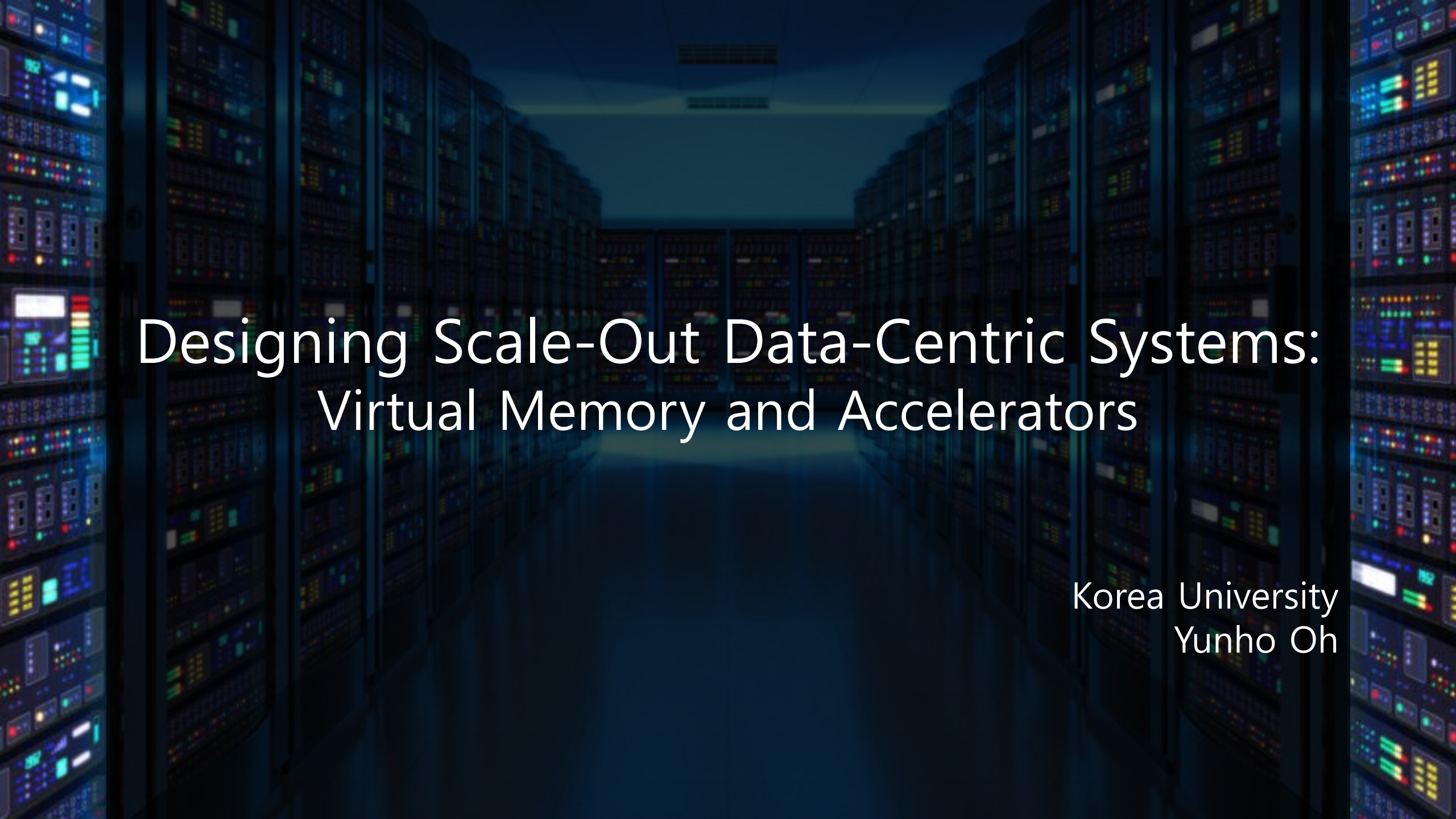
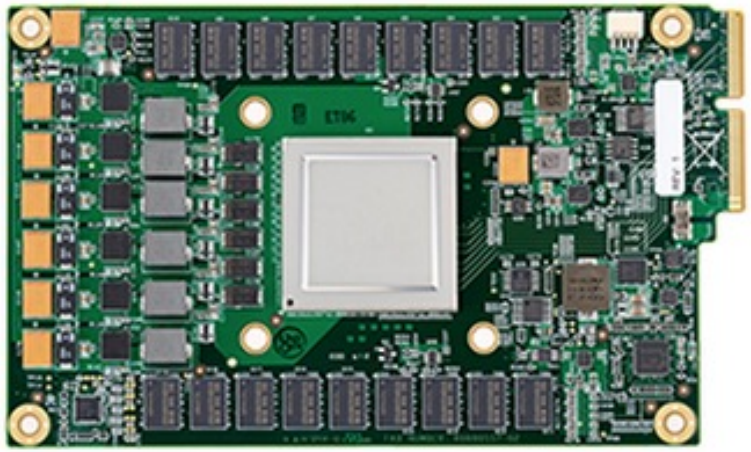


Designing Scale-Out Data-Centric Systems: Virtual Memory and Accelerators

Korea University
Yunho Oh

Market-Leading Accelerators

Graphics Processing Unit
(GPU)



Customized DNN Accelerators
(Google TPU)

What are challenges for energy-efficient accelerators?

A Key Question

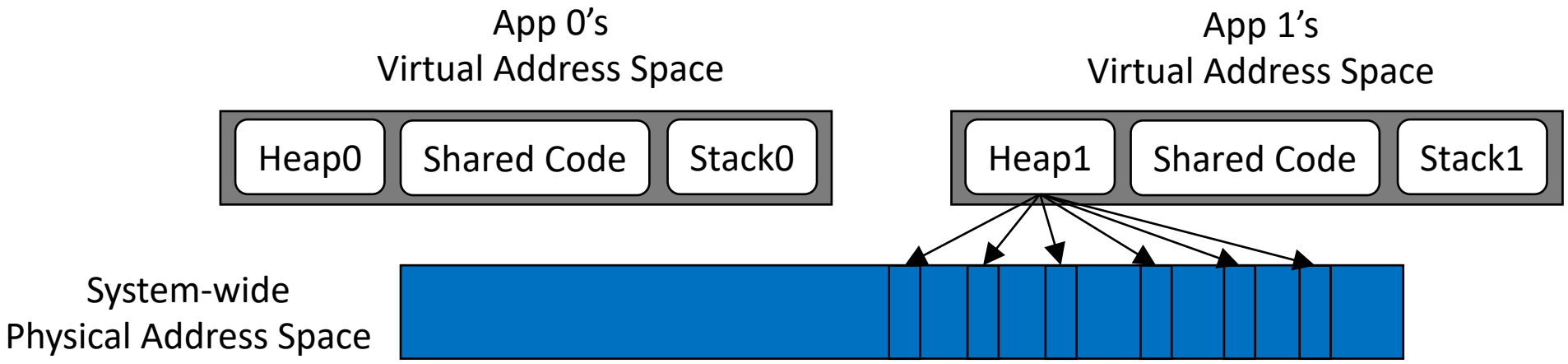
How can we achieve sustainable scalability?

Agenda

- Introduction
- Midgard
- Scale-Out Systolic Array
- Conclusion

Address Spaces in VM

- Virtual address space consists of Virtual Memory Areas (VMAs)
- Protection is defined at a VMA granularity

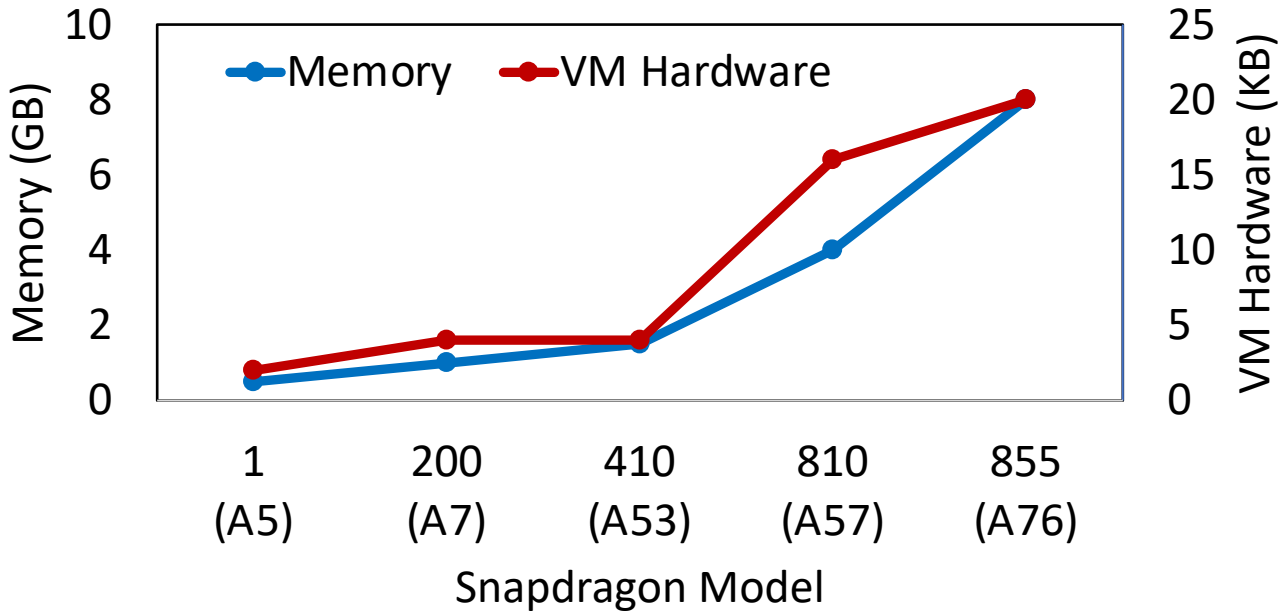


- Physical address space consists of fixed-size pages
- Pages are required for efficient capacity management

VMAs are divided into pages and placed in the physical address space

Virtual Memory is Broken!

- VM implementations are based on fixed-size pages.

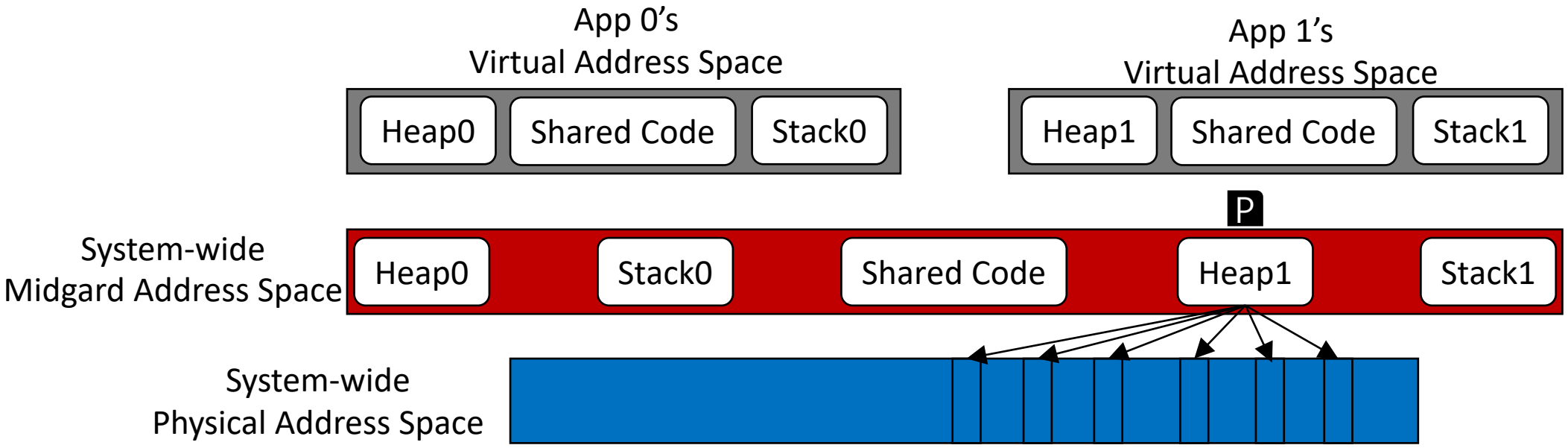


- High silicon cost (~10% of core's silicon area)
- Cannot keep adding resources because of the end of Moore's law

Virtual Memory do not scale with memory capacity.

Midgard

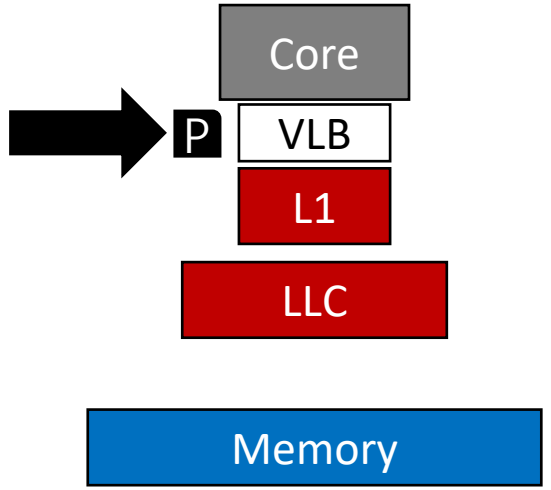
- Introduce an intermediate address space that retains VMAs
  - Protection check and contiguous translation at VMA granularity
  - Use the intermediate address space to address the cache hierarchy



Retaining VMAs simplifies memory protection and translation

Virtual-to-Midgard Translation

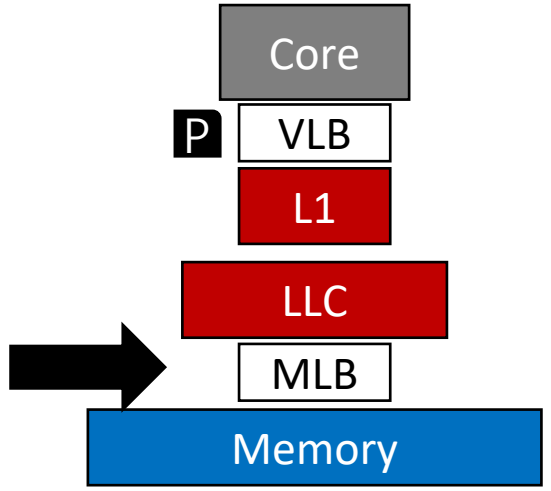
- Translation and protection at VMA granularity
  - Process-private VMA table contains mappings
  - Each process typically contains ~100 VMAs
- Virtual Lookaside Buffer (VLB)
  - Cache VMA mappings to benefit from locality
  - Range-based VLB entry: Wide coverage
  - Only ~10 VMAs are frequently accessed



Only ~10 VMA entries required per core

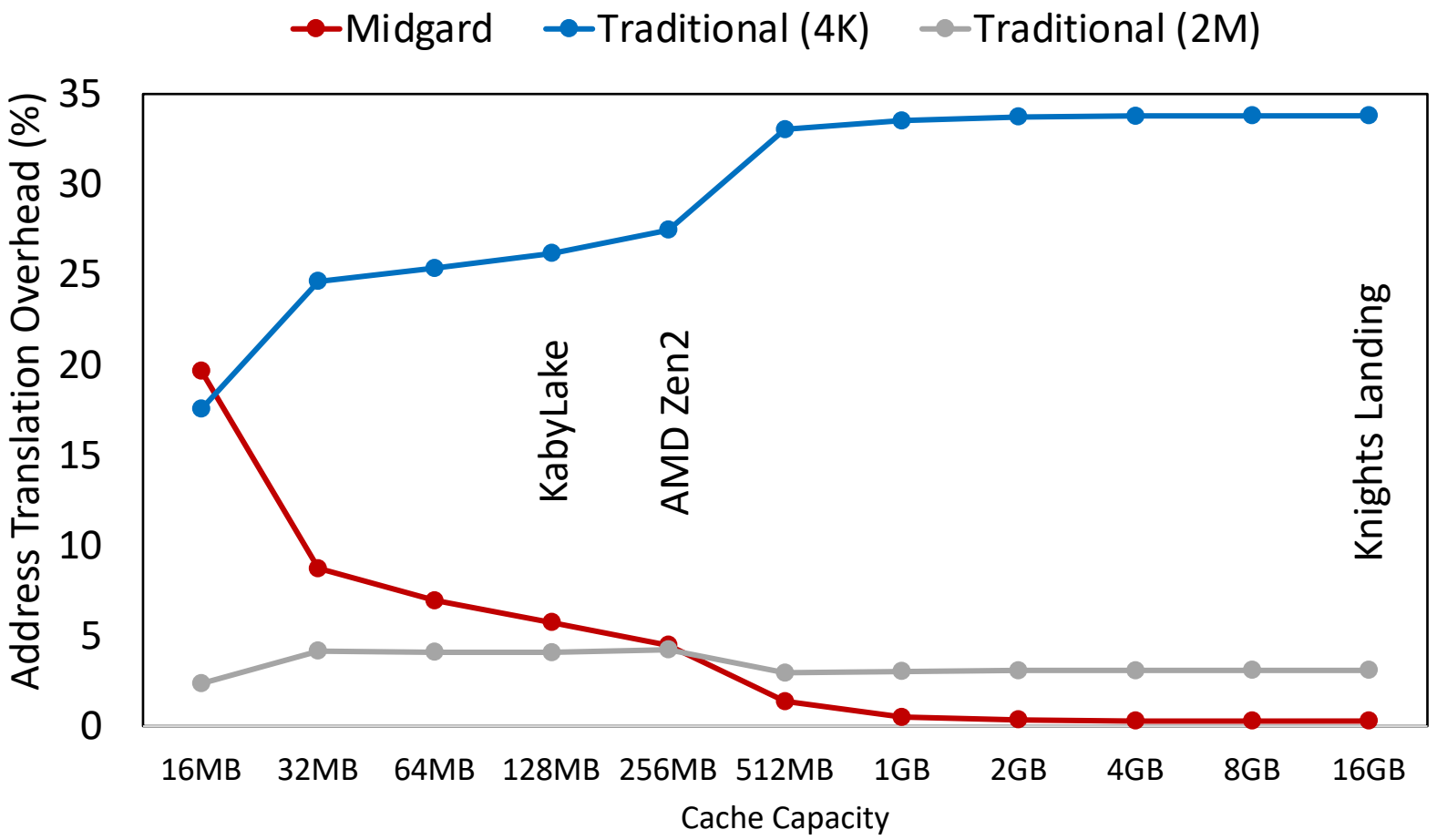
Midgard-to-Physical Translation

- Cache hierarchy filters most of the memory accesses
  - Translation required only for cache misses
  - Little locality left in the translation requests
- Translations stored in Midgard Page Table
  - Shared by all the processes/cores
  - Optionally cache in Midgard Lookaside Buffers (MLBs)



Required only for cache misses

Future-Proofing VM with Midgard

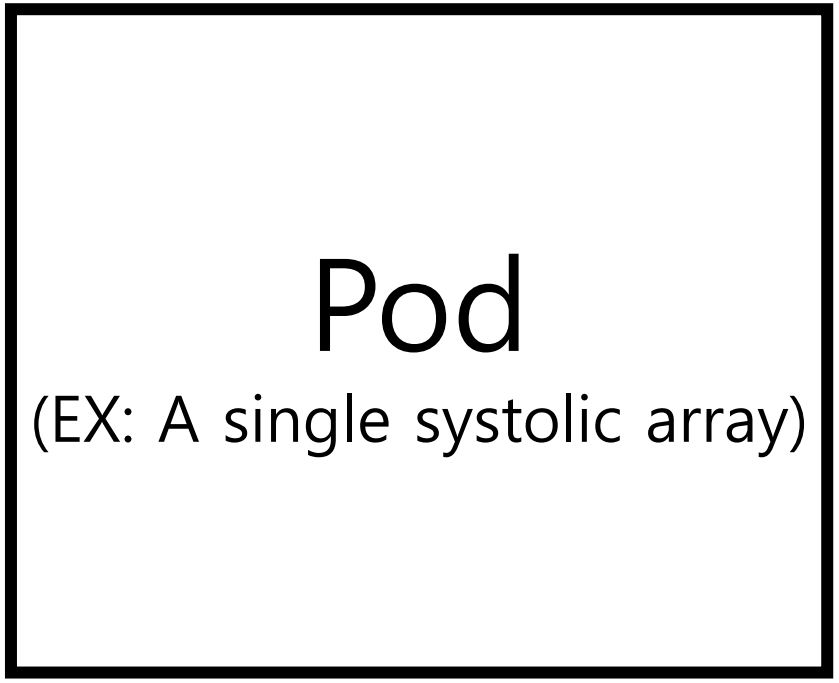


Midgard performance improves with the cache hierarchy capacity

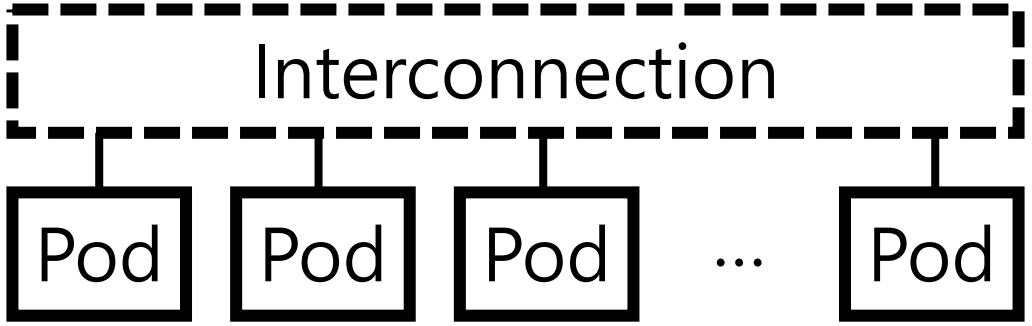
Agenda

- Introduction
- Midgard
- Scale-Out Systolic Array
- Conclusion

Spectrum of DNN Accelerator Designs



Monolithic Architecture



Multi-Pod Architecture

Design Space

Monolithic Architecture

- Accelerator having a big systolic array

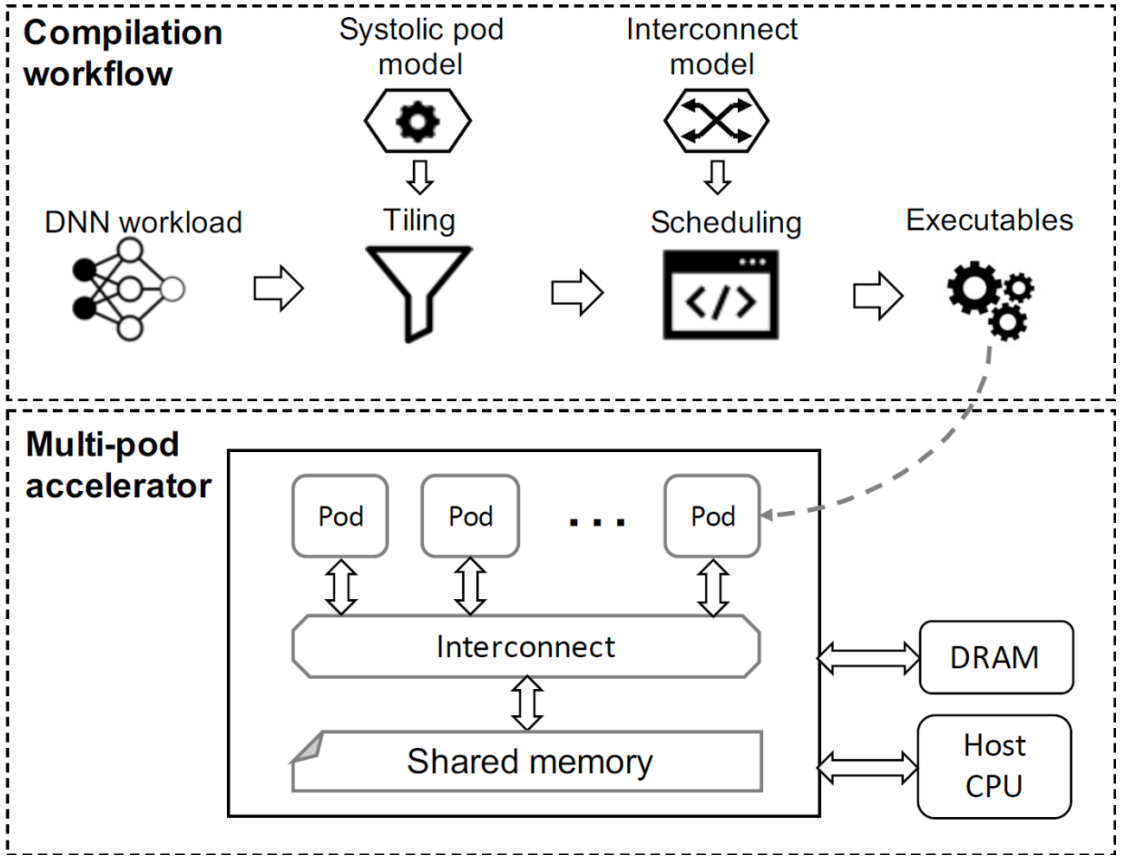
- Example: Size of Systolic Array per Core in Google TPUs
  - v1: 256x256 (1 core)
  - v2: 128x128 (1 cores)
  - v3: 128x128 (2 cores)
  - v4: 128x128 (4 cores)

Data from Ten Lessons From Three Generations Shaped Google's TPUv4i, ISCA 2021

Multi-Pod Architecture

- Academia is actively working on it.

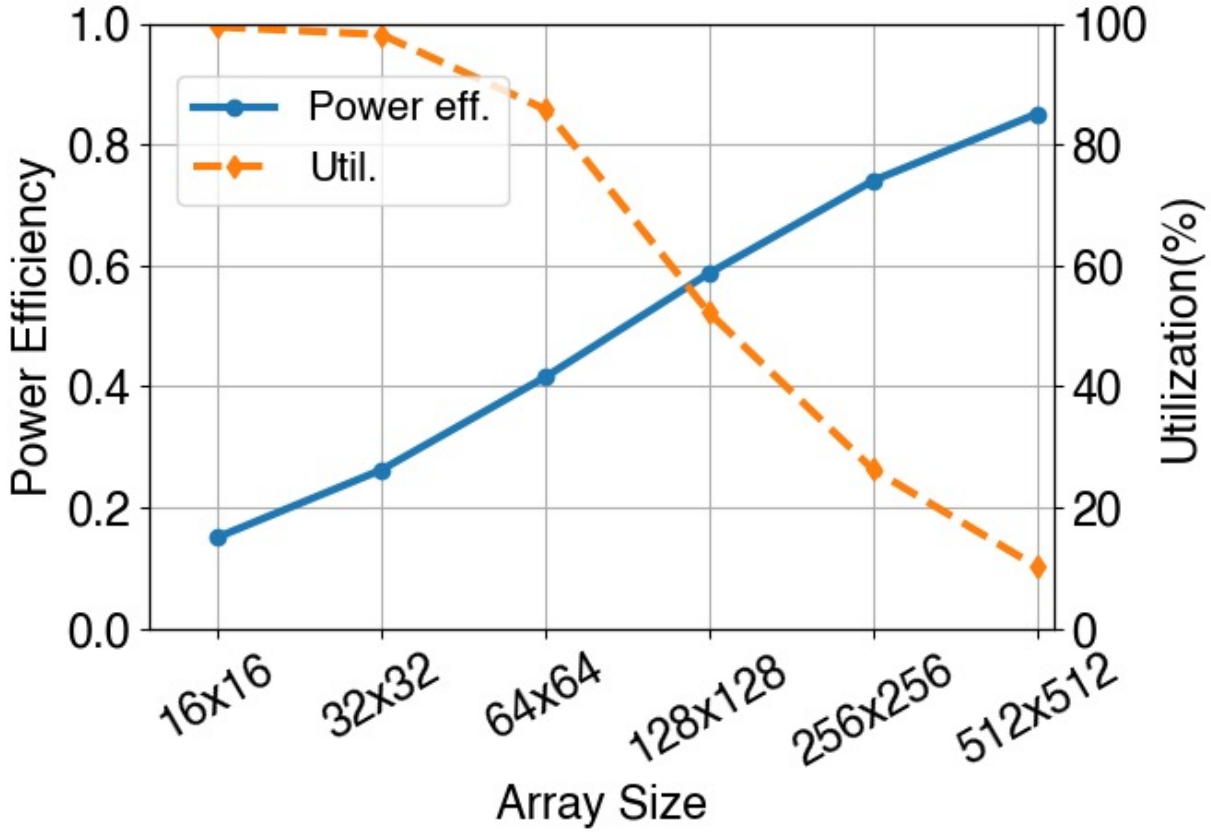
- Example: MAESTRO [Kung et al. ASAP 2019]
  - Single pod: 8x8 systolic array
  - A single accelerator may have 1~4096 systolic arrays
  - Improves utilization compared to TPUs.



Effect on Systolic Array Size

- Power efficiency:
  Peak Throughput/watt

- If utilization goes down, power efficiency does not reflect real performance.



No Too Big or Too Small Systolic Arrays

Key Question

How do we determine optimal systolic array size?

What Should We Focus?

- Sizing systolic arrays (Granularity)
- Interconnect
- Tiling

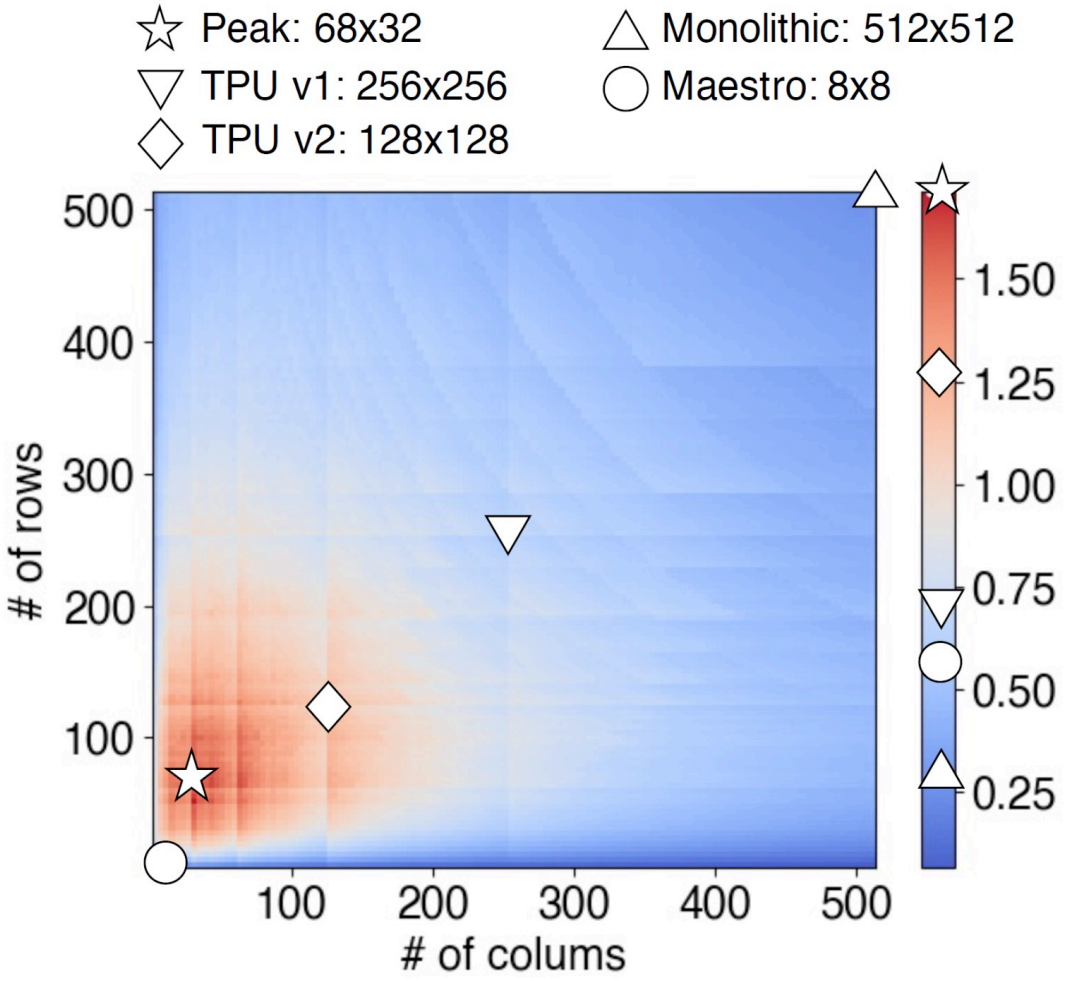
Sizing Systolic Arrays

- A new metric called **"Effective Throughput per Watt"**
  - Utilization x power efficiency

- DNN model behavior characteristics
  - Number of filter reuse
  - Number of filters

Array Size for CNN Model

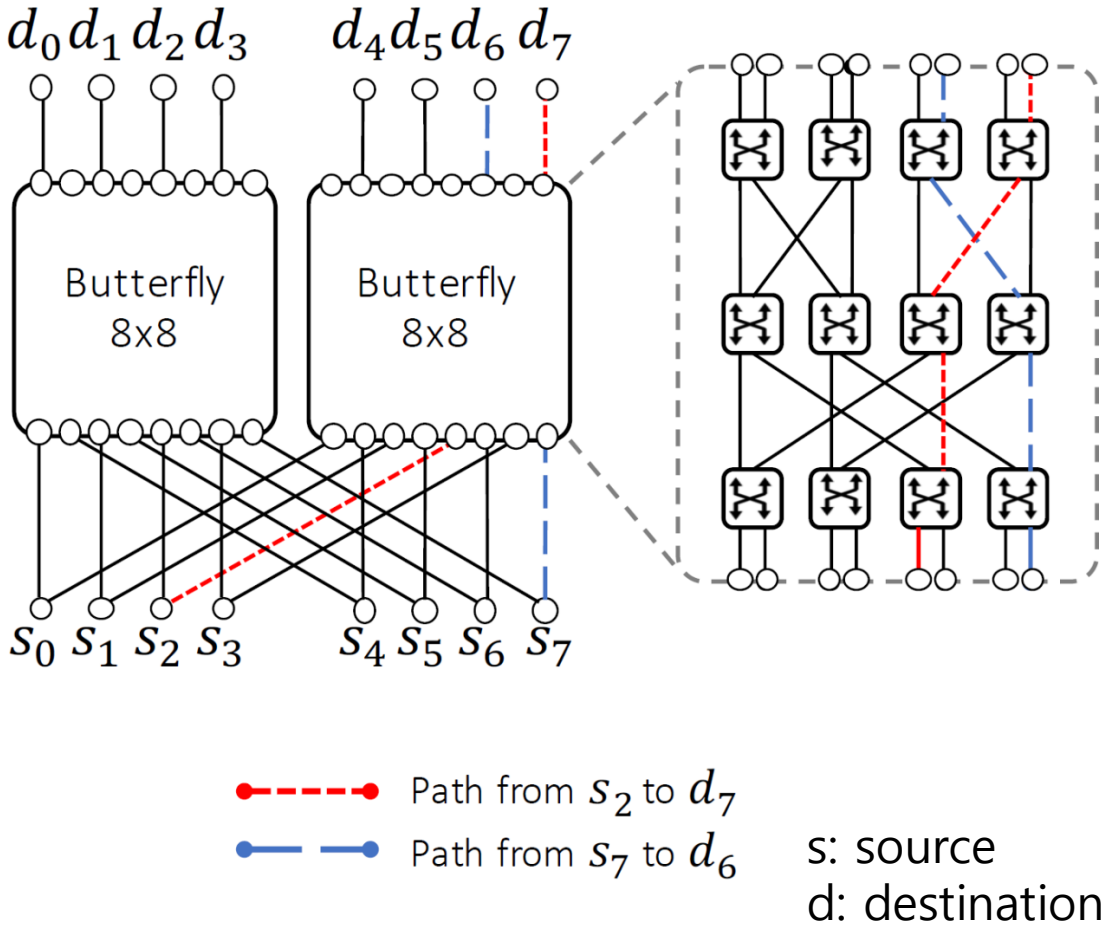
- Effective throughput varying array dimensions
- Red is better
- More filter reuses, more rows are effective. (68x32 in our analysis)



Interconnection

- Four design requirements
  - Sufficient bisection bandwidth
  - High combinatorial power with multicasting capability
  - Short latency
  - Scalability (It should be easily enhanced as we need more systolic arrays)

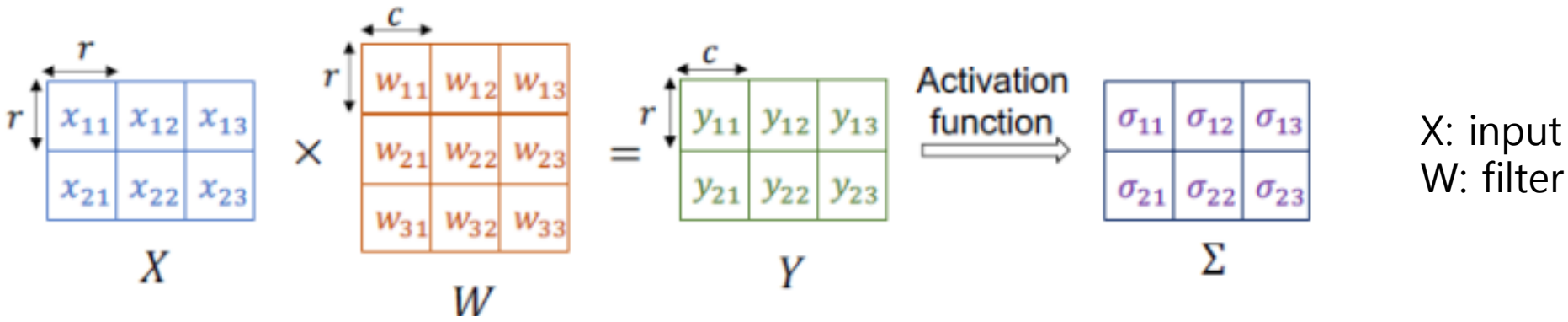
- We use the butterfly network.



Tile-Based Scheduling

- Matrix access with tiling

- Tile dimension: # of rows and columns of a single systolic array
  - r: # of rows
  - c: # columns

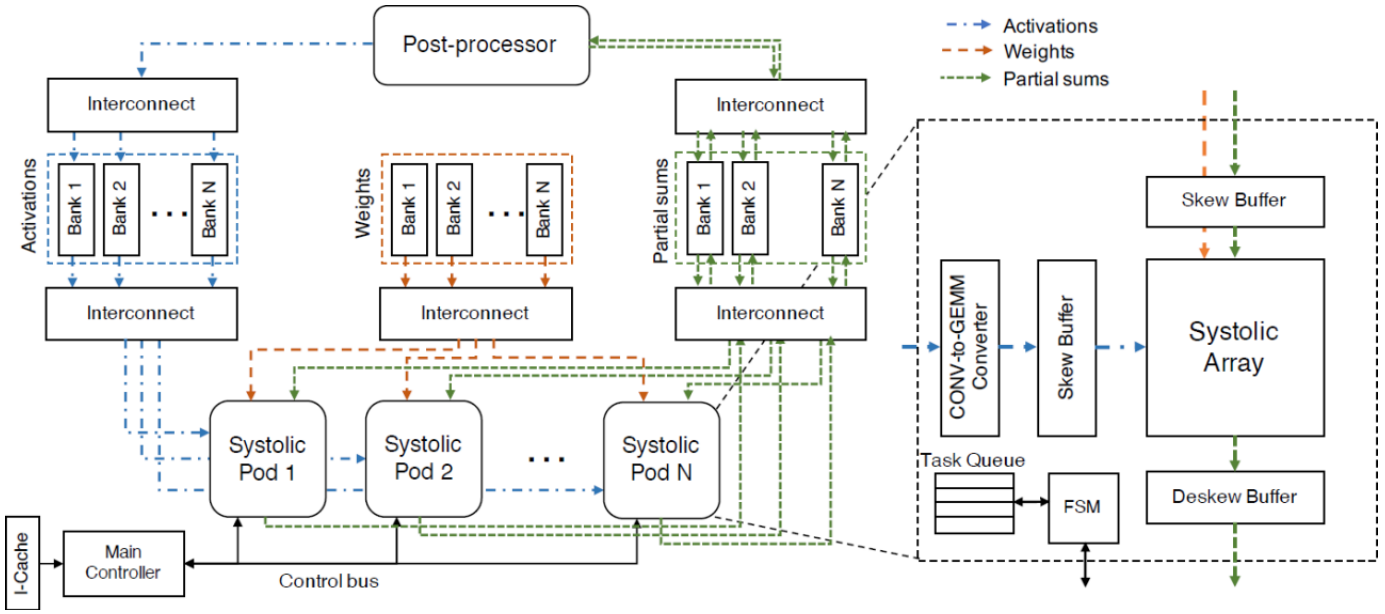


- Main controller detects an idle systolic array and sends tiles (from input and filter) to perform calculation.

Scale-Out Systolic Array (SOSA)

- A multi-pod architecture with optimally-sized systolic arrays (32x32 in our design)

- Data tiling and scheduling for multi-pod architecture



Performance Results

- Baselines
  - A: An upper bound of monolithic architecture
  - B: Google TPU v1
  - C: Google TPU v2~v4
  - D: MAESTRO
- Workloads: Bert, Densenet, inception, and Resnet (varying scales)

	SA Size	# of SA	Peak Power (Watts)	Peak Throughput @250W (TeraOps/s)	Util. (%)	Effective Throughput @250W (TeraOps/s)
Baseline A	512×512	1	117	1115	7.15	79.8
Baseline B	256×256	4	130	1007	14.0	141.4
Baseline C	128×128	16	155	844	24.8	209.2
Baseline D	8×8	1024	228	144	100*	143.7
SOSA	$\mathbf{32 \times 32}$	**128**	**153**	**428**	**62.1**	**265.5**

Concluding Remark

- Achieving high energy efficiency is an important mission for achieving sustainable scalability in datacenters.
- High utilization and high memory system performance are the fundamental challenges.
- Resolving memory system performance bottlenecks can be done with microarchitectural renovation of accelerators.

Thank you!