Toward Energy-Efficient Generative AI on Open-Source RISC-V Heterogeneous SoCs

Integrated Systems Laboratory (ETH Zürich)

Victor J.B. Jung jungvi@iis.ee.ethz.ch
Dr. Moritz Scherer scheremo@iis.ee.ethz.ch
Philip Wiese wiesep@iis.ee.ethz.ch
Gamze İslamoğlu gislamoglu@iis.ee.ethz.ch
Dr. Alessio Burrello alessio.burrello@polito.it
Dr. Francesco Conti f.conti@unibo.it
Prof. Dr. Luca Benini lbenini@iis.ee.ethz.ch

PULP Platform
Open Source Hardware, the way it should be!

@pulp_platform
pulp-platform.org
youtube.com/pulp_platform

GenAI & Foundation Models for TinyML?

What are Foundation Models and LMs?

Foundation Models advantages:

- One backbone for multiple tasks
- Support mixed-modalities (*i.e.* images + voice)

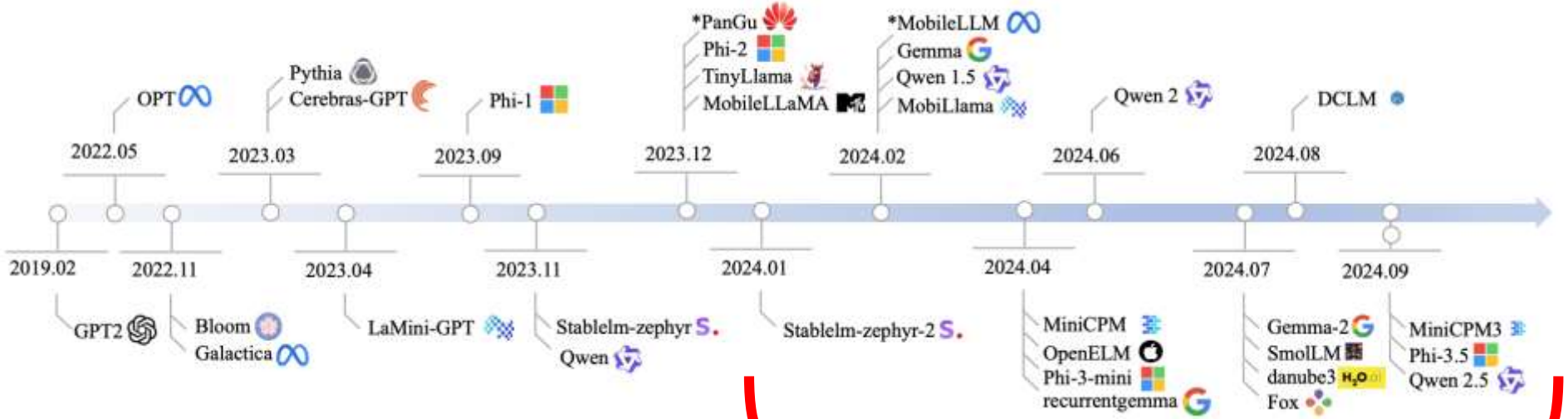
Embedded Foundation Models use cases:

AR Glasses

Nano-Drone

Generative and Foundation Models at Edge

Small Language Models (SLMs) release frequency is increasing:



Many have been released this year!

Most of these SLMs are too large for TinyML: ~1B to ~125M Parameters

- Still OOMs higher than TinyML requirements!
- But Tiny Language Models (TLMs) are coming → **And we need to be ready!**

Figure from: Lu, Zhenyan, et al. "Small Language Models: Survey, Measurements, and Insights." *arXiv preprint arXiv:2409.15790* (2024).

The Generative Inference Process

One Token

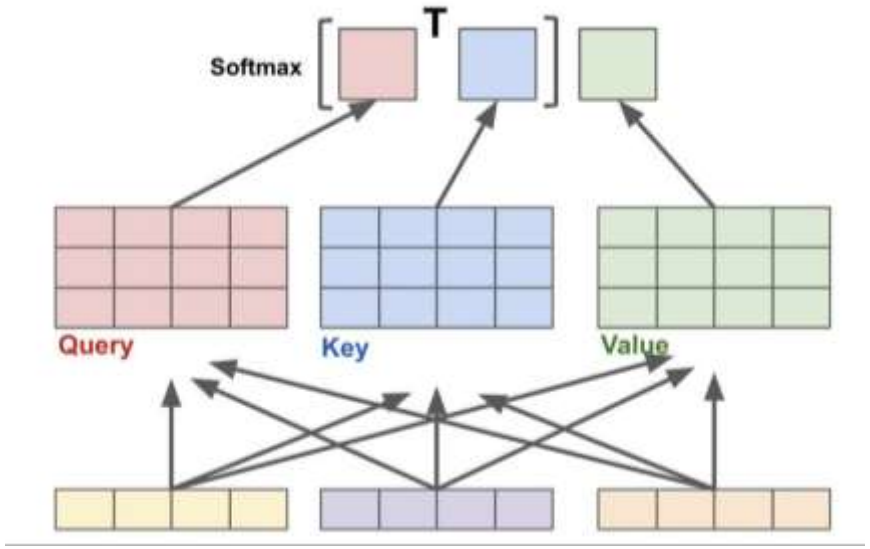
What does PULP mean? Parallel Ultra Low Power

The Prompt

Next Token: [illegible]

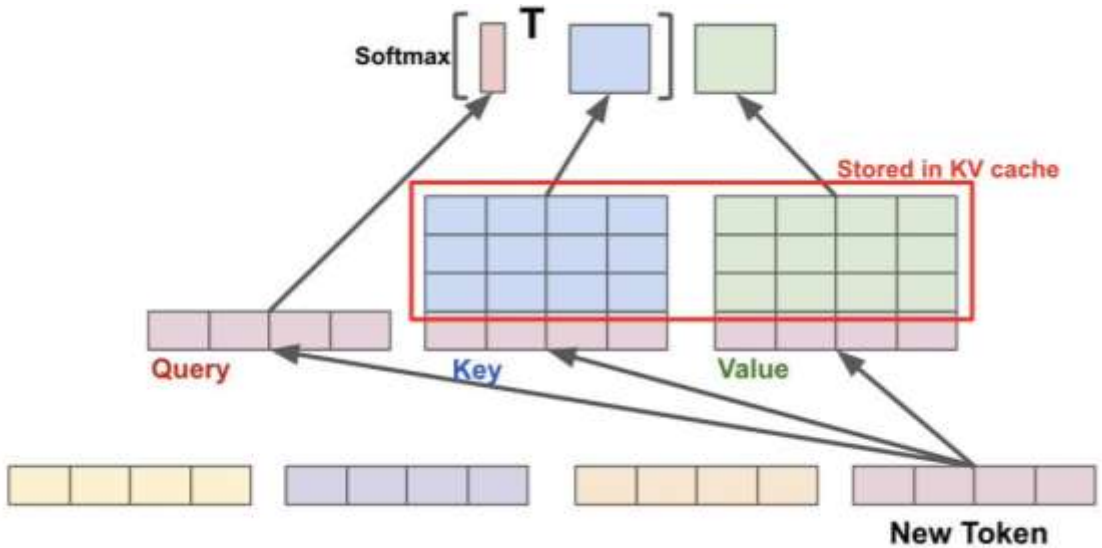
Generative Models: Parallel and Autoregressive Modes

Parallel Mode



- Processes the Prompt
- Parallel execution on each token of the prompt
- Composed of GEMM

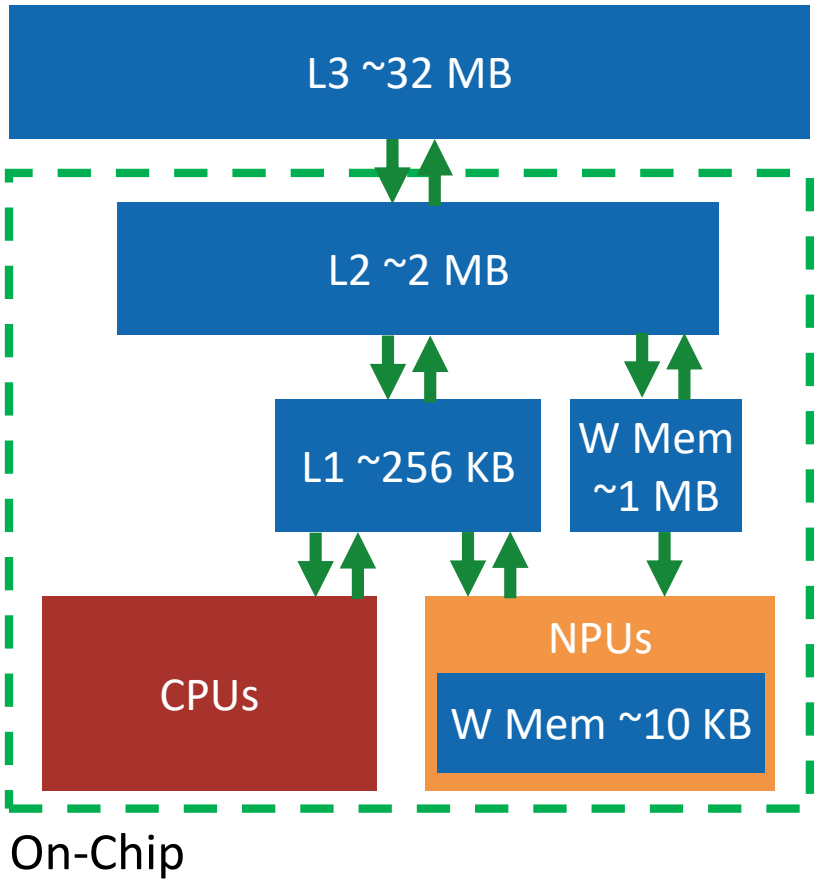
Autoregressive Mode



- Generates the Answer
- Store context in KV cache
- Composed of GEMV

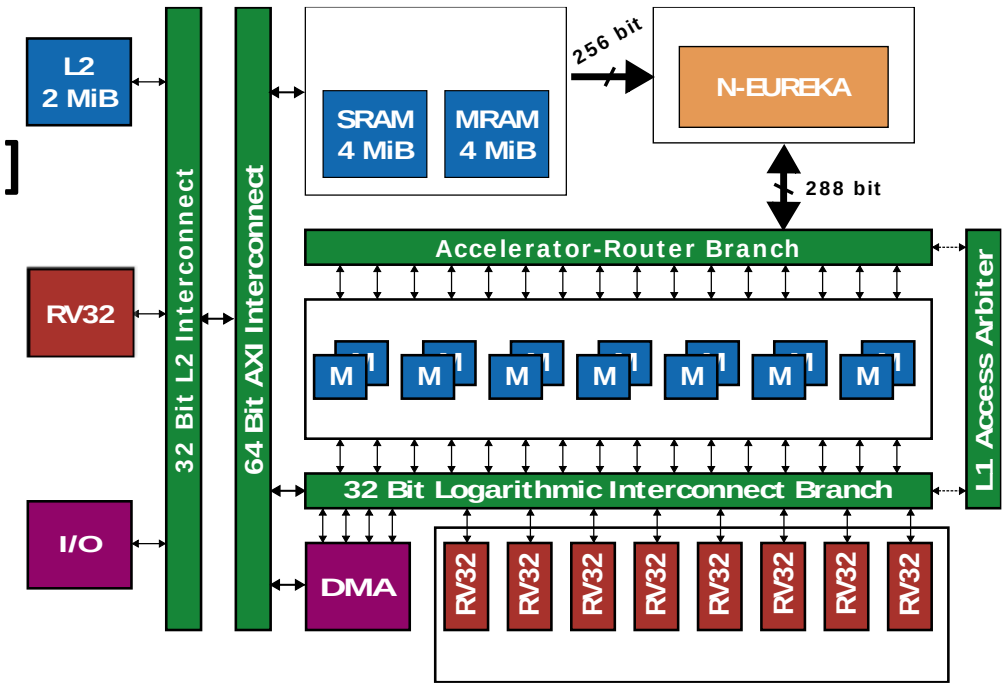
TinyML Heterogeneous SoCs Constraints

- **Limited off-chip capacity**
  - Maximum 64 Mbytes for MCUs
  - High read/write energy and latency
- **Small on-chip memory**
  - Typically 1 to 4 Mbytes
- **Memory hierarchy made of software-managed caches**
  - Suitable for workload known at compile time
  - Increased energy efficiency compared to regular caches
  - Offloads complexity to the compiler



Cambrian Explosion of Heterogeneous SoCs

- **Many SoC flavours developed and they need to run end-to-end workloads**
- **Heterogeneous SoC contains:**
  - Cores with different Instruction Set Architectures (ISAs)
  - Neural Processing Units (NPUs) with different capabilities
  - Different memory systems & hierarchies
- **One example: The Siracusa Microcontroller [1]**
  - Conv/GEMM Accelerator
  - Cluster of 8 RISC-V 32bit cores
  - Weight memory attached to the NPU



[1] Prasad, Arpan Suravi, et al. "Siracusa: A 16 nm Heterogenous RISC-V SoC for Extended Reality With At-MRAM Neural Engine." *IEEE JSSC.*

SotA TinyML Deployment Tools

Many tools exist but each are focusing on one specific SoC

STM32 MCUs

GAP8 & GAP9 SoC

MAX78000 SoC

Makes sense for the industry, but <u>inconvenient for researchers</u>!

Additionally they don't support GenAI patterns such as KV caching!

Our Deeployment Stack

Training

TensorFlow

PyTorch

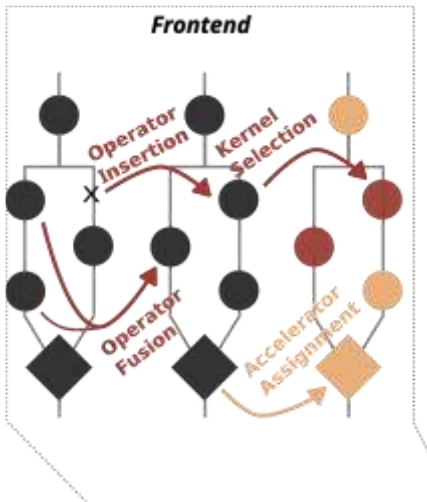
Keras

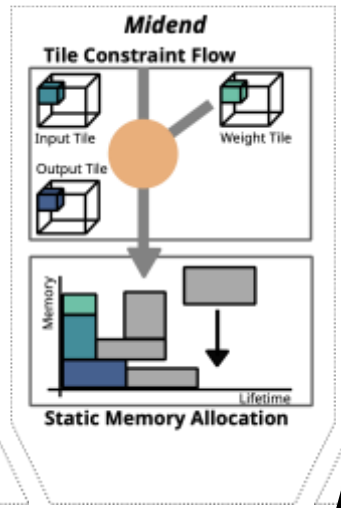
Model Export

ONNX

This Talk

Deeploy





ESWEEK 2024

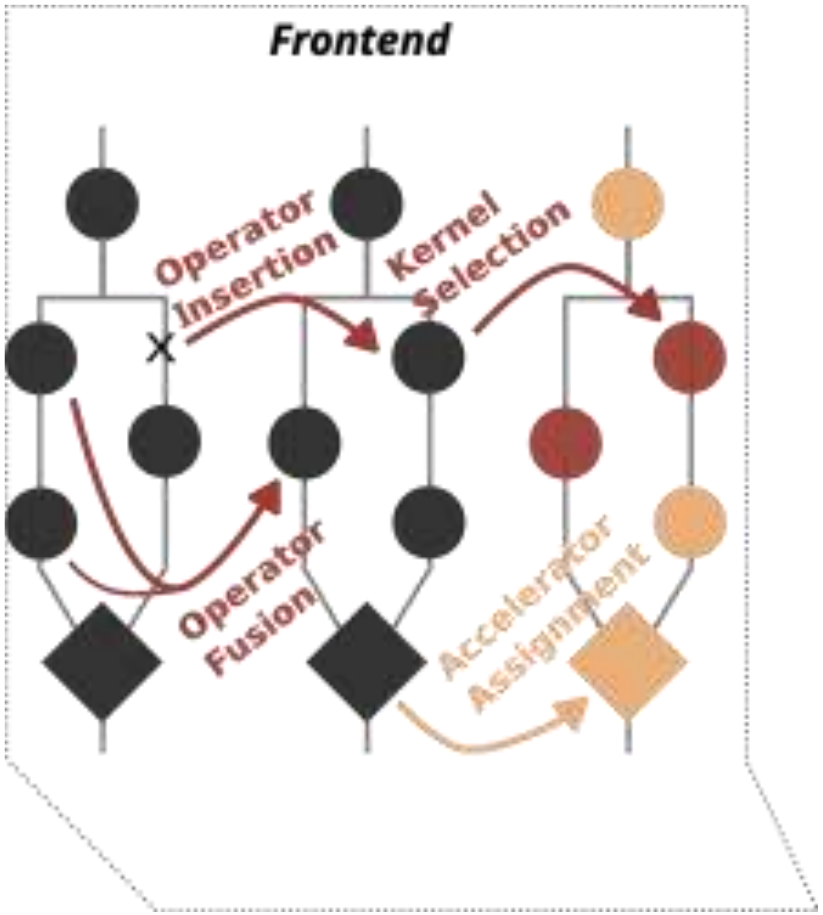
ference

SC-V

arm

- **Generates bare-metal code requiring minimal runtime support**
- **General structure makes it easy to add new platforms and complex networks**

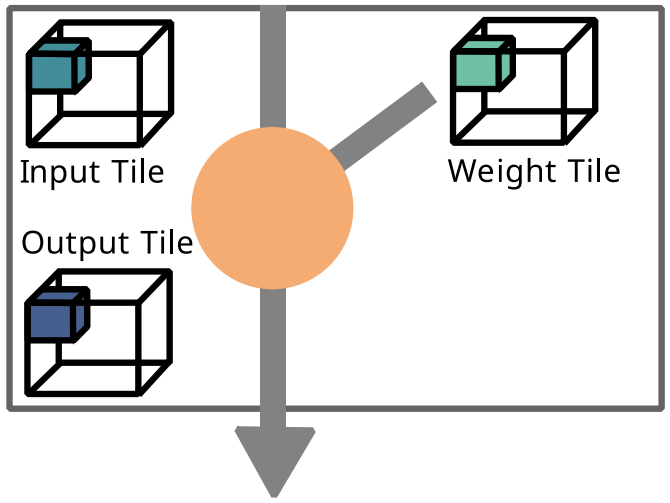
Deeploy's Frontend – Engine-aware lowering



- **Graph lowering is platform-dependent**
  - Taking into account engine mapping options
  - Graph remains fully ONNX compliant
- **Nodes are matched with low-level kernels**
  - Kernel libraries like CMSIS-NN, PULP-NN are supported
  - Custom kernels may be added as well
- **<u>How do we manage on-chip memory?</u>**

Deeploy's Midend – First Stage: Tiling

Tile Constraint Flow

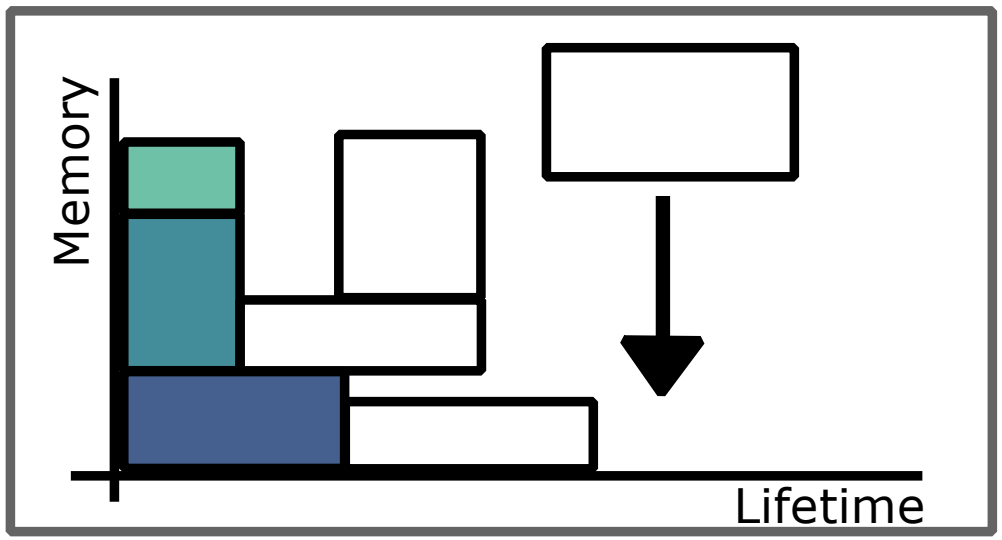


Operator Constraints:
Constrain inputs
to produce valid outputs

Target Constraints:
Constrain inputs to work with HW target

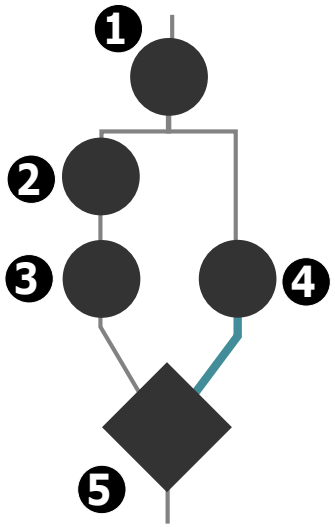
Core Idea: Get Symbolic Buffer sizes from the Tiler

Represents the block's height in Tetris Memory Scheduling



Deeploy's Midend – Second Stage: Memory Allocation

Core Idea: Solve ILP for joint tiling & allocation solution!



Encodes the order of the block placement

Tetris Scheduling: Topological order leads to suboptimal solutions!

Deeploy's Midend – Second Stage: Memory Allocation

Solution: Use Permutation Matrices

A valid permutation matrix P respects the following properties:

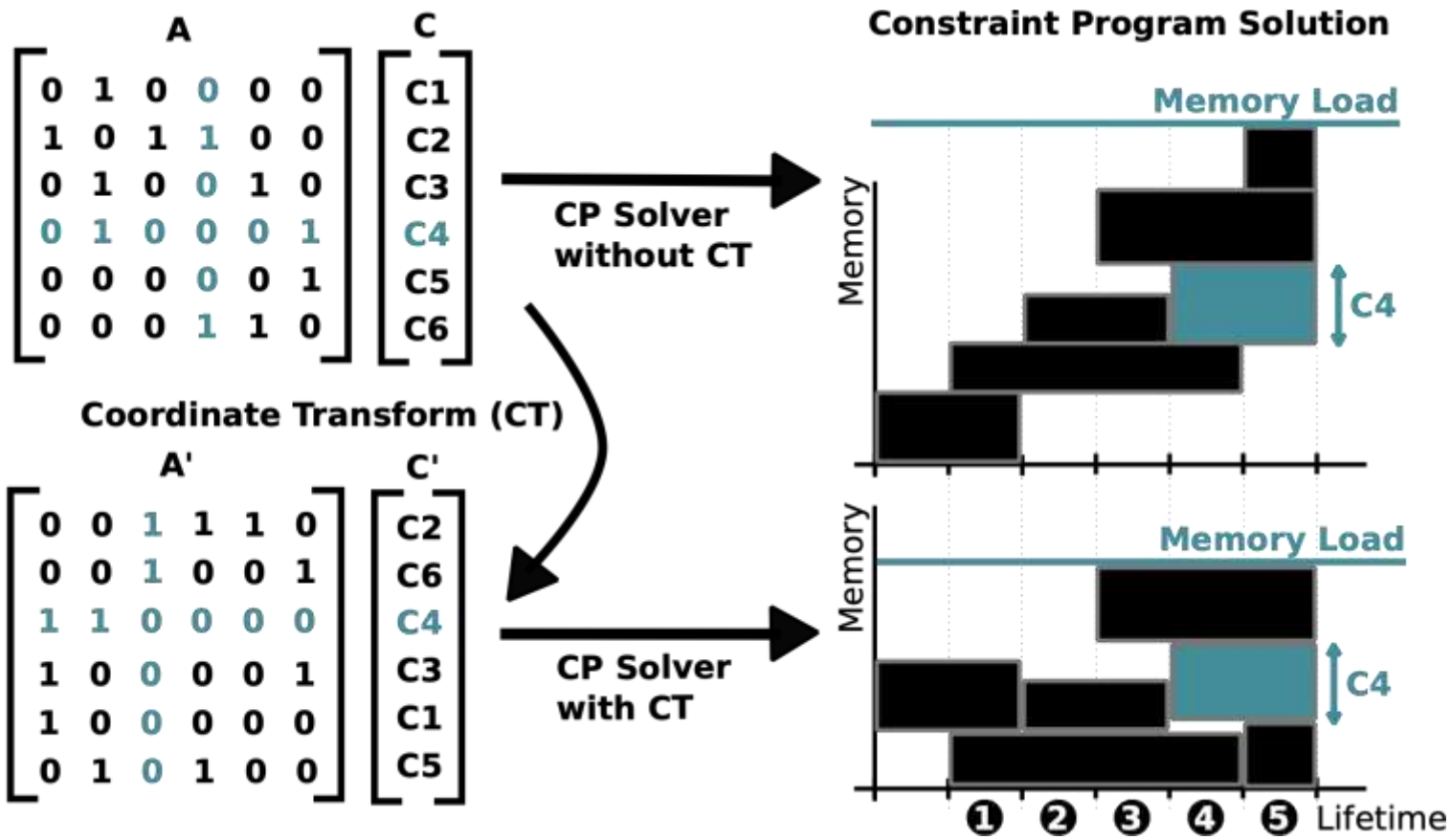
- Its elements are booleans: $p_{i,j} \in [0, 1]$
- The sum of rows is 1: $\sum_{i=0}^{N-1} p_{i,j} = 1 \,, \forall j \in [0, N-1]$
- The sum of cols is 1: $\sum_{i=0}^{N-1} p_{j,i} = 1 \,, \forall j \in [0, N-1]$

$$
\begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
C1 \\ C2 \\ C3 \\ C4 \\ C5 \\ C6
\end{bmatrix}
$$

A C

For any permutation matrix, $\boldsymbol{A' = P \times A \times P^T}$ is a valid adjacency matrix with associated cost vector $\boldsymbol{C' = P \times C}$

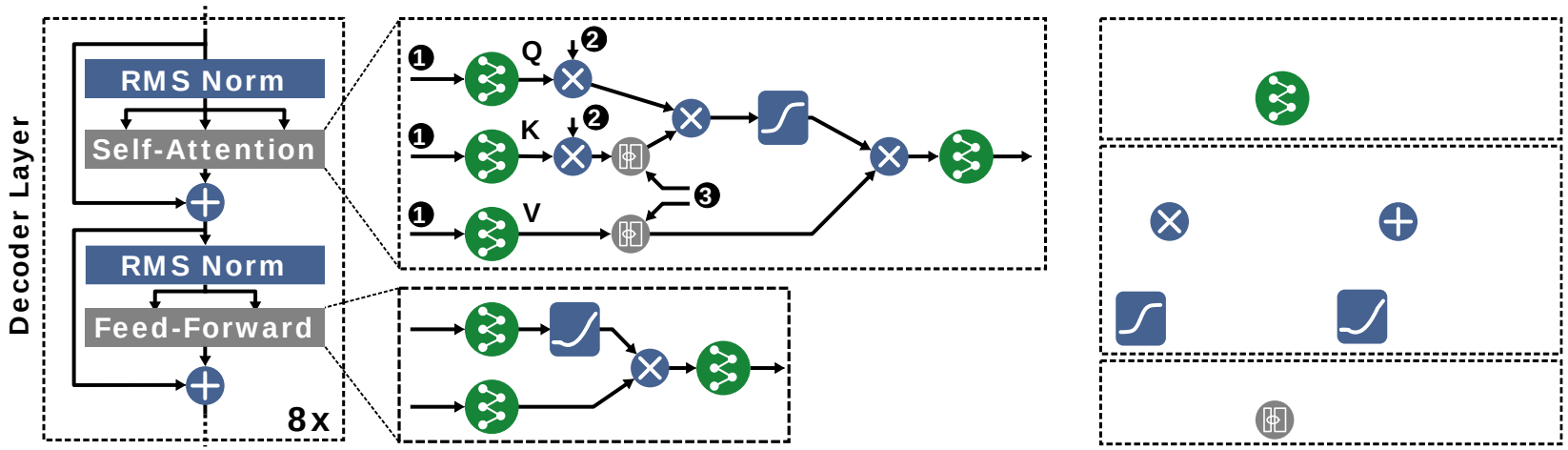
Deeploy's Midend – Second Stage: Memory Allocation



The solver can co-optimize the tiling solution with the memory load schedule

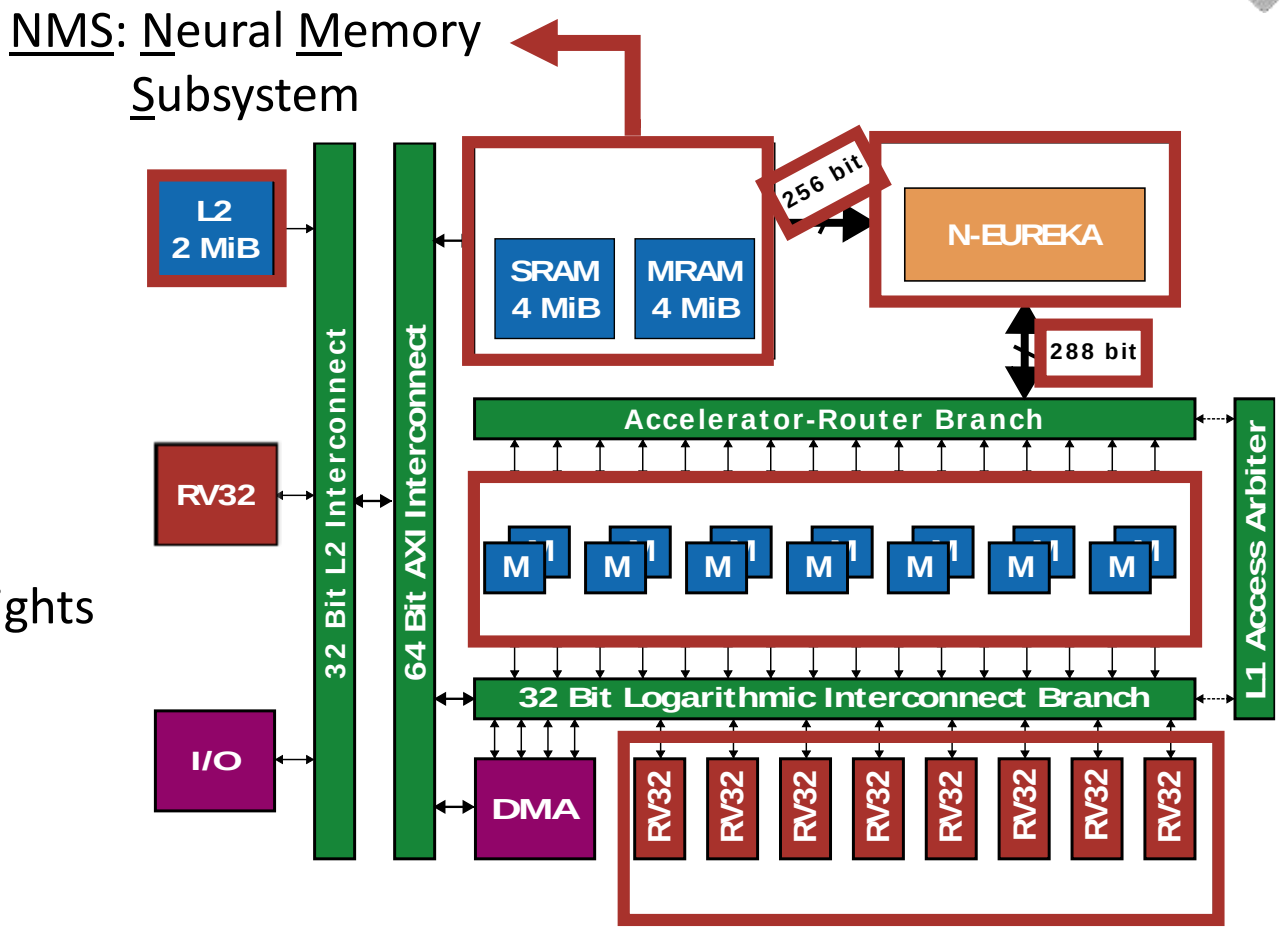
Let's go through an example – TLM on Siracusa

- **TLM trained on the TinyStory dataset**
- **4.6 Million parameters**
- **Llama 2 architecture**
- **Complete int8 quantization with QuantLib**
- **Polynomial approximation for Softmax and RMS Norm**



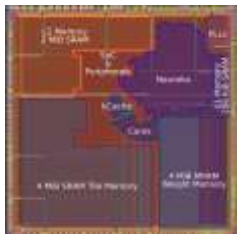
Siracusa – Heterogeneous RISC-V SoC

- **N-EUREKA: CNN Accelerator**
- **Lots of memory on-chip**
  - 4 MiB SRAM weight memory
  - 2 MiB SRAM
  - 256 KiB L1 TCDM
- **High-Bandwidth links to NPU**
  - From Neural Memory Subsystem for Weights
  - From L1 for Activations
- **Octa-Core RISC-V Cluster**
  - RISC-V cores with DSP extension
  - Integer SIMD support
  - Post-Increment and Hardware Loop extension



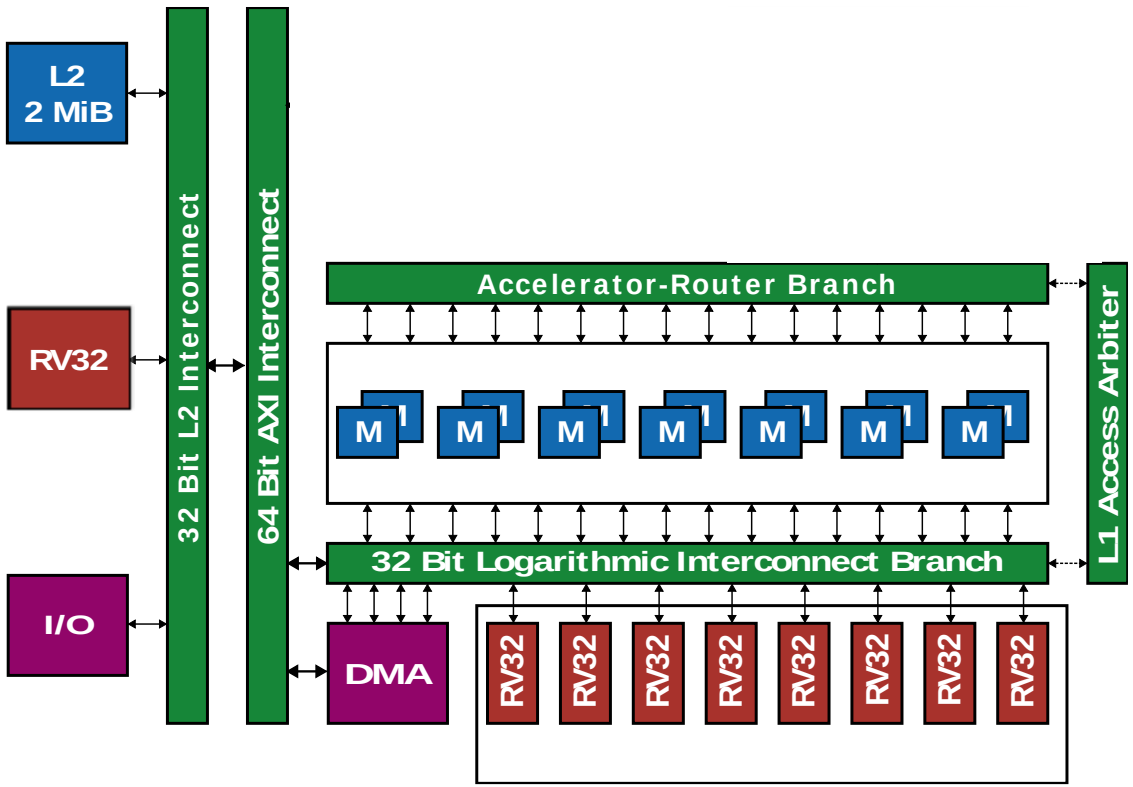
Our Benchmark Setup

Siracusa is Silicon Proven



We perform measurements on board

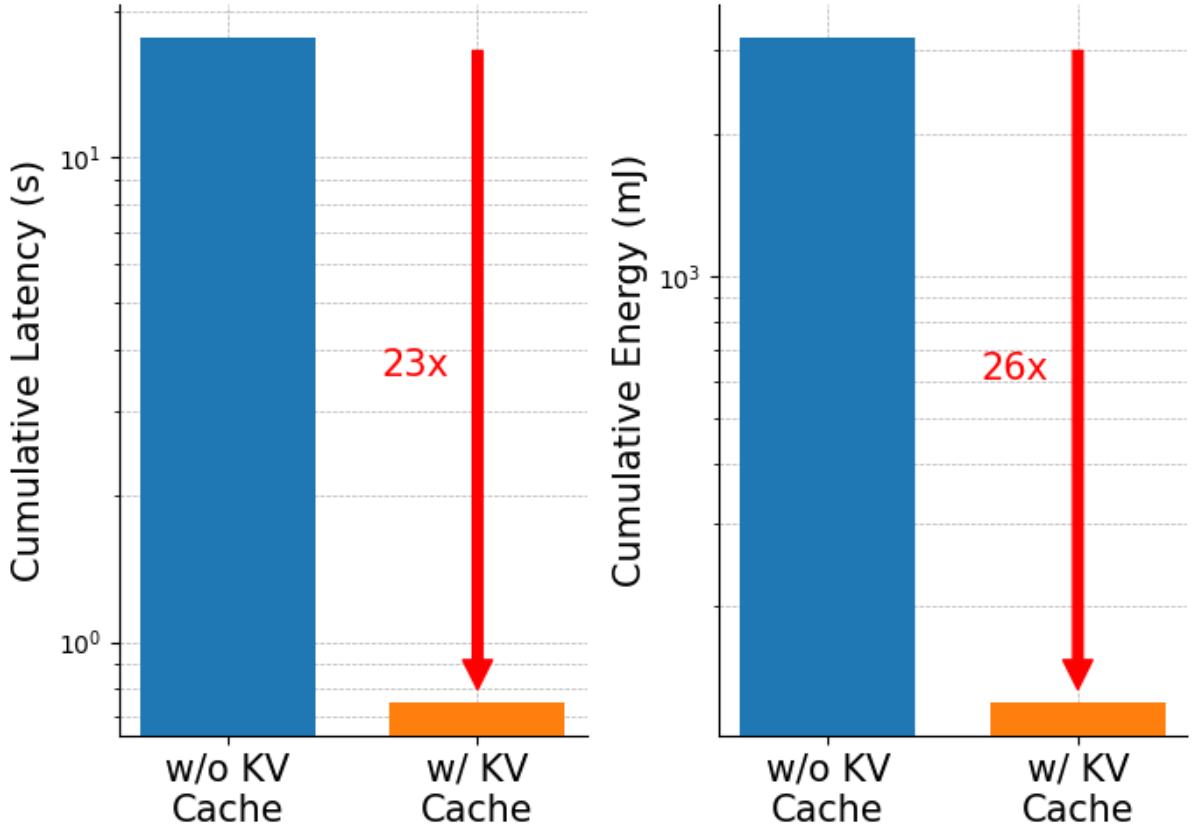
- **We benchmark 3 configurations:**
  - **Octa-Core Cluster**
  - **NPU w/o NMS**
  - **NPU w/ NMS**



The Benefits of KV Caching

- **Cumulative latency/energy on 256 step inference**
- **Using KV caching avoids costly recomputations**

	w/o KV Cache	w/ KV Cache
Latency (s)	17.6	0.75
Energy (mJ)	3193	125

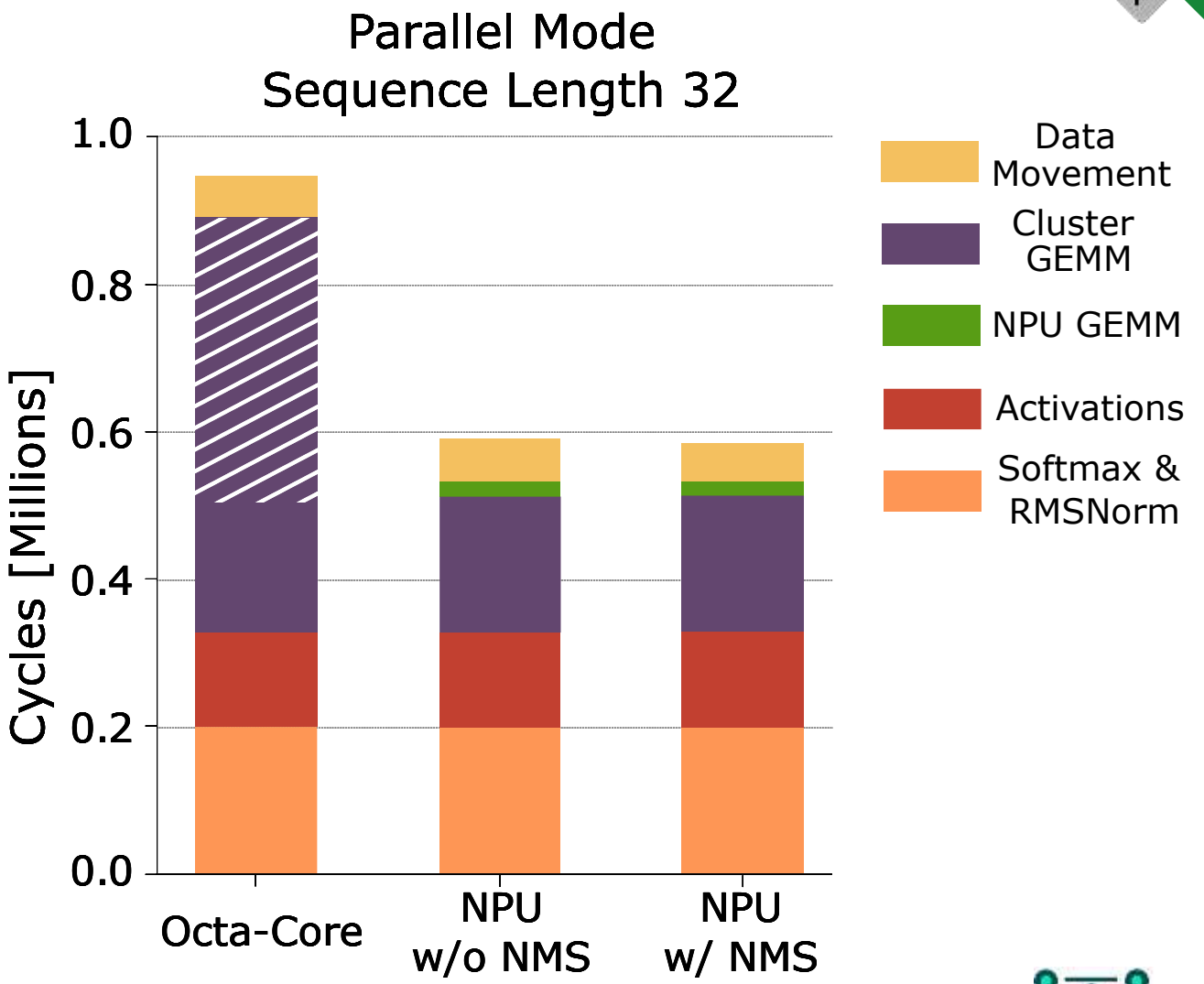


Parallel Mode E2E Inference Breakdown

- **GEMMs take 59% of the runtime in Octa-Core regime.**
- **We Offload Linear layers to the NPU**

Parallel Mode Properties

- **Compute bound regime**
- **Benefits maximally from NPU**

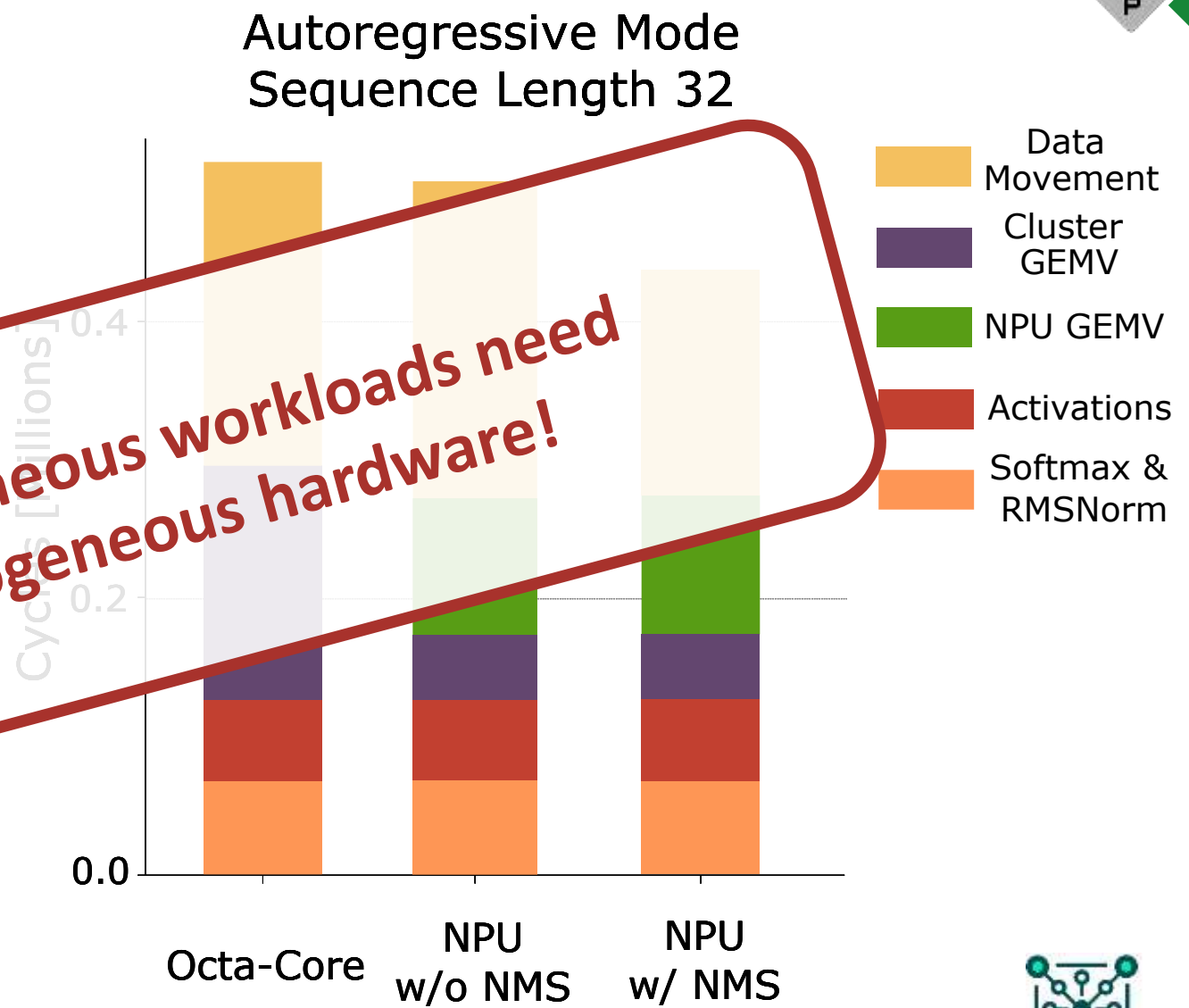


Autoregressive Mode E2E Inference Breakdown

- **NPU still provides 16% speedup on GEMV**
- **NMS relaxes the NPU memory boundness**

Autoregressive Mode Properties

- **Memory bound regime**
- **Leverages mostly the Octa-Core Cluster**



Heterogeneous workloads need heterogeneous hardware!

Heterogeneous workloads need heterogeneous hardware

- **Deeploy demonstrated End-to-End Inference of TLM on Heterogeneous SoCs**
- **With Deeploy, we benchmark different configurations of the SoC for the two modes of the network**
- **We reach a throughput of 340 Token/s at 490 μJ/Token**
- **And the best part? It's all open-source!**

ESWEEK 2024 Paper

`github.com/pulp-platform/Deeploy`

`github.com/pulp-platform/quantlib`

PULP Platform

Open Source Hardware, the way it should be!

Victor J.B. Jung	jungvi@iis.ee.ethz.ch
Dr. Moritz Scherer	scheremo@iis.ee.ethz.ch
Philip Wiese	wiesep@iis.ee.ethz.ch
Gamze İslamoğlu	gislamoglu@iis.ee.ethz.ch
Dr. Alessio Burrello	alessio.burrello@polito.it
Dr. Francesco Conti	f.conti@unibo.it
Prof. Dr. Luca Benini	lbenini@iis.ee.ethz.ch

Institut für Integrierte Systeme – ETH Zürich
Gloriastrasse 35
Zürich, Switzerland

DEI – Università di Bologna
Viale del Risorgimento 2
Bologna, Italy

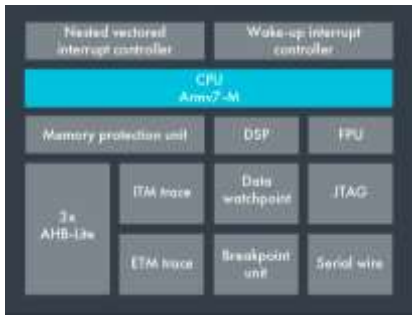
@pulp_platform

pulp-platform.org

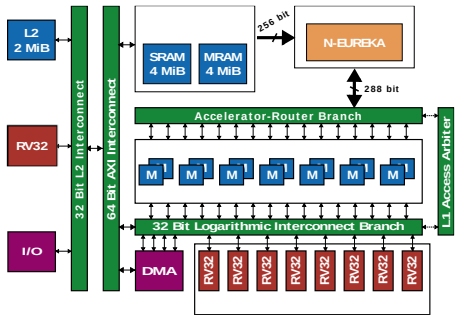
youtube.com/pulp_platform

Deeploy's Supported Platforms

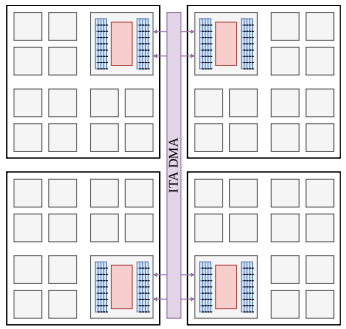
- **Already supports several TinyML platforms:**



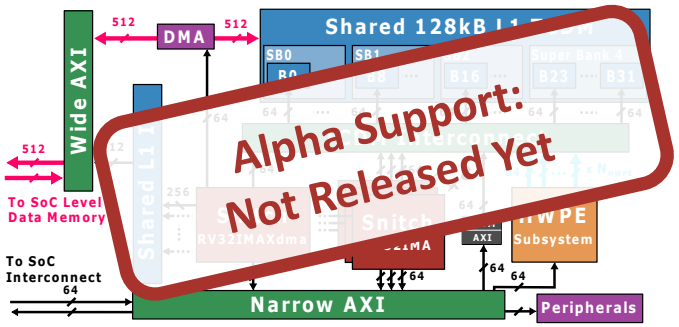
ARM Cortex
Generic ARM MCUs



Siracusa SoC
8-cores RISC-V 32 bit Cluster
NPU with specialized Memory System



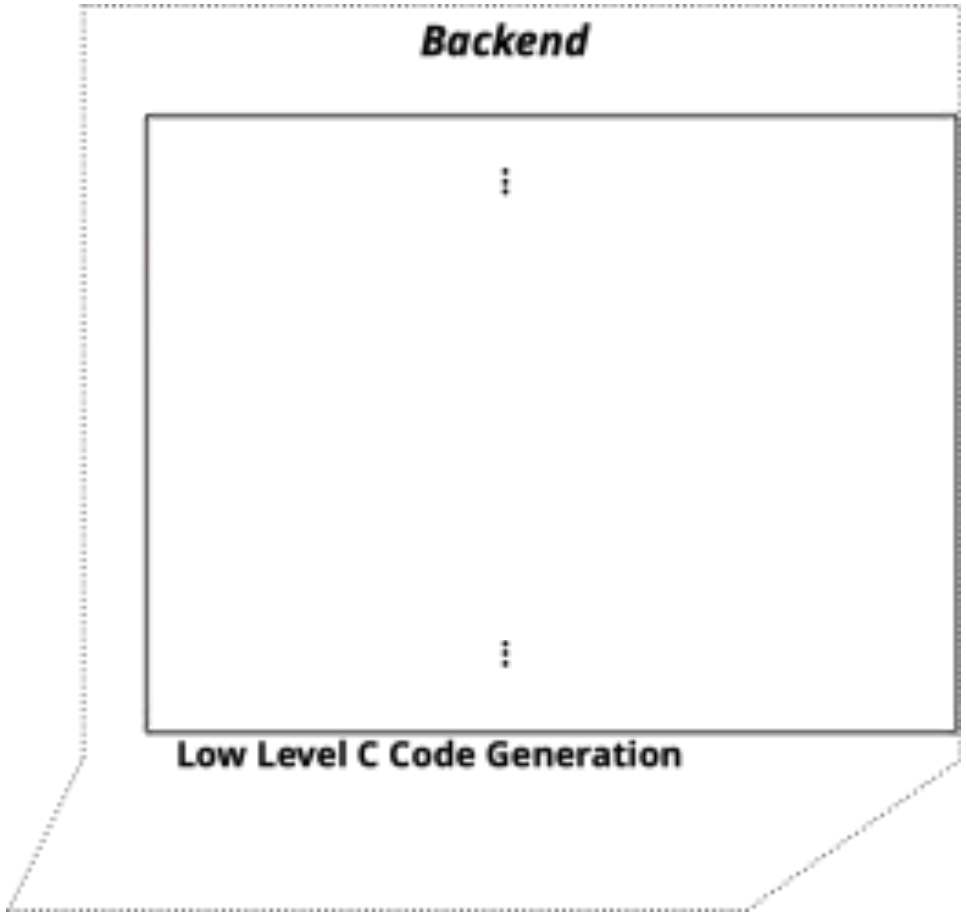
ITAPool
192-cores Manycore
NPU for Integer Attention



Snitch + ITA Cluster
8-cores Snitch Cluster
Extended NPU for Integer Attention

Deeploy's Backend – Vendor agnostic code generation

- **Backend generates low-level C Code**
  - Tiling & memory allocation from midend
  - Using operator mapping from frontend
- **Deeploy provides code generation primitives**
  - Device offloading
  - Double-buffered DMA transfers
  - Fork-synch based multi-core programming
- **Bottom-up code generation accelerates reuse**
  - Bring your own expert-optimized kernel templates!



Linear Layers Acceleration

- **We benchmark 4 configurations:**
  - Single core of the cluster
  - Octa-cores of the cluster
  - Cluster + NPU without NMS
  - Cluster + NPU with NMS
- **On large linear layers, the NPU brings up to 25x speedup**
- **The NMS reduces congestion and brings 2.1x speedup**