In-Datacenter Performance Analysis of a Tensor Processing Unit

CS 533, SPRING 2025

JOVAN STOJKOVIC

Motivation

Main sources of improved performance in the past

Today

Future

Motivation

Main sources of improved performance in the past

- Architectural innovations: Wider-bit Processors, ILP, Multicore systems
- Technology improvements: Moore's Law and Dennard's Scaling

Today

- Moore's law and Dennard's scaling do not hold anymore
- ILP wall
- Limited benefits from parallelism (Amdahl's law)
- Products switched from PC/Server model to Client/Cloud model

Future

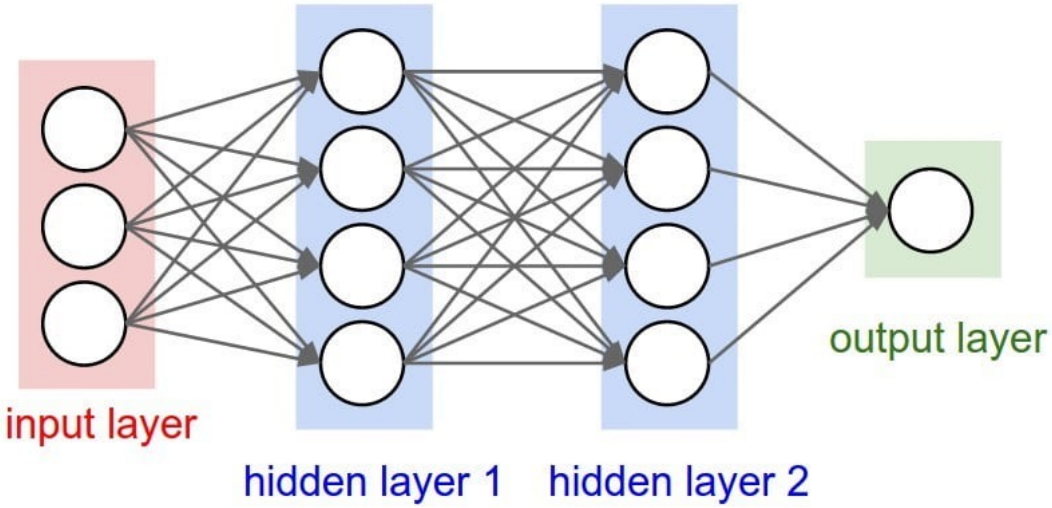
- Domain specific architectures – capable of doing only a few tasks but incredibly well

TPU – domain specific architecture for Deep Neural Networks

Outline

1. Neural Networks
2. Tensor Processing Unit – Architecture, Implementation and Software
3. Evaluation – Performance, TCO, Energy Proportionality
4. Conclusion

Deep Neural Networks



Widely used due to their high accuracy

Composed of up to 1000s layers

Layer is a set of neurons

Neuron is a non-linear function of a weighted-sum of inputs

Two Phases of Neural Networks (NNs)

TRAINING

INFERENCE

Two Phases of Neural Networks (NNs)

TRAINING

1. Learn weights
2. Supervised-learning + back propagation
3. Offline, emphasis on throughput
4. 16-bit FPs

INFERENCE

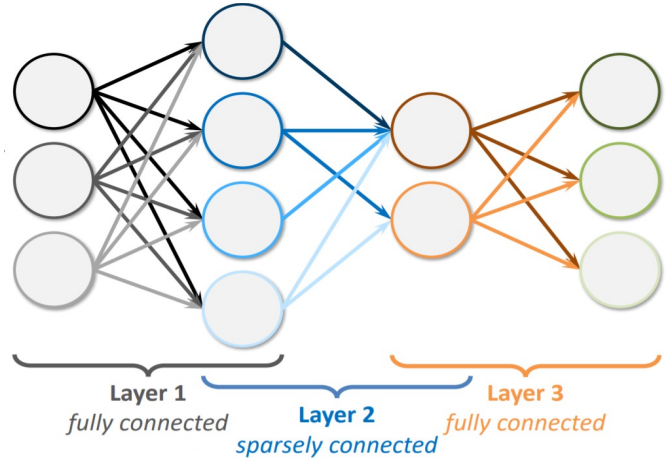
1. Read-only weights
2. Feed NN with new input and get the output
3. Online, emphasis on response time
4. 8-bit Integers (quantization)

Types of Neural Networks

Multi-Layer Perceptrons (MLPs)

Convolutional Neural Networks (CNNs)

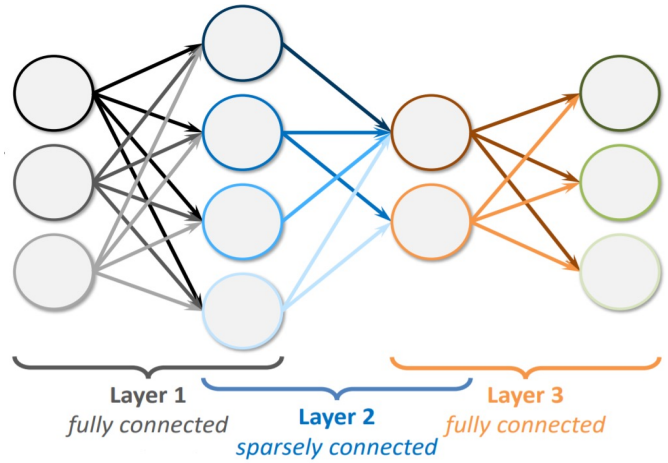
Recurrent Neural Networks (RNNs)



Types of Neural Networks

Multi-Layer Perceptrons (MLPs)

- Fully connected neural network - all outputs from layer i are inputs to all neurons in layer $i + 1$



Convolutional Neural Networks (CNNs)

- Feed forward, but sparsely connected
- Weight sharing: multiple synapses share the same weight

Recurrent Neural Networks (RNNs)

- Include feedback
- Long Short Term Memory (LSTMs) – decide what state to propagate and what to forget

NN Inference Workload

Name	LOC	Layers	Nonlinear Function	Weights	TPU Batch Size	Popularity
MLP0	100	5	ReLU	20M	200	61%
MLP1	1000	4	ReLU	5M	168	
LSTM0	1000	58	sigmoid, tanh	52M	64	29%
LSTM1	1500	56	sigmoid, tanh	34M	96	
CNN0	1000	16	ReLU	8M	8	5%
CNN1	1000	89	ReLU	100M	32	

95% of NN workload

Large weights, batched execution

CNNs target of research – only 5%

TPU Origin

2006 – no need to deploy GPUs, FPGAs and ASICs in Google datacenters

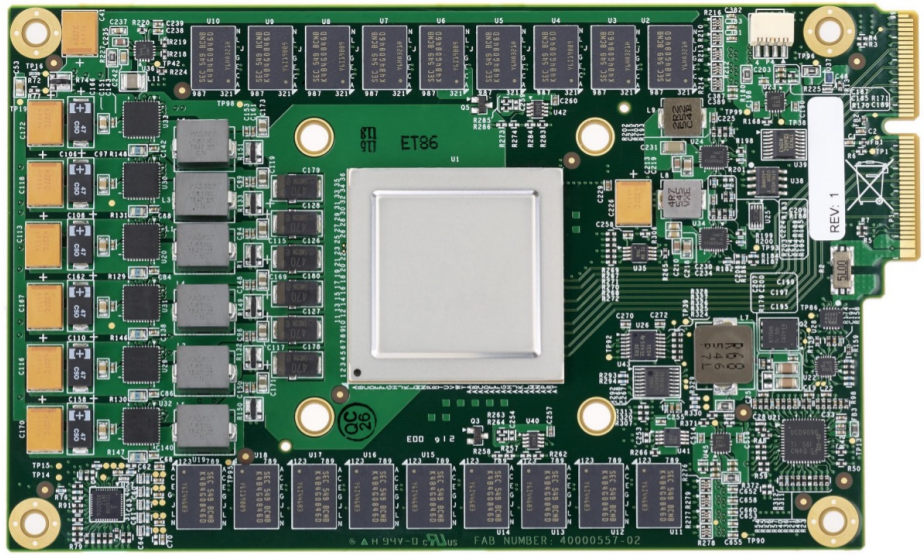
- the few applications that could run on special hardware could be done virtually for free using the excess capacity of their large datacenters

2013 – projection that with users searching by voice for three minutes a day using DNNs, computational demand would double

Custom ASIC for inference and GPU for training

Goal: improve cost-performance by 10x over GPUs and maintain backward compatibility

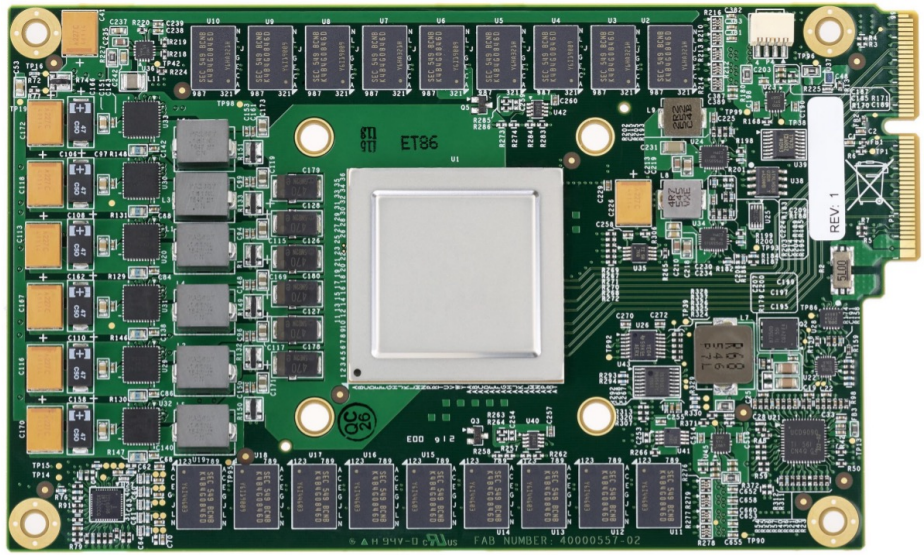
What is a TPU?



Designed as a **coprocessor** on the PCIe I/O bus

TPU **does not fetch instruction** by itself

What is a TPU?



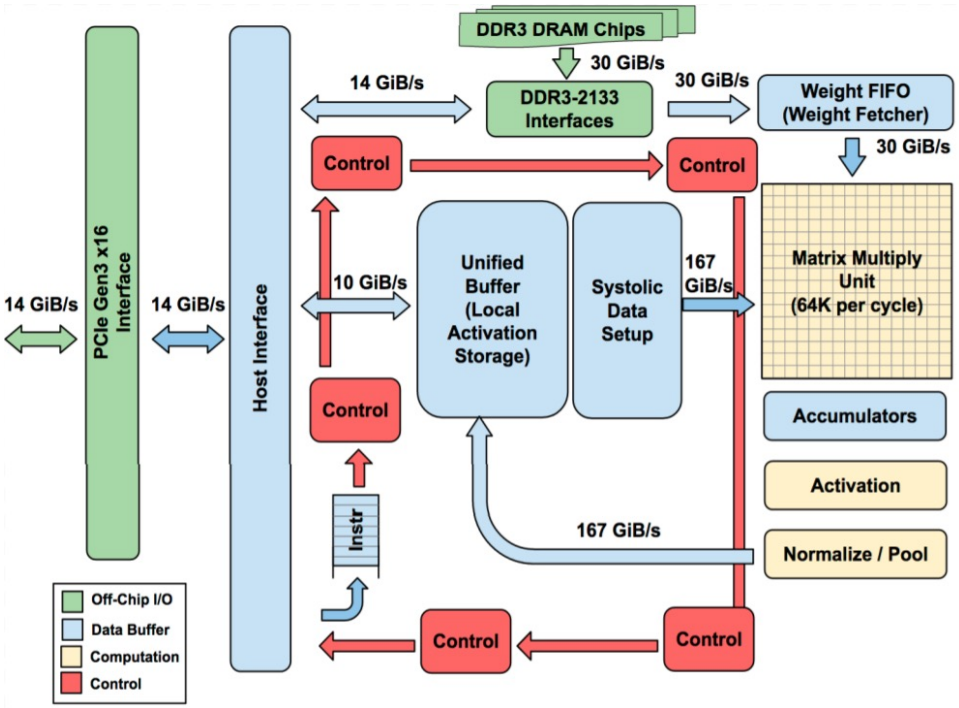
Designed as a **coprocessor** on the PCIe I/O bus

- Not tightly connected with a CPU
- Can plug into existing servers
- Reduce the chance of delaying deployment

TPU **does not fetch instruction** by itself

- CPU fetches the instructions and sends them over PCIe bus
- Simplify hardware design and debugging

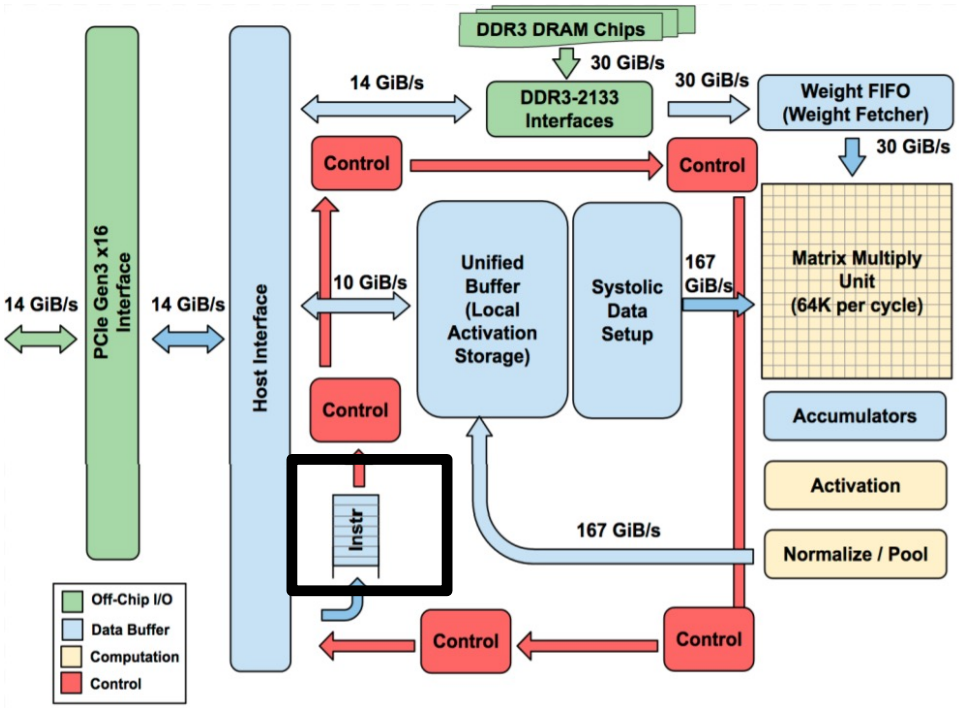
TPU Block Diagram



Main components:

1. *Matrix Multiply Unit*
2. *Unified Buffer*
3. *Weight FIFO*
4. *Accumulators*
5. *Instruction Buffer*

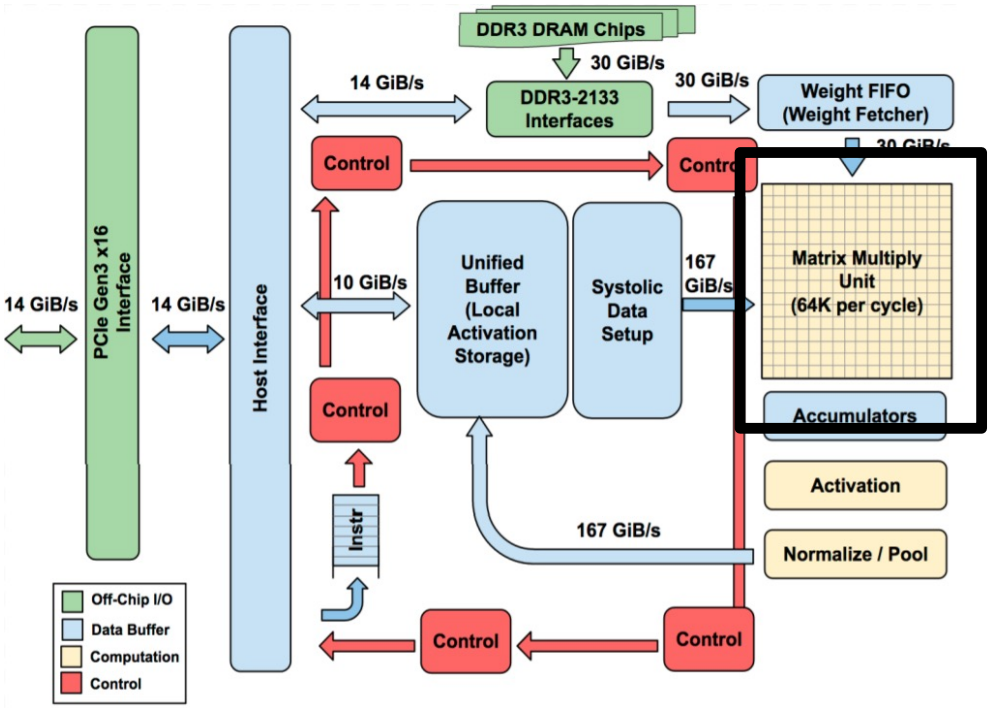
Instruction Buffer



CPU fetches the instructions and sends them over PCIe bus

Instructions are stored in Instruction Buffer

Matrix Multiply Unit



256 x 256 8-bit MACs

Wider inputs with slower operations

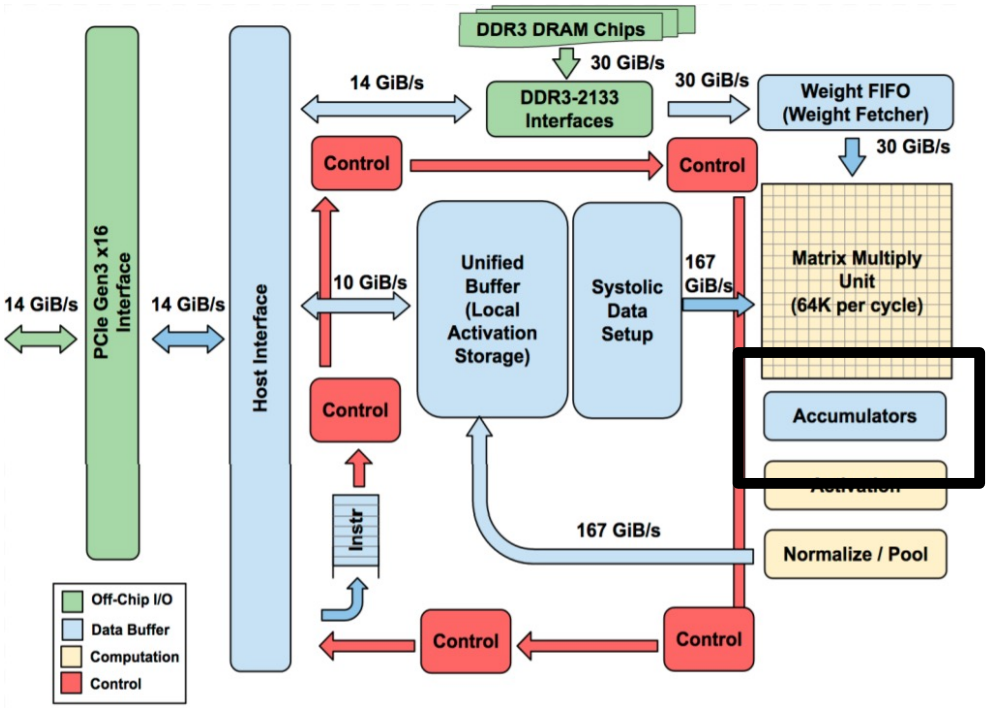
Reads and writes 256 values per clock cycle

Able to perform

- Matrix multiplication
- Convolution

Designed for dense matrices

Accumulators

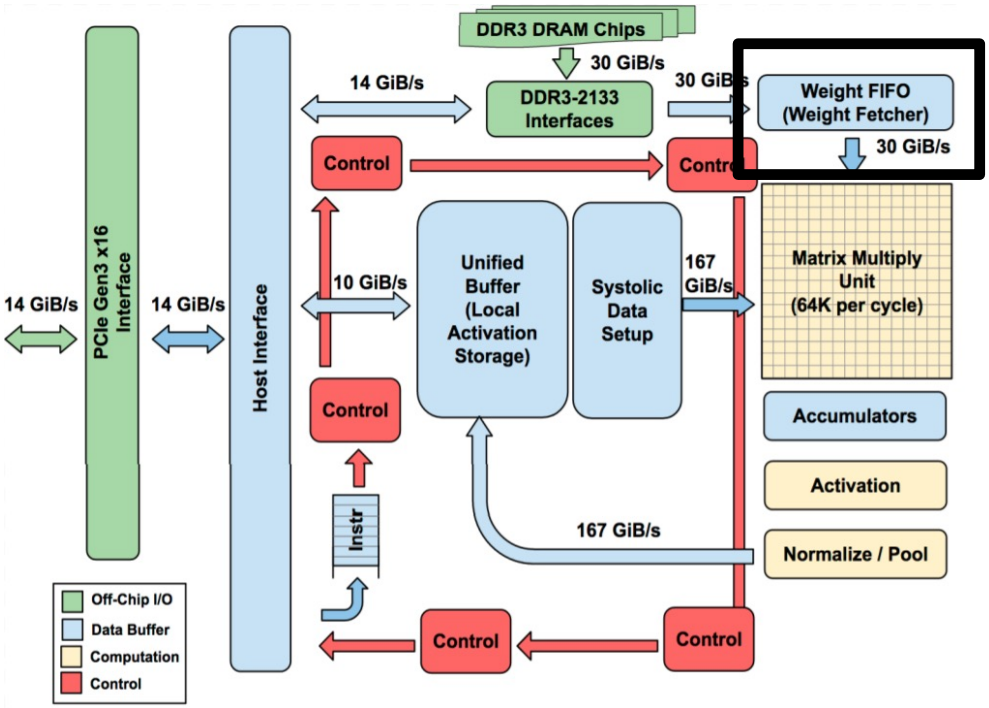


16-bits products of MMU are stored in 4MiB of 32-bit accumulators

4096 256-elements of 32-bit accs

- 1350 needed for peak performance
- Round to 2K
- Double buffering – 4K

Weight FIFO



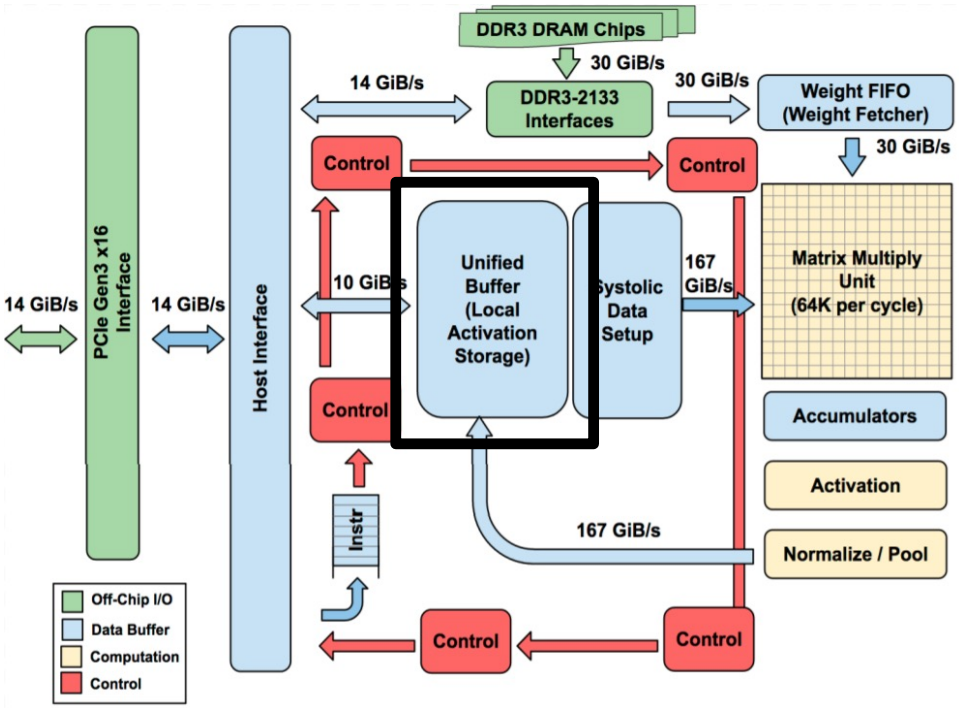
On-chip storage

Fetches data from Weight Memory

256x256 size

4 tiles deep

Unified Buffer

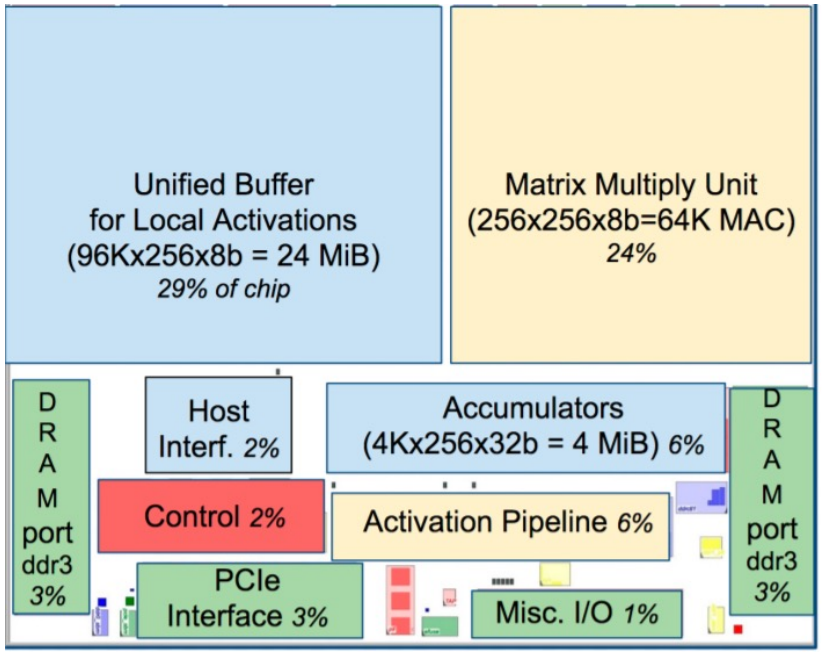


24MB on-chip storage

Store intermediate results and input activations

Communicate with host memory

Floorplan of a TPU Die



Hardware specialized for matrix multiplication

Two thirds of a die are data path

Only 2% is control

Legend:

- Blue and yellow is data-path
- Red is control
- Green is I/O

TPU Instructions

5 main **CISC** instructions

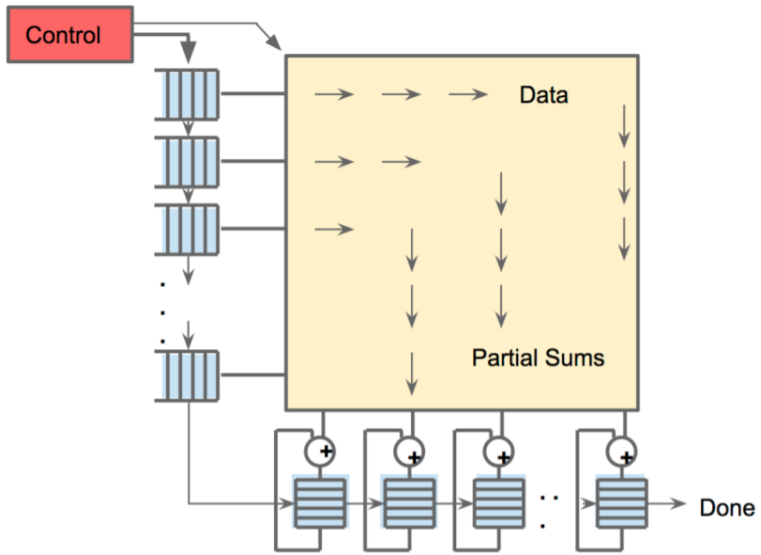
- *Read Host Memory*
- *Write Host Memory*
- *Read Weights*
- *Matrix Multiply / Convolve*
- *Activate*

4-stage overlapped execution, 1 instruction per stage

- Execute other instruction while MMU busy – Read Weights follows decoupled-access/execute

Complexity in SW: no branches, in-order issue, SW controlled buffers and pipeline synch

Systolic Execution in MMU



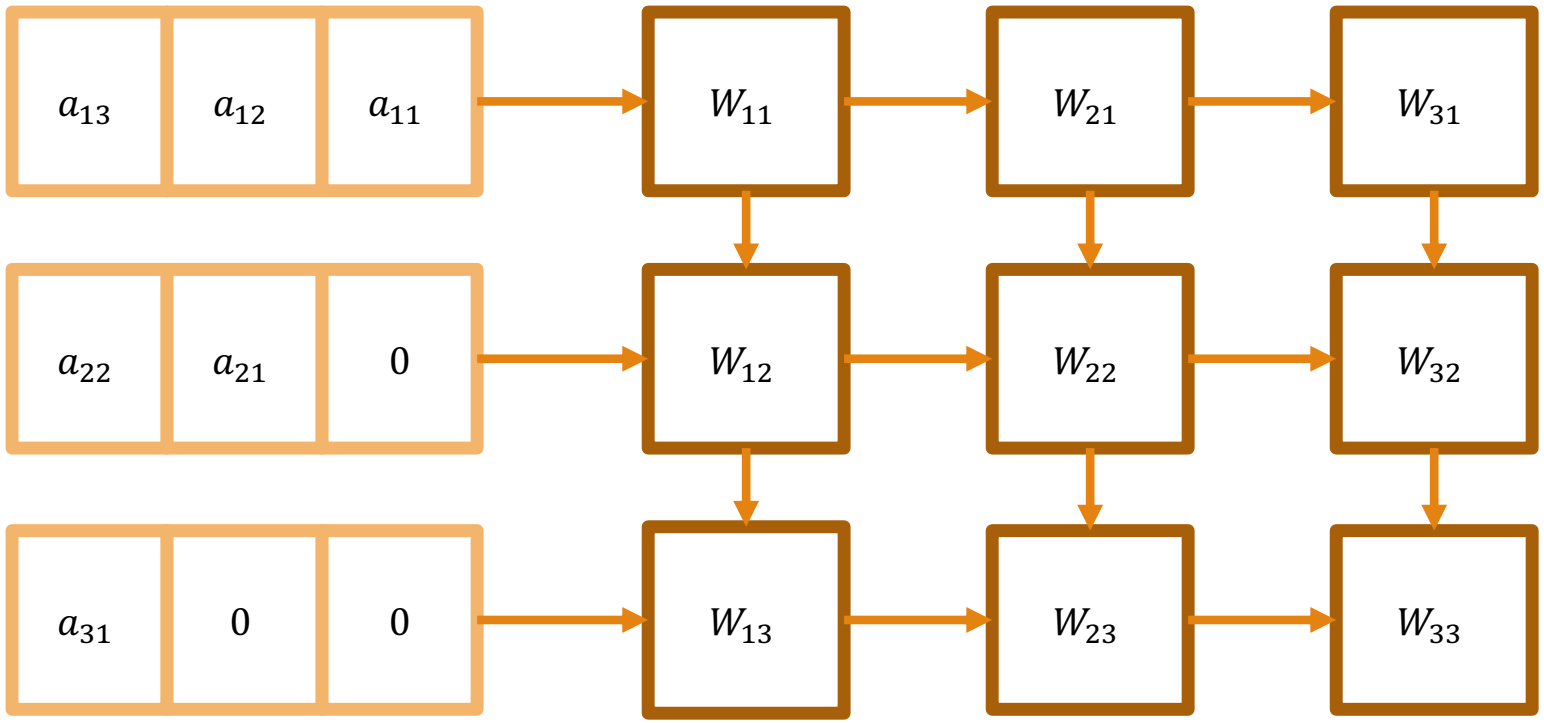
Minimize the reads and writes to save energy

The matrix unit uses **systolic execution**

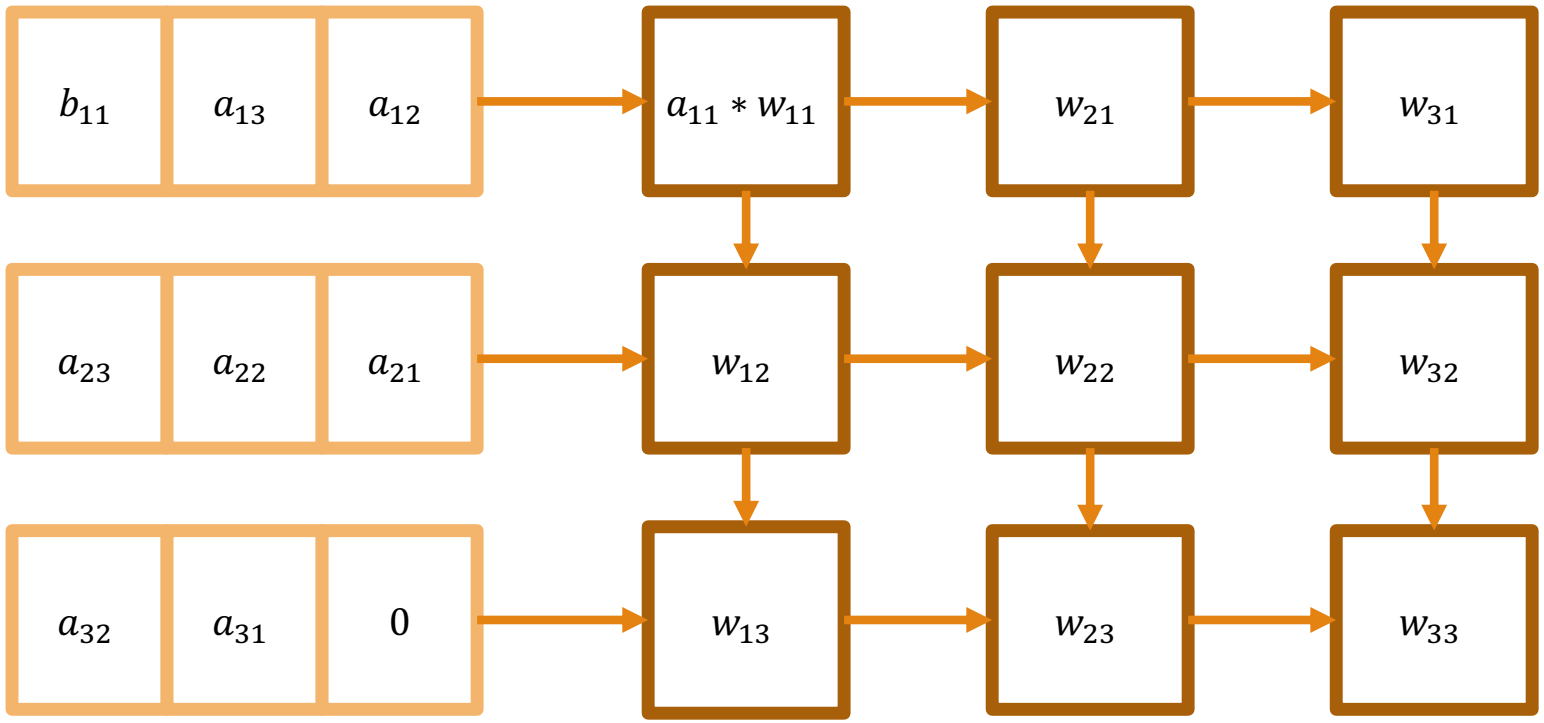
Data flows in from the left, weights loaded from the top

256-element MAC moves through the matrix as a diagonal wavefront

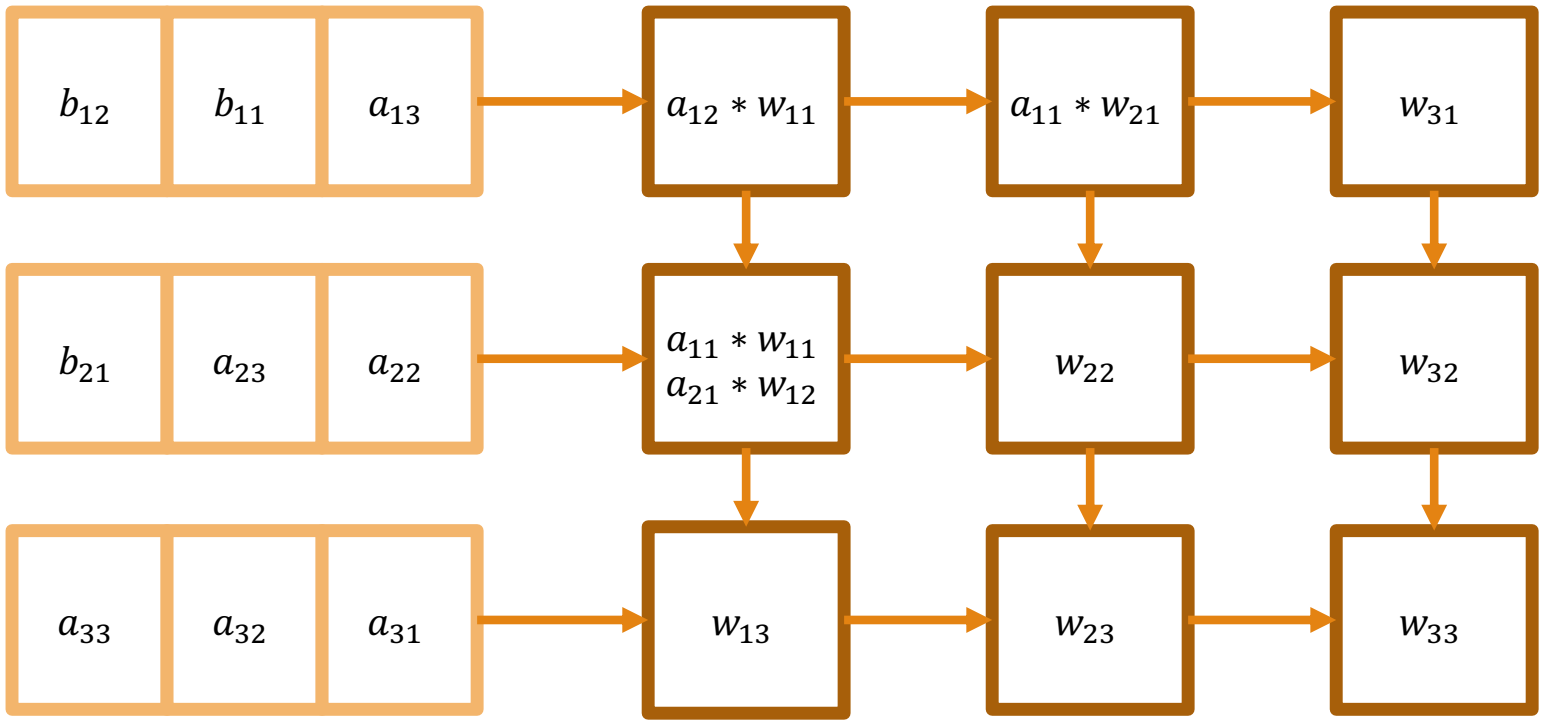
Systolic Execution – Example



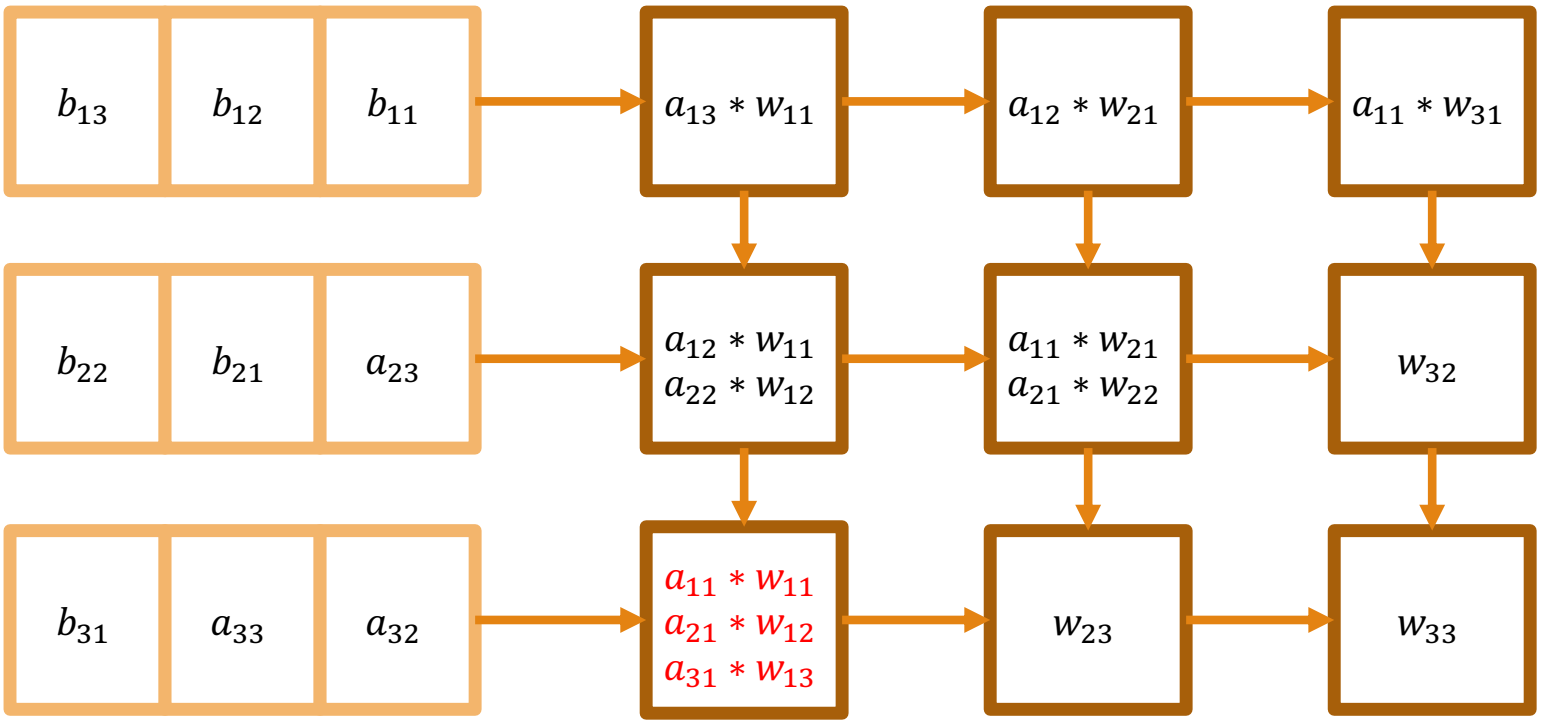
Systolic Execution – Example



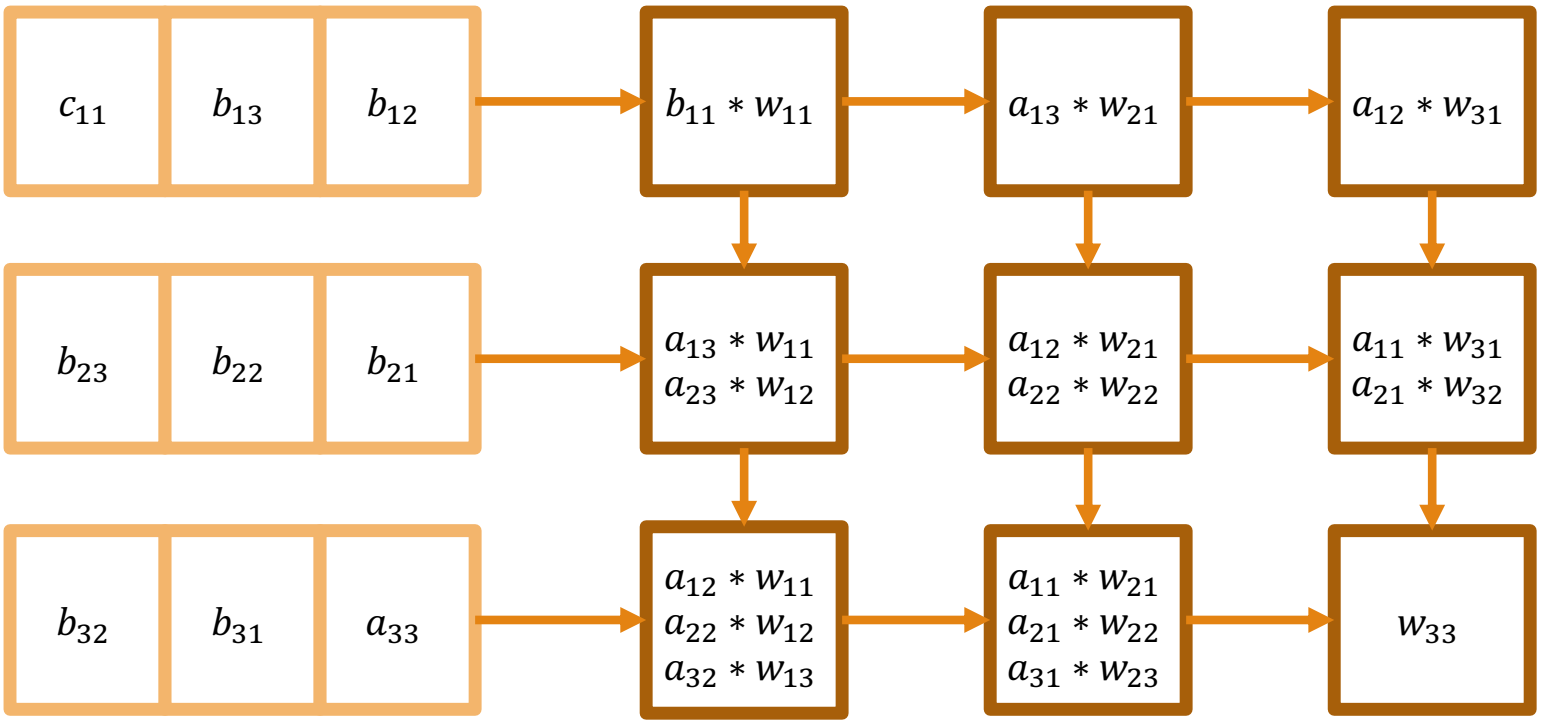
Systolic Execution – Example



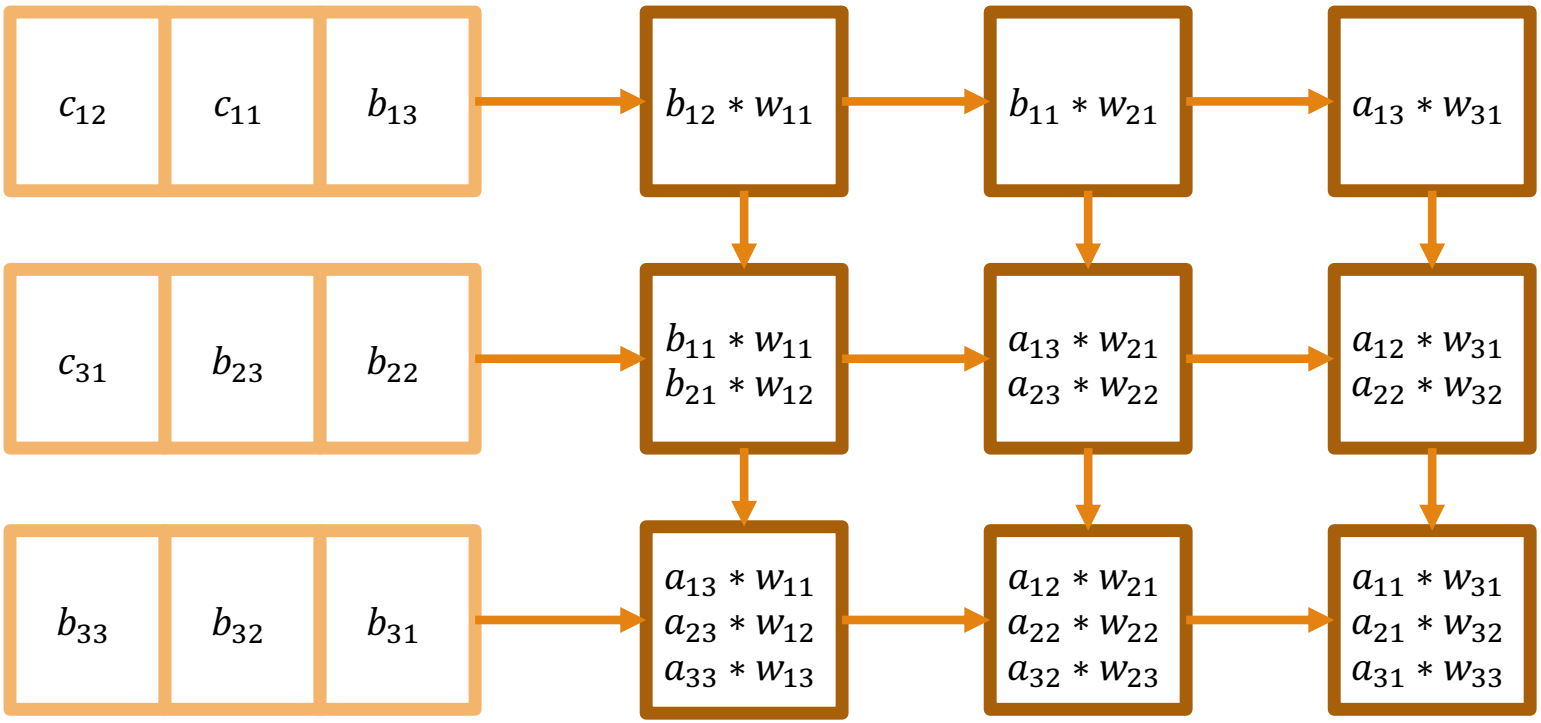
Systolic Execution – Example



Systolic Execution – Example



Systolic Execution – Example



Configuration Comparison

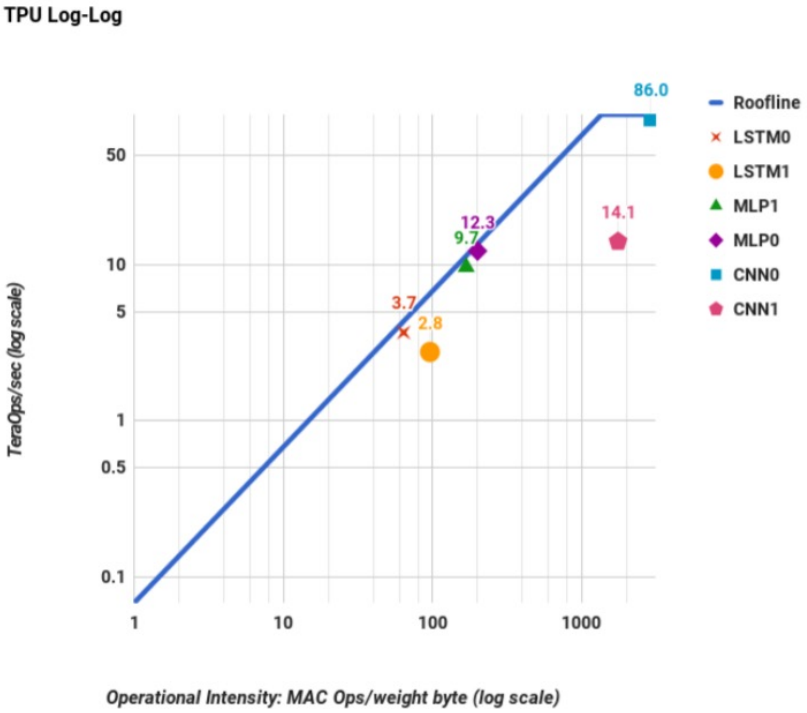
2x reduction in area over both CPU and GPU

2x reduction in TDP over both CPU and GPU

3.5x increase in on-chip memory over GPU

	Die Size (mm^2)	TDP	On-Chip Memory (MB)
CPU	662	145	51
GPU	561	150	8
TPU	331	75	28

TPU Roofline



Operational intensity vs. Performance

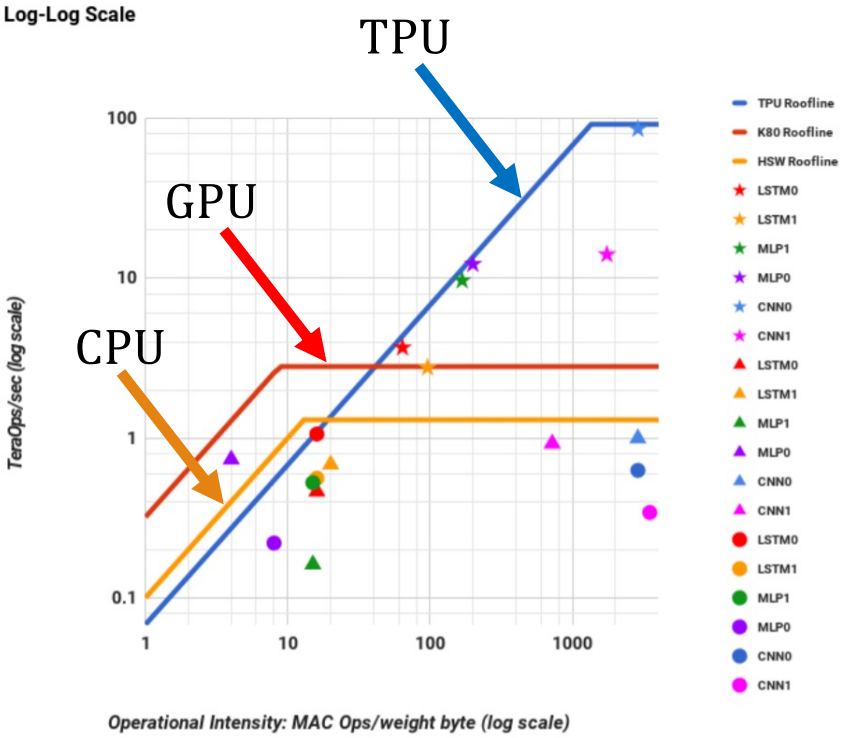
Peak computation rate determines flat line

Memory bandwidth is slanted line

LSTMs and MLPs are memory bound

CNNs are compute bound

Log Rooflines for CPU, GPU, TPU



Peak performance for CPU and GPU significantly lower than peak performance of TPU

With CPU and GPU applications generally further below their ceiling

Increases in response time cause customers to use a service less

Inference prefers latency over throughput

Why so far below the Roofline?

	Batch Size	99th Percentile Response Time (ms)
CPU	16	7.2
CPU	64	21.3
GPU	16	6.7
GPU	64	8.3
TPU	200	7.0
TPU	250	10.0

99th percentile response time limit was 7ms

CPU improves average latency

GPU improves throughput

TPU cares about 99th percentile response time

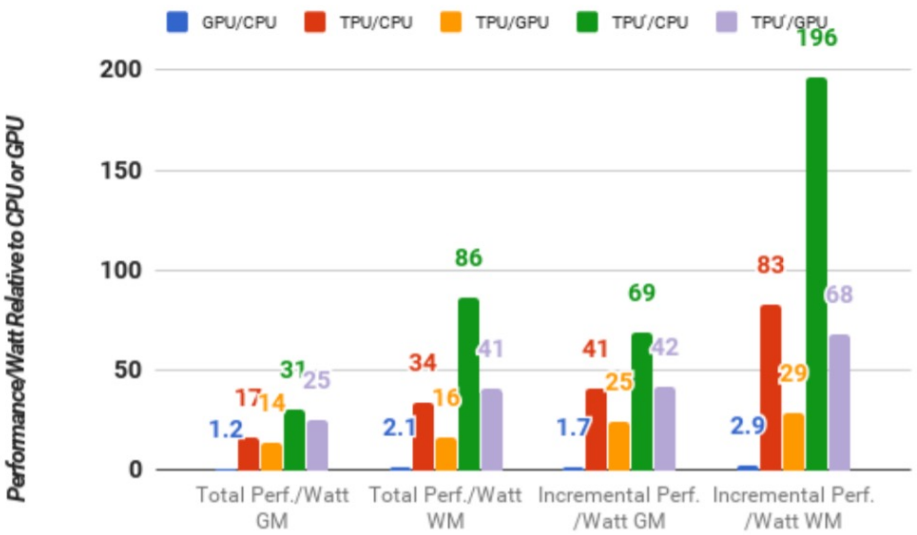
CPUs and GPUs must use less-efficient, smaller batch sizes (16 vs 200)

TPU Outperforms CPU and GPU

Geometric mean when the exact mix of applications is not known
GPU improves performance by 1.1x
TPU improves performance by 14.5x
→ TPU/GPU = 13.2

Weighted mean is computed since the authors know the exact mix of applications
GPU improves performance by 1.9x
TPU improves performance by 29.2x
→ TPU/GPU = 15.4

Performance/Power Improvement



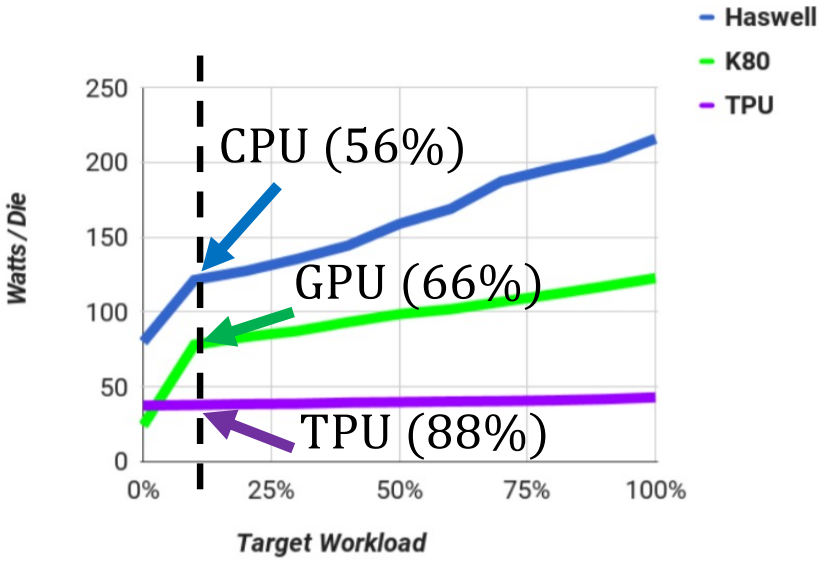
Performance/Watt as proxy for performance/TCO

Total includes the power consumed by the host CPU server

Incremental subtracts the host CPU server power from the GPU and TPU beforehand

Green bar is improved TPU (with GDDR5)

Energy Proportionality



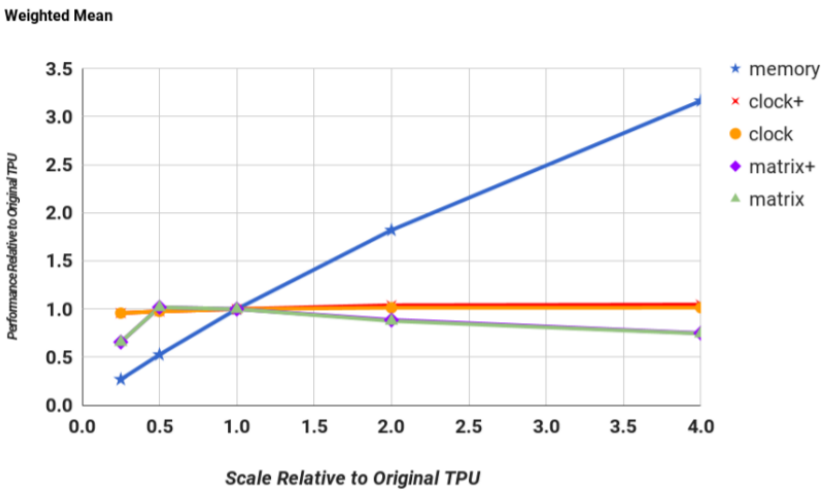
The cost of electricity is based upon the average consumed as the workload varies during the day

TPU has the lowest power consumption

But it has the worst energy proportionality

Lack of energy-saving features

Alternative TPU Designs



1. Increasing memory bandwidth – the biggest impact
2. Cock rate has little benefit on average with or without more accumulators
3. Matrix unit expansion causes performance degradation

Why is TPU successful?

Large, but not too large matrix multiply unit

Substantial software-controlled on-chip memory

The ability to run the whole inference (no dependences on host CPU)

Single-threaded deterministic execution model

Use of 8-bit integers through quantization

Omission of general-purpose features

Applications are easy to port since they are written in TensorFlow

Conclusion

Inference prefers latency over throughput – users care about response time

TPU can run applications written using TensorFlow

- Performance improvement: 15x over NVIDIA K80, 29x over Intel Haswell
- Performance/Watt improvement: 29x over NVIDIA K80, 83x over Intel Haswell

TPU using 2015 GPU memory would go two to three times as fast and boost the performance/Watt advantage to nearly 70 over the K80 and 200 over Haswell

Thank you!
Questions?