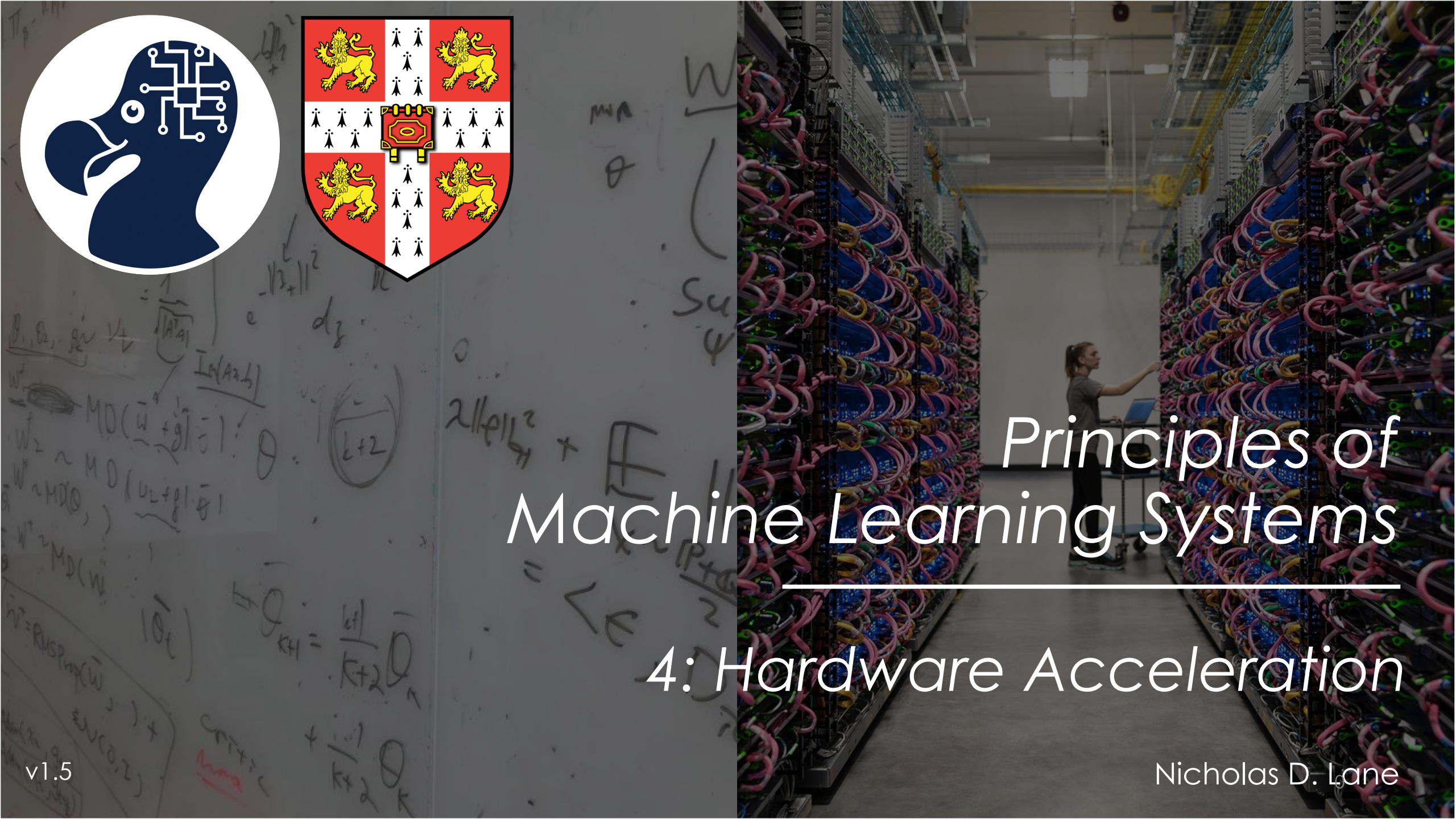


Principles of Machine Learning Systems

4: Hardware Acceleration

v1.5

Nicholas D. Lane

Roadmap for Today

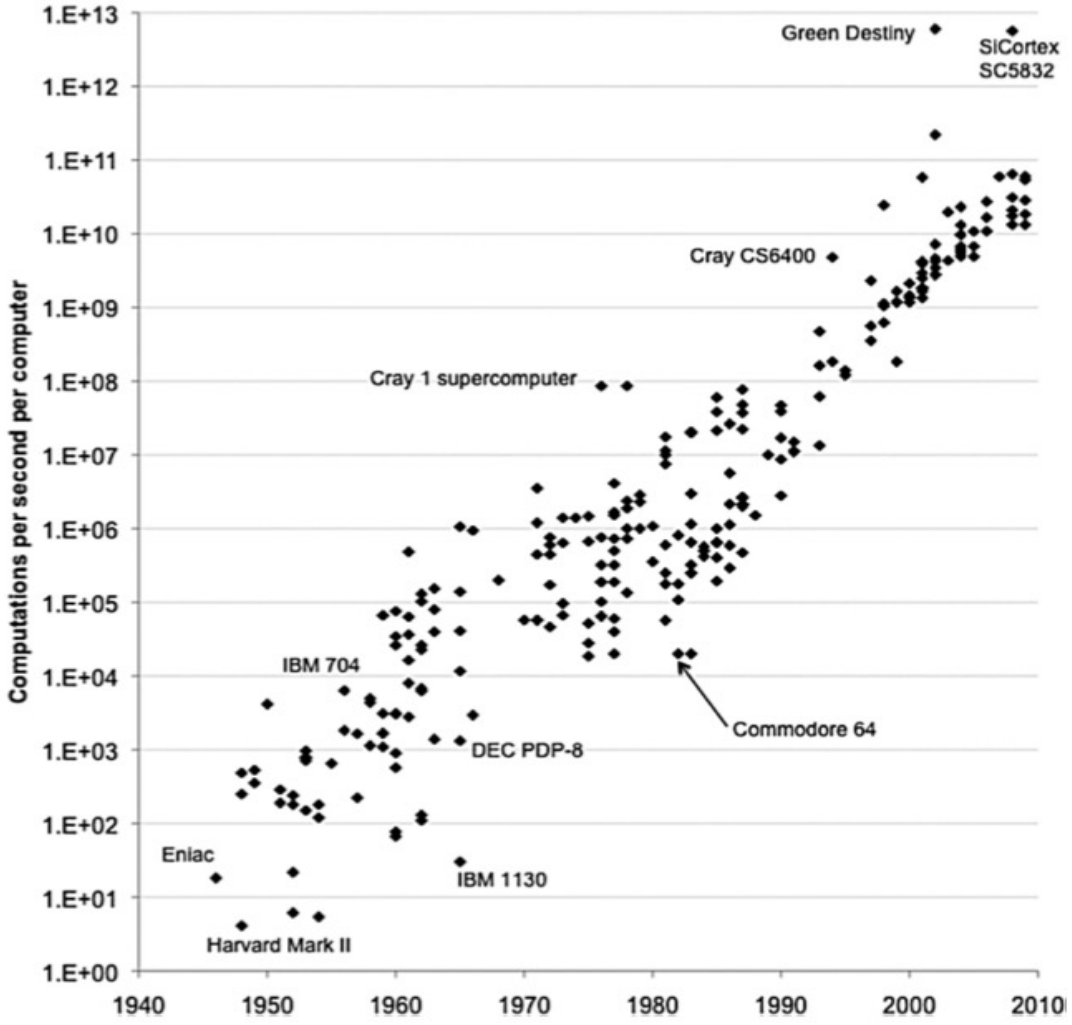
- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- Hardware Efficiency Options
- Hardware Case Studies

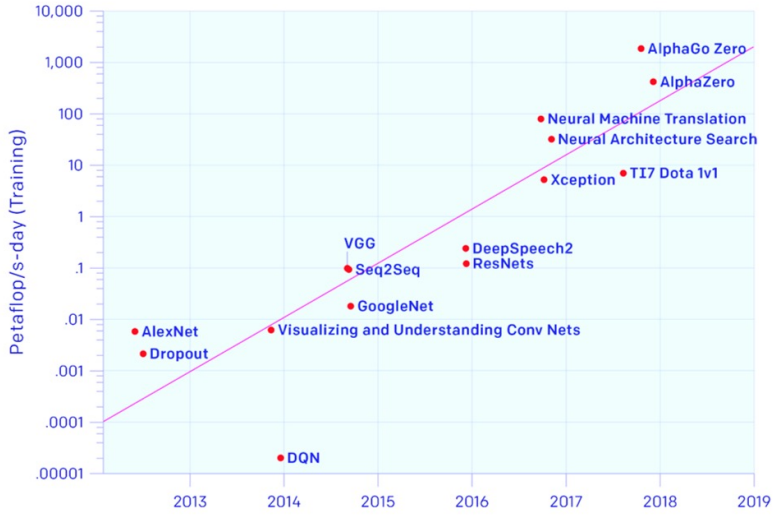
Mohamed Abdelfattah

Roadmap for Today

- **HW enabling Deep Learning**
- Performance Metrics
- Where does Energy Go?
- Hardware Efficiency Options
- Hardware Case Studies

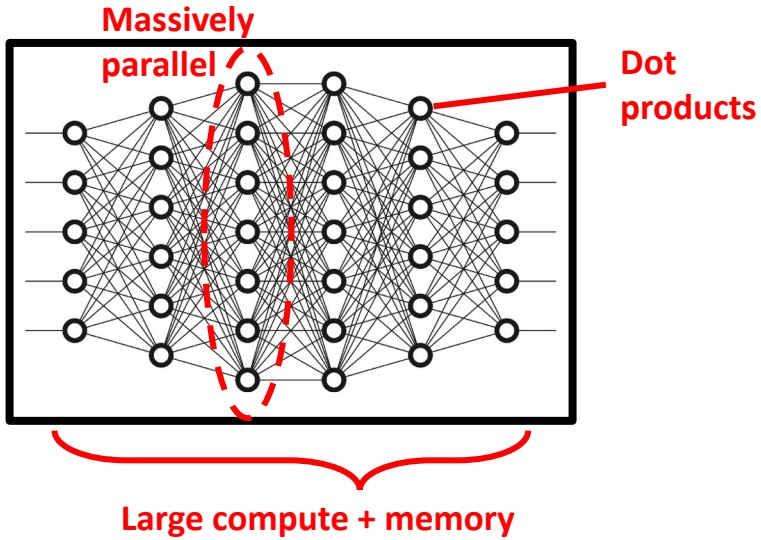
HW enables Deep Learning

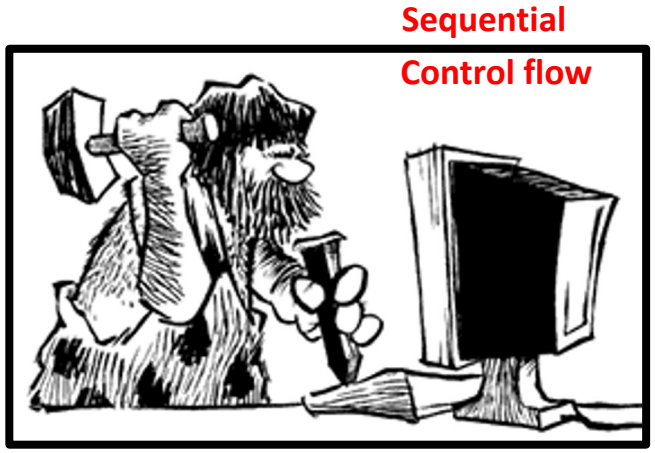




Source: OpenAI

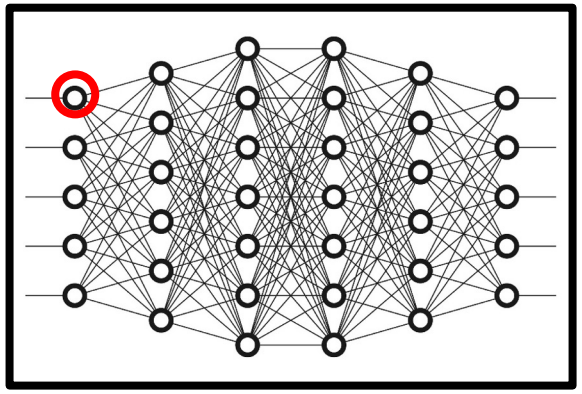
HW & Deep Learning Basics

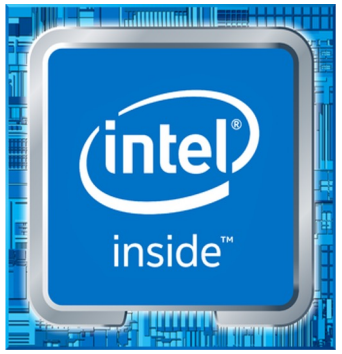




- 1986: Backpropagation published

HW & Deep Learning Basics

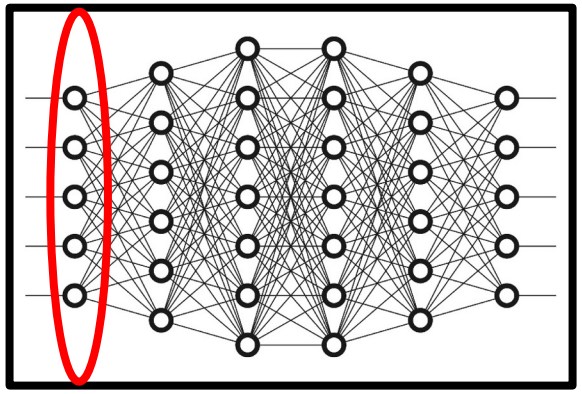




Sequential
Complex instruction set
Great for control flow

- 1986: Backpropagation published
- ~30 years of trying this on CPUs

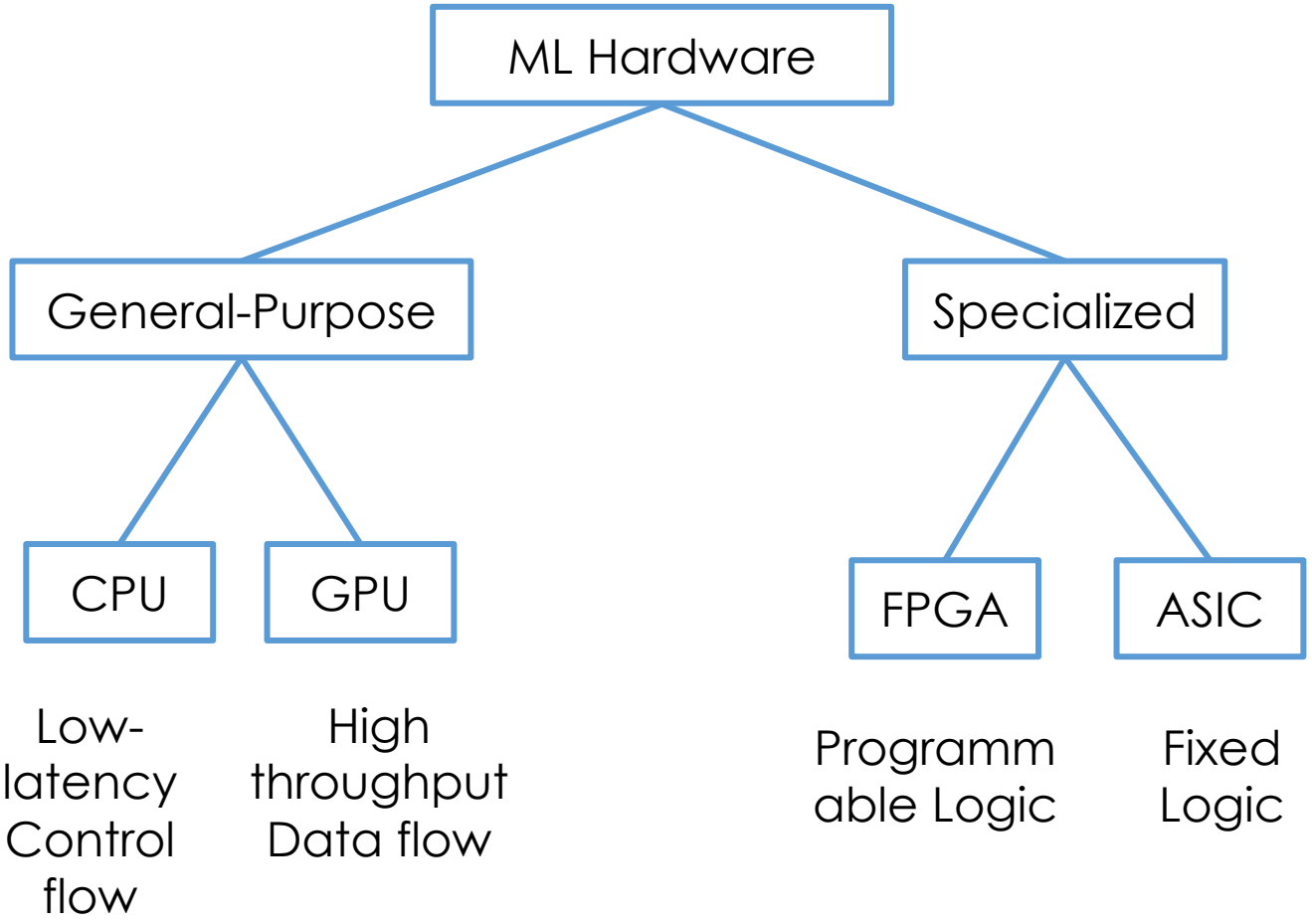
HW & Deep Learning Basics



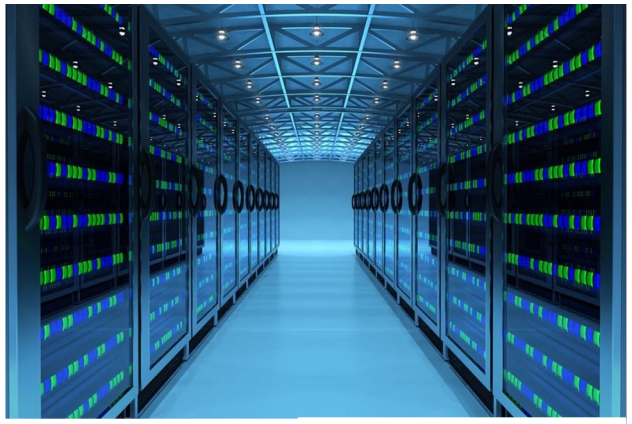
Parallel
Great for matrix math
Bad for control flow

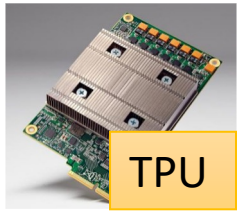
- 1986: Backpropagation published
- ~30 years of trying this on CPUs
- 2012: AlexNet paper ⇨ +10% accuracy ⇨ Deep learning explosion

Hardware types

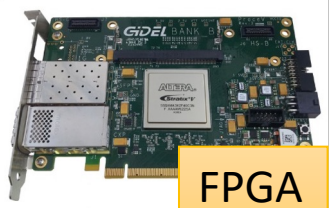


Specialized Hardware



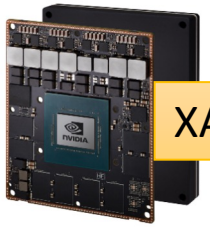


TPU

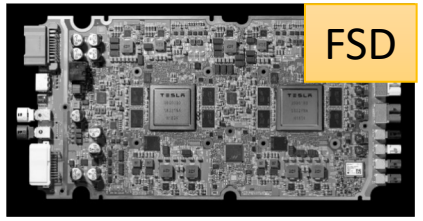


FPGA

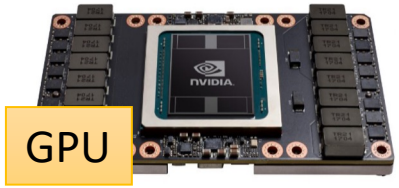
NPU



XAVIER



FSD

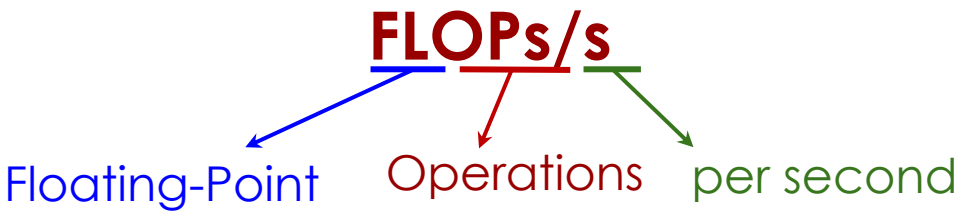


GPU

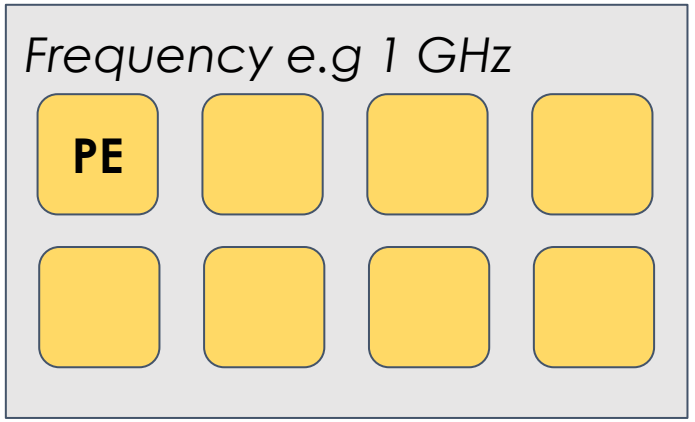
Roadmap for Today

- HW enabling Deep Learning
- **Performance Metrics**
- Where does Energy Go?
- Hardware Efficiency Options
- Hardware Case Studies

Compute Performance Metrics



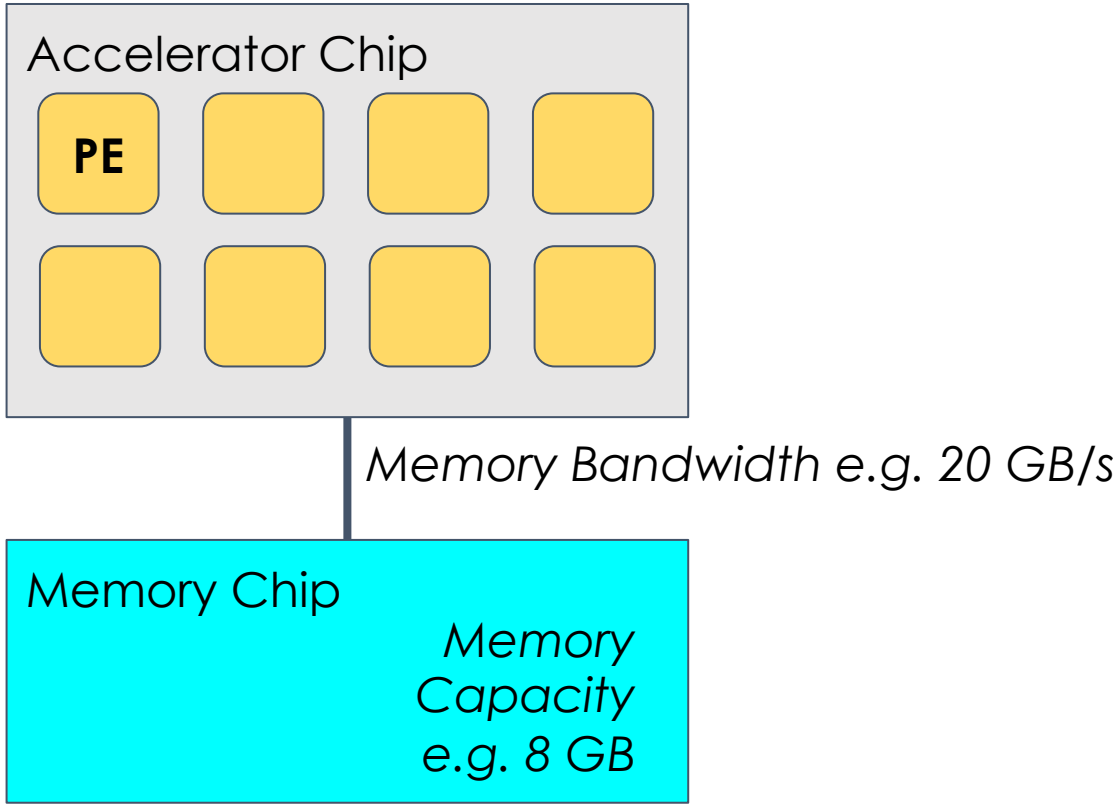
- MACs/s: Multiply-accumulate Ops/s
  - Half FLOPs/s
- OPs/s: for non floating-point operations
- Chips are often labeled with "peak FLOPs/s"
  - Not achievable under normal workloads
  - Very rough indication of performance



$$\frac{\text{operations}}{\text{second}} = \underbrace{\left(\frac{1}{\frac{\text{cycles}}{\text{operation}}} \times \frac{\text{cycles}}{\text{second}}\right)}_{\text{for a single PE}} \times \text{number of PEs}$$

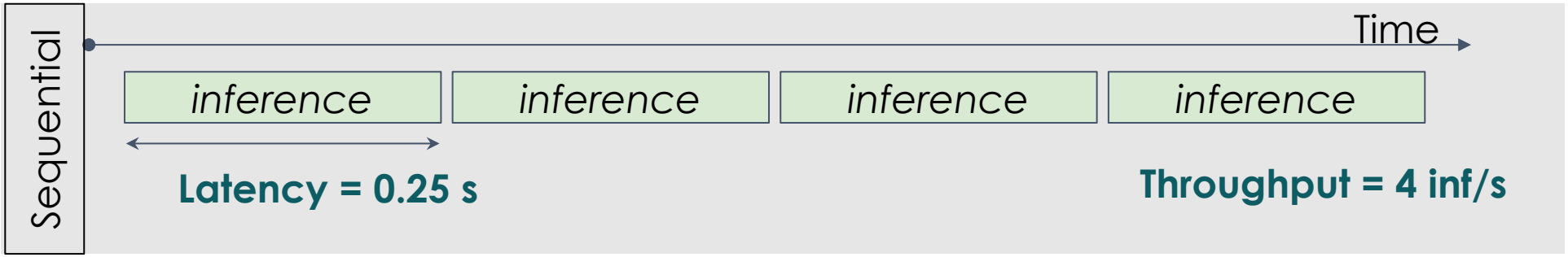
Memory Performance Metrics

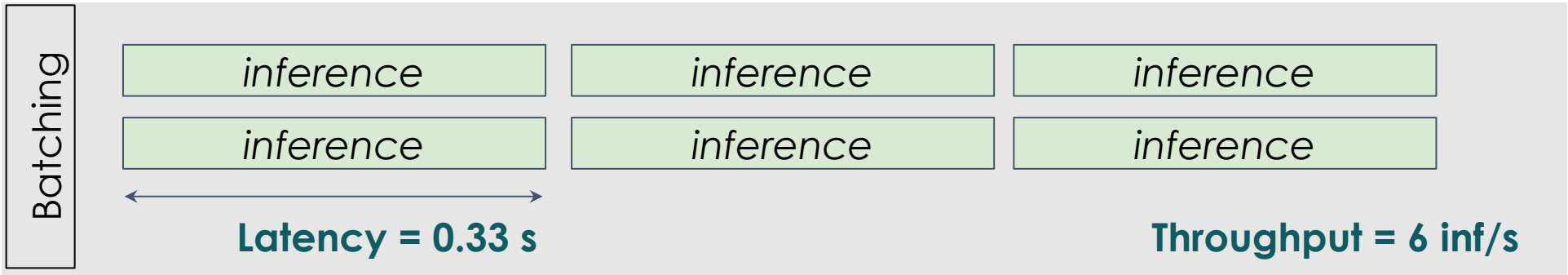
- Memory capacity [GB]
- Memory bandwidth [GB/s]
  - Transfer speed from memory chip to compute chip
- More complicated because there is a *memory hierarchy*
  - Showing "external"/"main" memory
  - Can have caches, local memory, registers with much higher bandwidth



DNN Performance

- **Latency**: Number of seconds per inference (unit = seconds)
- **Throughput**: Number of inferences per second (unit = inference/second)



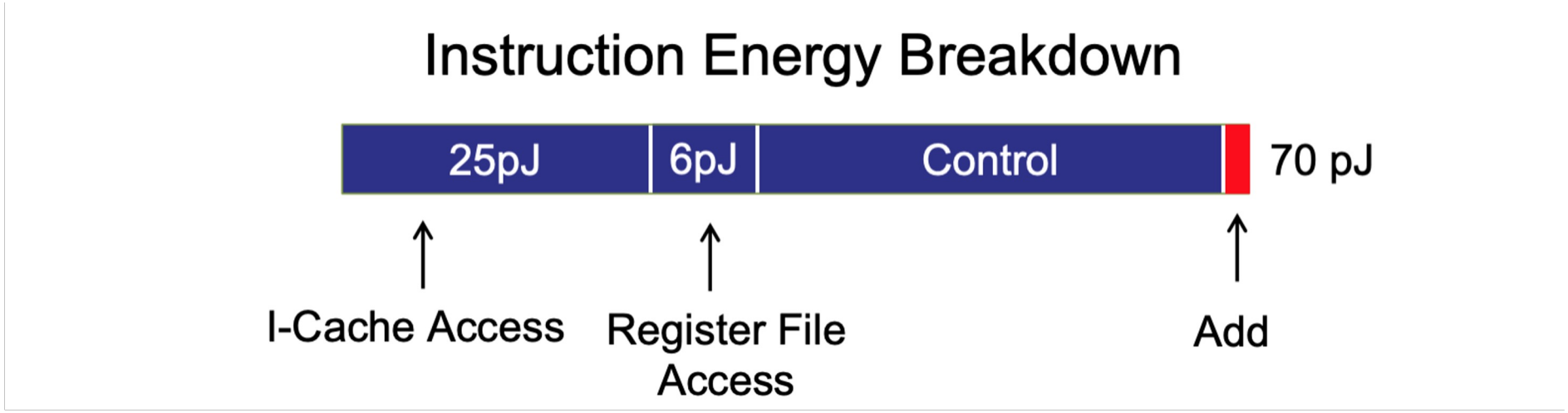


Roadmap for Today

- HW enabling Deep Learning
- Performance Metrics
- **Where does Energy Go?**
- Hardware Efficiency Options
- Hardware Case Studies

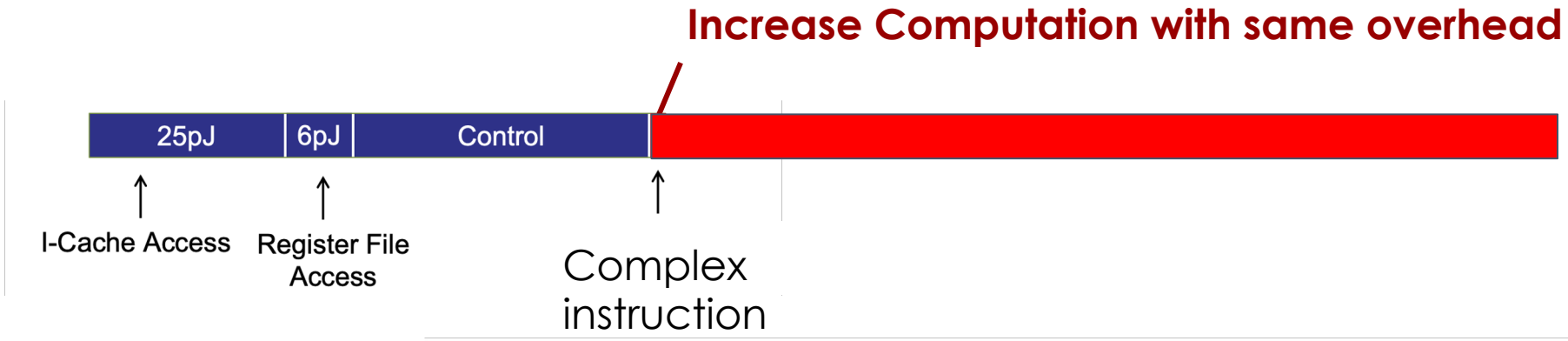
Where does the energy go?

- Energy breakdown of an add instruction in a 45nm CPU
- How can we optimize this?



Source: Horowitz

Amortize Overhead



Half-precision Fused Multiply-Add

4-way dot-product

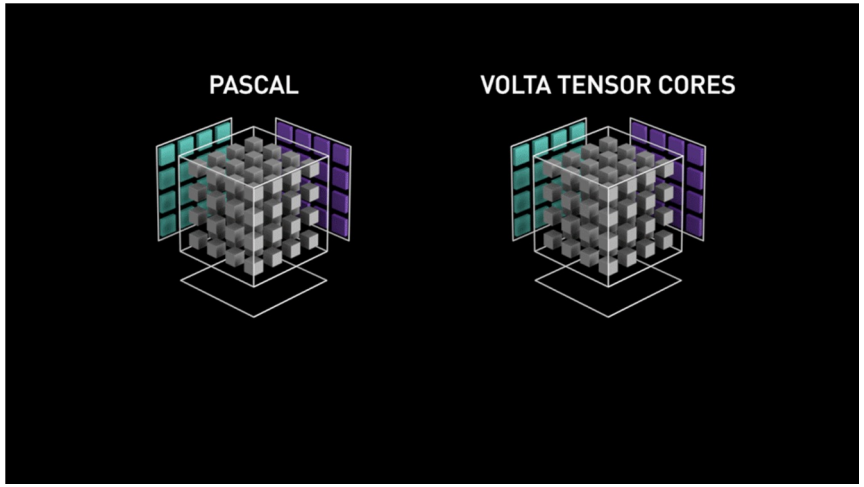
16x16 matrix multiplication

Operation	Energy**	Overhead*
HFMA	1.5pJ	2000%
HDP4A	6.0pJ	500%
HMMA	110pJ	27%

Source: Dally

"Special" Instruction Examples

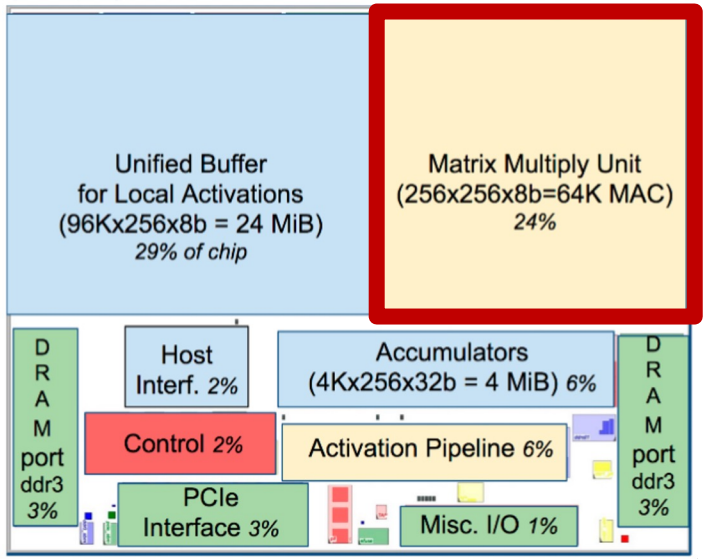
GPU



16x16 = 256*
MAC/cycle

*~ 500 tensor cores per GPU

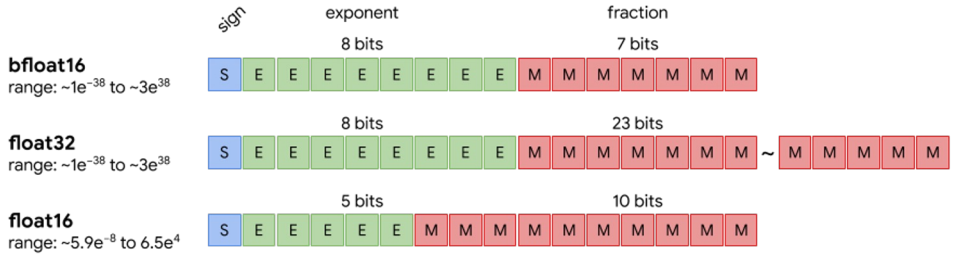
ASIC (TPUv1)



256x256 = 64
kMAC/cycle

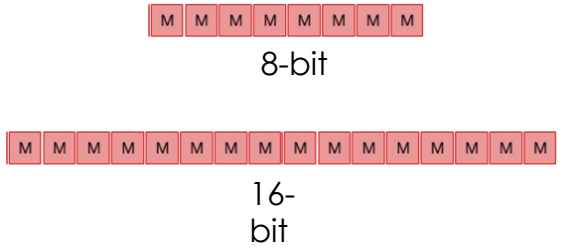
Numerical Format & Precision

Floating Point



- IEEE standard includes FP32 and FP16
- Many exotic FP numbers in DNN
  - E.g. bfloat, minifloat

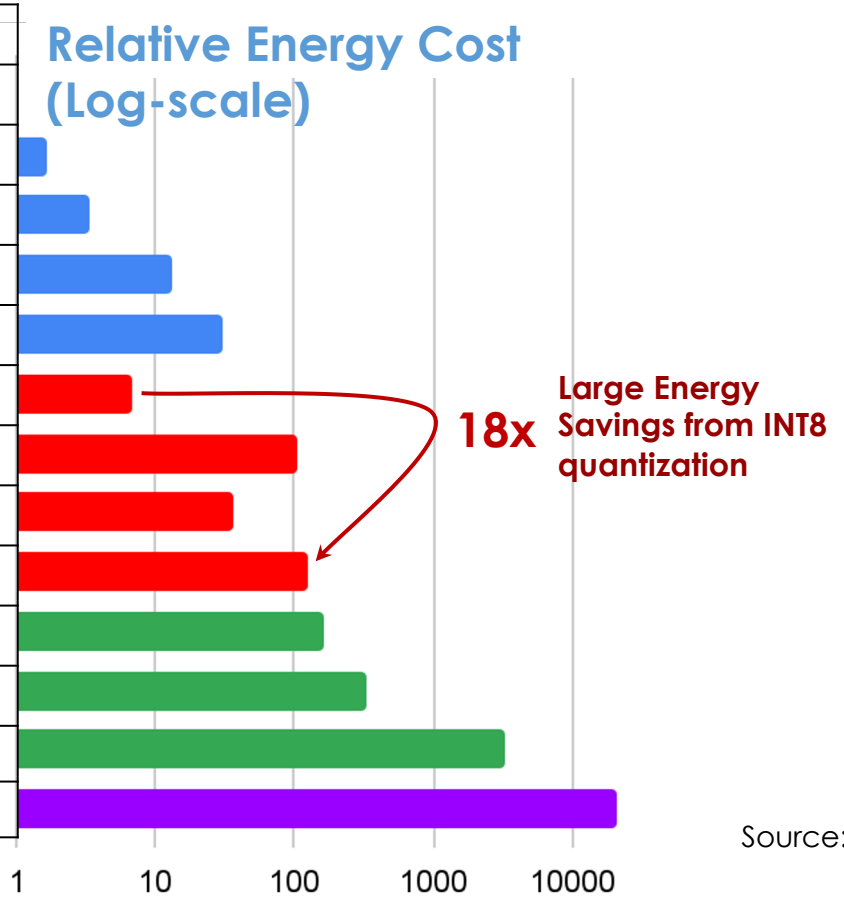
Integer



- Whole numbers only
- (typically) much cheaper circuit area and power

Cost of Arithmetic Operations

Operation	Precision	Energy (pJ)
4x Addition	INT8	0.03
	INT16	0.05
	INT32	0.1
	FP16	0.4
	FP32	0.9
16x Multiplication	INT8	0.2
	INT32	3.1
	FP16	1.1
	FP32	3.7
32-bit SRAM Read (8 kB)		5
32-bit SRAM Read (32 kB)		10
32-bit SRAM Read (1 MB)		100
32-bit DRAM Read		640



Source: Horowitz

Roadmap for Today

- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- **Hardware Efficiency Options**
- Hardware Case Studies

Roadmap for Today

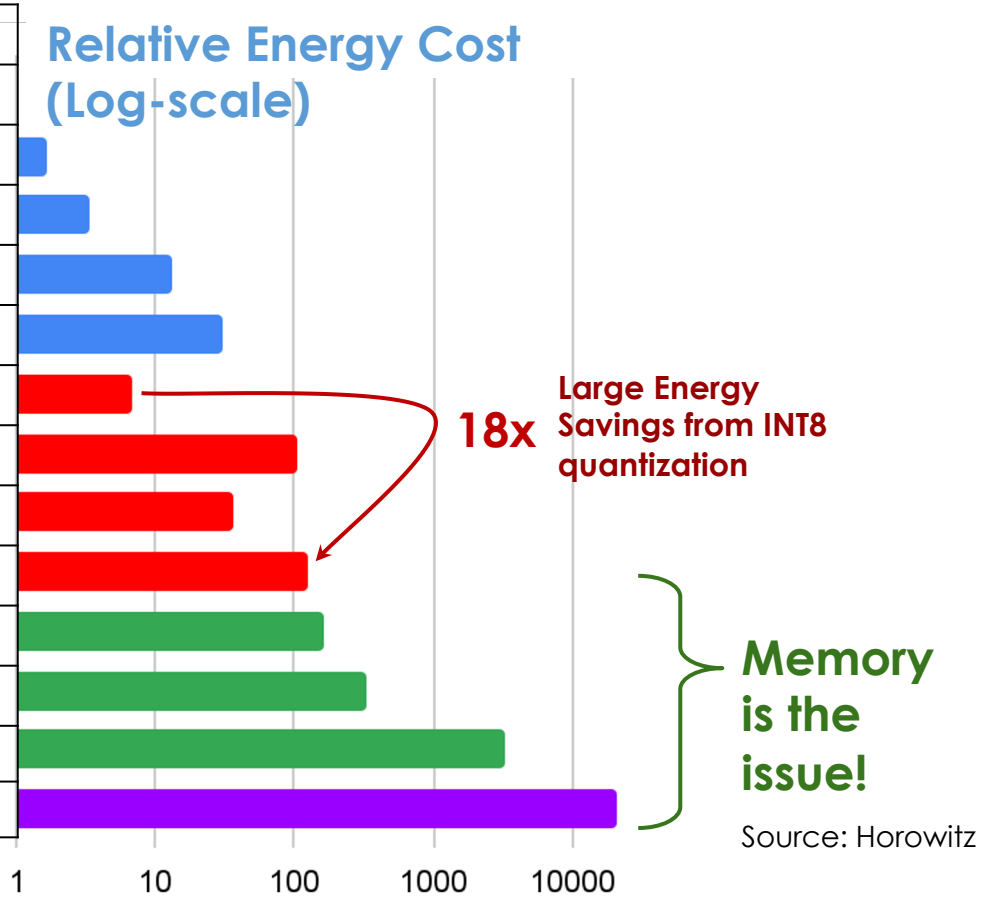
- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- **Hardware Efficiency Options**
  - **Arithmetic**
  - **Memory**
  - Ineffectual Operation
- Hardware Case Studies

Roadmap for Today

- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- **Hardware Efficiency Options**
  - Arithmetic
  - **Memory**
  - Ineffectual Operation
- Hardware Case Studies

Memory is the bottleneck

Operation	Precision	Energy (pJ)
4x Addition	INT8	0.03
	INT16	0.05
	INT32	0.1
	FP16	0.4
	FP32	0.9
16x Multiplication	INT8	0.2
	INT32	3.1
	FP16	1.1
	FP32	3.7
32-bit SRAM Read (8 kB)		5
32-bit SRAM Read (32 kB)		10
32-bit SRAM Read (1 MB)		100
32-bit DRAM Read		640

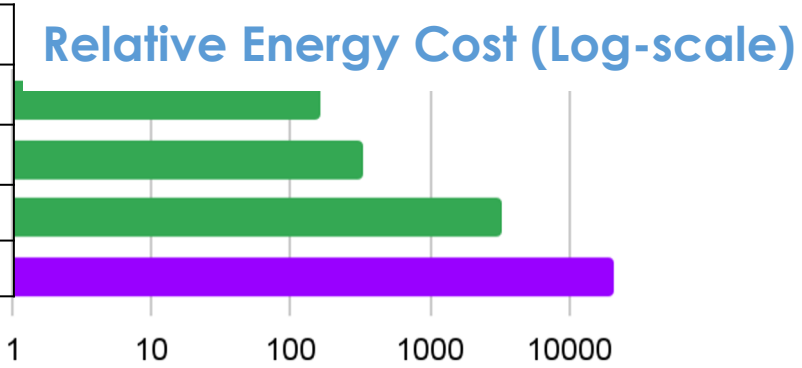


Source: Horowitz

Mem. Hierarchy Optimizations

1. Get data close to the computation. (LOCALITY)
2. Once data is close - perform all computations with this data. (REUSE)

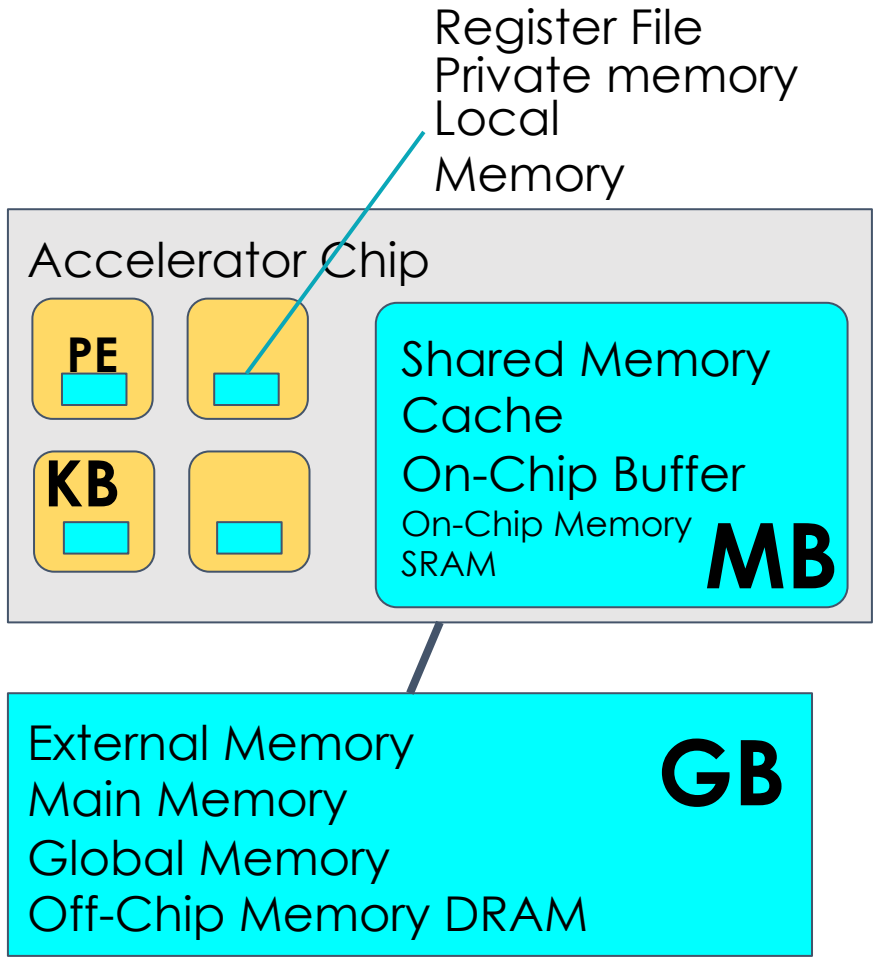
Operation	Energy (pJ)
32-bit SRAM Read (8 kB)	5
32-bit SRAM Read (32 kB)	10
32-bit SRAM Read (1 MB)	100
32-bit DRAM Read	640



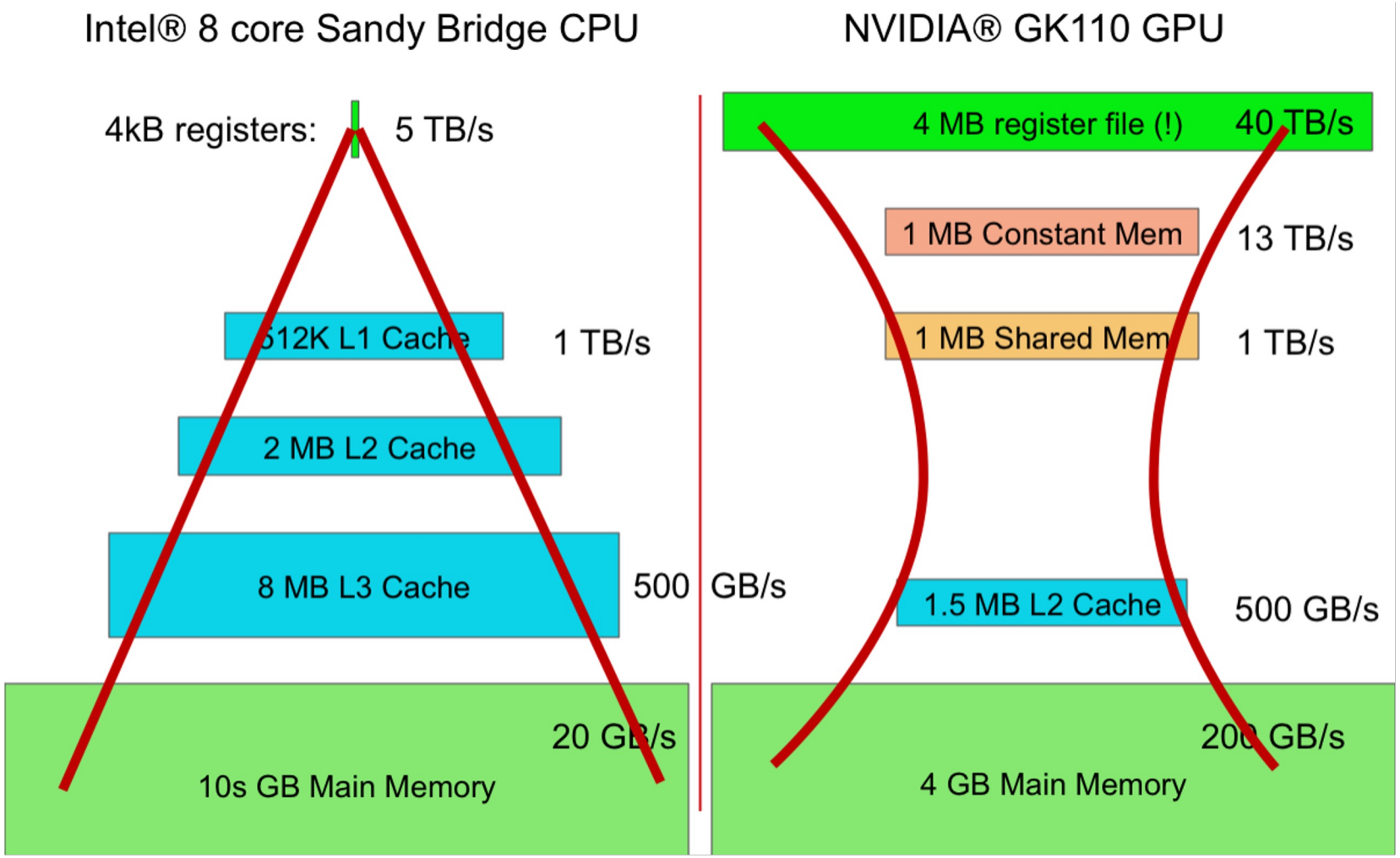
Memory Hierarchy

Why do we have a memory hierarchy?

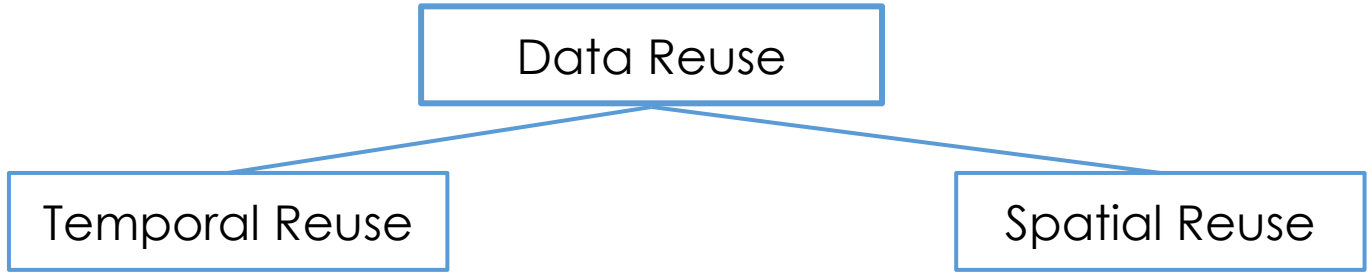
- The closer you get to compute, the more $$ and scarce the memory resource becomes
- In *most* cases, the DNN parameters live off chip and are fetched layer-by-layer or tile-by-tile
- Data locality: how to get data close to the PEs (to keep them fully utilized)



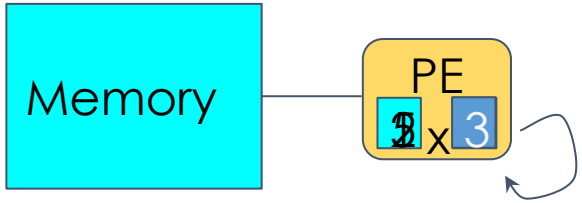
Memory Hierarchy Examples



Data Reuse

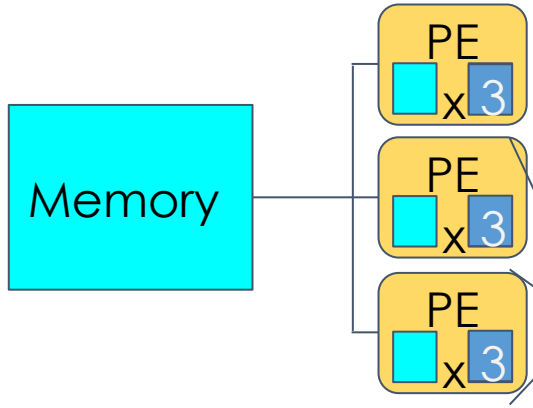


Read once from memory, use same data multiple times by same PE



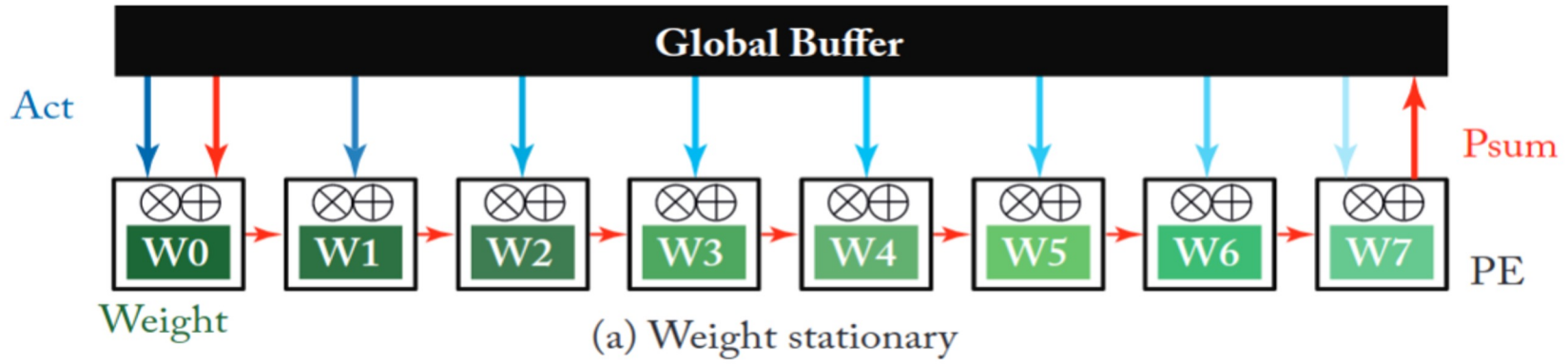
Reuse entry multiple times from a single memory read

Read once from memory, use same data multiple times by multiple PEs



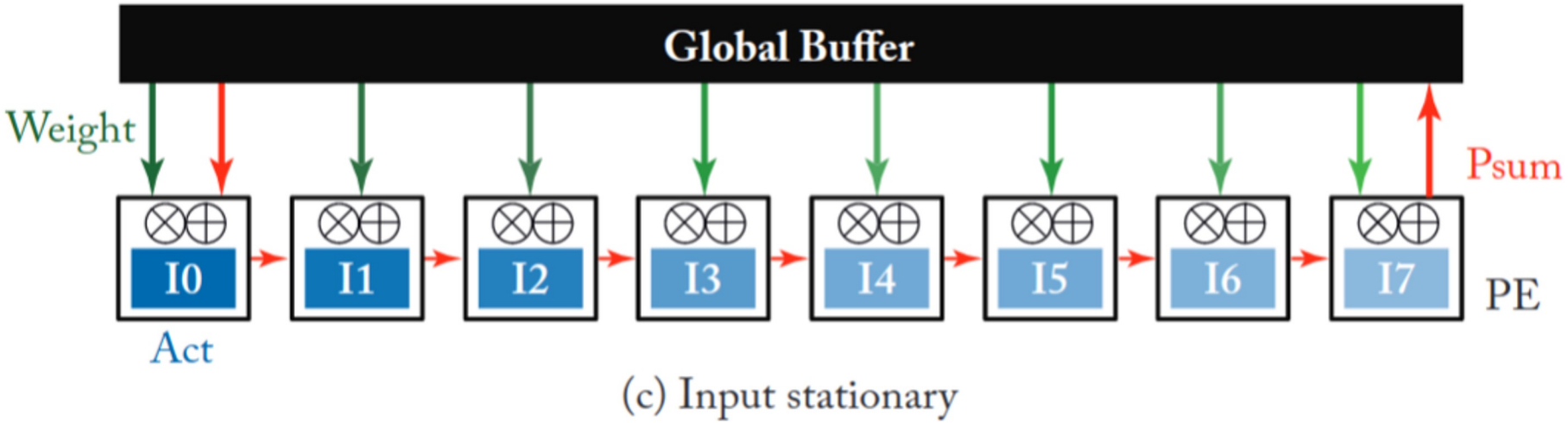
Reuse entry multiple times from a single memory read

Stationary Weights?



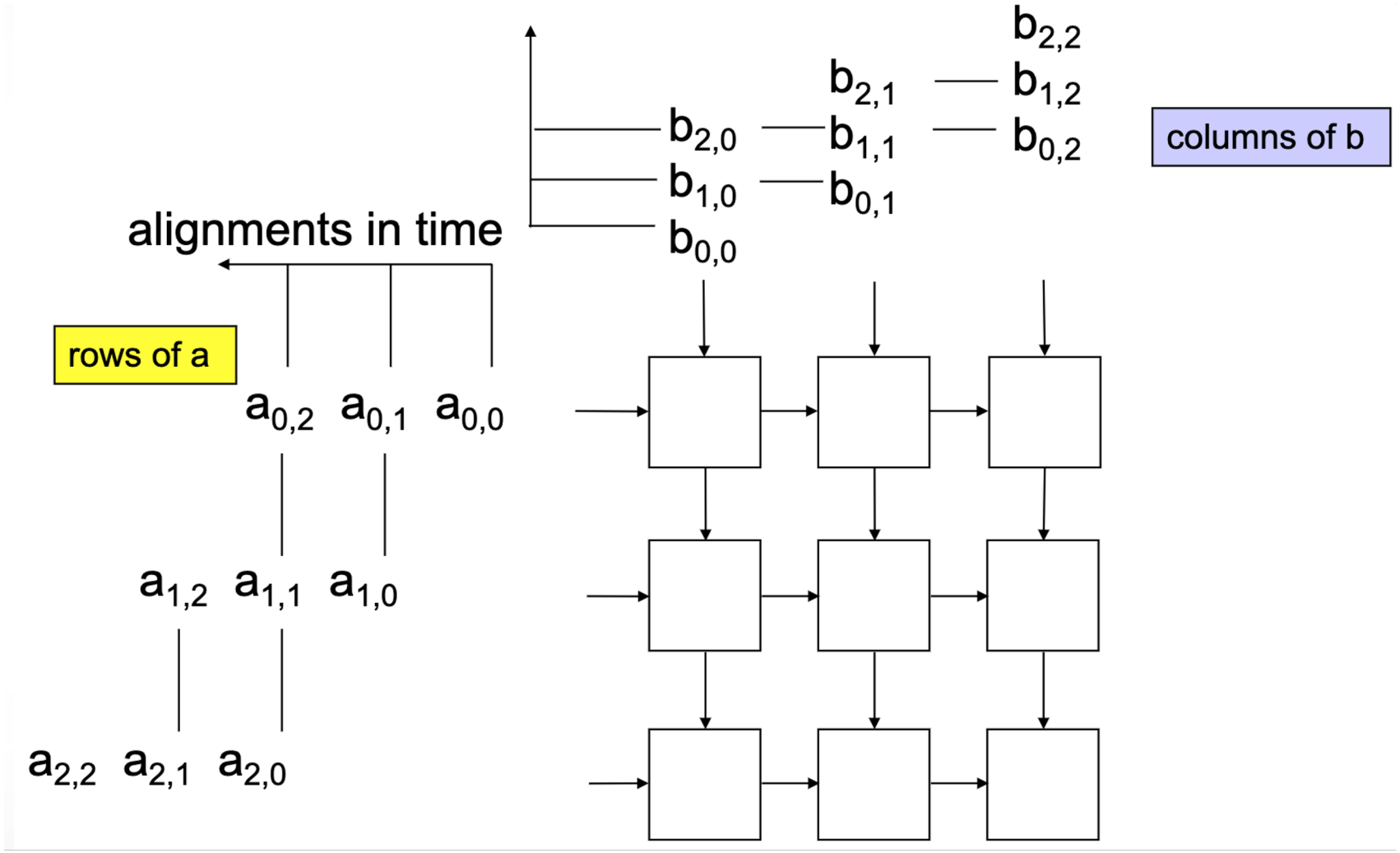
a0w0 + a1w1 + a2w2 + a3w3 + a4w4 + a5w5 + a6w6 + a7w7

Stationary Inputs?

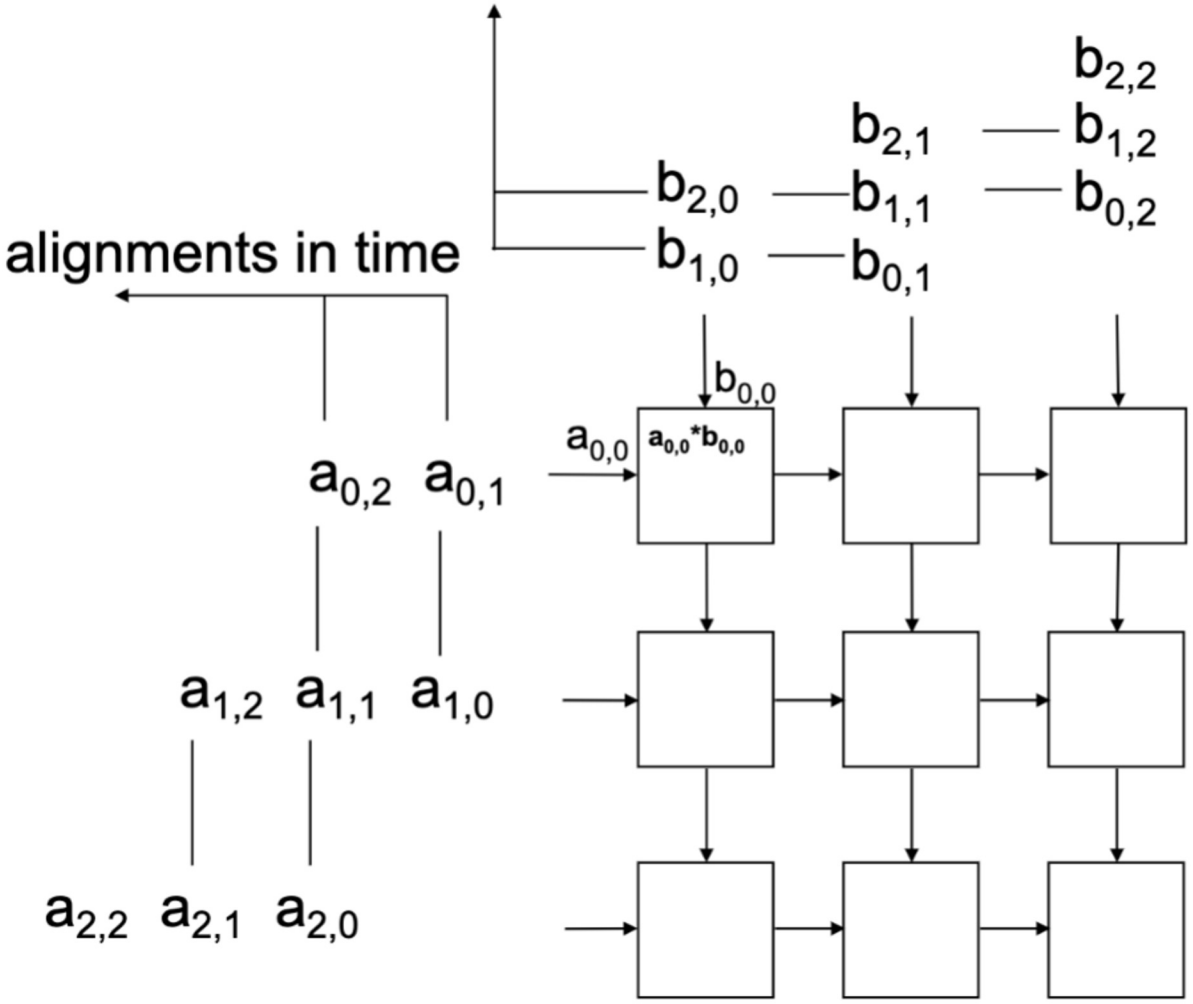


a0w0 + a1w1 + a2w2 + a3w3 + a4w4 + a5w5 + a6w6 + a7w7

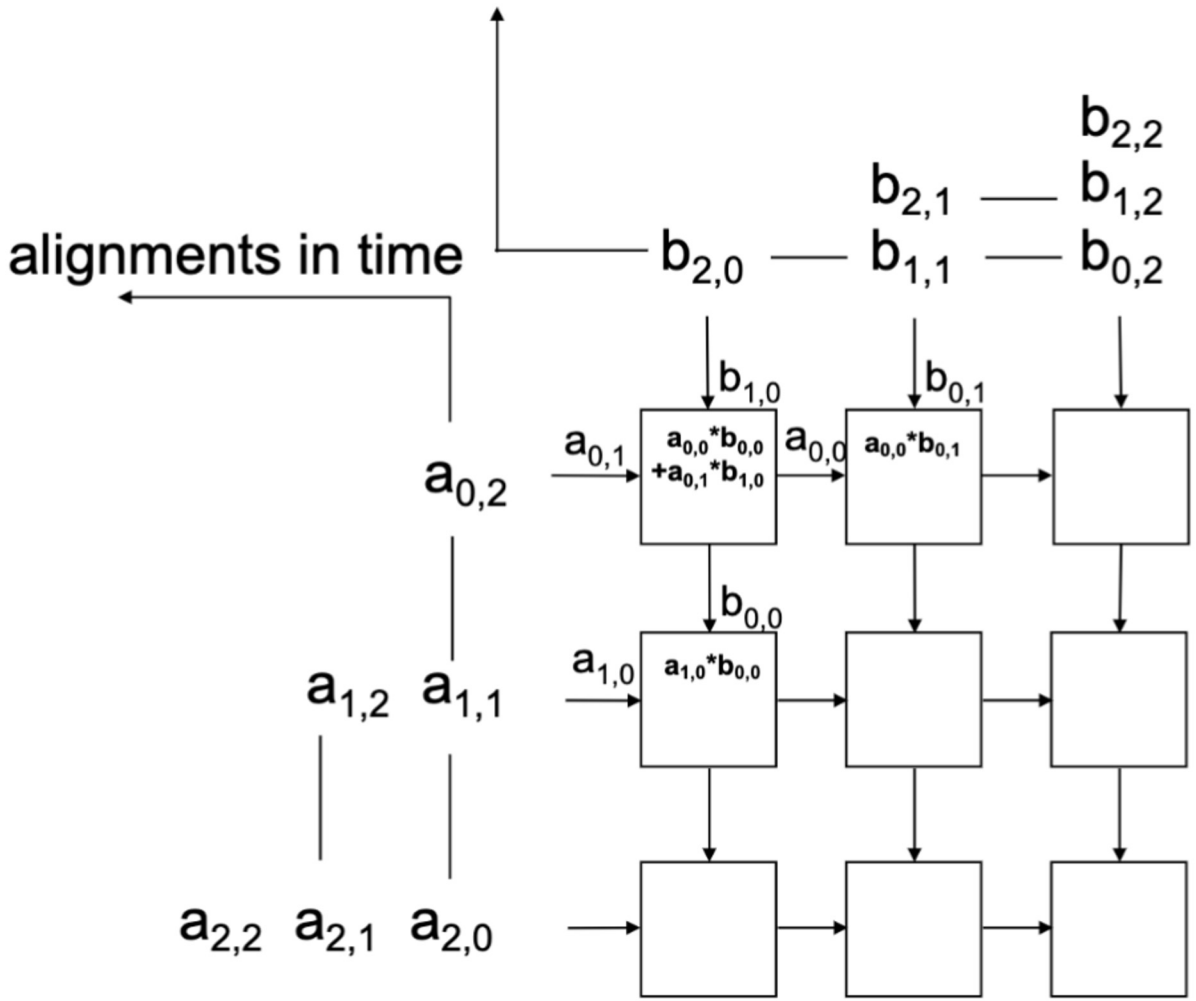
Systolic Array: Matrix Mults.



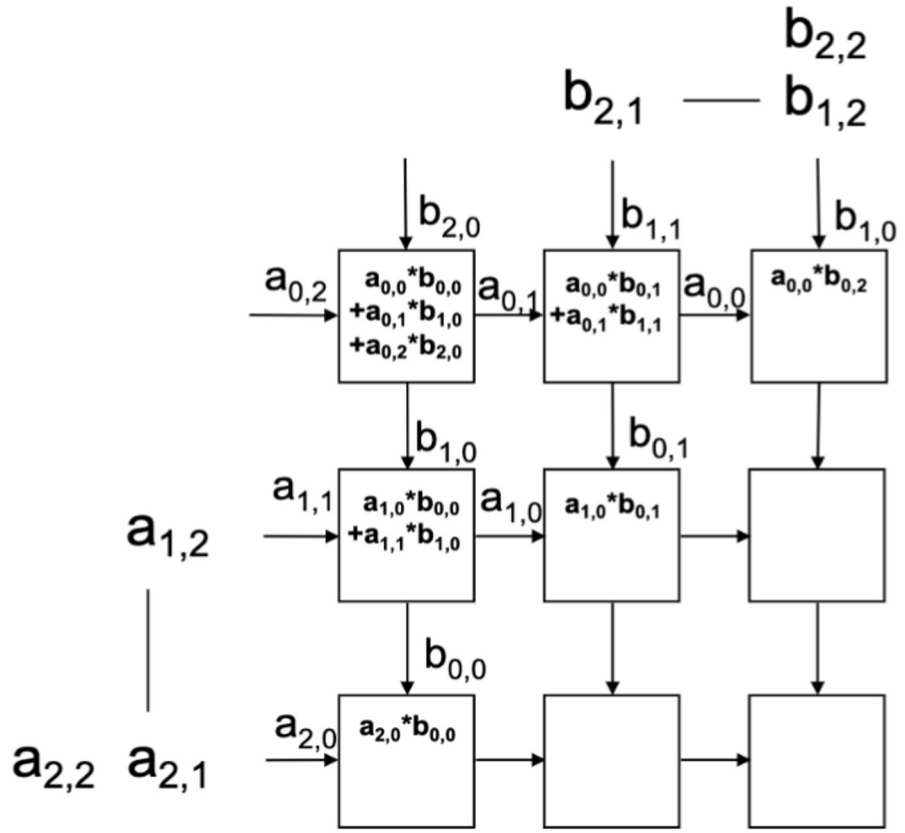
Systolic Array: Matrix Mults.



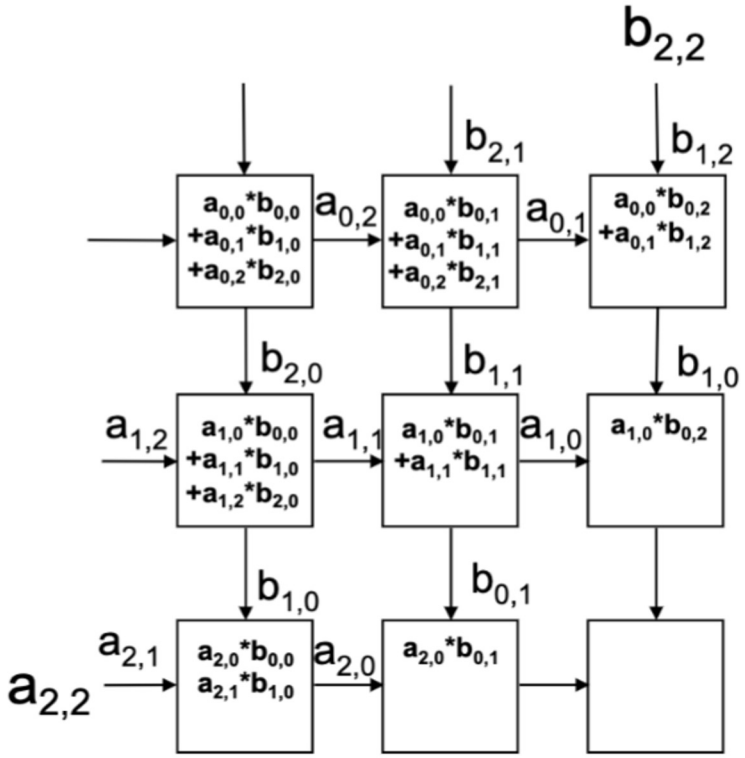
Systolic Array: Matrix Mults.



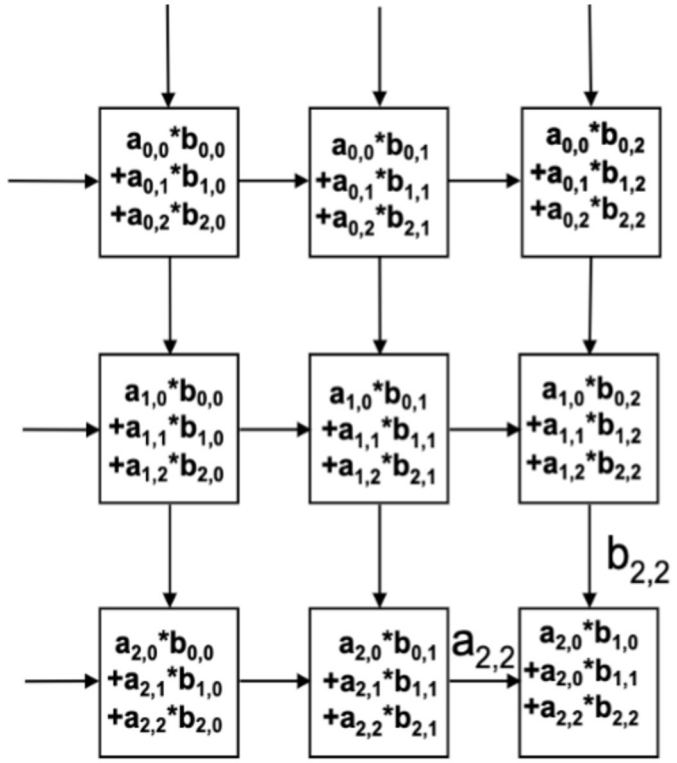
Systolic Array: Matrix Mults.



Systolic Array: Matrix Mults.



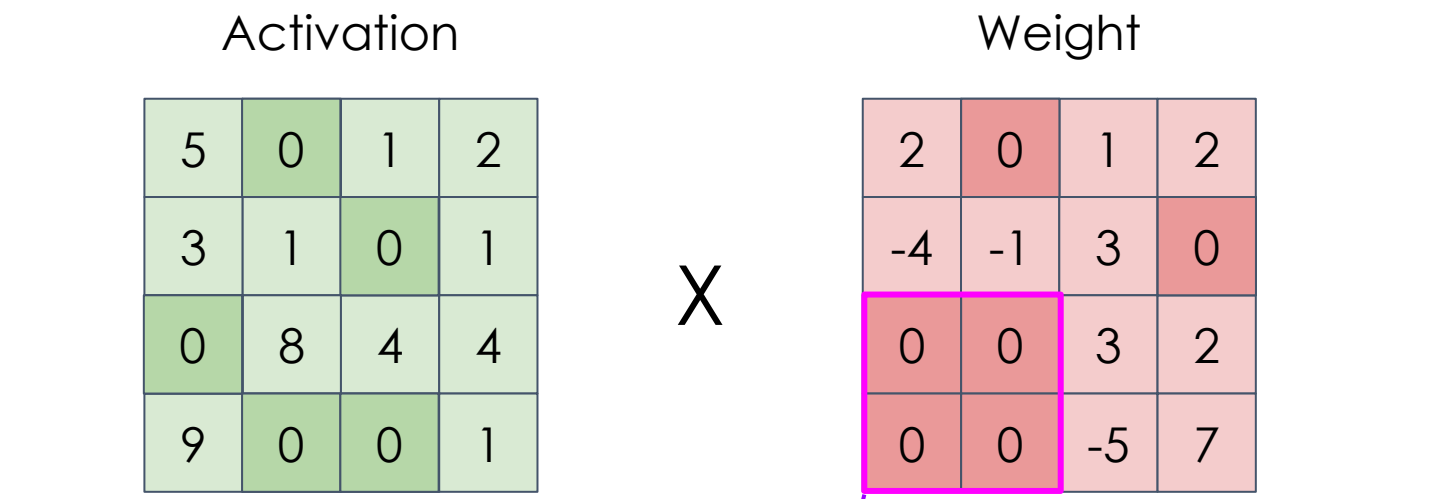
Systolic Array: Matrix Mults.



Roadmap for Today

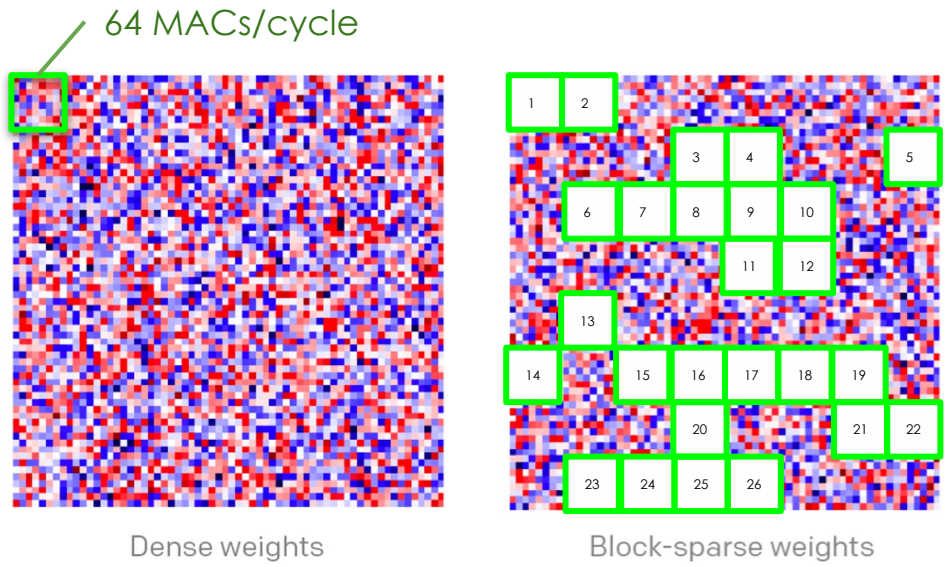
- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- **Hardware Efficiency Options**
  - Arithmetic
  - Memory
  - **Ineffectual Operation**
- Hardware Case Studies

Kinds of Sparsity



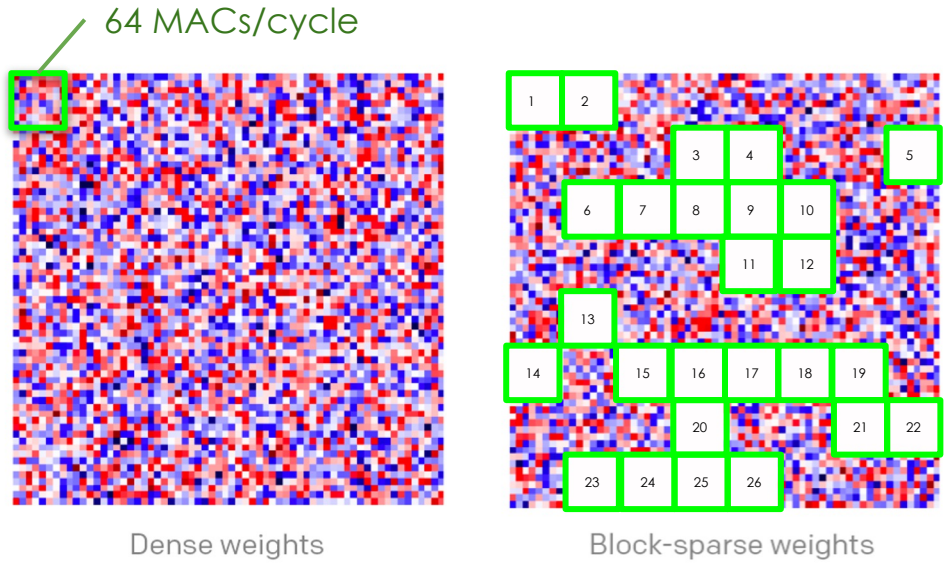
Coarse-grained "Block" Sparsity

- All DNN accelerators are parallel
  - Multiple MACs/cycle
- The smallest unit of computation that can be skipped is a large block (*recall amortized overhead*)
- Example:
  - Systolic array with 64 MACs/cycle
    - 8x8 pattern
  - 64x64 matrix = 4096 MACs
  - Total # cycles = 64 cycles
  - Block sparsity pattern needs to skip blocks of 8x8
  - Speedup = 64/(64-26) = 1.7X faster



Coarse-grained "Block" Sparsity

- All DNN accelerators are parallel
  - Multiple MACs/cycle
- The smallest unit of computation that can be skipped is a large block (*recall amortized overhead*)
- Example:
  - Systolic array with 64 MACs/cycle
    - 8x8 pattern
  - 64x64 matrix = 4096 MACs
  - Total # cycles = 64 cycles
  - Block sparsity pattern needs to skip blocks of 8x8
  - Speedup = 64/(64-26) = 1.7X faster



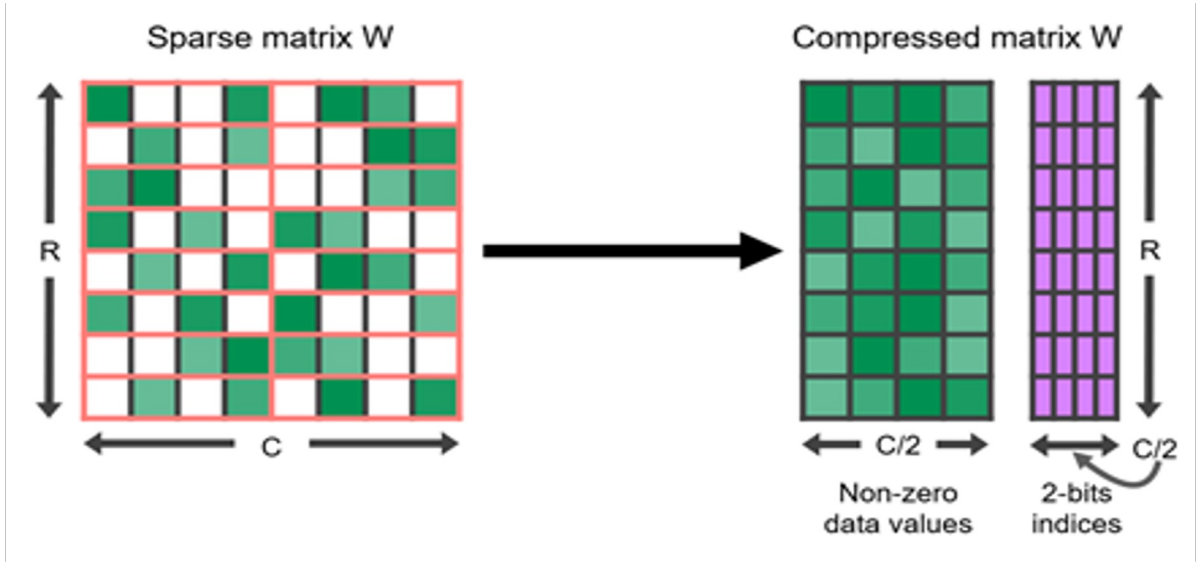
Simplest way to leverage sparsity with low overhead

⇨ Single bit per 8x8 block (1/64 = 1.6% overhead)

⇨ Simple control logic because entire block is skipped

Fine-grained Sparsity (Ampere GPUs)

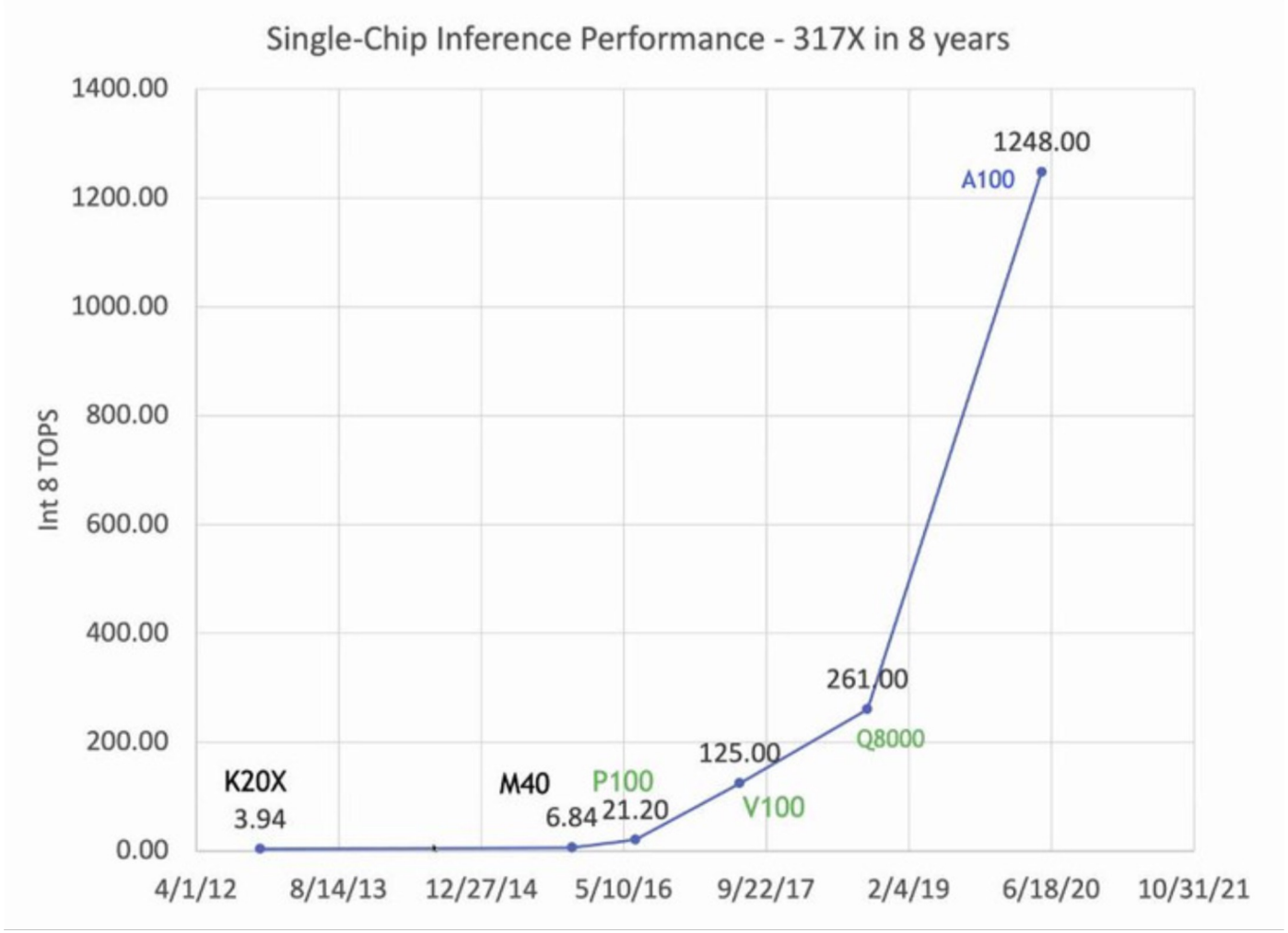
- Very recently, fine-grained sparsity was added to Tensor Cores on Nvidia GPUs
- 2 elements for every block of 4 elements can be zero
- Requires retraining to regain accuracy
- Overhead?
  - 2 bits per 8-bit element
  - 12.5% memory overhead
  - Control logic? Performance improvement? Power savings?



Roadmap for Today

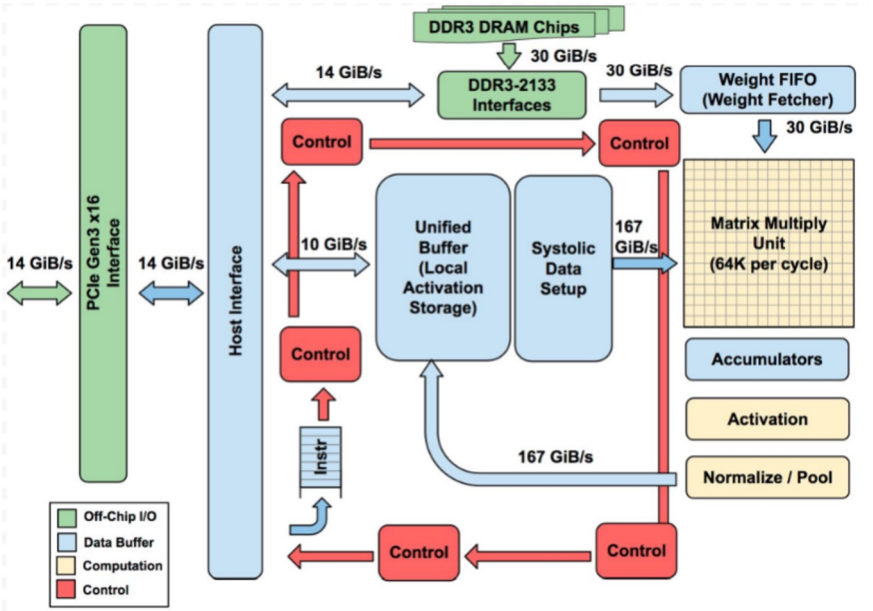
- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- Hardware Efficiency Options
- **Hardware Case Studies**

Nvidia GPU Progression

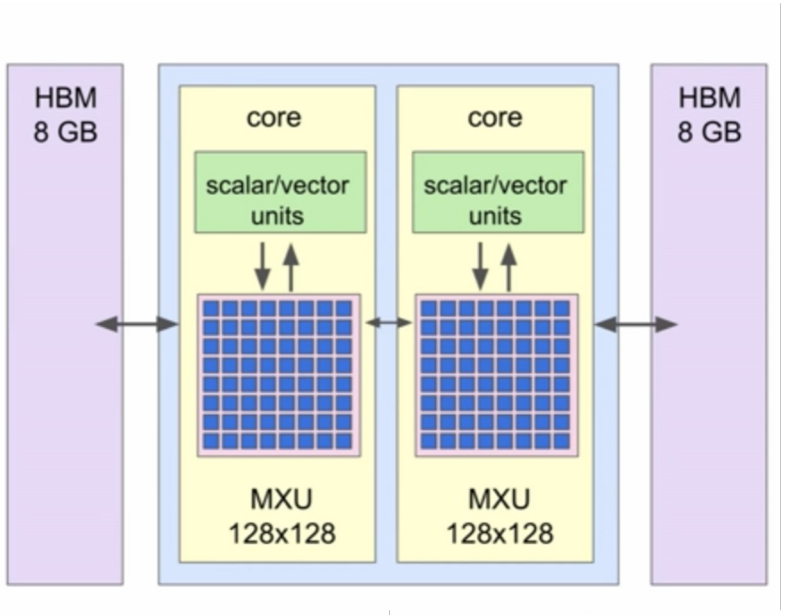


Source: Dally

Google TPU



v2



- 16 GB of HBM
- 600 GB/s mem BW
- Scalar/vector units: 32b float

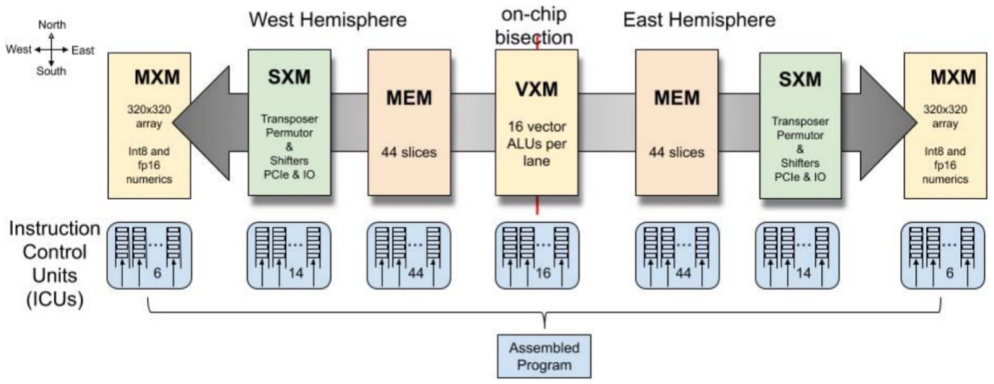
- MXU: 32b float accumulation but reduced precision for multipliers
- 45 TFLOPS

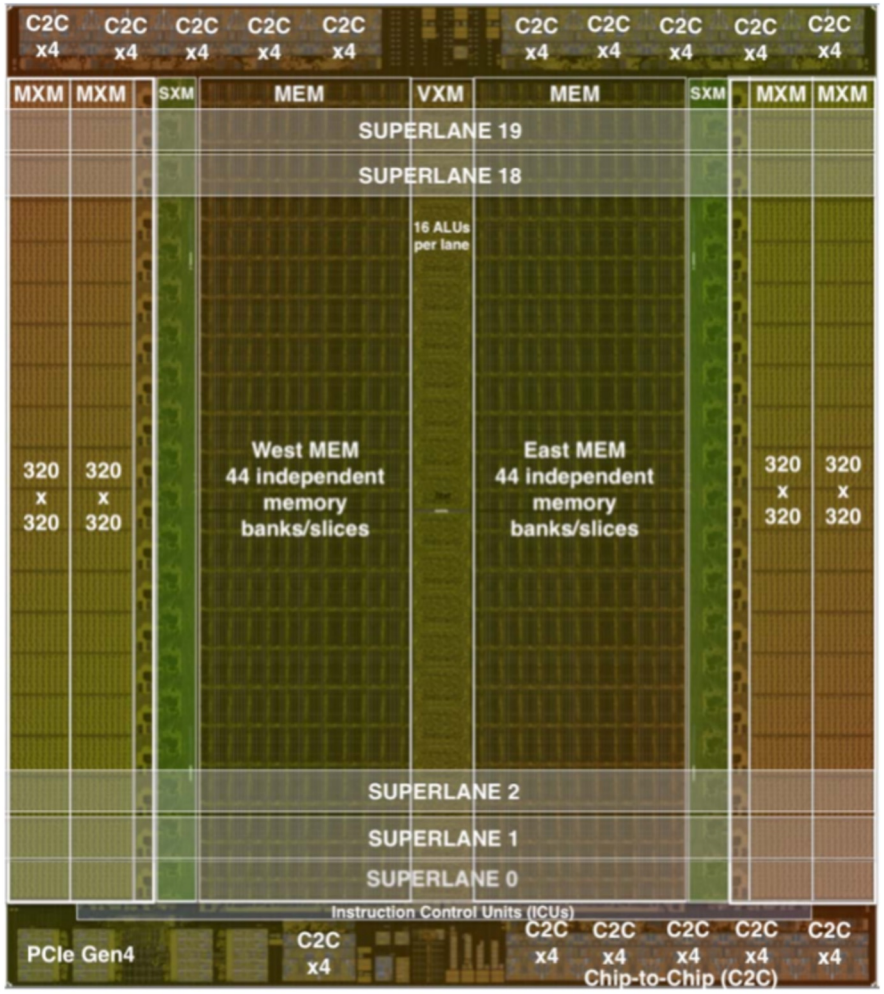
Source: Google

Source: D. Harris

Groq

- Programmable dataflow architecture
- 1000 TOPs/s peak INT8 performance
- 200 MB on-chip SRAM (80 TB/s)
  - No external memory, scales by increasing number of chips
- FP16 and INT8 precision
- Philosophy: "unroll" a multicore architecture on-chip spatially to allow for custom instructions





Source: Groq

Groq

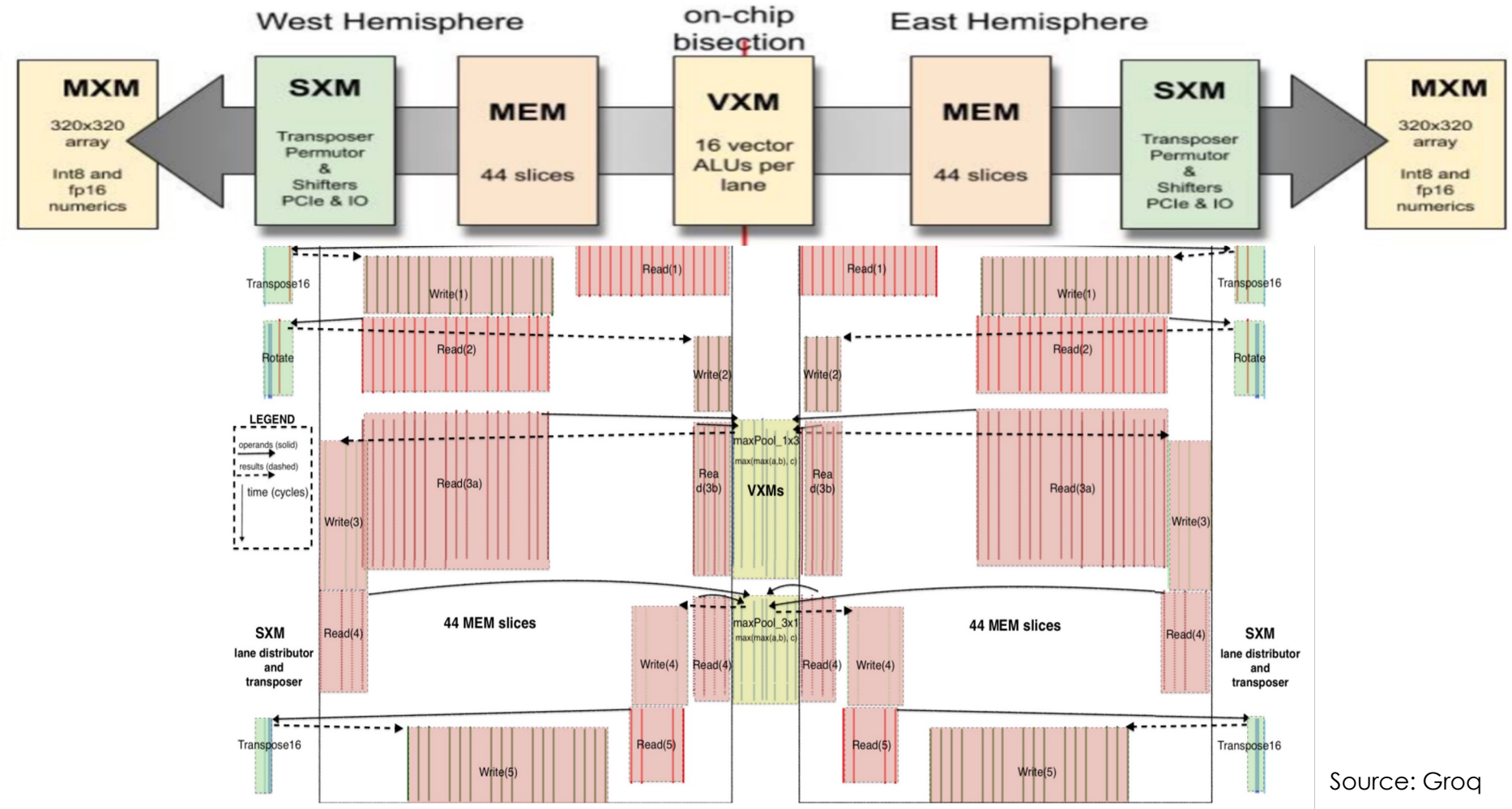
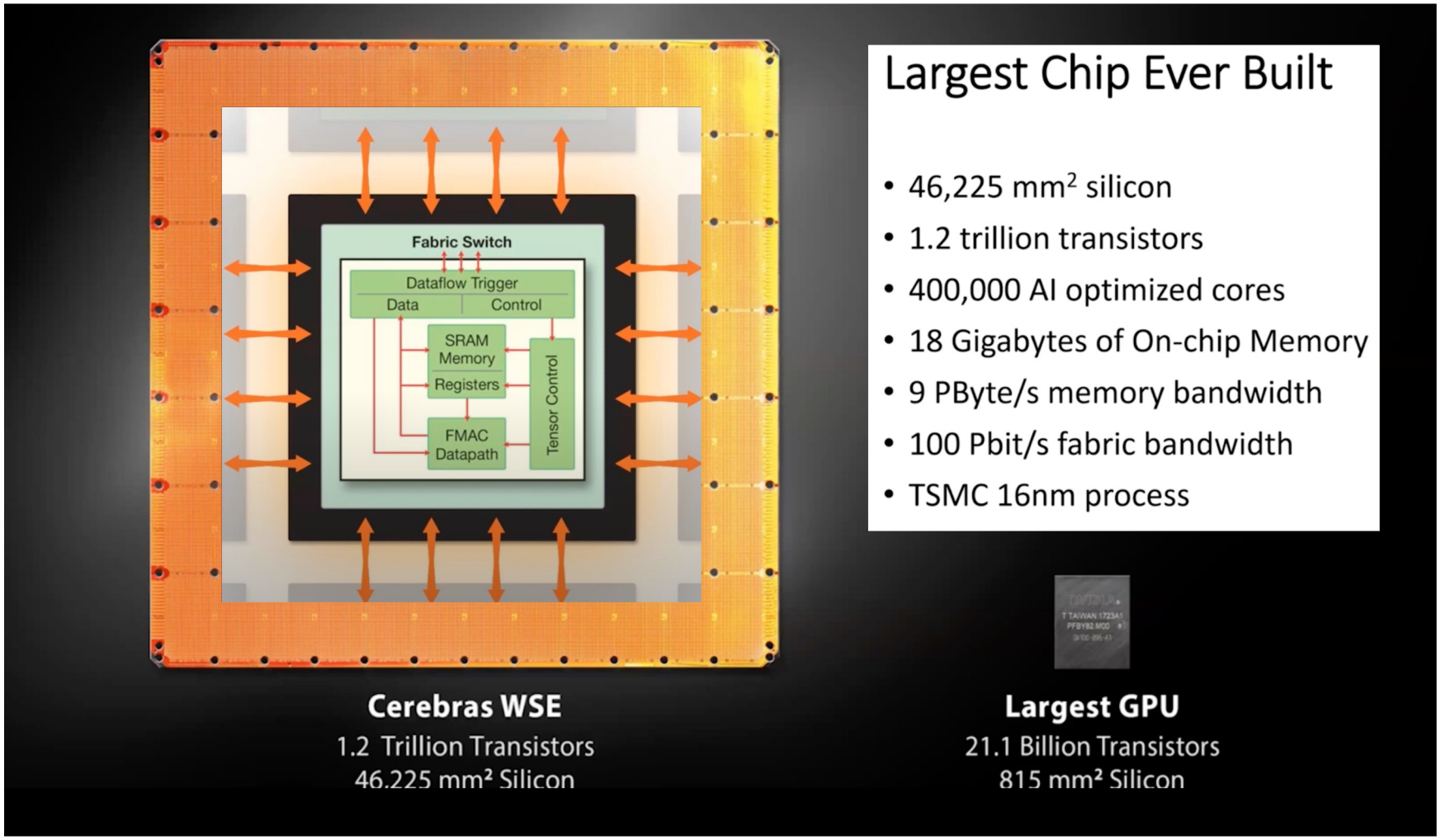


Fig. 11. Example instruction schedule for 3x3 max pool in ResNet50.

Source: Groq

Cerebras



Source:Cerebras

Summary of the Day

- HW enabling Deep Learning
- Performance Metrics
- Where does Energy Go?
- Hardware Efficiency Options
- Hardware Case Studies