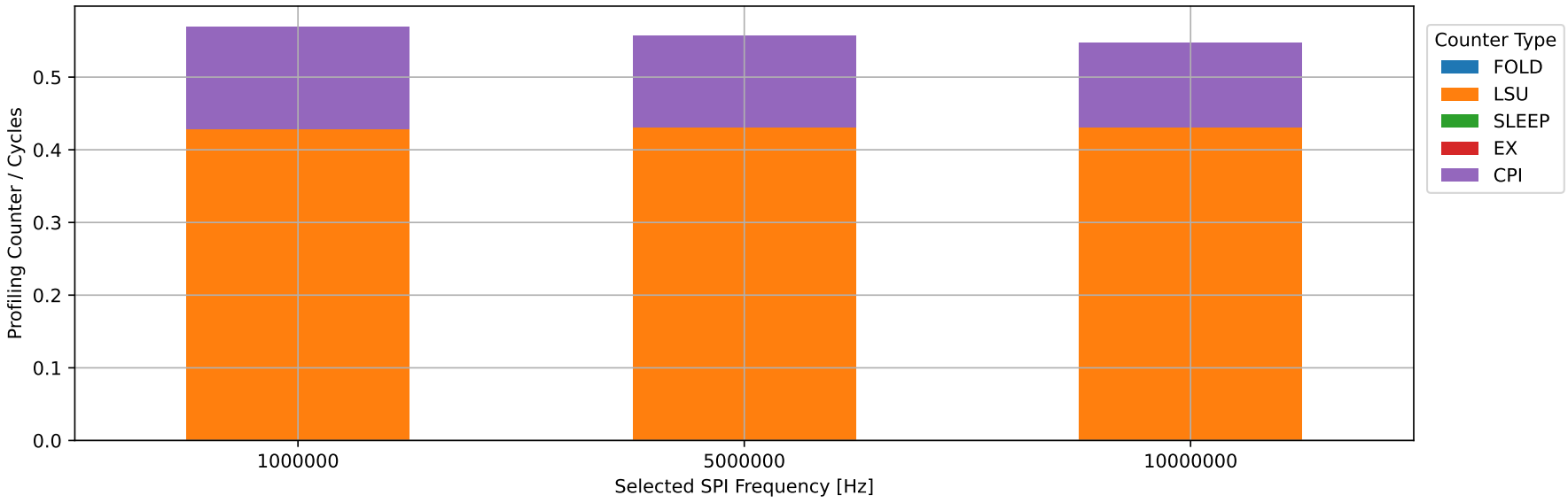
Packet Delay



For Ranking: Context Switch (80 MHz) 2760 ns

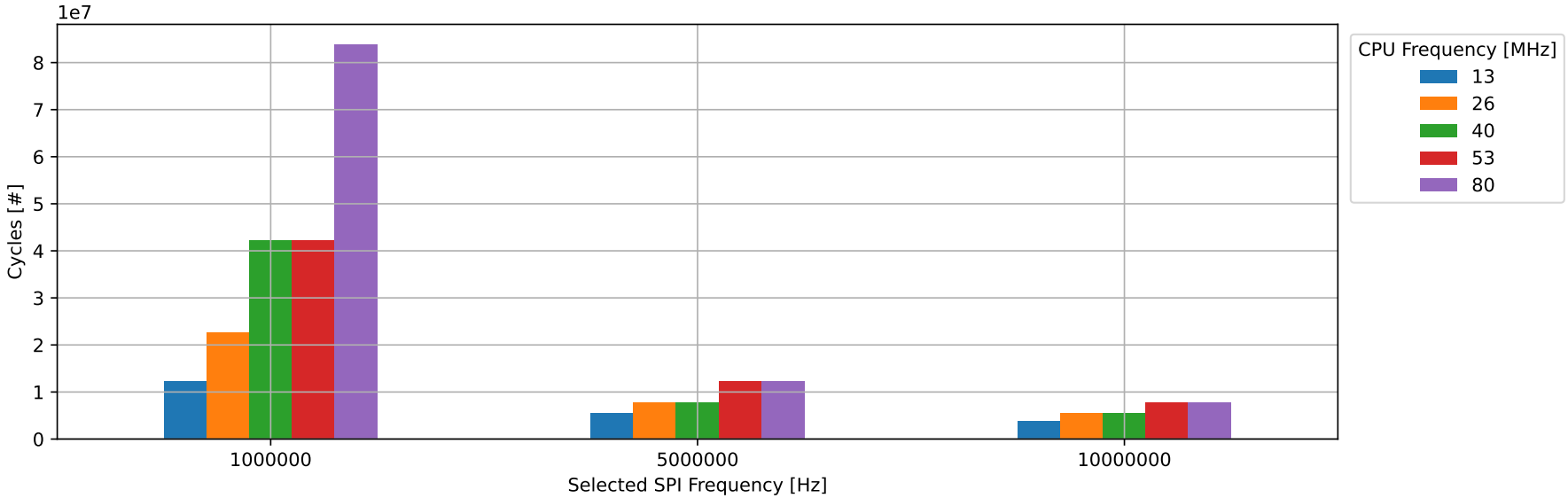
I/O Workload - SPI





- LSU high due to Ram Access → higher MEEF setting

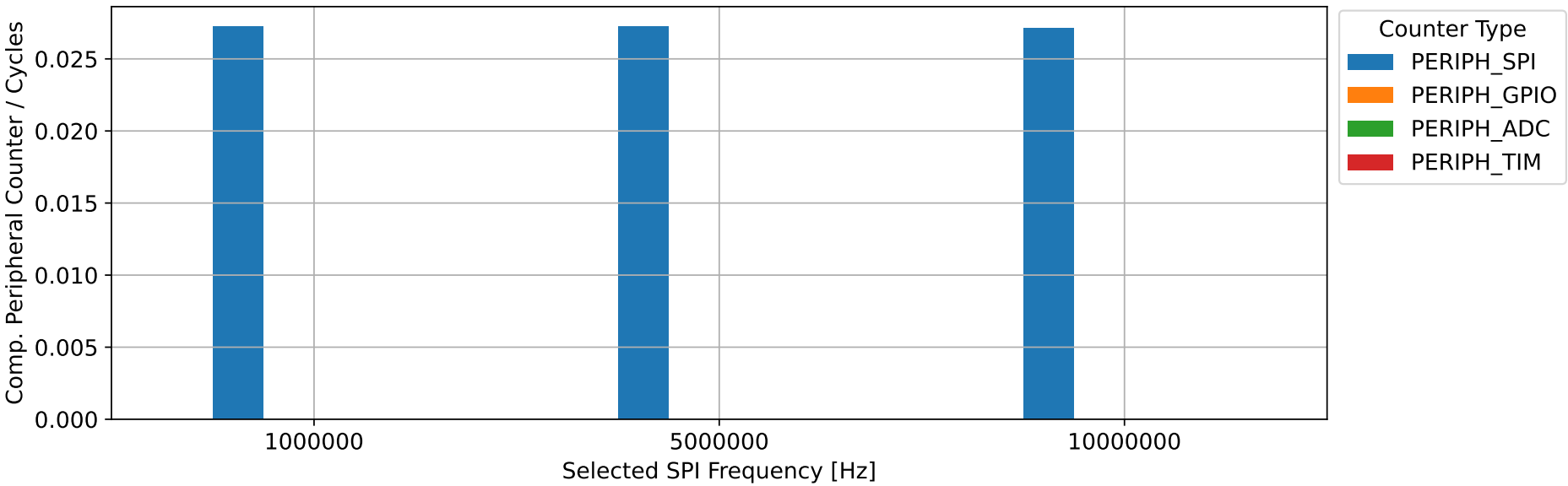
I/O Workload - SPI



- Cycle Reduction with lower frequencies/ FWS
- MEEF setting = 13MHz

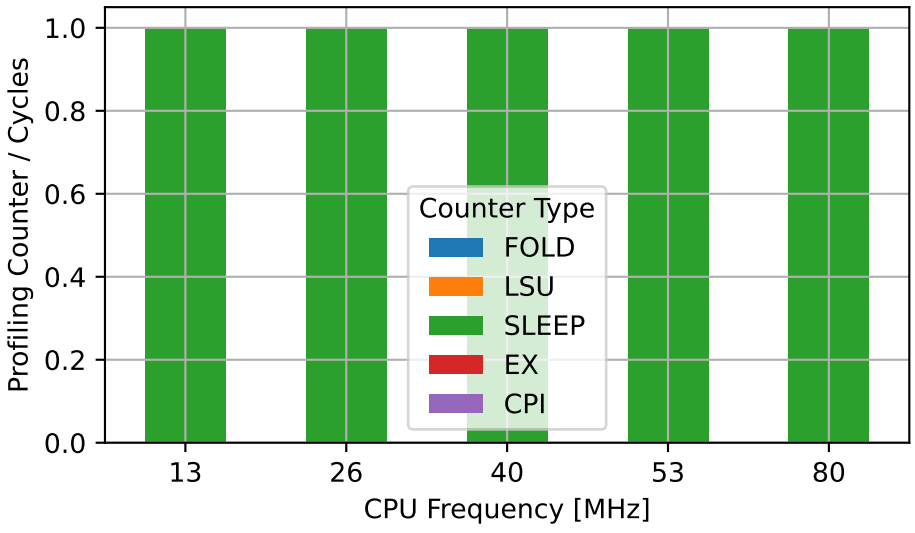
```
volatile uint8_t *DR = (volatile uint8_t*)&(dev(bus)->DR);
/* sending data */
for (size_t i = 0; i < len; i++) {
    while (!(dev(bus)->SR & SPI_SR_TXE));
    *DR = outbuf[i];
}
```

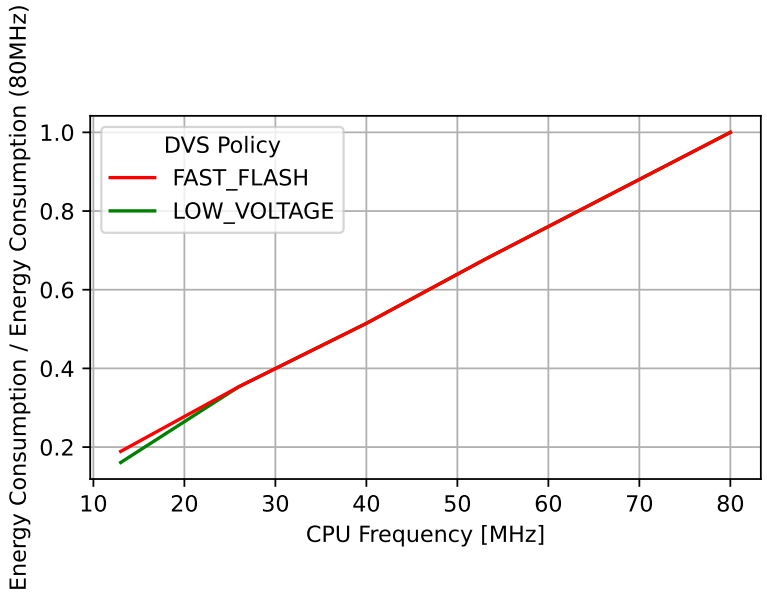
I/O Workload - SPI



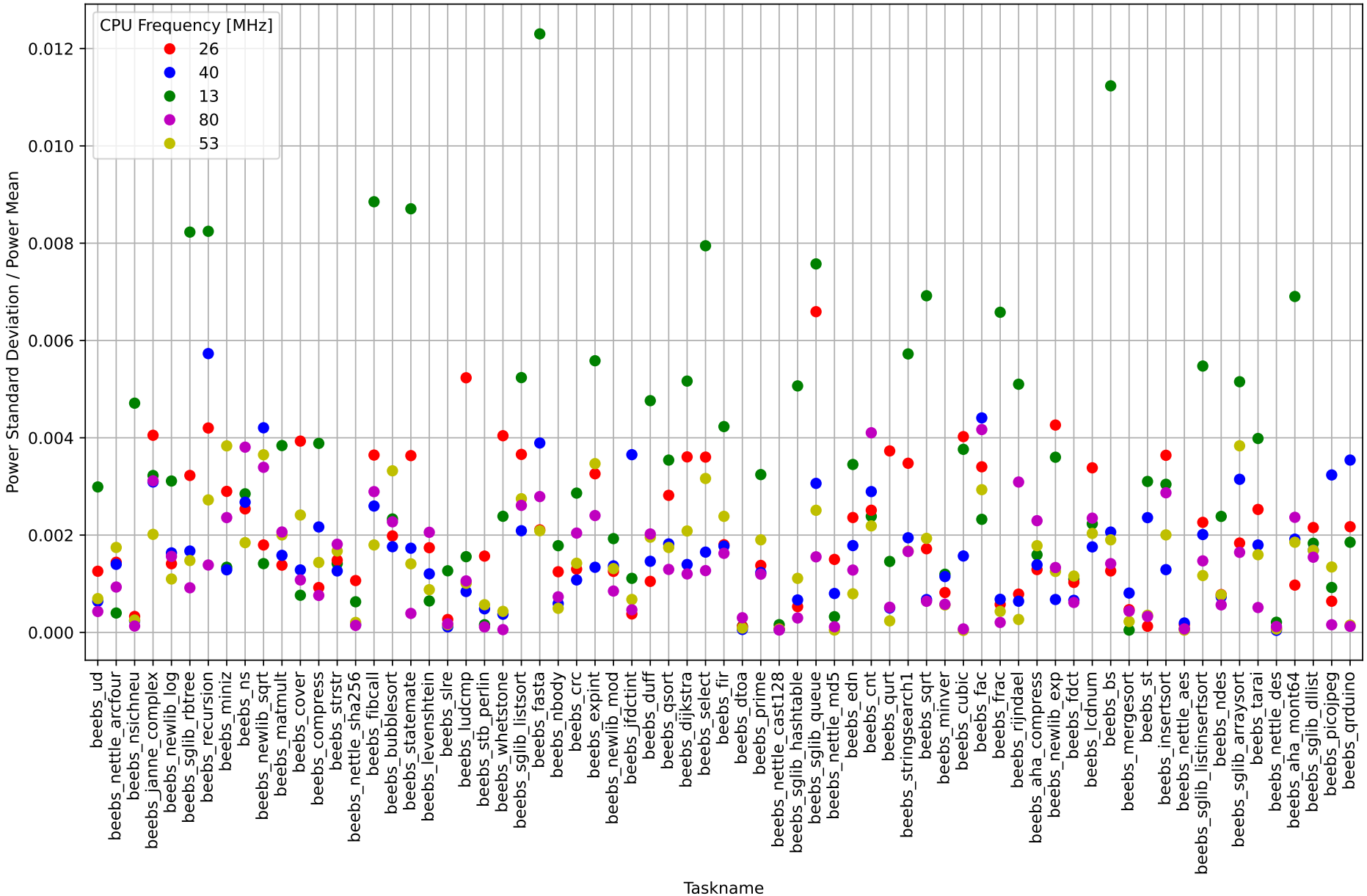
- I/O usage difficult to detect with Profiling Counter and RAM/Flash Access
- Comparator: Trace access to peripheral device

Sleep Workload

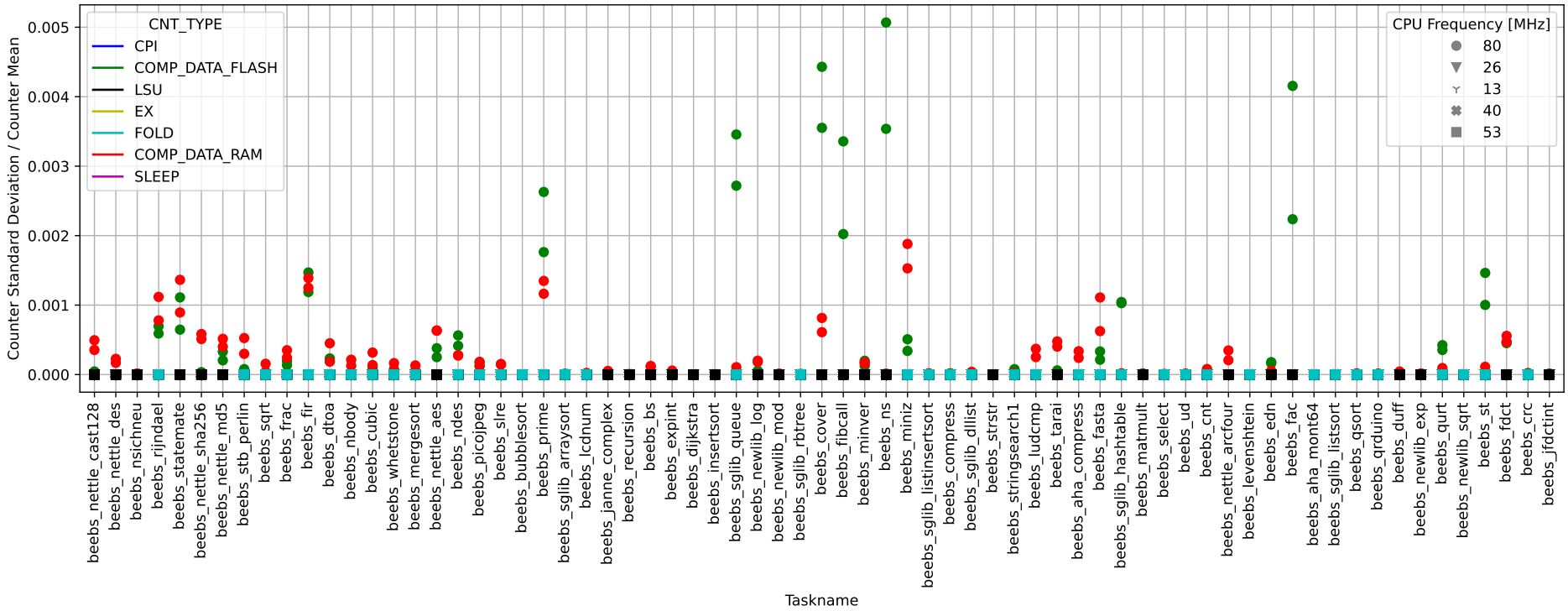




Power Standard Deviation / Mean

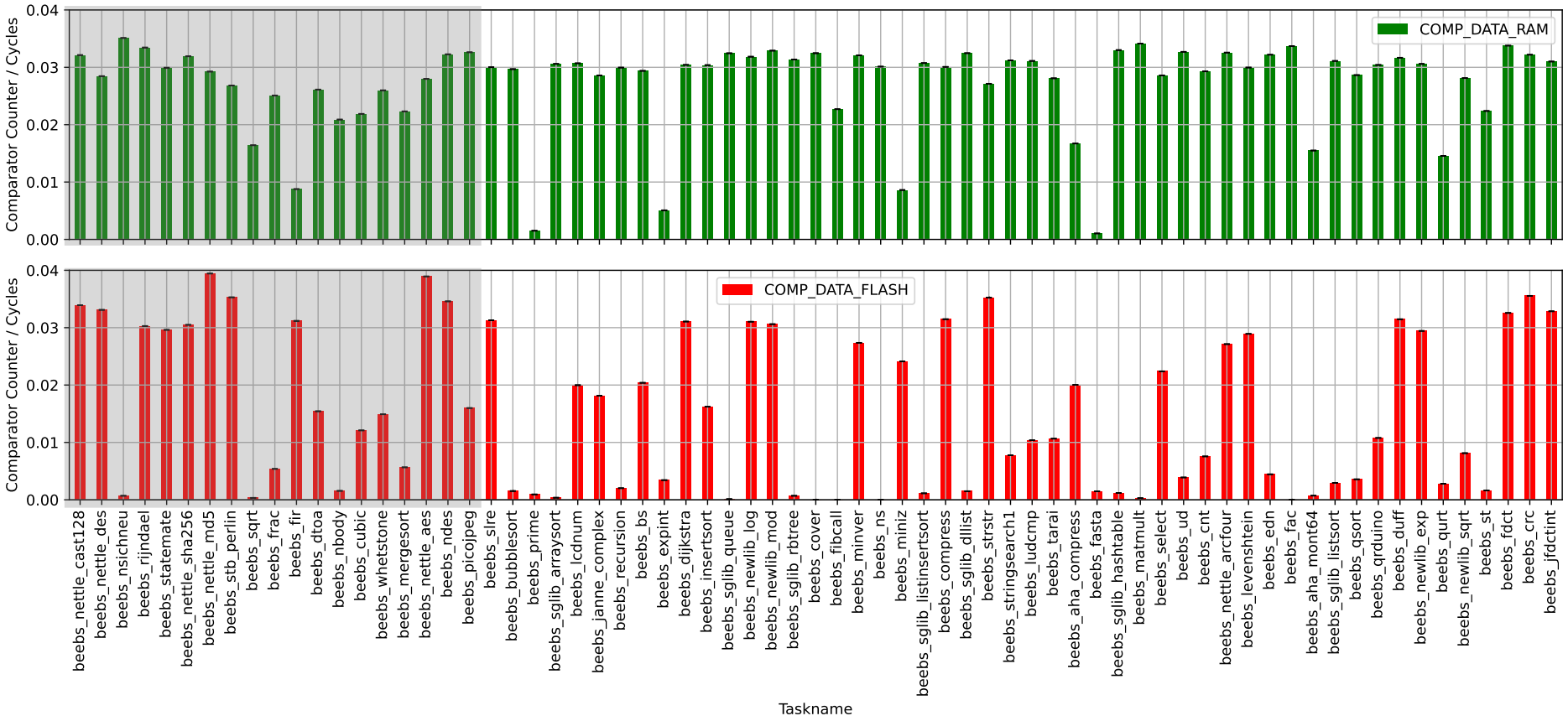


BEEBS Tracing Inaccuracies



- $\frac{StandardDeviation}{Mean} \rightarrow$ independence of absolute values
- Tracing with comparators only shows divergence between repetitions of up to 0.5%

BEEBS Tracing with Comparators



- No high correlation to saved cycles
- Many RAM/flash counter results limit at 0.03 access per cycle