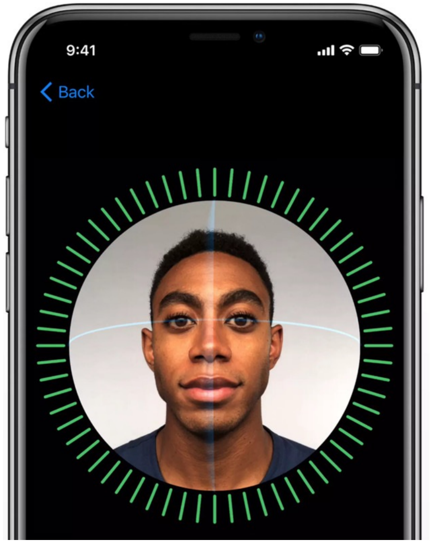
Gemmini: An Open-Source, Full-System DNN Accelerator Design and Evaluation Platform

Hasan Genc, Seah Kim, Vadim Vadimovich Nikiforov, Simon Zirui Guo, Borivoje Nikolić, Krste Asanović and Yakun Sophia Shao

DNNs are exploding in popularity…

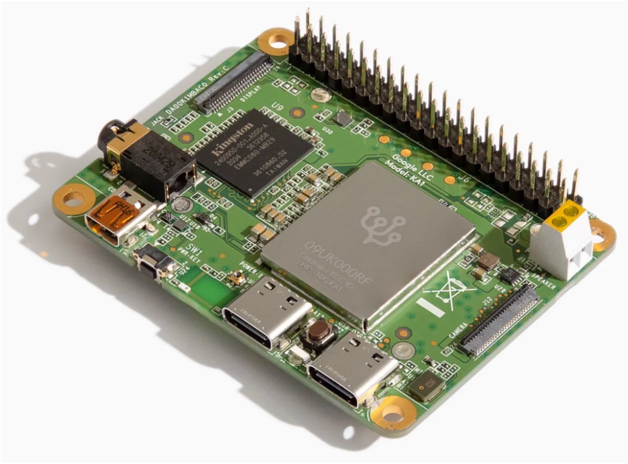
Matt Christenson/BLM/2017

By Dllu - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=64517567

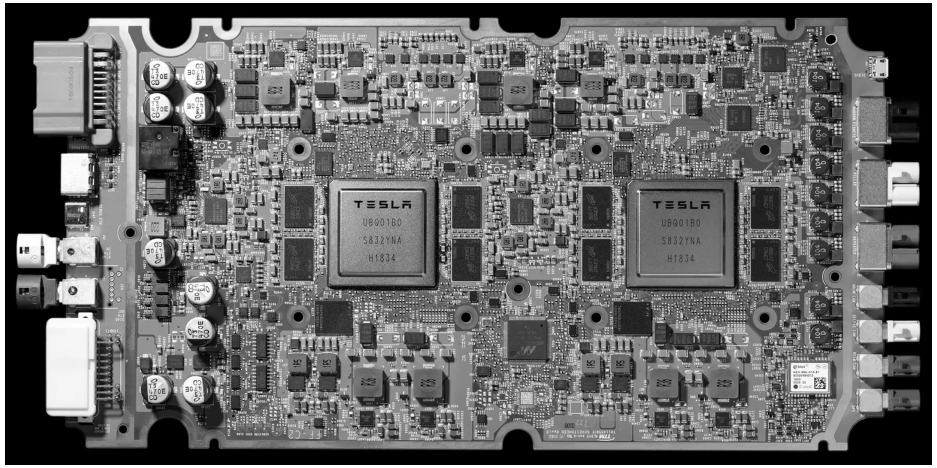


Apple Support

Which means DNN **ACCELERATORS** are exploding in popularity...



Edge TPU

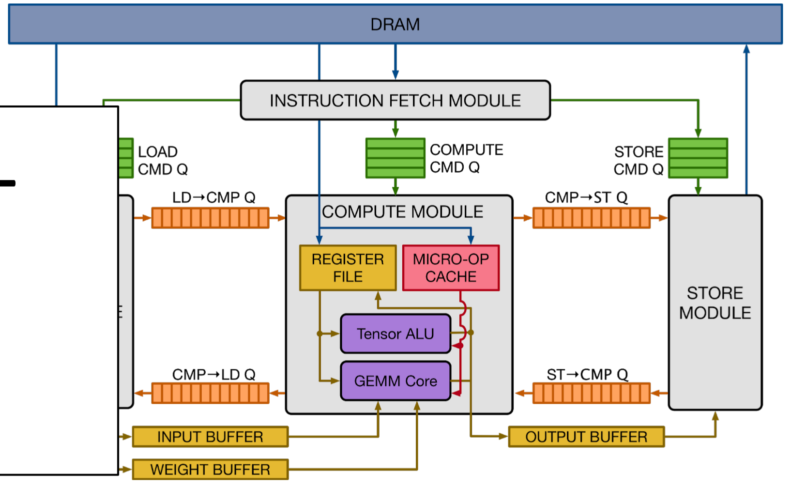


Tesla FSD

Cloud TPU

Which means DNN accelerator **GENERATORS** are exploding in popularity…

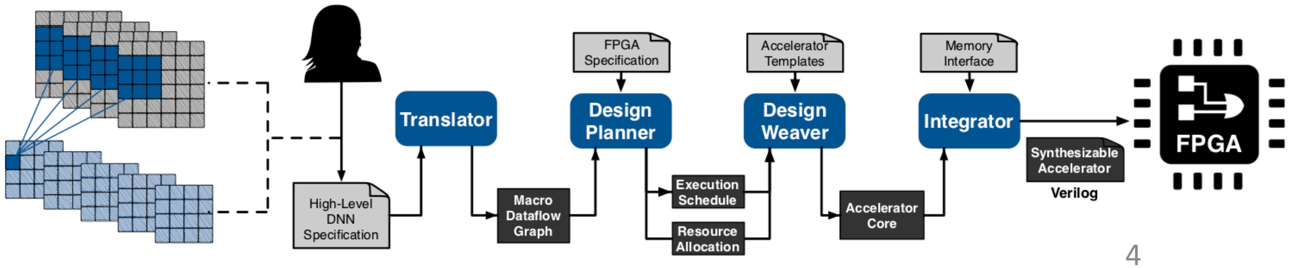
VTA



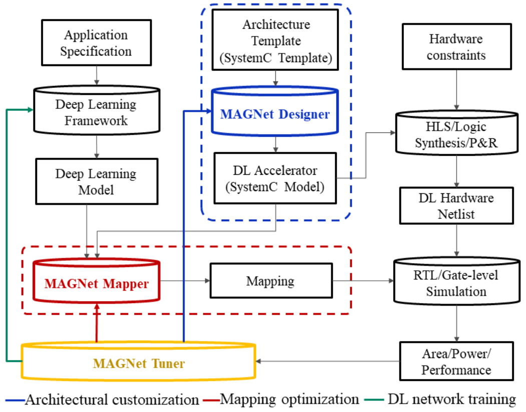
However, they lack full-system and full-stack visibility

NVDLA.org

DNNWeaver

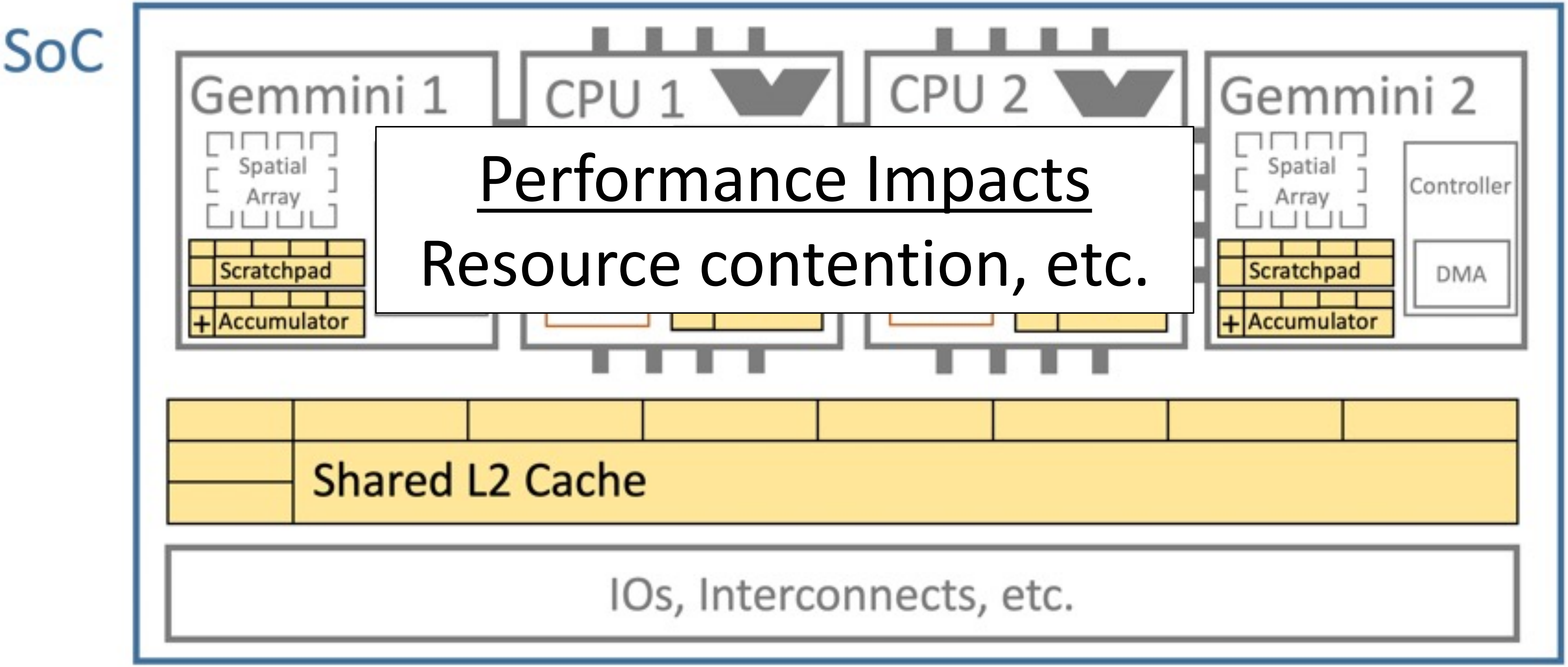


MAGNet



Full-System Visibility

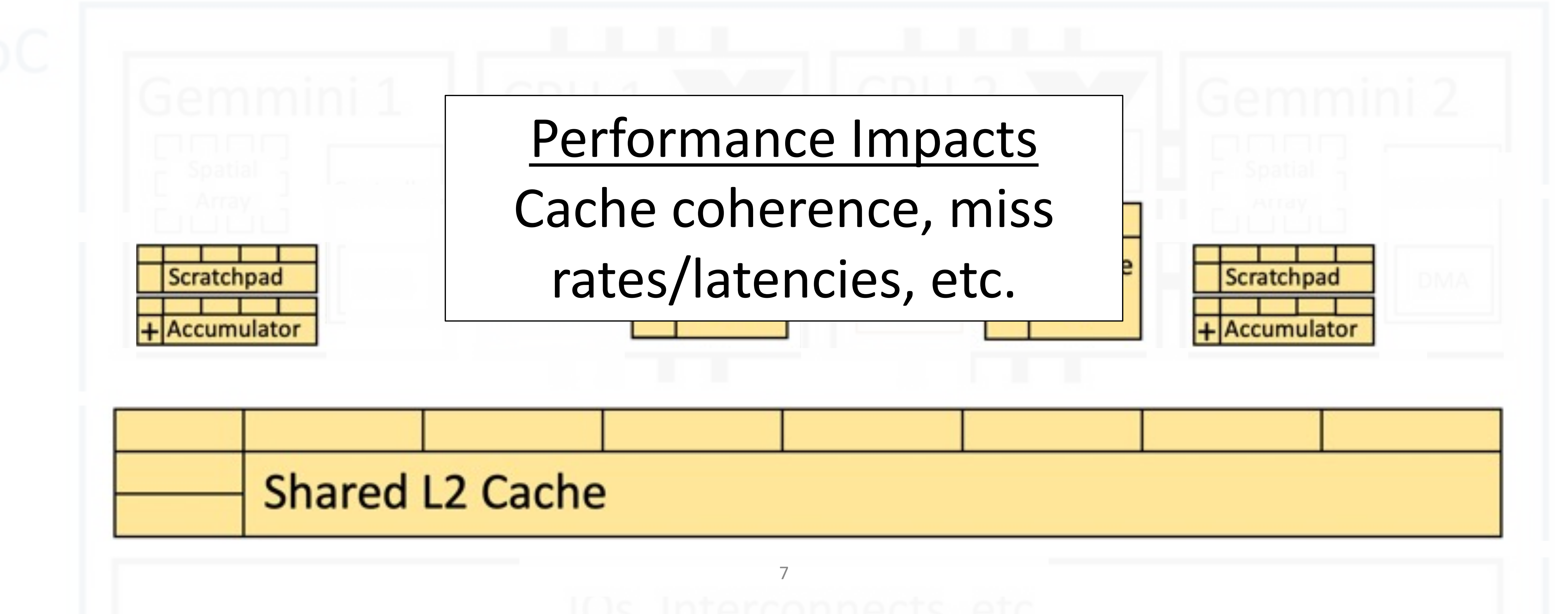
Full-System Visibility: SoC



Performance Impacts

Resource contention, etc.

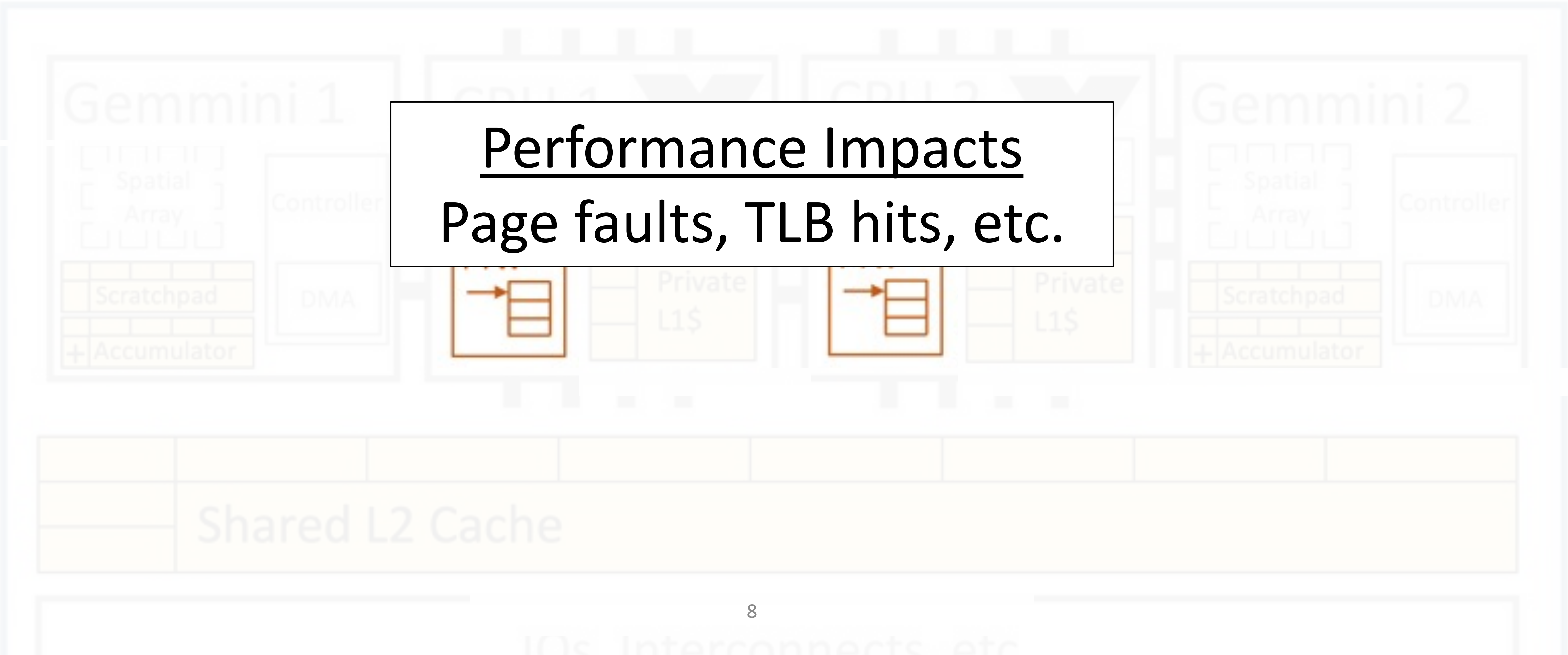
Full-System Visibility: Memory Hierarchy



Performance Impacts

Cache coherence, miss rates/latencies, etc.

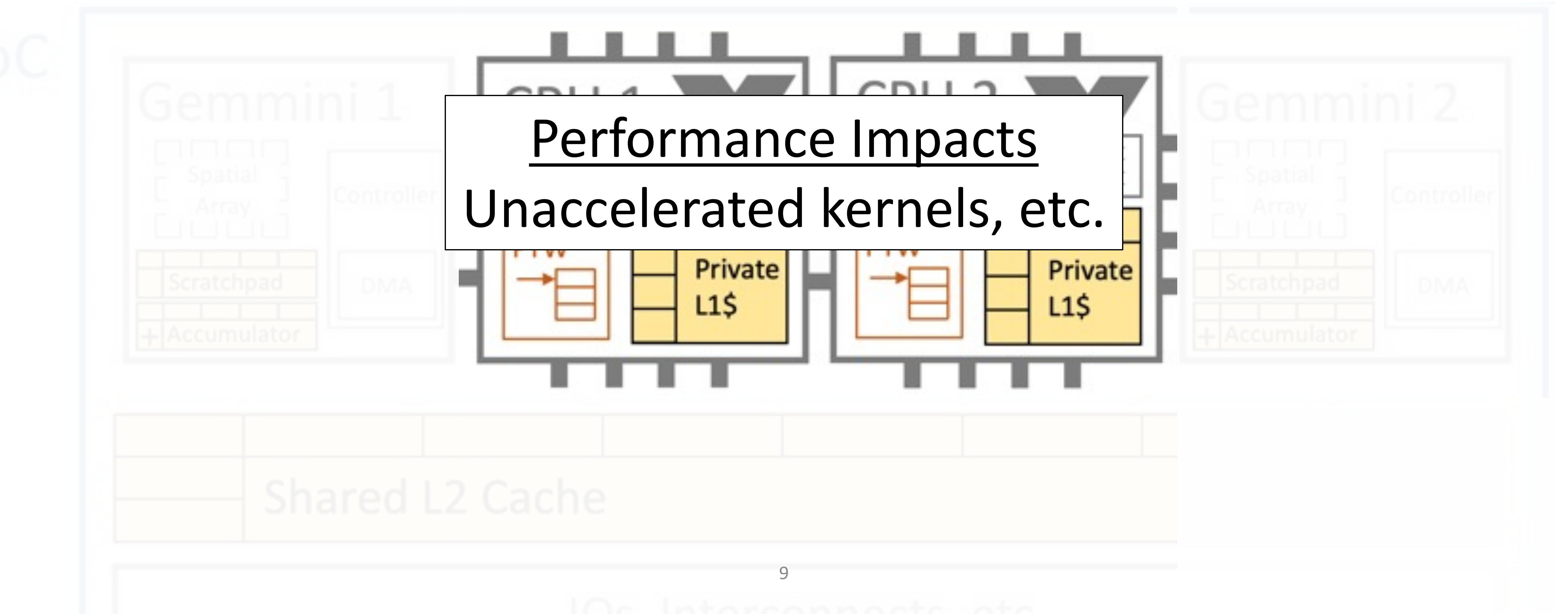
Full-System Visibility: Virtual Addresses



Performance Impacts

Page faults, TLB hits, etc.

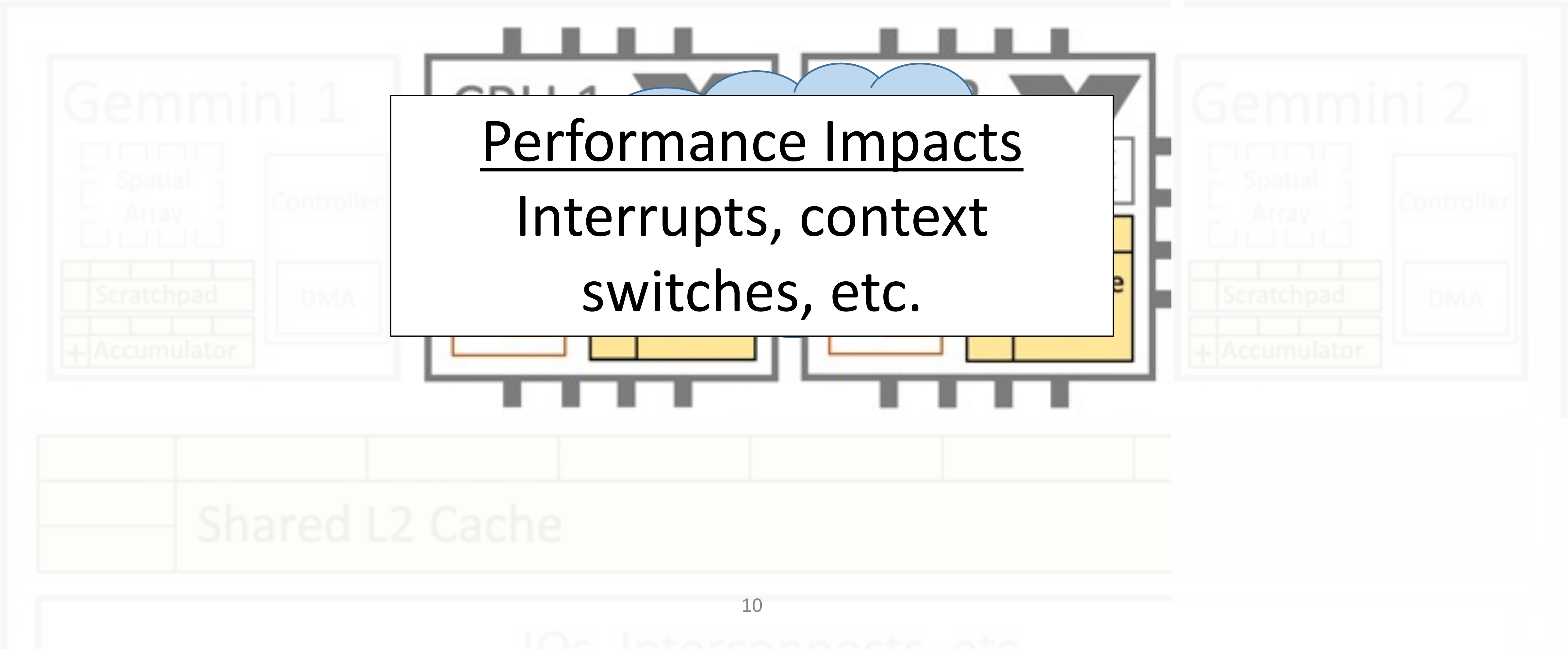
Full-System Visibility: Host CPUs



Performance Impacts

Unaccelerated kernels, etc.

Full-System Visibility: Operating System



<u>Performance Impacts</u>

Interrupts, context switches, etc.

Full-Stack Visibility

ONNX

cuDNN

cuBLAS

Direct hardware configuration, low-level ISA

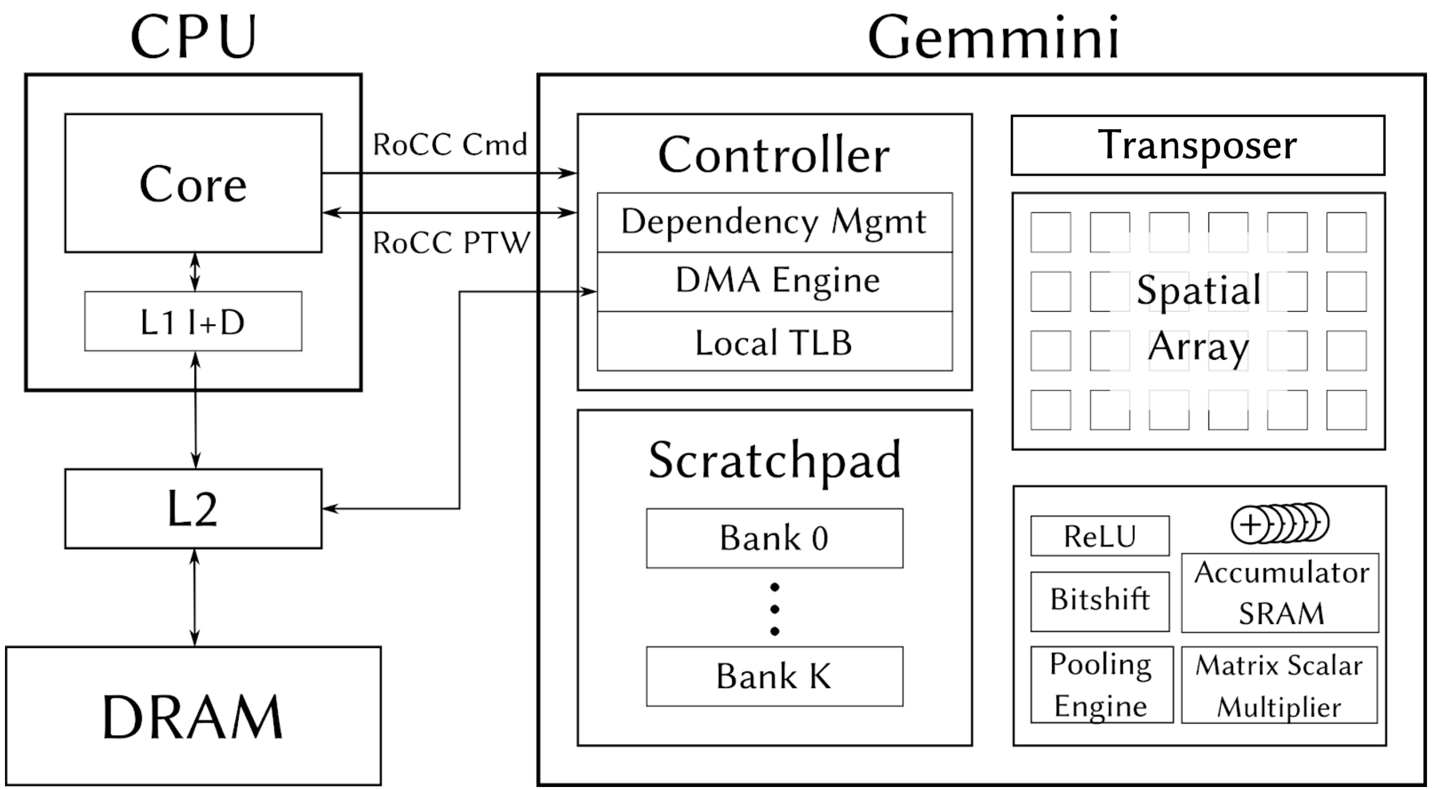
High

Medium

Low

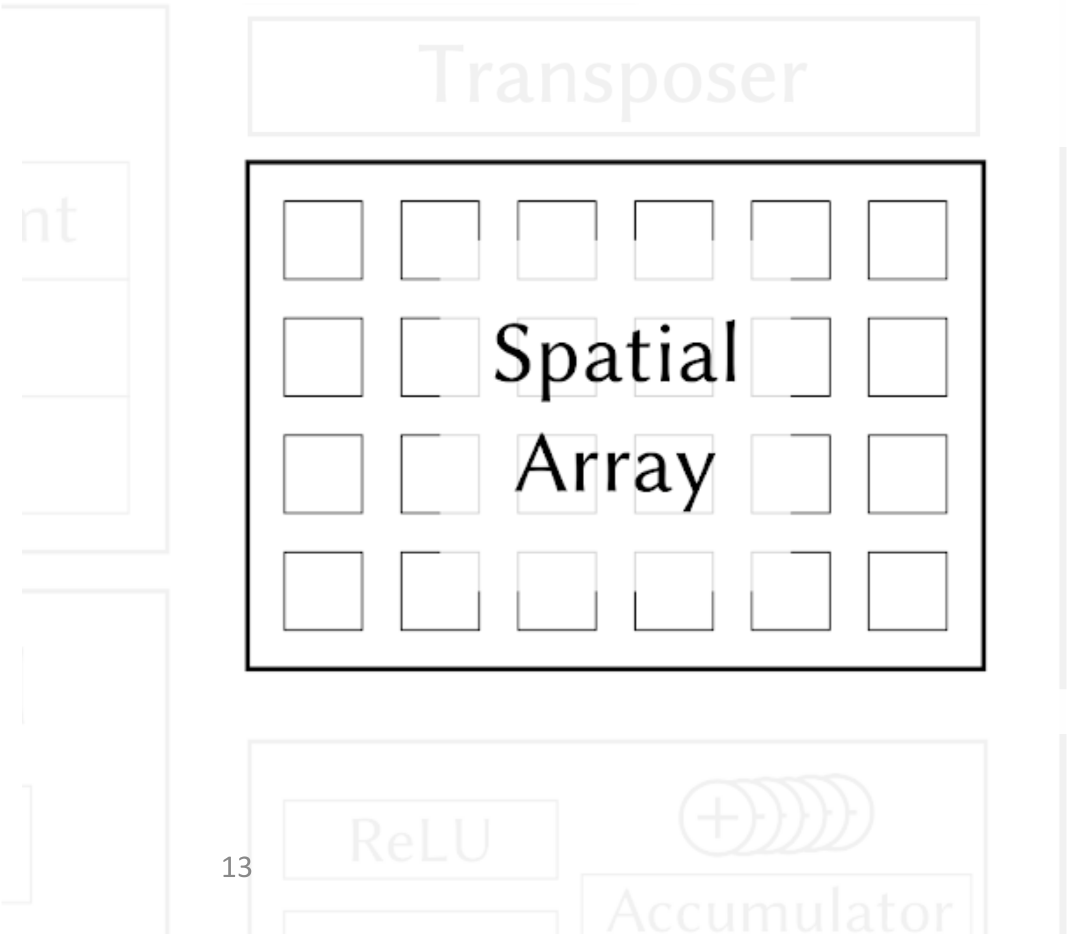
Gemmini

- DNN accelerator generator
  - RTL
  - Simulations
  - Runs Linux
- Flexible hardware template
- Full-stack
- Full-system



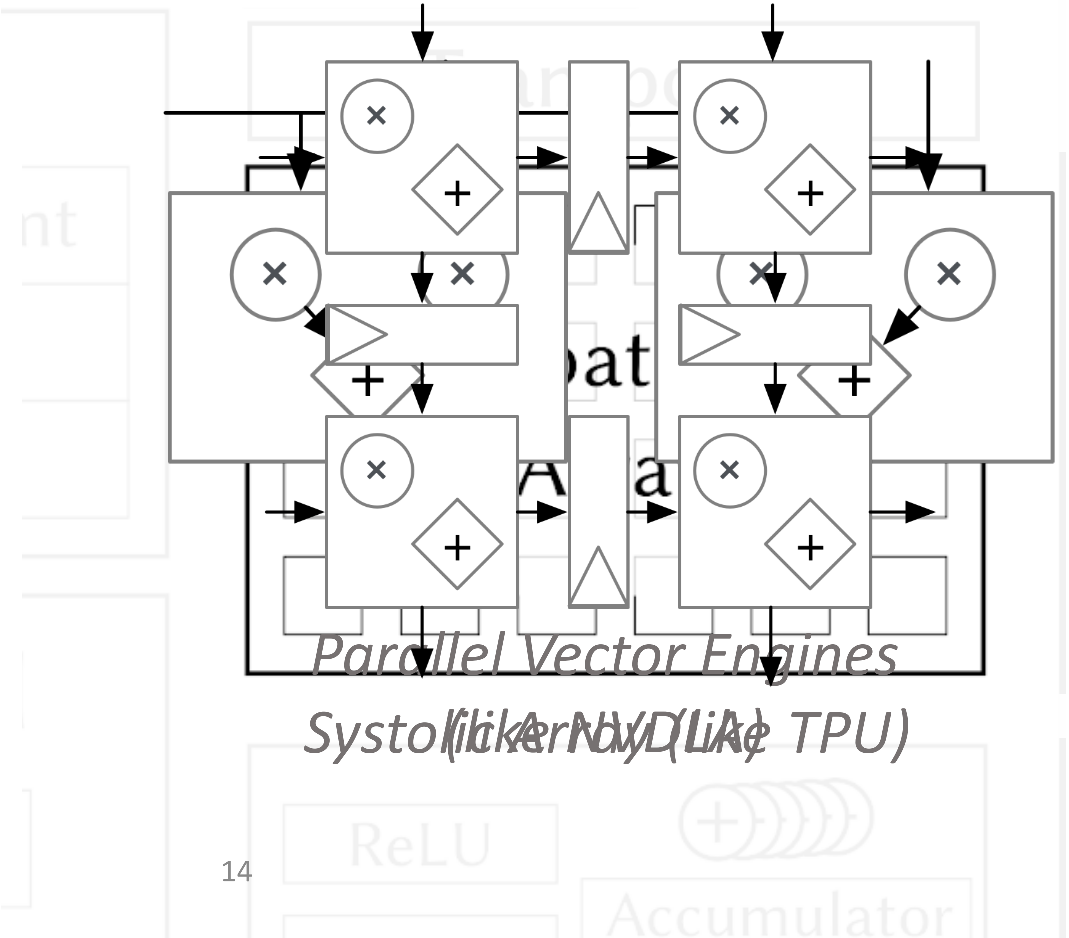
Gemmini: Spatial Array

- Parameters:
  - Dataflow
  - Datatypes
  - Dimensions
  - Pipelining



Gemmini: Spatial Array

- Parameters:
  - Dataflow
  - Datatypes
  - Dimensions
  - Pipelining

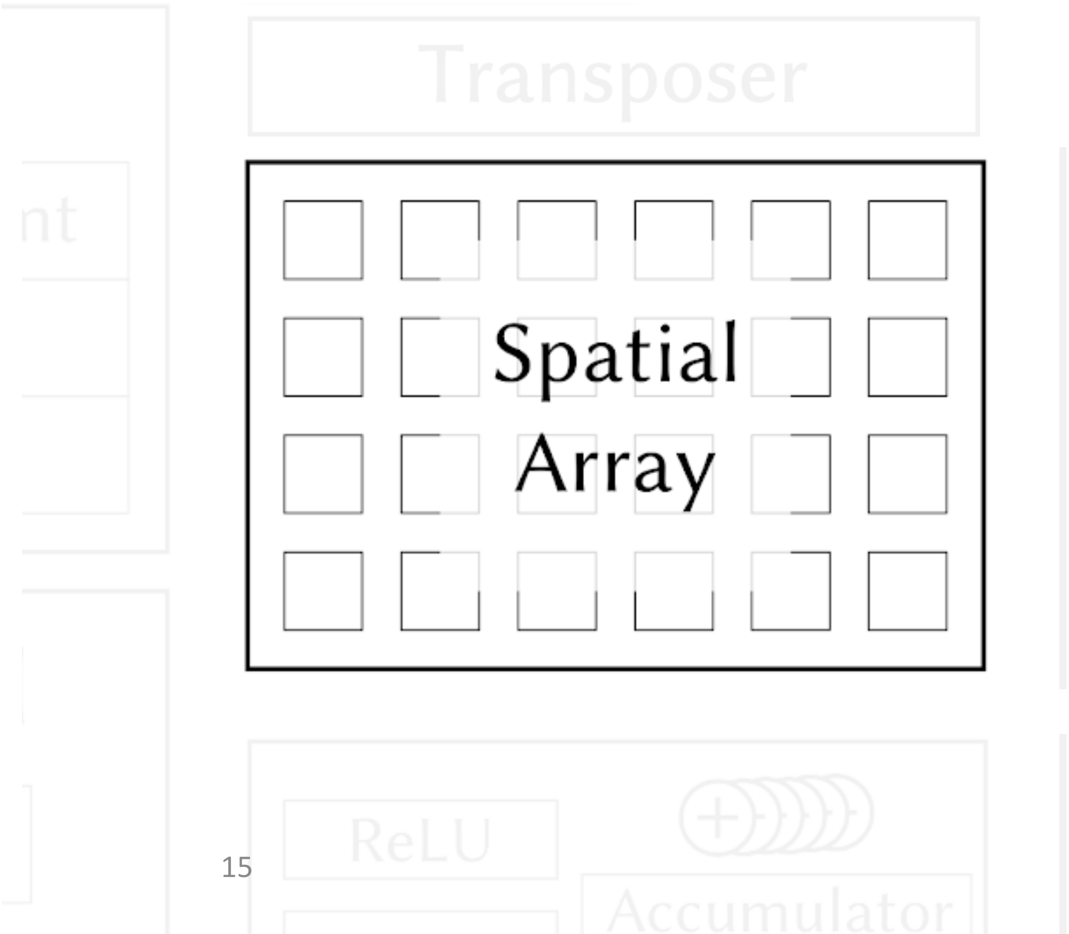


Parallel Vector Engines (like NVDLA)

Systolic Array (like TPU)

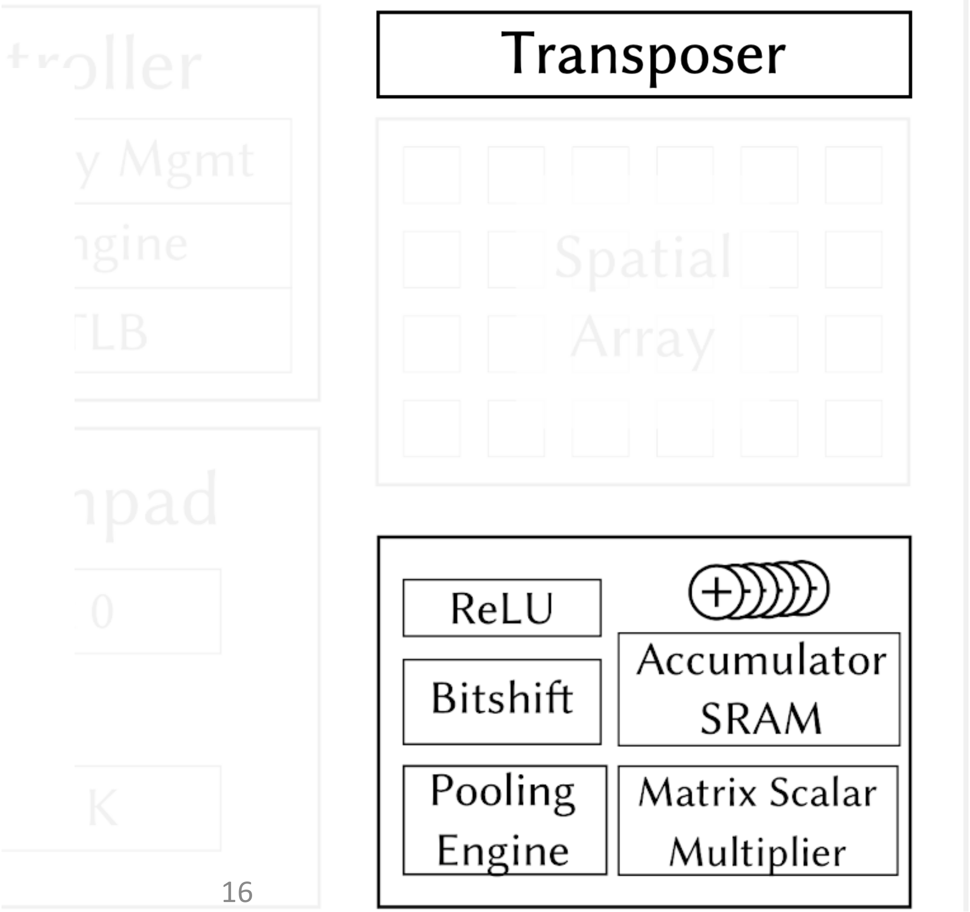
Gemmini: Spatial Array

- Parameters:
  - Dataflow
  - Datatypes
  - Dimensions
  - Pipelining



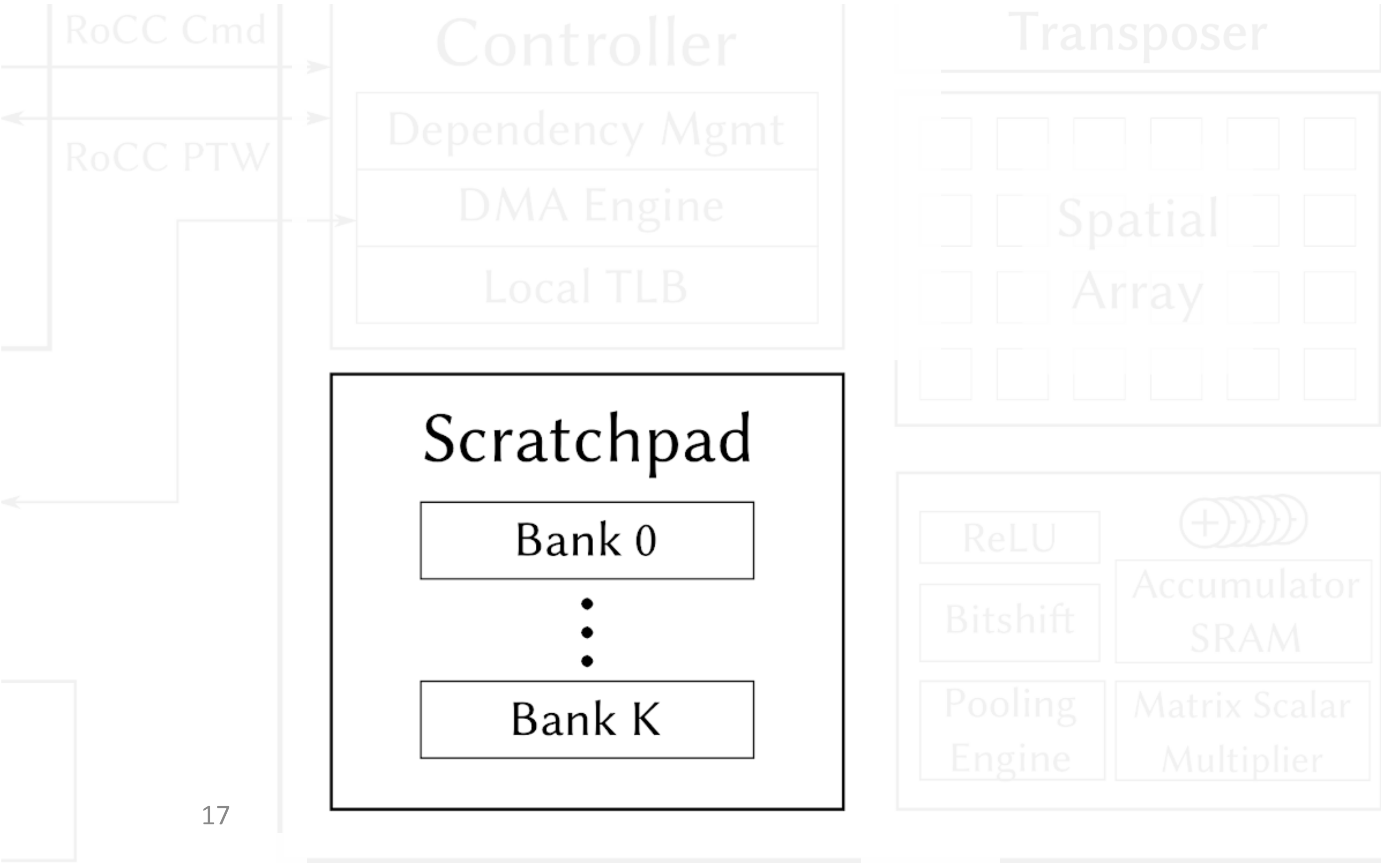
Gemmini: Non-GEMM Functionality

- Can be optimized out at elaboration-time



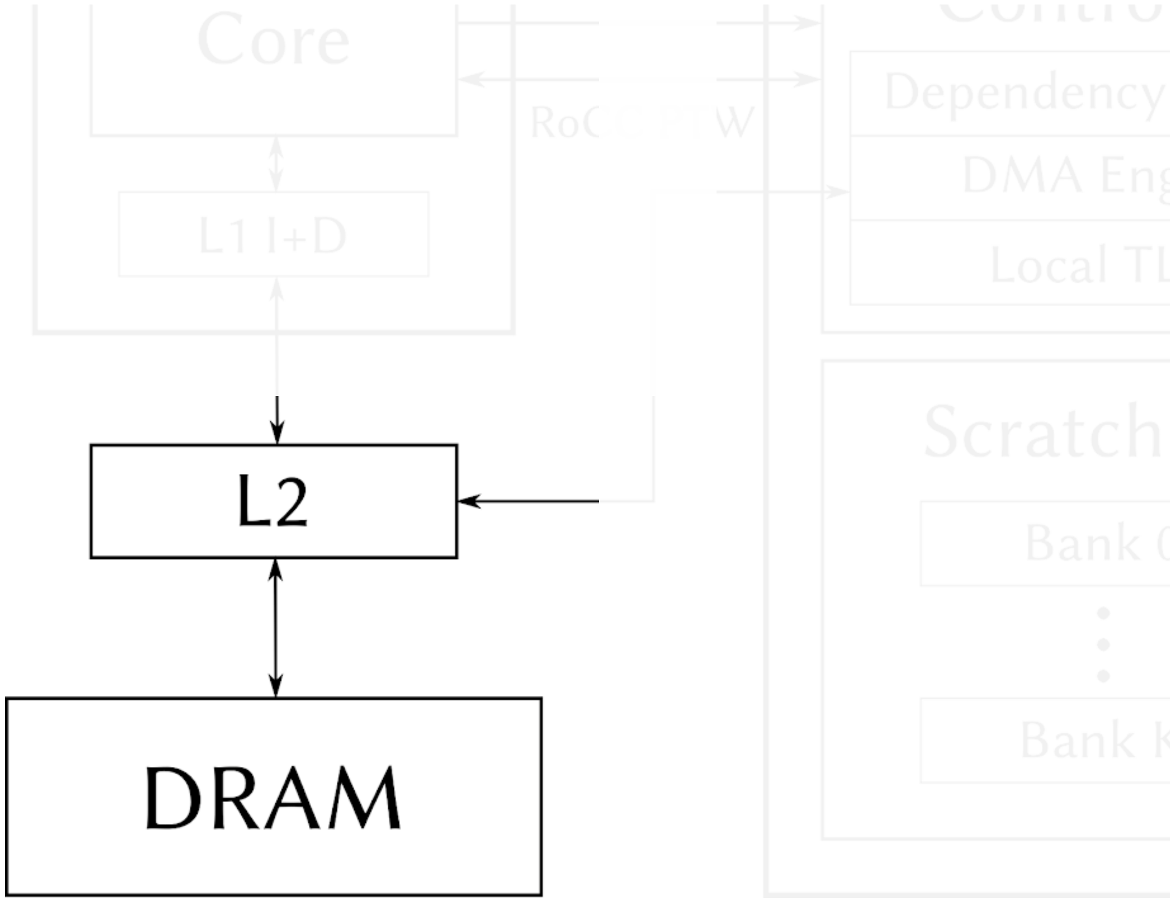
Gemmini: Local Scratchpad

- Parameters:
  - Capacity
  - Banks
  - Single- or dual-port



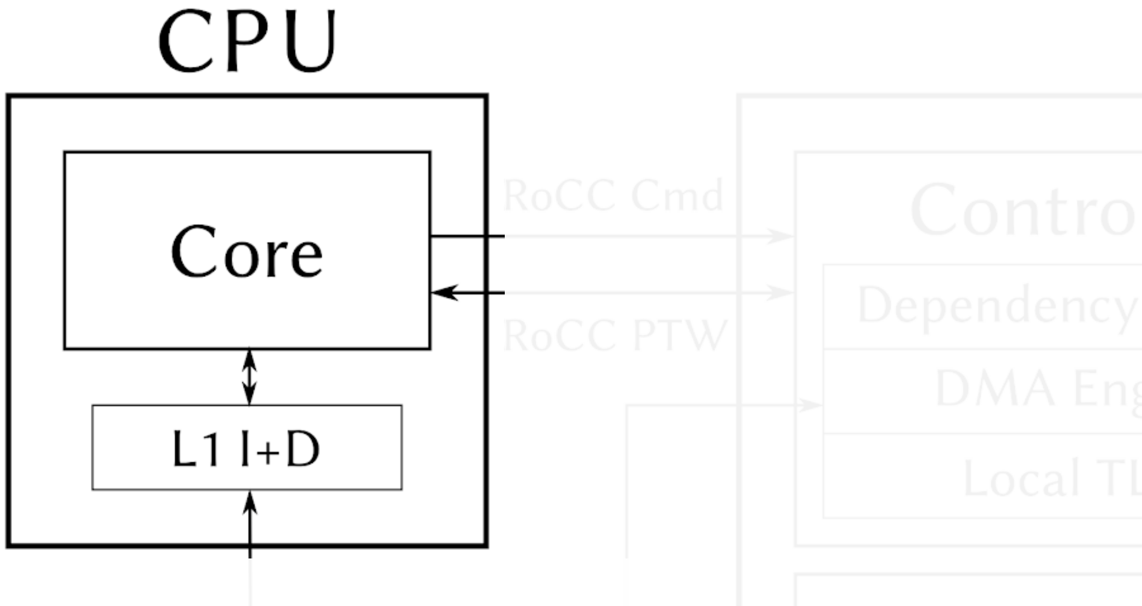
Gemmini: Global Memory

- Parameters:
  - Capacity
  - Banks
  - DRAM controller



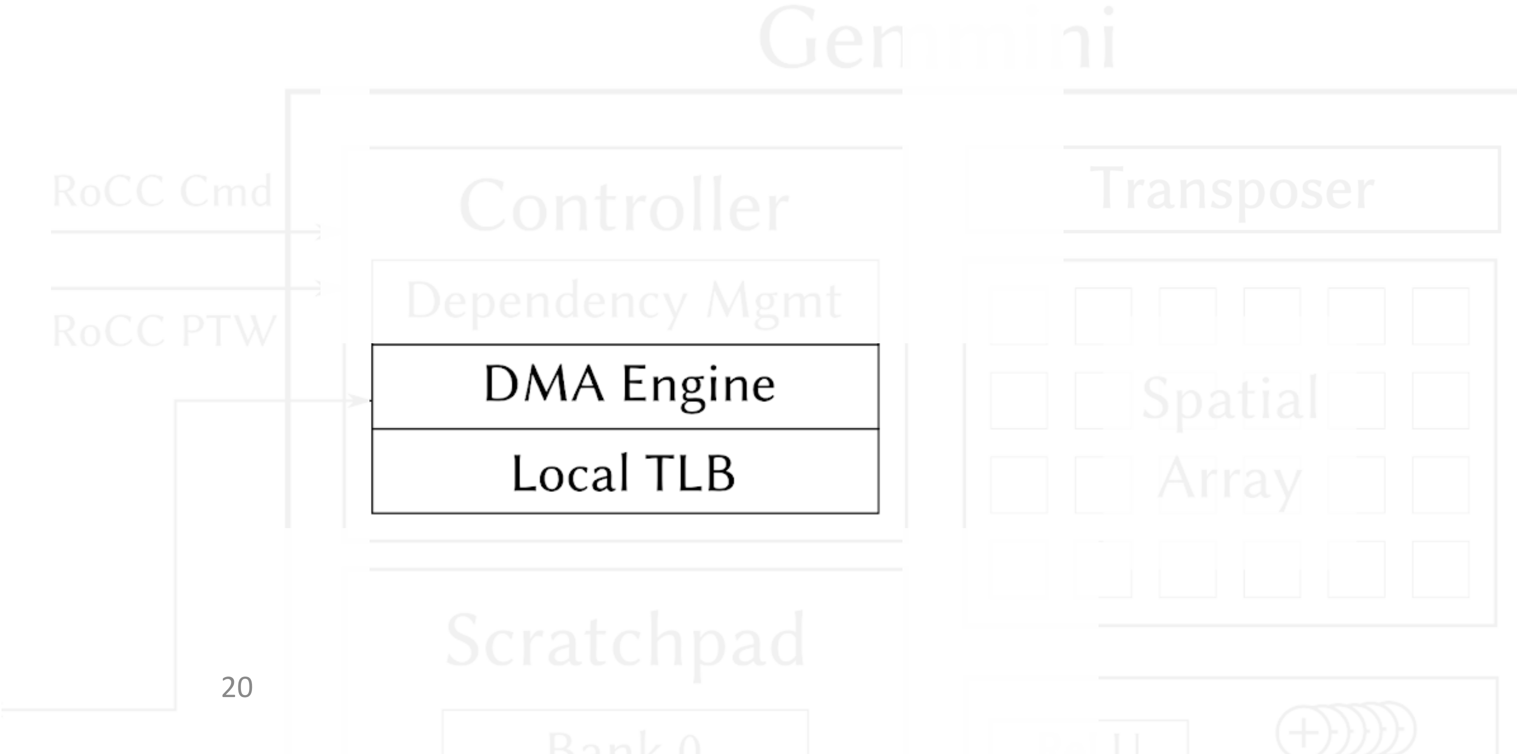
Gemmini: Host CPU

- Parameters:
  - In-order/out-of-order
  - ROB capacity
  - L1 capacity
  - Branch predictor

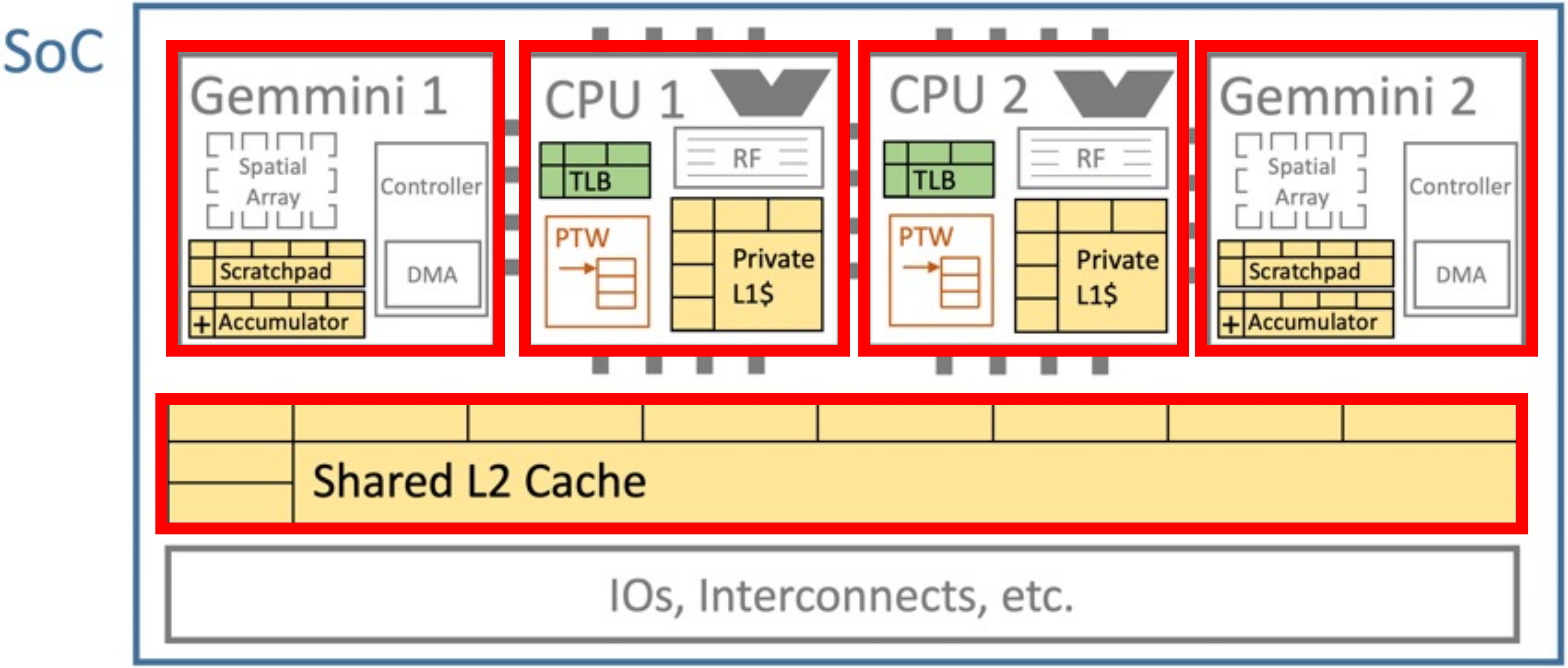


Gemmini: Virtual Address Translation

- Parameters:
  - TLB capacity
  - TLB hierarchy
    - e.g. L2 TLB

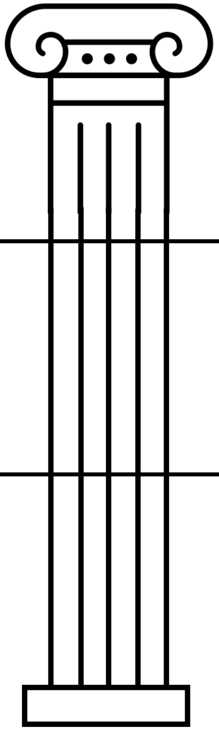


Gemmini: Full SoC



Gemmini: Programming Model

ONNX



Hand-tuned C library for DNNs

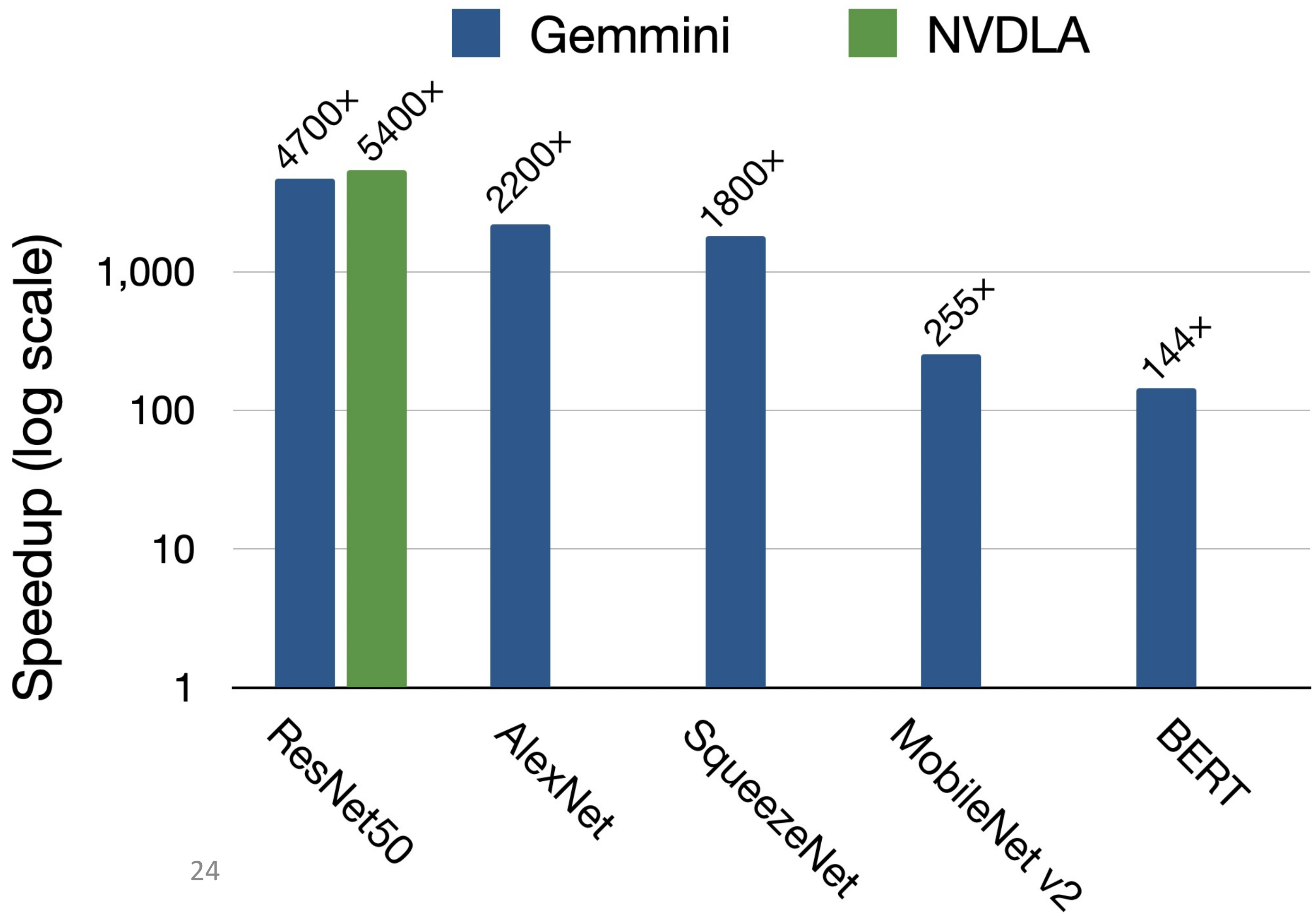
```
matmul(…); conv(…); residual_add(…);
max_pool(…); global_averaging(…)
```

Direct hardware configuration, low-level ISA

```
configure_loads(…); configure_stores(…);
preload_spatial_array(…); feed_spatial_array(…)
```

Performance: Overall

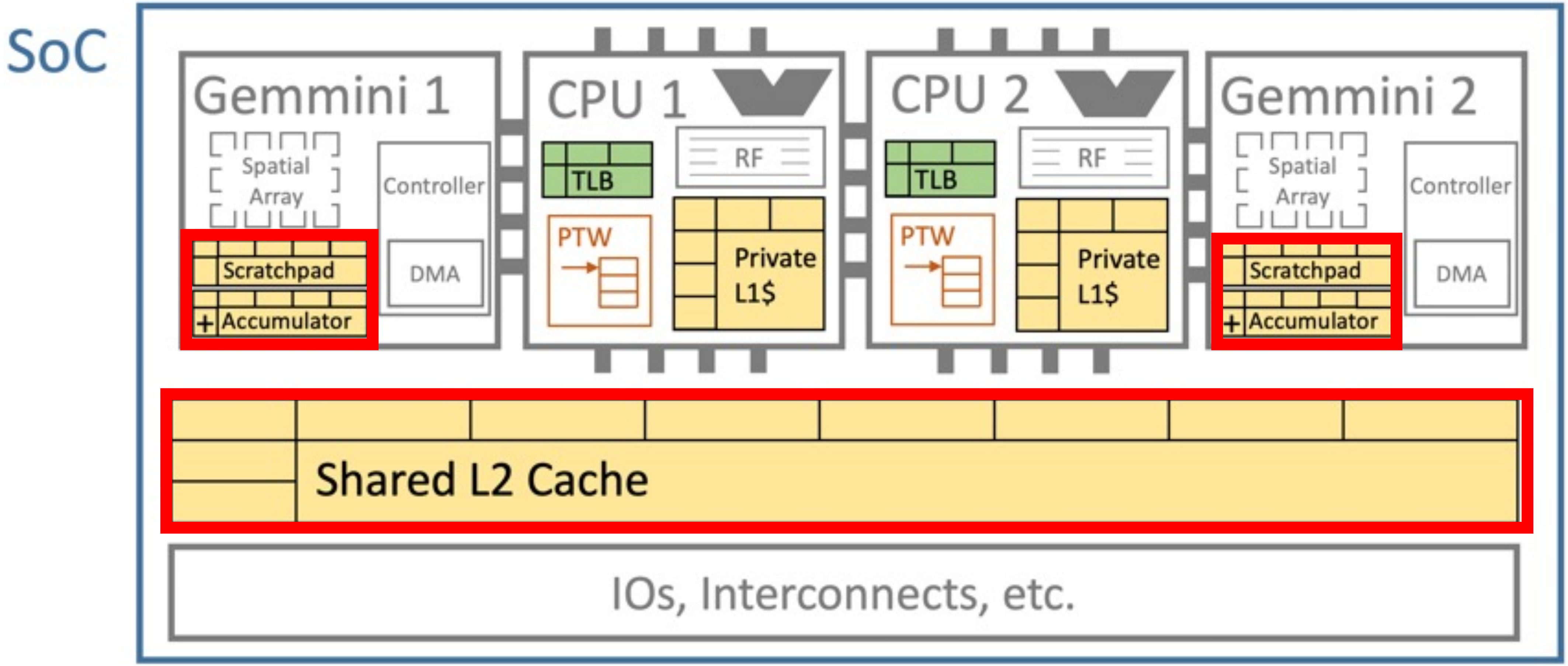
- DNNs:
  - ResNet50: 40.3 FPS
  - AlexNet: 79.3 FPS
  - MobileNet: 37.5 FPS
  - BERT: 167x speedup
- About 80% as fast as NVDLA



How Does the Full System and Full Stack Affect Performance?

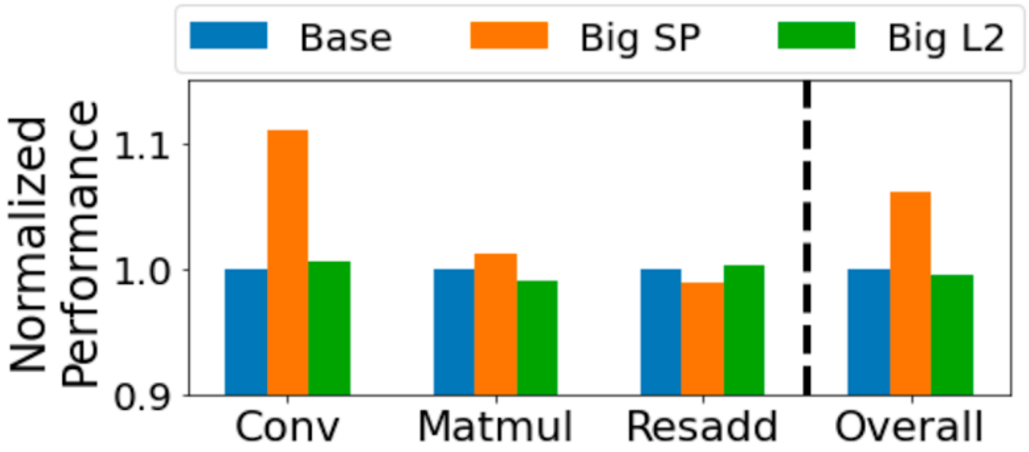
Case Study: Memory Partitioning Schemes for Multi-Accelerator SoCs

Case Study: Memory Partitioning



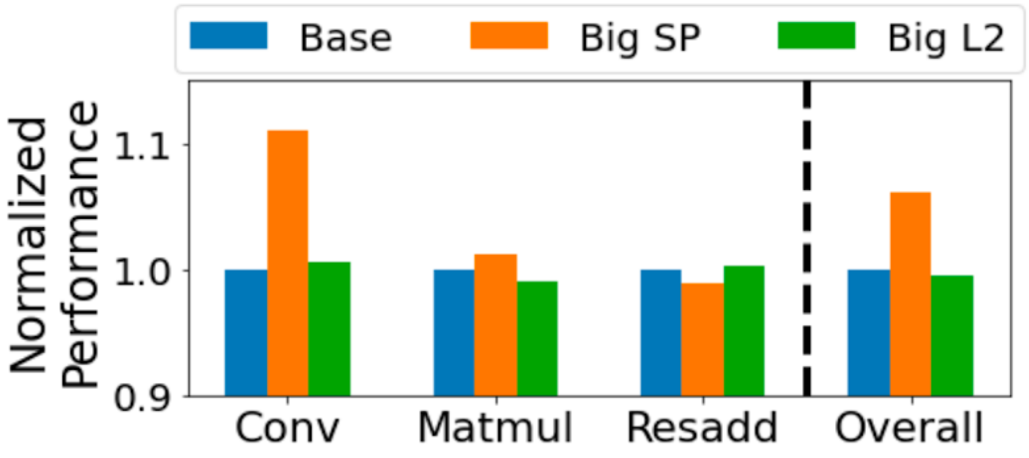
Case Study: Memory Partitioning

- Single core
  - Private scratchpad more helpful
  - Much better for convs

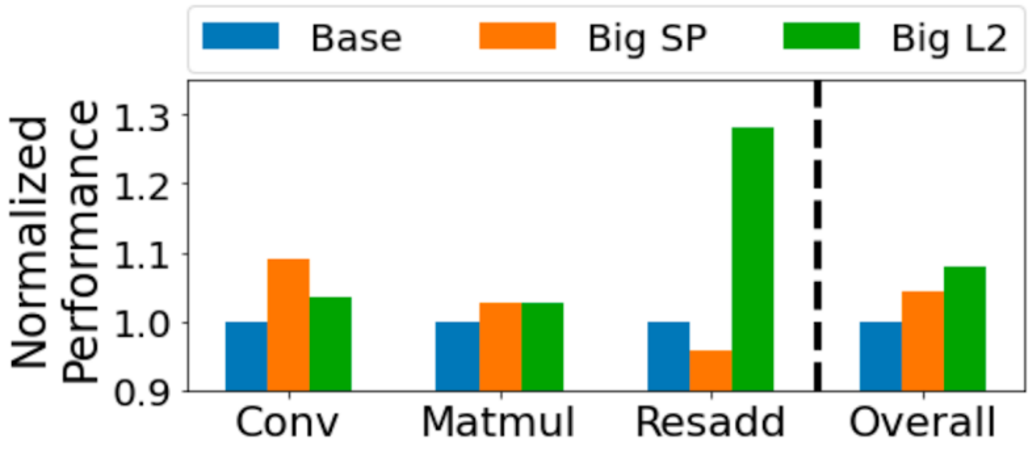


Case Study: Memory Partitioning

- Single core
  - Private scratchpad more helpful
  - Much better for convs



- Dual core
  - Shared L2 more helpful
  - Much better for residual additions

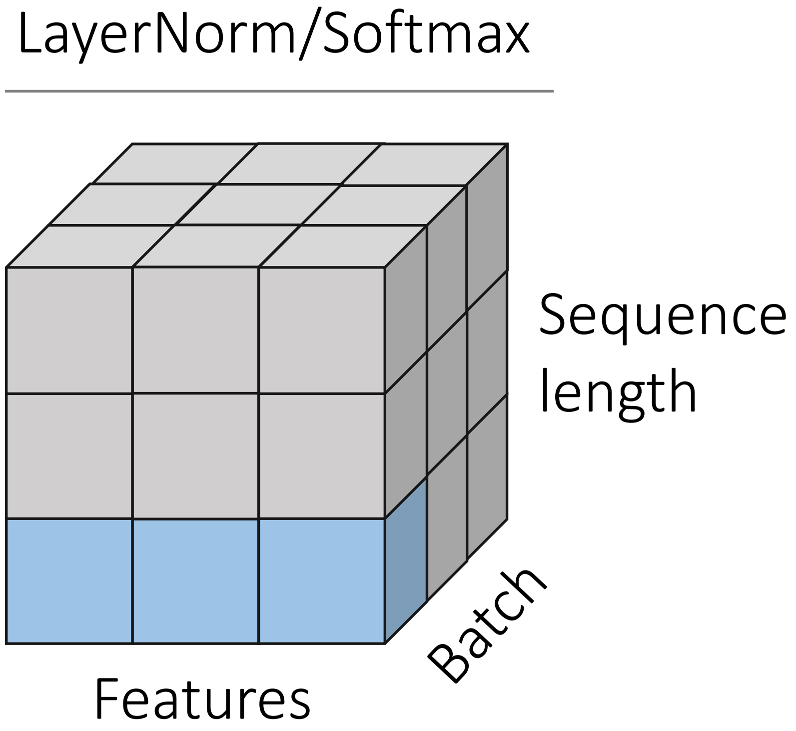


What New Features Are Coming to Gemmini?

Ongoing Work: Transformers

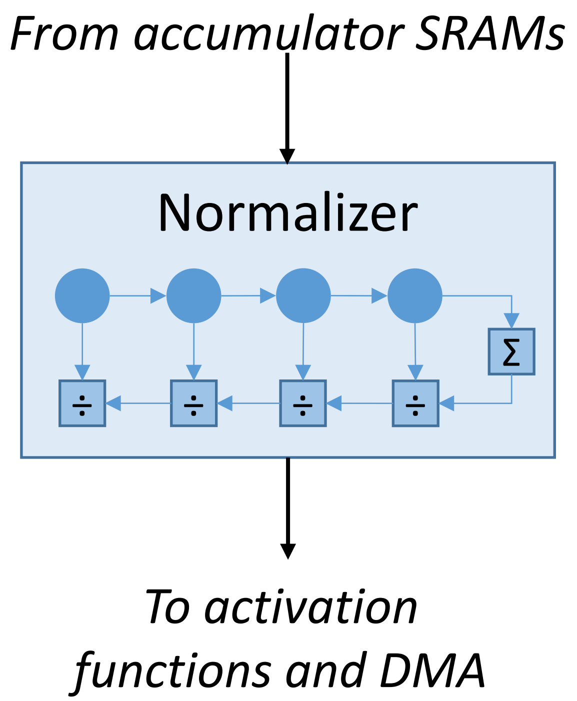
Why Are Transformers Challenging on Gemmini?

- Optimized for dot-products and element-wise operations
- Normalization layers ran on CPU
  - Dequantization cost also incurred
- LayerNorms and Softmaxes were **expensive**



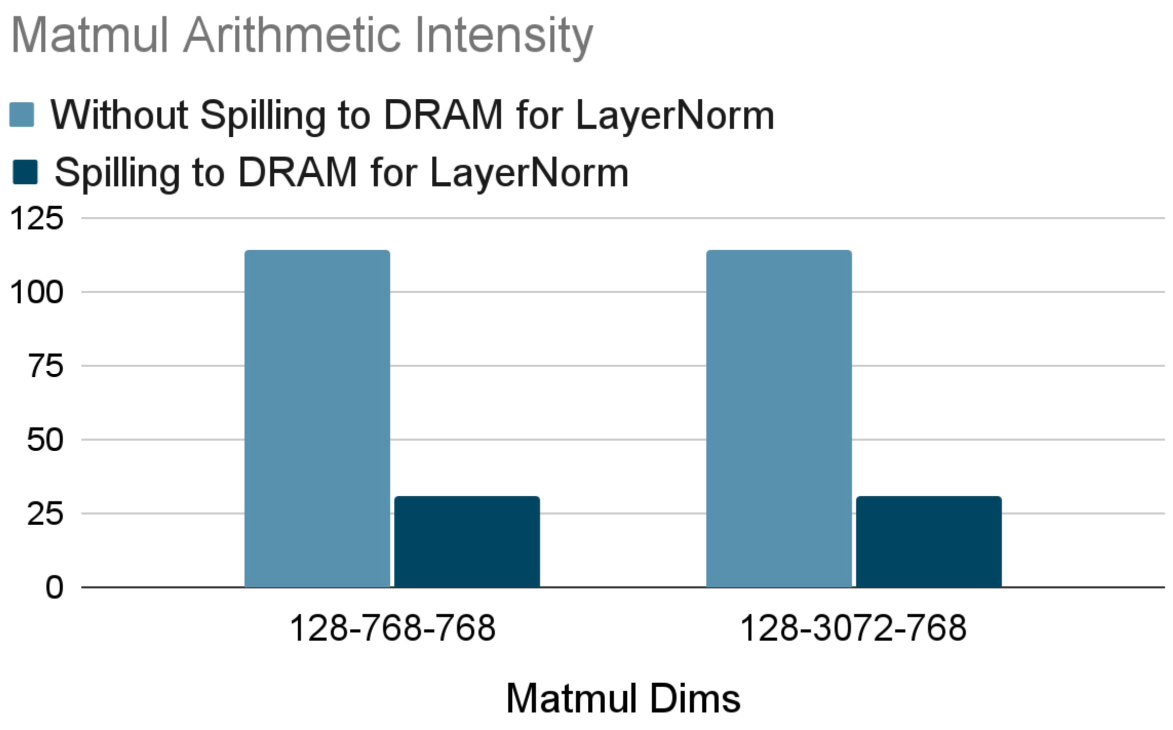
New Normalization Module

- New reduction operations supported
    - LayerNorm
        - Sum/mean
        - Variance/std-dev
    - Softmax
        - Max
        - Sum of exponentials



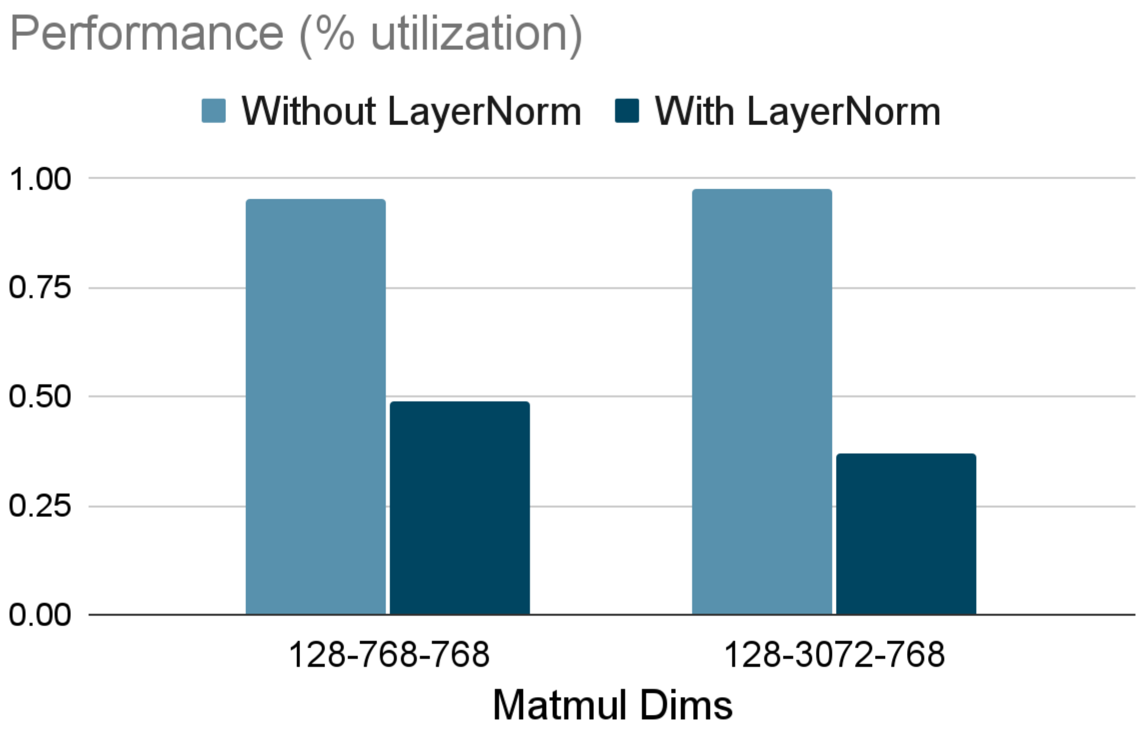
Transformers: New Trade-Offs

- Should the **entire** dimension being normalized be in scratchpad memory?
- Avoiding spilling to DRAM reduces arithmetic intensity
  - -73% reduction for BERT layers with LayerNorm



Transformers: **Preliminary** Results

- Implemented support for I-BERT
  - Integer-only variation of BERT
- **40x faster** than prior BERT implementation
  - Prior implementation in DAC paper
- More performance tuning is required
  - Up to 2x performance degradation for normalized matmuls



Ongoing Work: Sparsity

Wide Design Space to Explore!

- Design space of sparse accelerators is very wide!
- Design points differ in:
  - NoC complexity
  - Data re-use
  - Supported sparse data formats
  - Memory access patterns
    - Scattered accesses
  - Key-matching, output-merging cost
    - May be larger than matmul unit!

Sparsity: Separation of Concerns

- We need separation of concerns!

Sparsity: Separation of Concerns

- We need separation of concerns!
- Functional behavior
  - E.g. matmul, convolution, sorting, etc.

Matmul

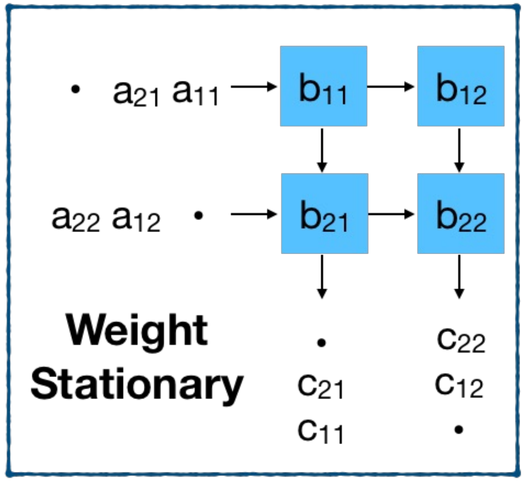
$$C_{ij} = \sum_k A_{ik} B_{kj}$$

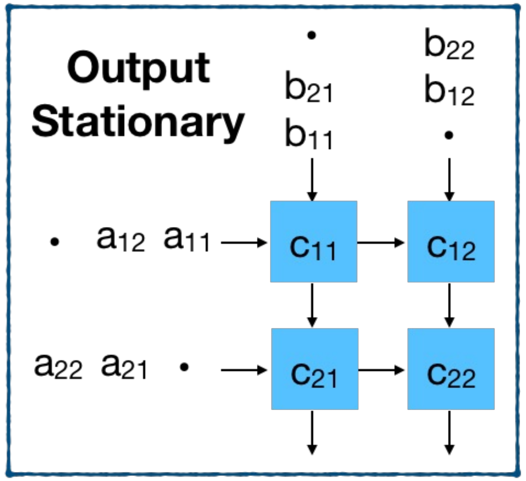
MTTKRP

$$A_{ij} = \sum_k \sum_l B_{ikl} D_{lj} C_{kj}$$

Sparsity: Separation of Concerns

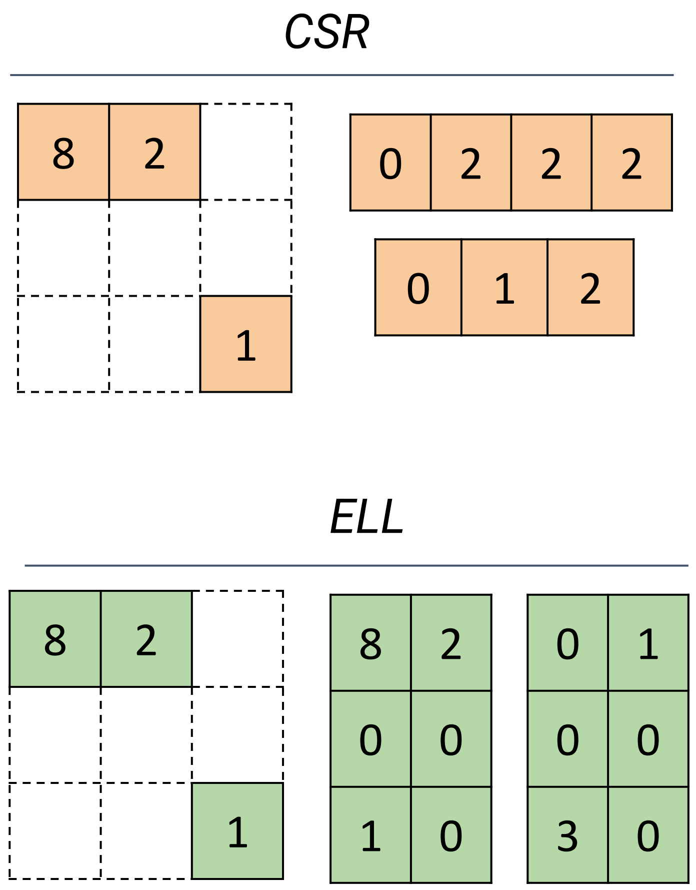
- We need separation of concerns!
- Functional behavior
  - E.g. matmul, convolution, sorting, etc.
- Dataflow
  - E.g. matmul, convolution, sorting, etc.





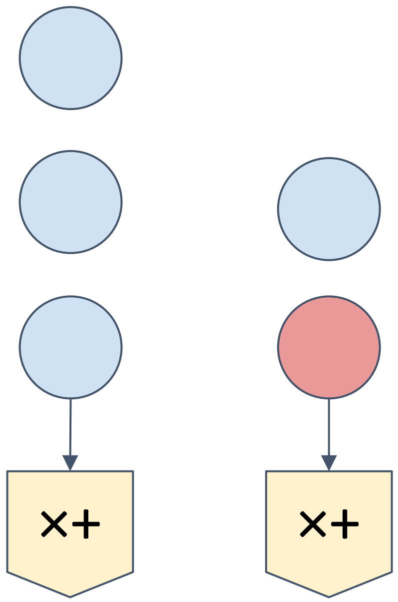
Sparsity: Separation of Concerns

- We need separation of concerns!
- Functional behavior
  - E.g. matmul, convolution, sorting, etc.
- Dataflow
  - E.g. matmul, convolution, sorting, etc.
- Data formats
  - E.g. CSR, ELL, DBB, diagonal, etc.

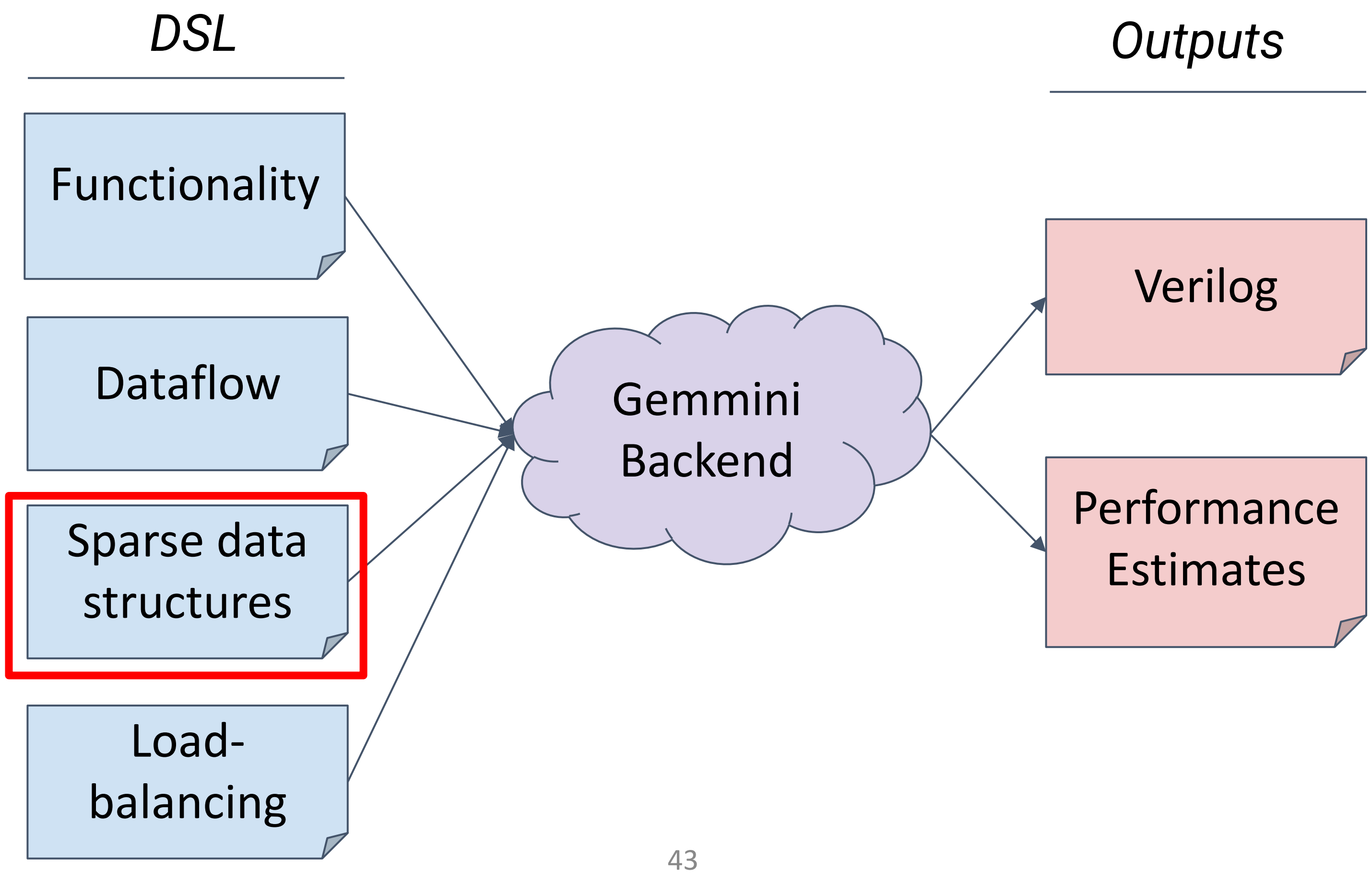


Sparsity: Separation of Concerns

- We need separation of concerns!
- Functional behavior
  - E.g. matmul, convolution, sorting, etc.
- Dataflow
  - E.g. matmul, convolution, sorting, etc.
- Data formats
  - E.g. CSR, ELL, DBB, diagonal, etc.
- Load balancing
  - Affects cost of NoC



Sparsity: Overview



Expressing Sparse Data Structures

- Express sparsity in terms of which iterators are "skipped"

<u>A * B = C where A and B are CSR</u>

Skip *k* if A(*i*,*k*) == 0

Skip *j* if B(*i*,*k*) == 0

<u>A * B = C where A is diagonal</u>

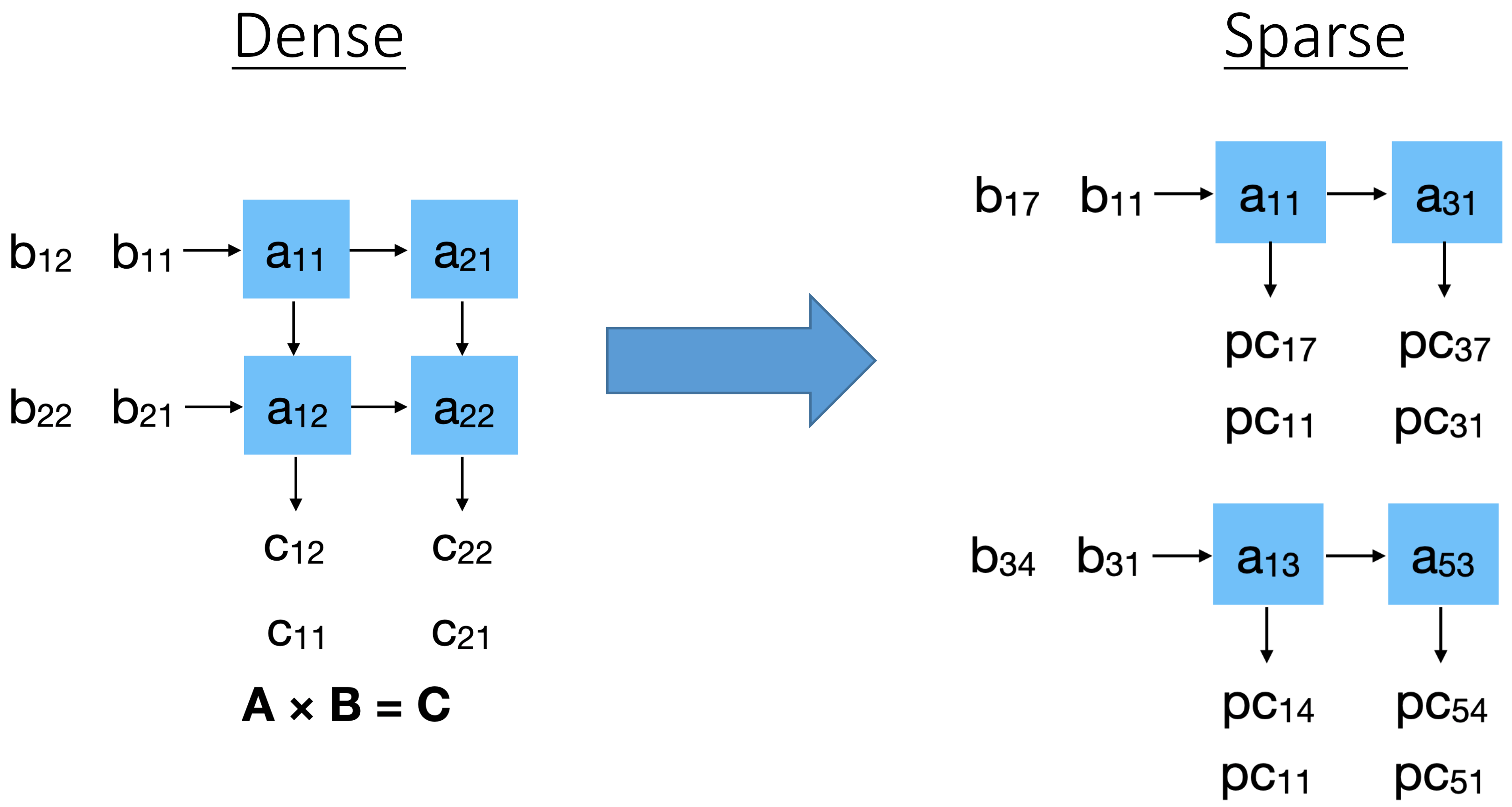
Skip *i* if *i* != *k*

Skip *k* if *i* != *k*

<u>Simple "row-sparsity" example</u>

Skip *k* if A(*i*,->) == 0

Sparsity: Spatial Array Generation



Upcoming Events: MLSys Tutorial

MLSys 2022 Tutorial

- New transformer features
- New parallelization features for ONNX-Runtime
  - Parallelizing across *multiple* accelerators
- New performance models
  - Uses Timeloop
  - Enables faster DSE
- New usability improvements
  - Run many simulations in parallel

Conclusion

- Gemmini is:
  - Full-system
  - Full-stack
- Enables DSE and hardware/software co-design
- Open-source!
  - github.com/ucb-bar/gemmini

- Upcoming events:
  - MLSys Tutorial
    - Santa Clara, CA
    - September 1st, 2022

Funded by DARPA RTML program (contract FA8650-20-2-7006)