CS 758: Advanced Topics in Computer Architecture

Lecture #17: ML: RNNs

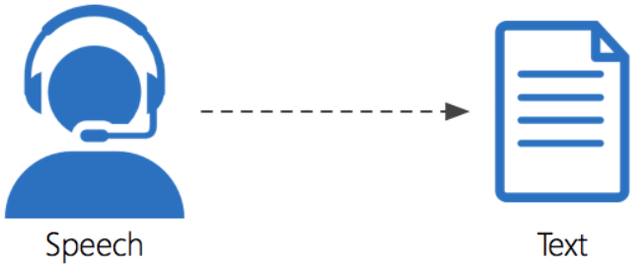
Professor Matthew D. Sinclair

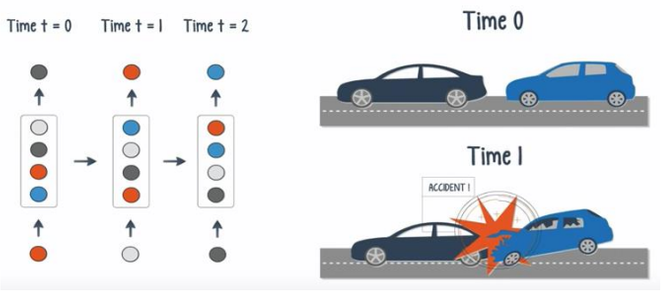
Some of these slides were developed by Tushar Krishna at Georgia Tech

Slides enhanced by Matt Sinclair

Recurrent Neural Network (RNN) Use Cases

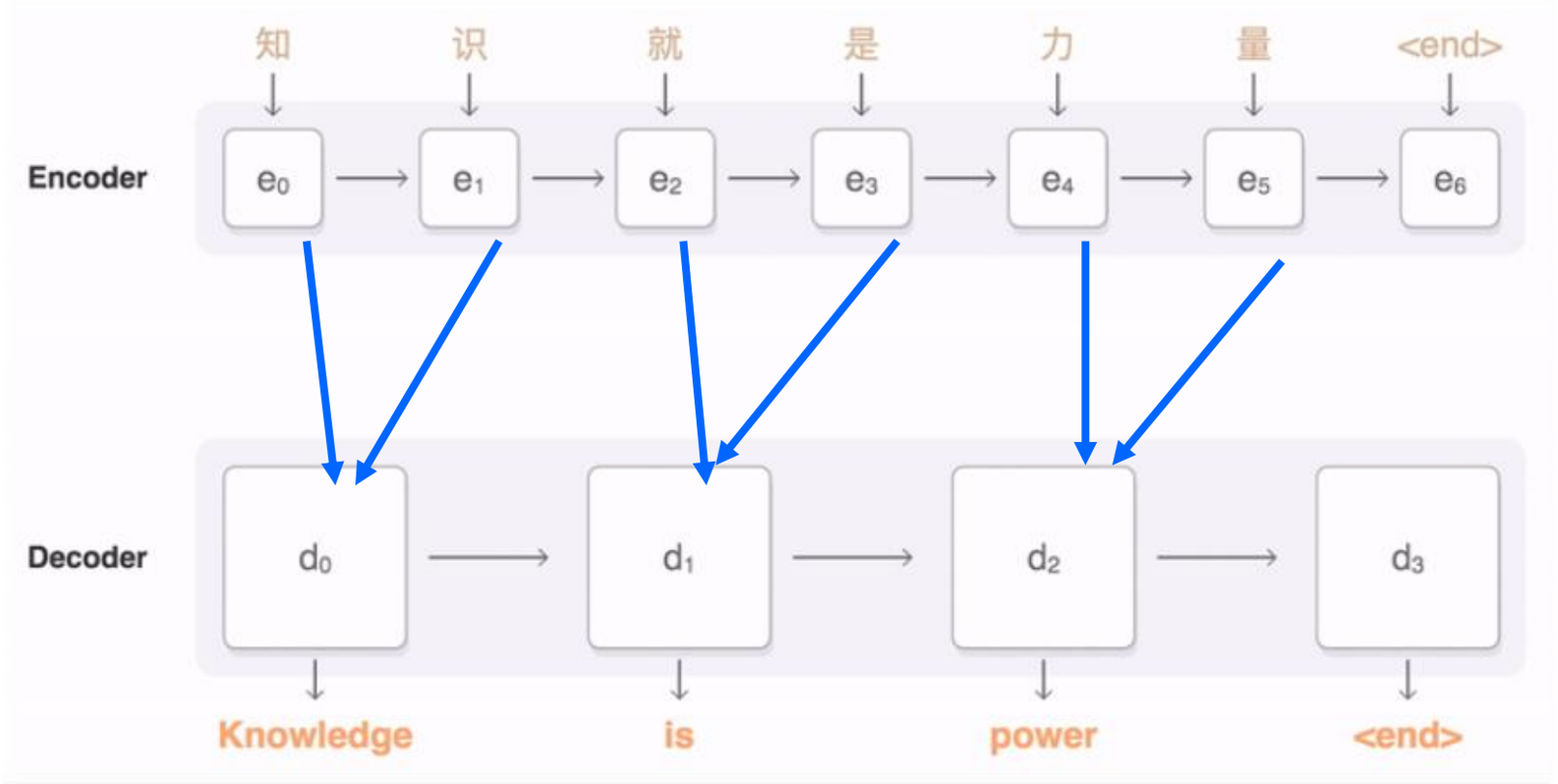
- RNNs heavily used in several important applications
- RNN architecture different than CNN
- Existing CNN optimizations are not very effective for RNNs





RNN Use Cases (Cont.)

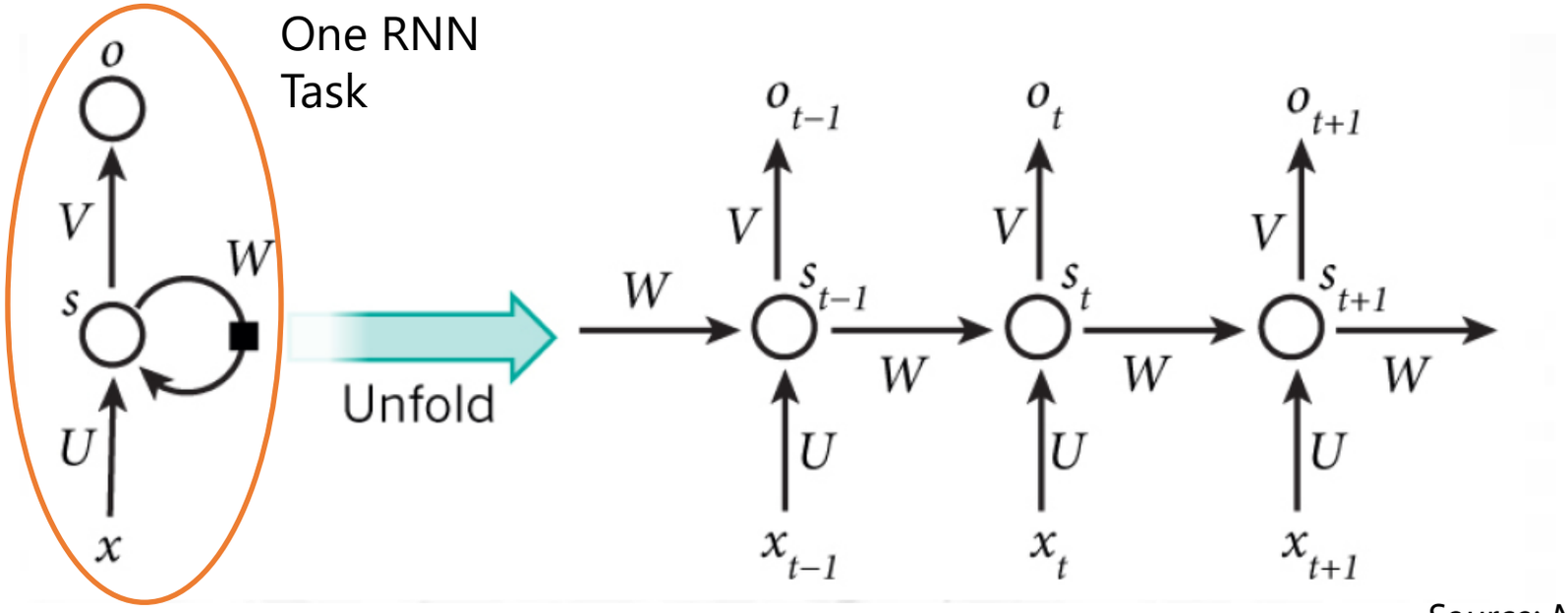
- RNNs are widely used in machine translation and speech recognition
- The memory cells of RNNs help to infer the sequence of inputs
  - Goal: understand long-term dependencies between multiple inputs



[Source: https://github.com/google/seq2seq]

RNN execution model

- *V*: the weight we multiply the output *o_t* by
- *U*: the weight we multiply the input *x_t* by
- *W*: the weight we multiply the previous hidden state's value *s_t-1* by

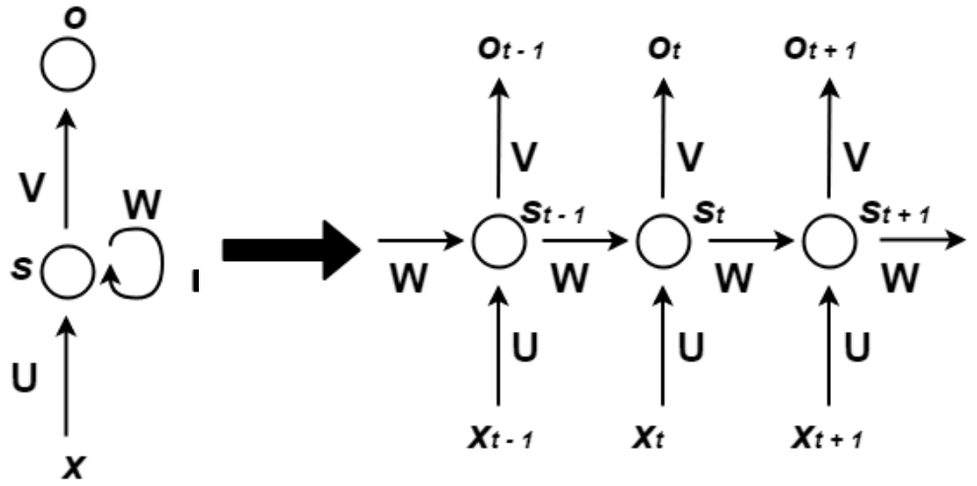


Source: Nature

- Each RNN task passes through multiple time steps (*t*)
- Each time step is dependent on the previous one (e.g., *t-1* and *t*)
- The hidden state size determines the degree of parallelism
  - Large hidden state sizes increase parallelism but may lead to overfitting

RNN Background

- RNNs used to recognize and predict sequences



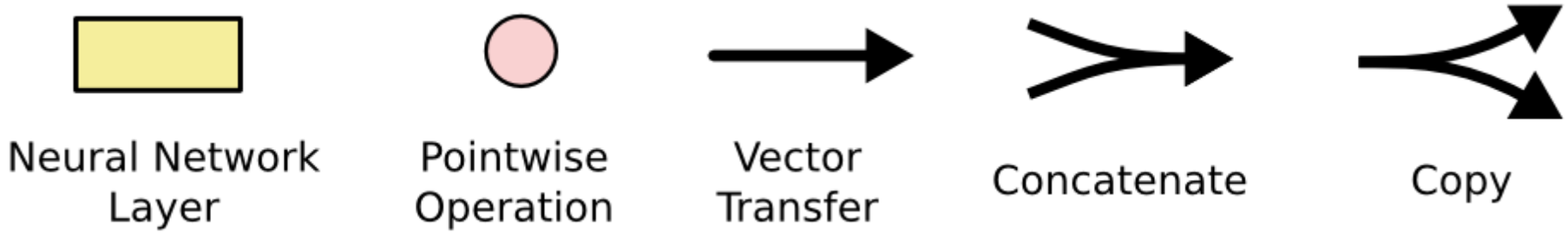
$x = input$ $U = input\ weight$

$s = hidden\ state$ $W = recurrent\ weight$ $t = timestep$

$o = output$ $V = output\ weight$

- They need to remember previous inputs
- Have real-time deployment constraints

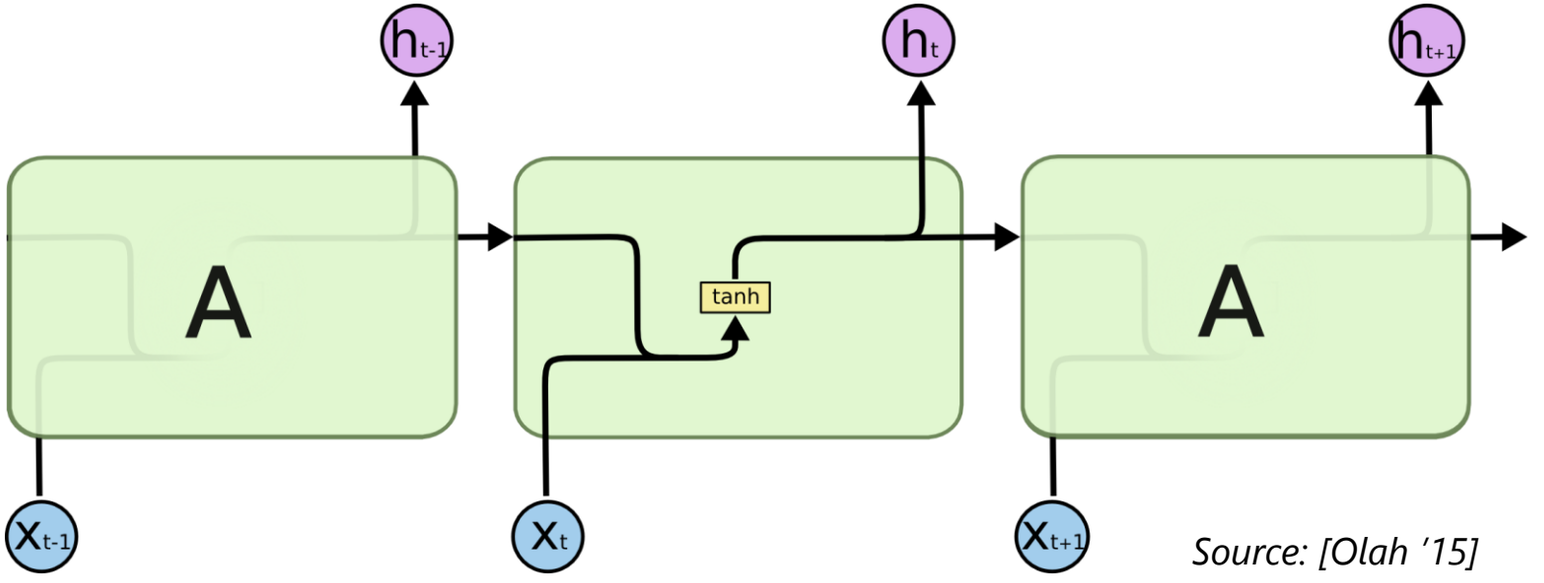
RNN Terminology (for next few Figures)



Source: [Olah '15]

RNN Variant #1: Vanilla

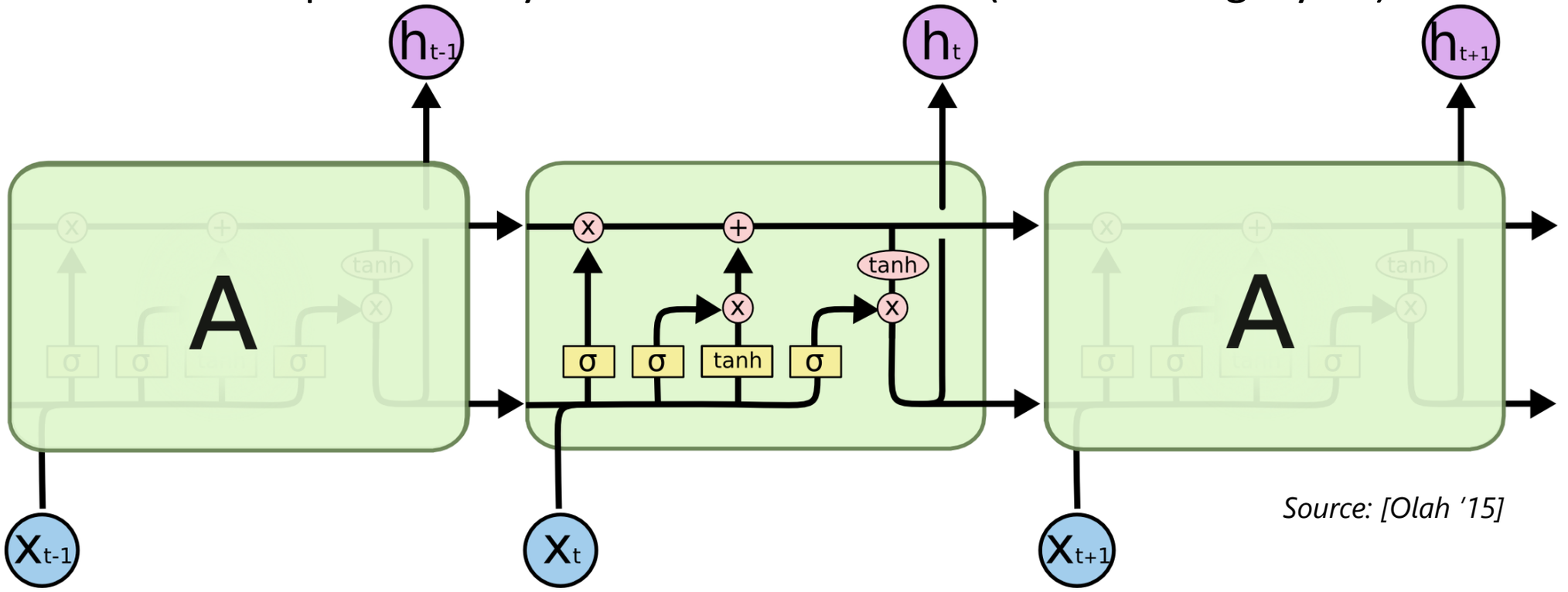
- 3 main RNN variants: Vanilla, LSTM, and GRU
- Vanilla
  - "Traditional" RNN – chain of repeating modules
  - Key distinguishing feature: no "memory" unit
  - Issue: hard to retain long-term dependencies because no "memory"



Source: [Olah '15]

RNN Variant #2: LSTM

- Long Short-Term Memory (LSTM)
  - Most widely used today
  - 4X more computationally intensive than Vanilla (4 interacting layers)

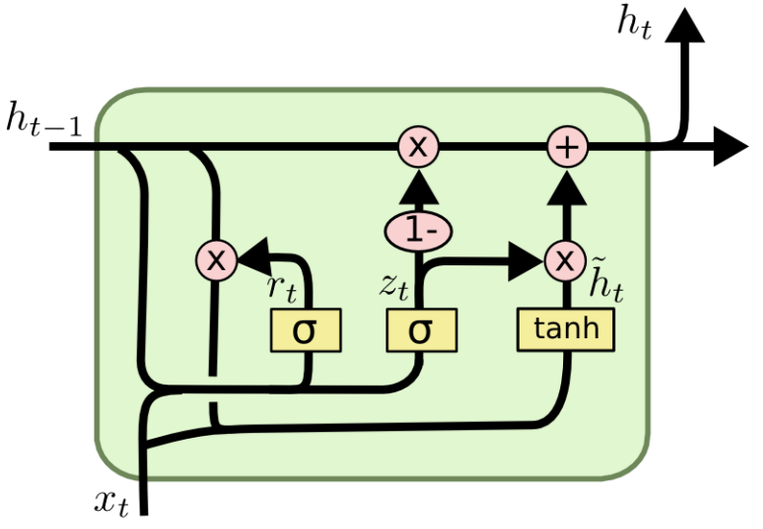


Source: [Olah '15]

Hidden slides have more details on each layer

RNN Variant #3: GRU

- Gated Recurrence Unit (GRU)
  - ~1/2 computational intensity of LSTM/~2X computation of Vanilla
    - Combines first two layers of LSTM into a single layer
    - Partially combines third and fourth layers of LSTM
  - Was popular, but LSTMs again more widely used
  - Which is best very case-by-case dependent



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

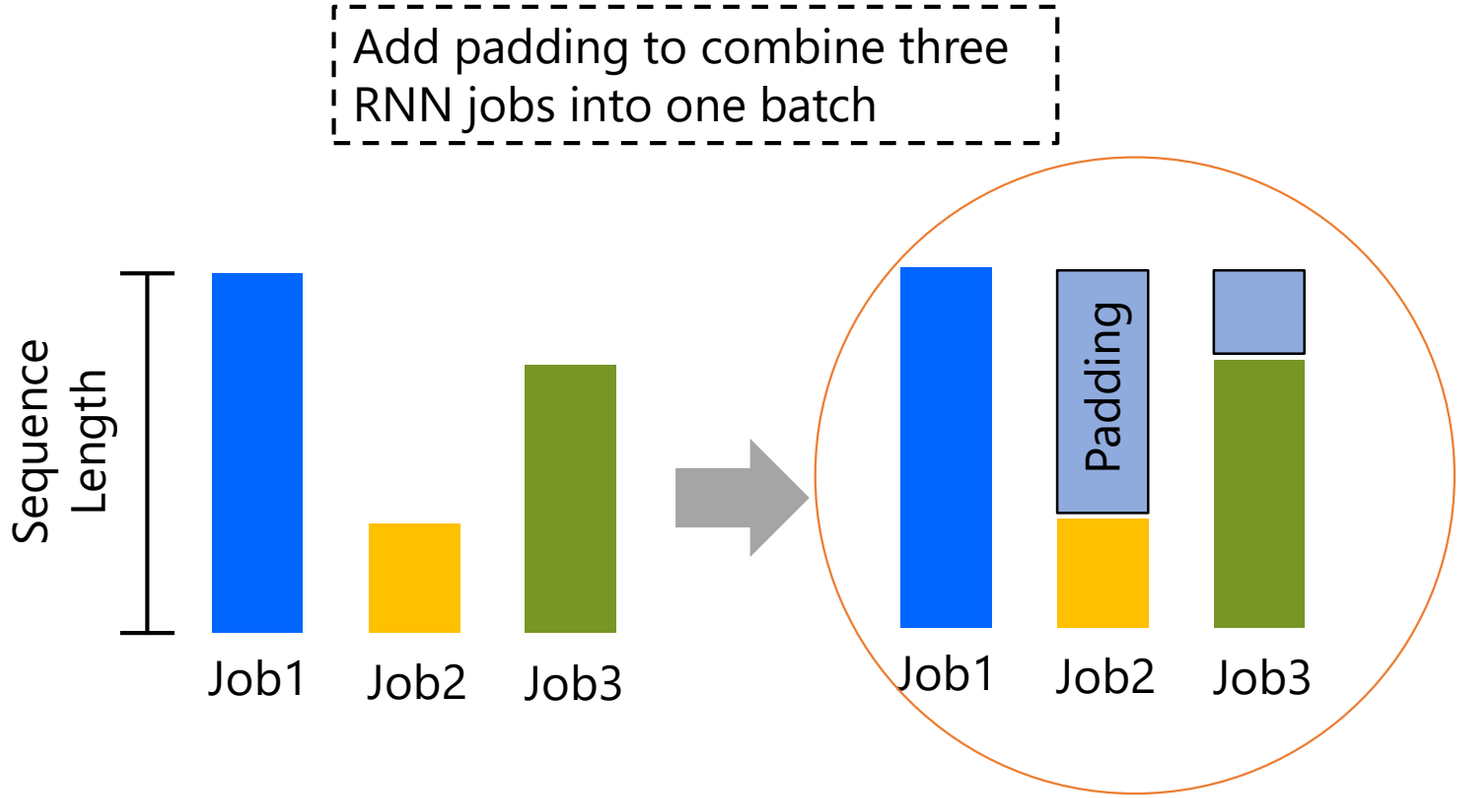
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Factors that impact RNN performance

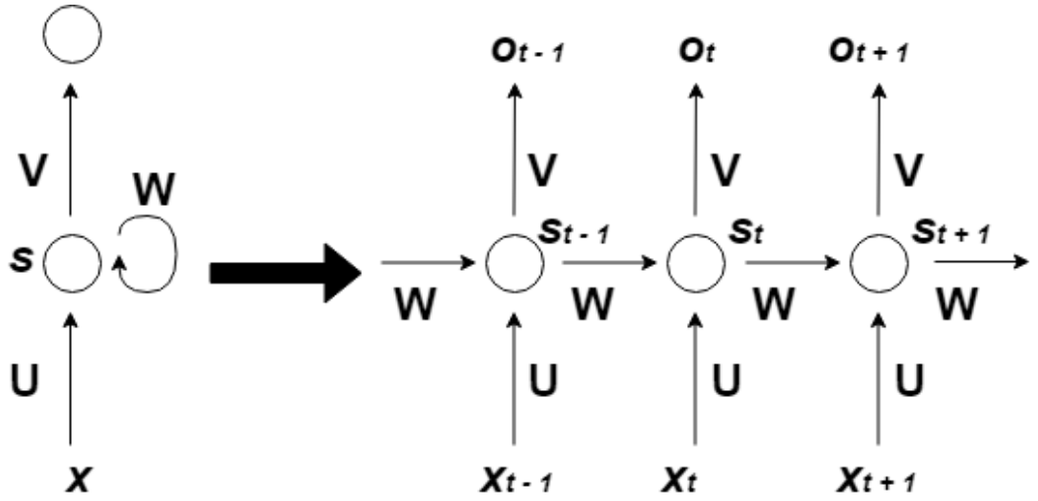
- Sequence length
  - Input/problem dependent
  - Varies from RNN job to RNN job
- Hidden state size
  - Impacts the accuracy of training data
  - Usually a small hidden state is sufficient to achieve good training accuracy (still true?)
- Arrival time of RNN jobs
  - Impacts the response time of all RNN jobs in a single batch
  - Can be sporadic
- The batch size of RNN jobs
  - May require additional padding if the RNN jobs have different sequence lengths

Why doesn't batching help RNN inference performance?

- Issues:
  - Implicit barrier of the batch operation → delays all RNN jobs in a batch
  - Jobs unlikely to arrive simultaneously → postpones start of all jobs in batch
  - Padding leads to unnecessary computation



Challenges



- Contain loops to remember information
  - sequential dependencies **limits parallelism**
- Batching difficult due to strict service level agreements (SLAs)
  - poor data reuse - **high memory bandwidth**
- Read and write activations between timesteps
  - requires **high memory bandwidth**

The Future

- Attention models/Transformer networks slowly replacing RNNs [NeurIPS ‘17]
  - Removes all convolutional and recurrent calculations
  - Just keeps the “attention” mechanism from RNNs
  - Advantages: less serial dependencies, faster training convergence, potentially less training data
- Likely the next “big” ML application

Microsoft

Serving DNNs in Real Time at Datacenter Scale with Project Brainwave

E. Chung* et al., Hot Chips 2018, ISCA 2018

*Microsoft

The Rise of Deep Learning in ML

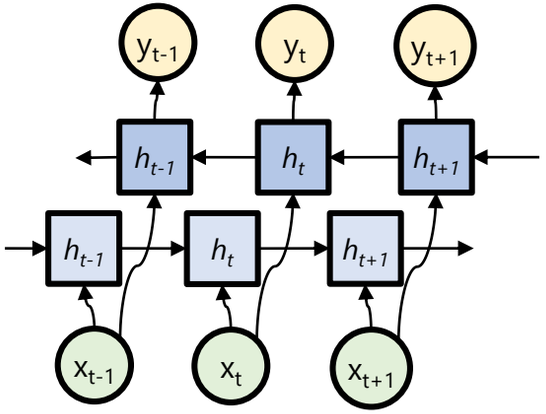
Deep neural networks have enabled major advances in machine learning and AI

- Computer vision
- Language translation
- Speech recognition
- Question answering
- And more…

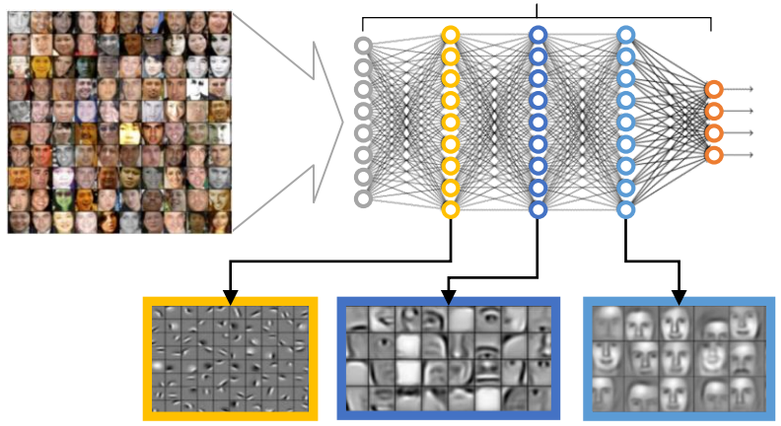
Problem: DNNs are challenging to serve and deploy in large-scale online services

- Heavily constrained by latency, cost, and power
- Size and complexity of DNNs outpacing growth of commodity CPUs

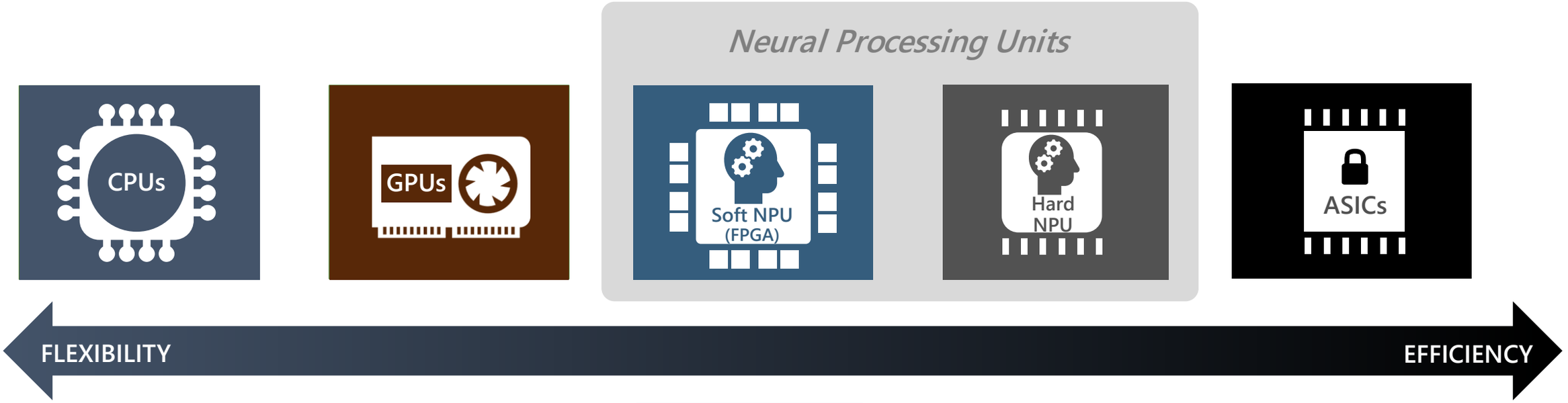
Recurrent Neural Networks



Convolutional Neural Networks



Silicon alternatives for DNNs



BrainWave
Baidu SDA
Deephi Tech
ESE
Teradeep
Etc.

Cerebras
Google TPU
Graphcore
Groq
Intel Nervana
Movidius
Wave Computing
Etc.

The power of Deep Learning on FPGA

Performance

- Excellent inference performance at low batch sizes
- Ultra-low latency serving on modern DNNs
  - >10X lower than CPUs and GPUs
- Scale to many FPGAs in single DNN service

Flexibility

- FPGAs ideal for adapting to rapidly evolving ML
- CNNs, LSTMs, MLPs, reinforcement learning, feature extraction, decision trees, etc.
- Inference-optimized numerical precision
- Exploit sparsity, deep compression for larger, faster models

Scale

- Microsoft has the world's largest cloud investment in FPGAs
- Multiple Exa-Ops of aggregate AI capacity
- BrainWave runs on Microsoft's scale infrastructure

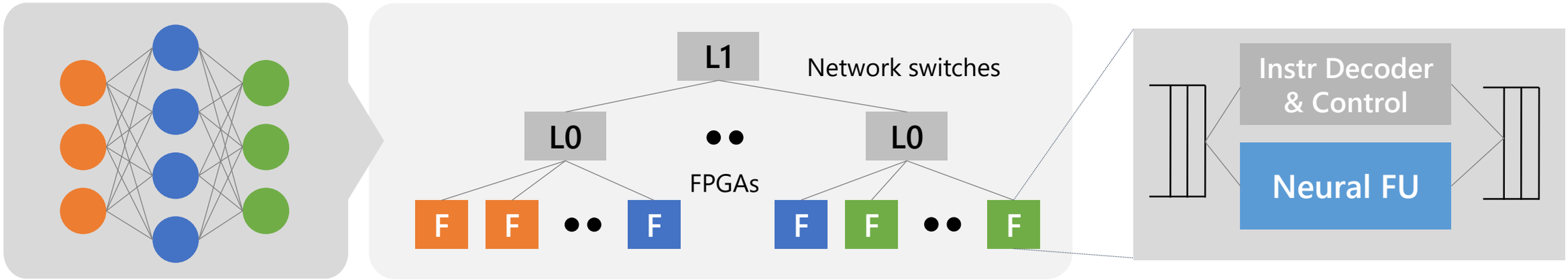
Project BrainWave

A Scalable FPGA-powered DNN Serving Platform

Fast: ultra-low latency, high-throughput serving of DNN models at low batch sizes

Flexible: adaptive numerical precision and custom operators

Friendly: turnkey deployment of CNTK/Caffe/TF/etc

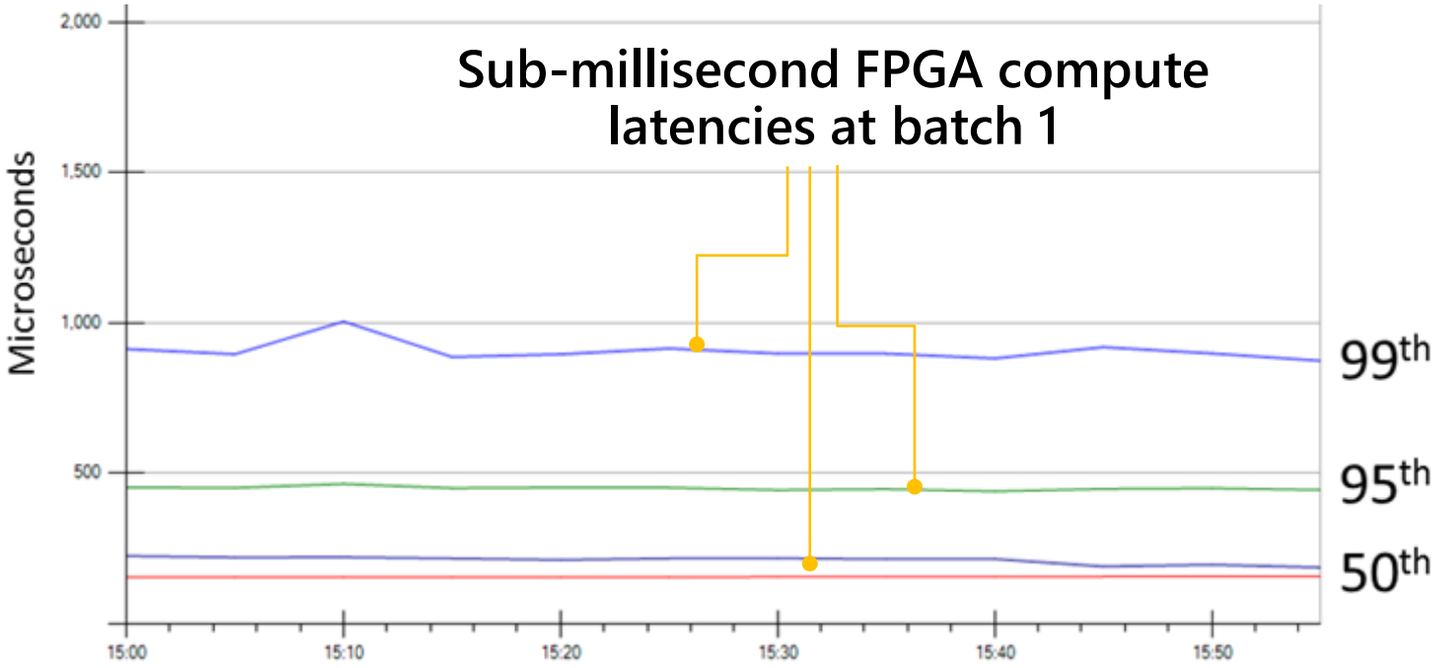


Pretrained DNN Model
in CNTK, etc.

Scalable DNN Hardware
Microservice

BrainWave
Soft NPU

Deployed in Production Datacenters

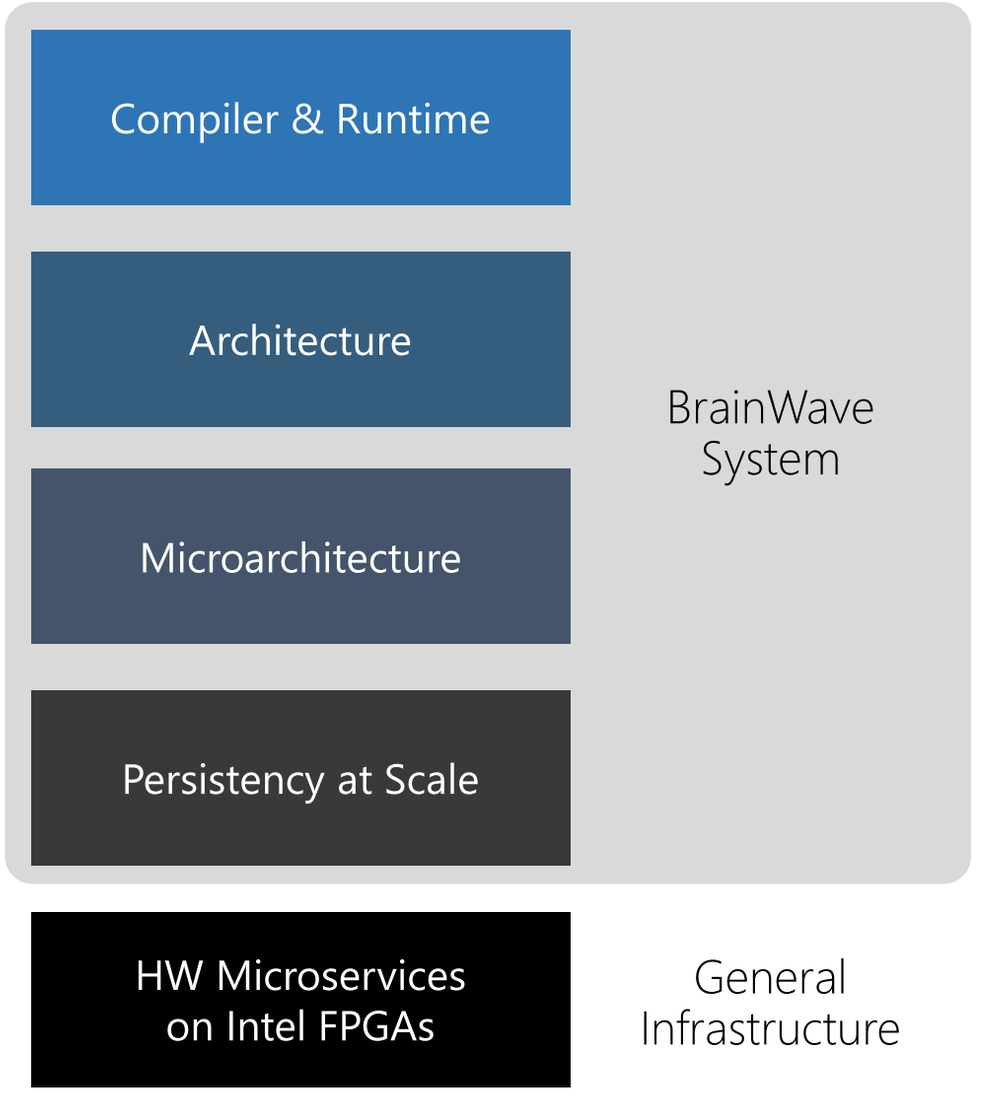


Deployment of LSTM-based NLP model (tens of millions of parameters)

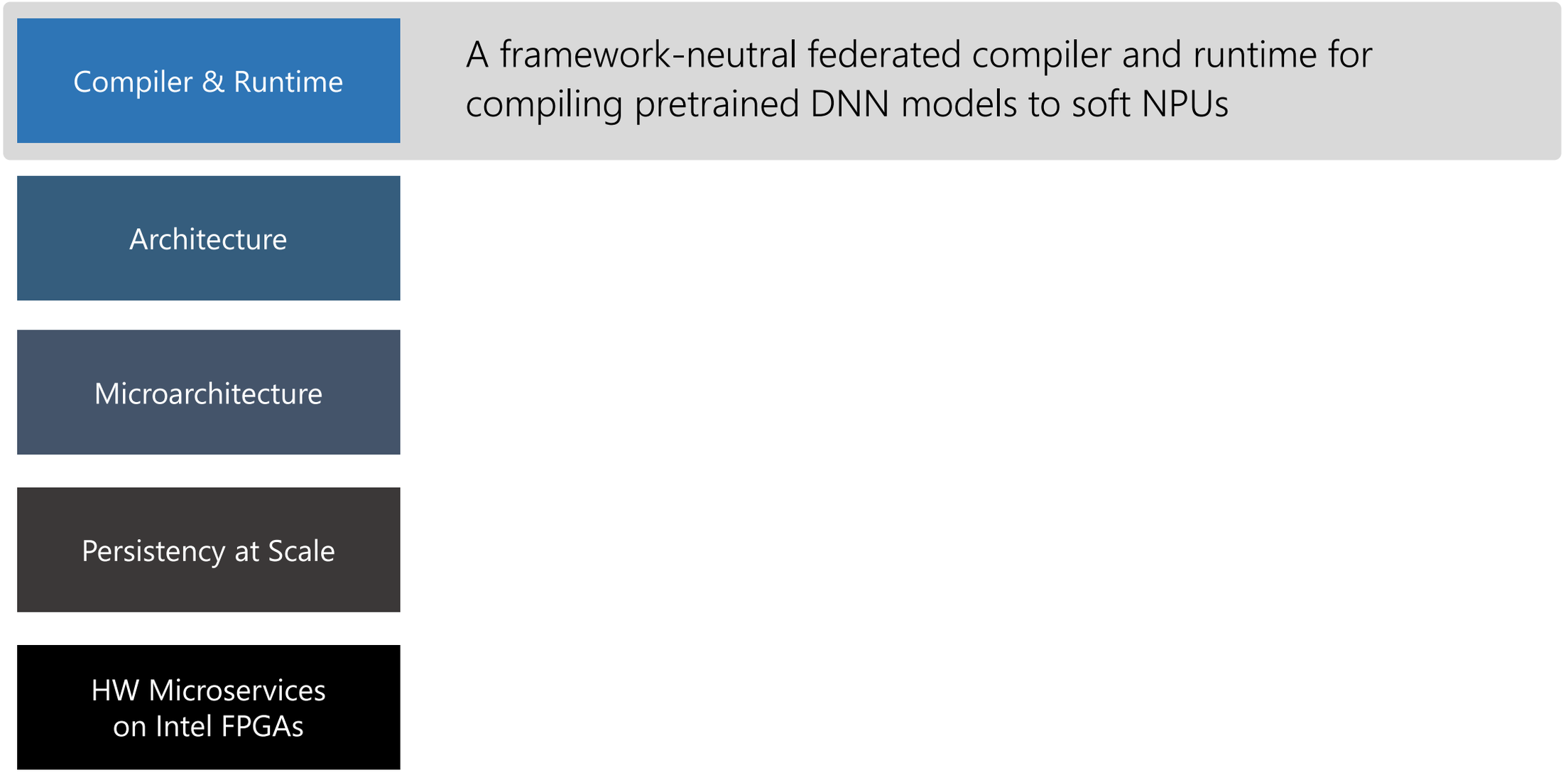
Takes tens of milliseconds to serve on well-tuned CPU implementations

Tail latencies in BrainWave-powered DNN models appear negligible in E2E software pipelines

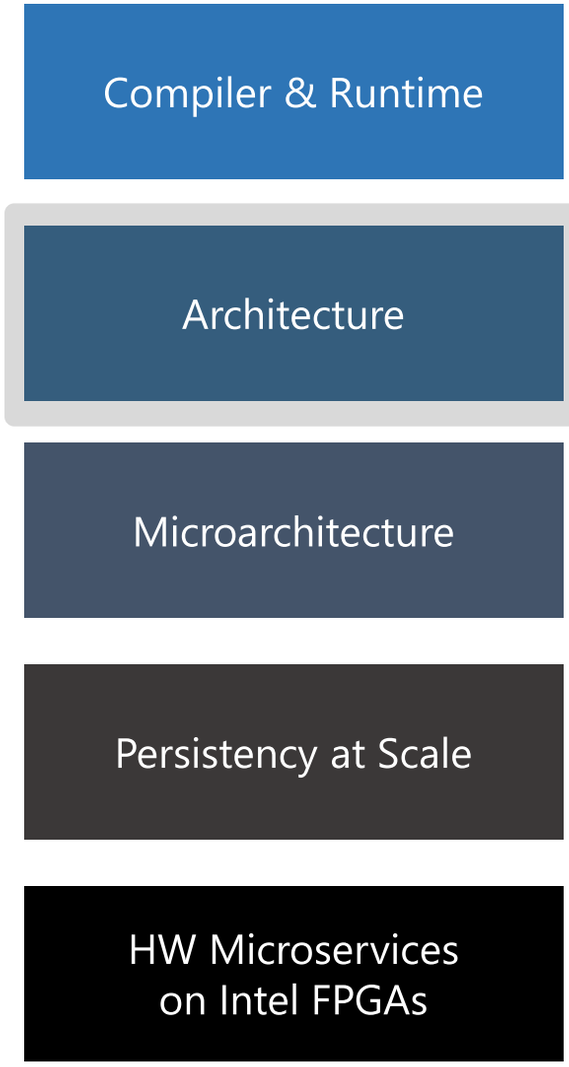
How It Works: The BrainWave Stack



How It Works: The BrainWave Stack



How It Works: The BrainWave Stack



A framework-neutral federated compiler and runtime for compiling pretrained DNN models to soft NPUs

Adaptive ISA for narrow precision DNN inference
Flexible and extensible to support fast-changing AI algorithms

How It Works: The BrainWave Stack

Compiler & Runtime	A framework-neutral federated compiler and runtime for compiling pretrained DNN models to soft NPUs
Architecture	Adaptive ISA for narrow precision DNN inference Flexible and extensible to support fast-changing AI algorithms
Microarchitecture	BrainWave Soft NPU microarchitecture Highly optimized for narrow precision and low batch
Persistency at Scale	
HW Microservices on Intel FPGAs	

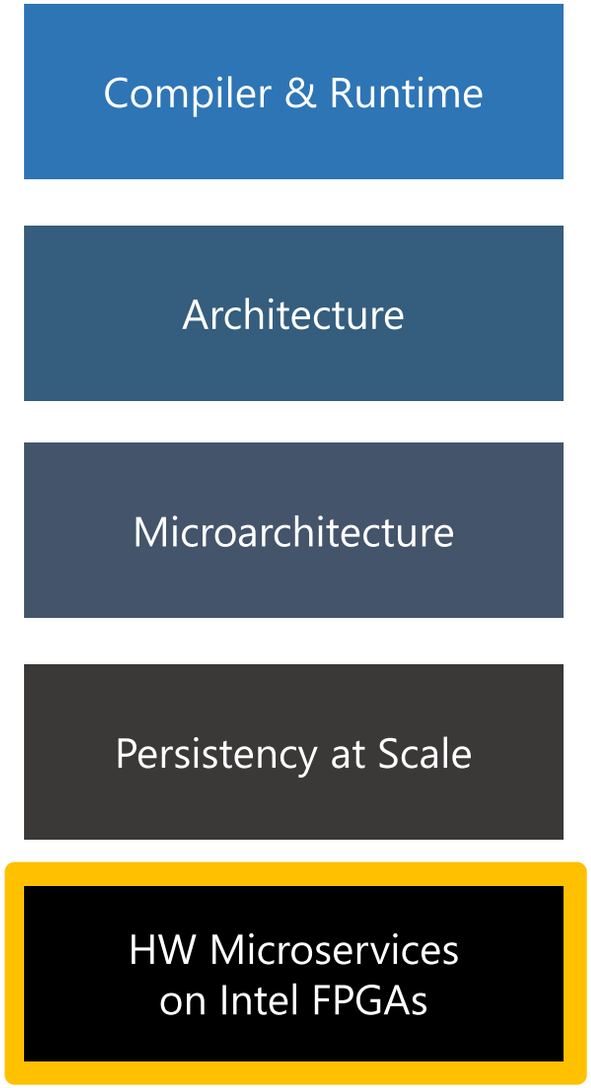
How It Works: The BrainWave Stack

Compiler & Runtime	A framework-neutral federated compiler and runtime for compiling pretrained DNN models to soft NPUs
Architecture	Adaptive ISA for narrow precision DNN inference Flexible and extensible to support fast-changing AI algorithms
Microarchitecture	BrainWave Soft NPU microarchitecture Highly optimized for narrow precision and low batch
Persistency at Scale	Persist model parameters entirely in FPGA on-chip memories Support large models by scaling across many FPGAs
HW Microservices on Intel FPGAs	

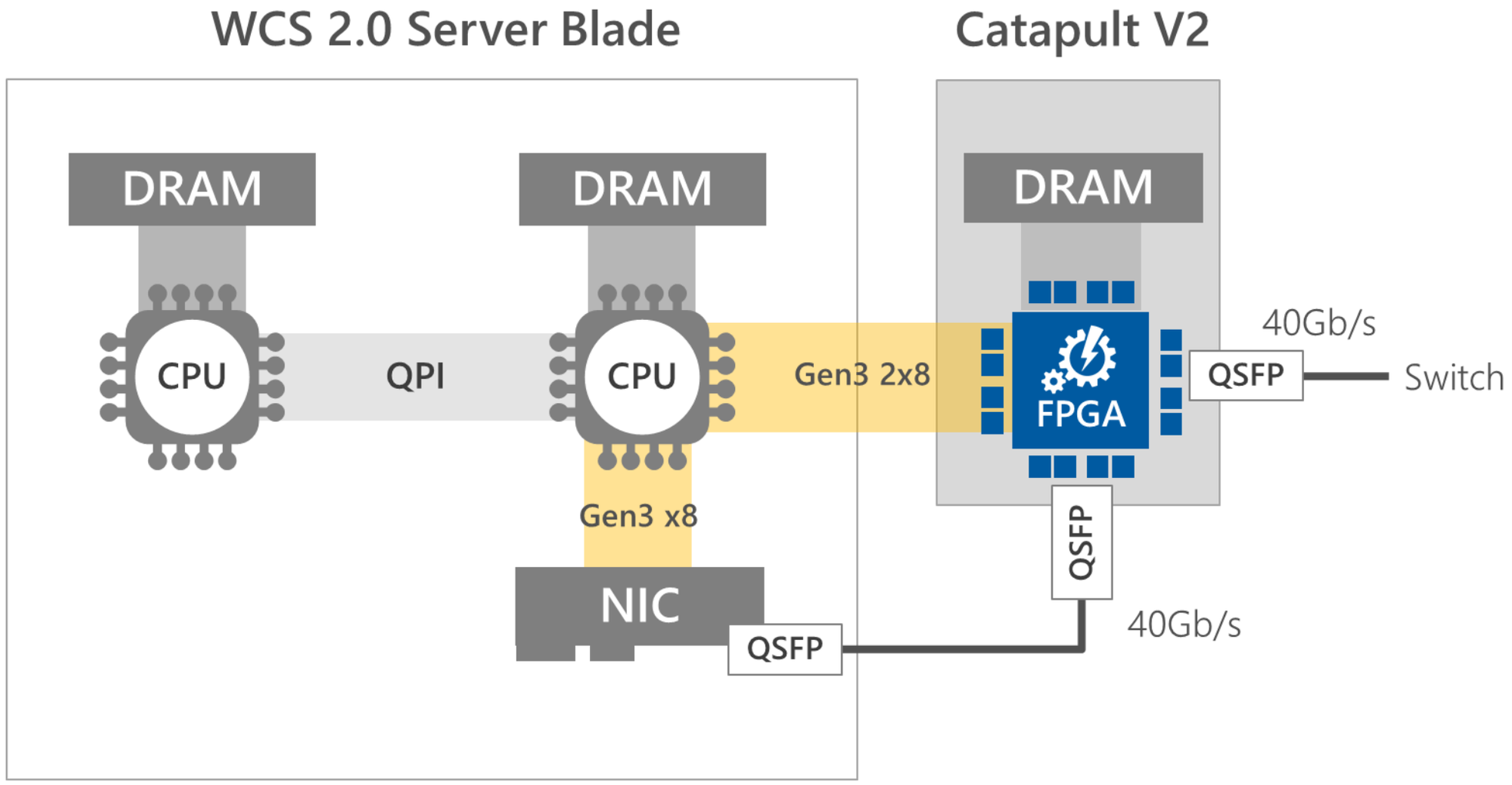
How It Works: The BrainWave Stack

Compiler & Runtime	A framework-neutral federated compiler and runtime for compiling pretrained DNN models to soft NPUs
Architecture	Adaptive ISA for narrow precision DNN inference Flexible and extensible to support fast-changing AI algorithms
Microarchitecture	BrainWave Soft NPU microarchitecture Highly optimized for narrow precision and low batch
Persistency at Scale	Persist model parameters entirely in FPGA on-chip memories Support large models by scaling across many FPGAs
HW Microservices on Intel FPGAs	Intel FPGAs deployed at scale with HW microservices [MICRO'16]

The BrainWave Stack

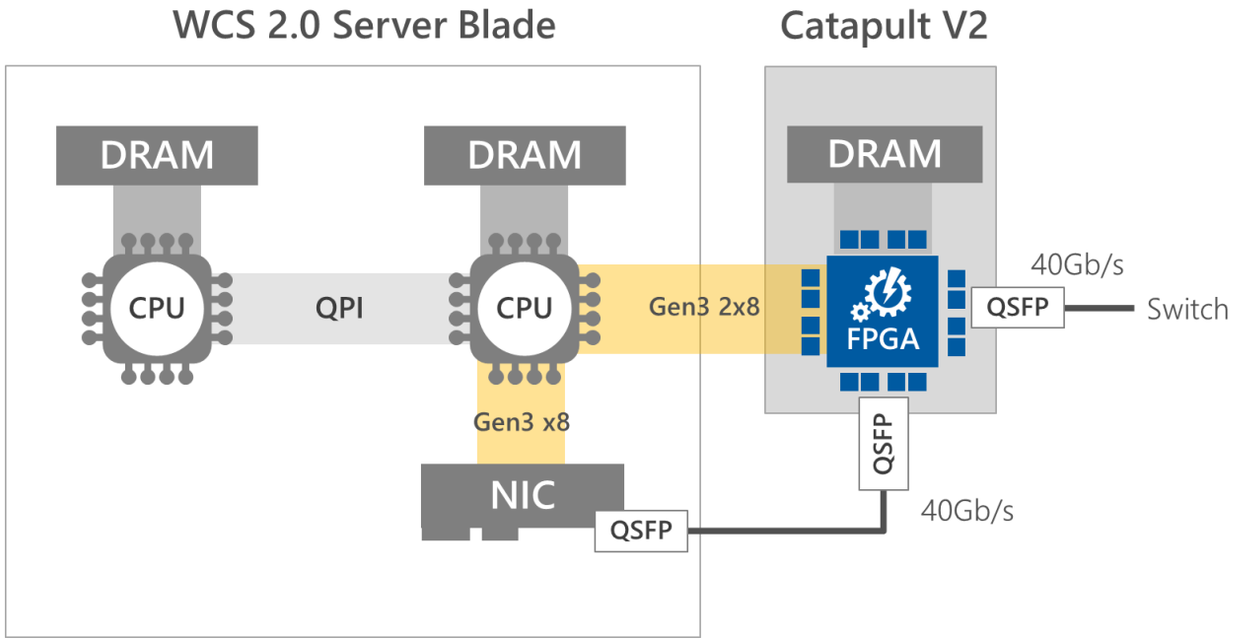


FPGAs Are Deployed in MSFT Servers Worldwide



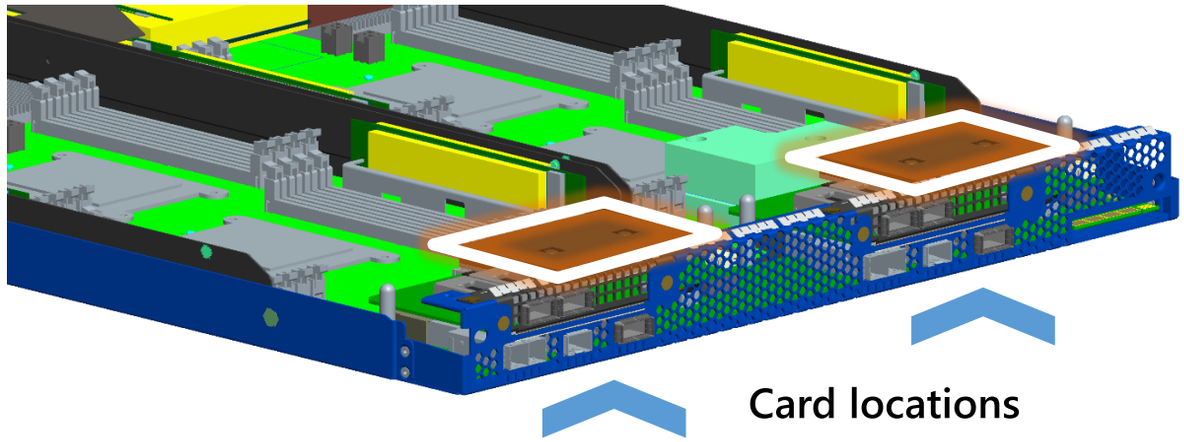
A Cloud-Scale Acceleration Architecture [MICRO'16]

FPGAs Are Deployed in MSFT Servers Worldwide



Catapult v2 Mezzanine card

WCS Gen4.1 Blade with NIC and Catapult FPGA

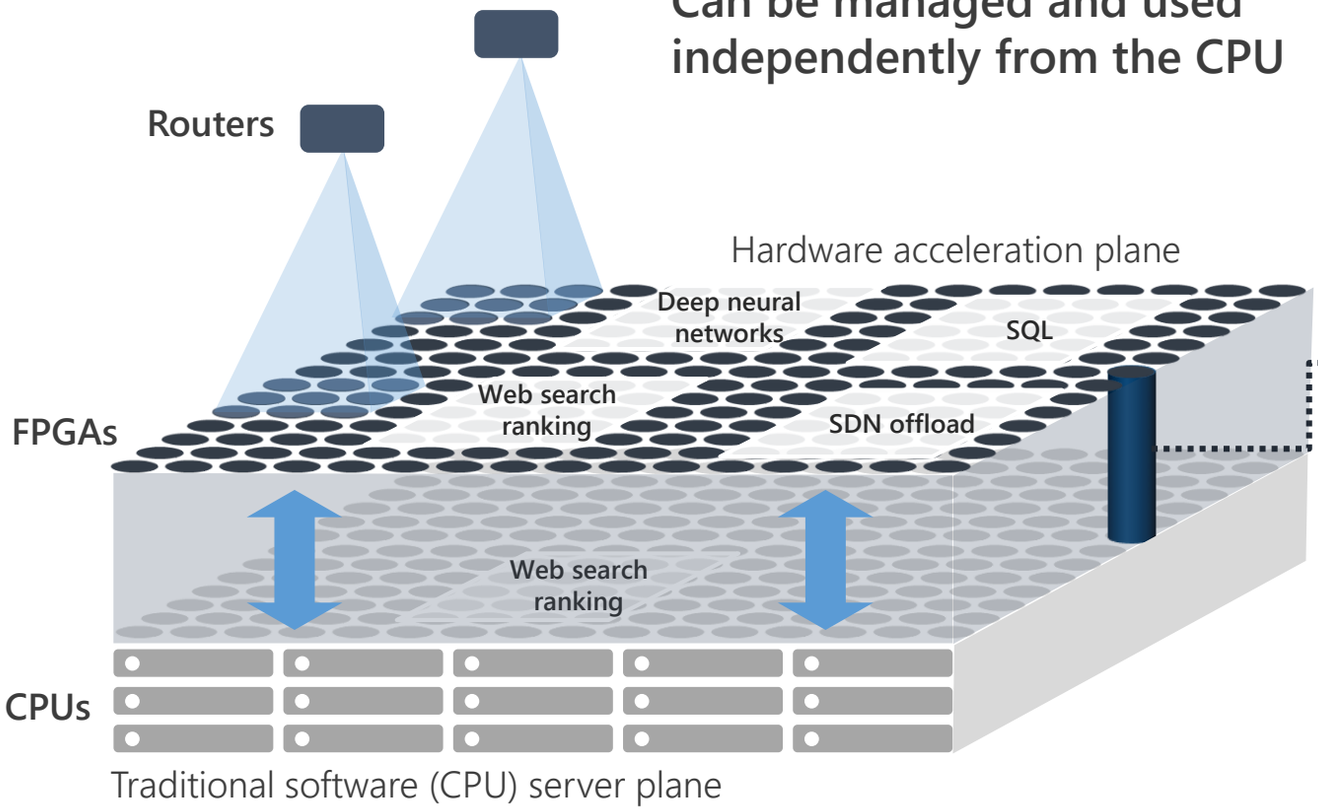


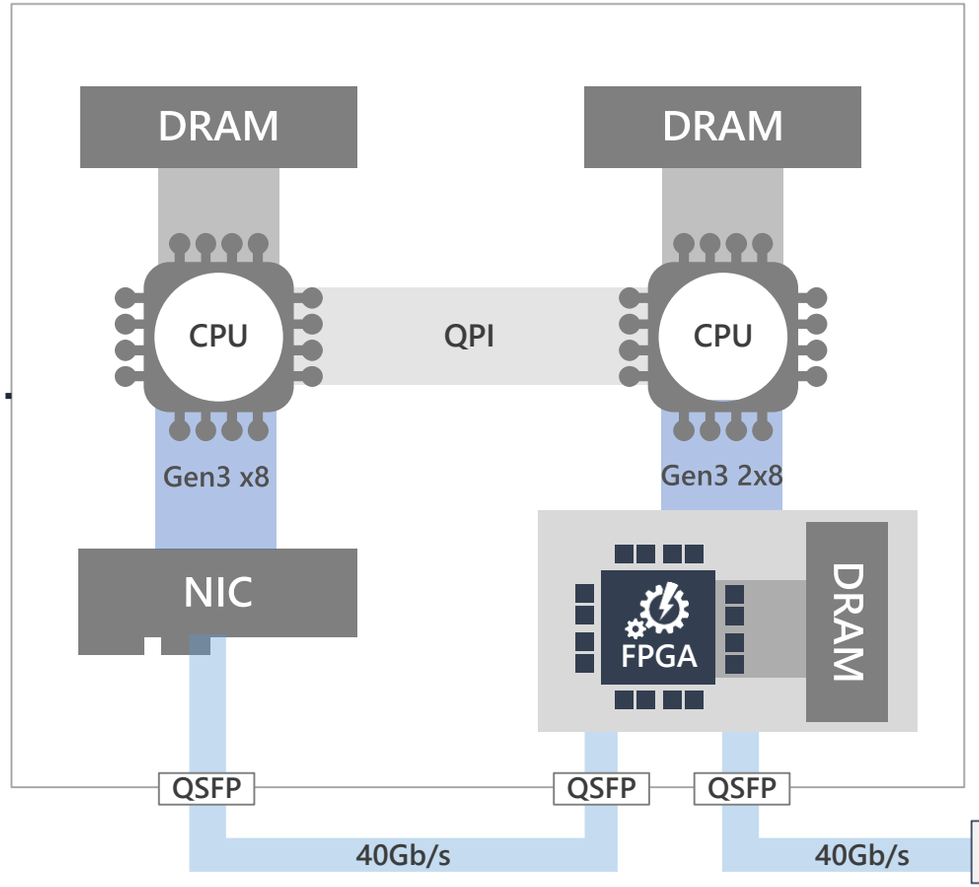
[ISCA'14, HotChips'14, MICRO'16]

Hardware Microservices on FPGAs [MICRO'16]

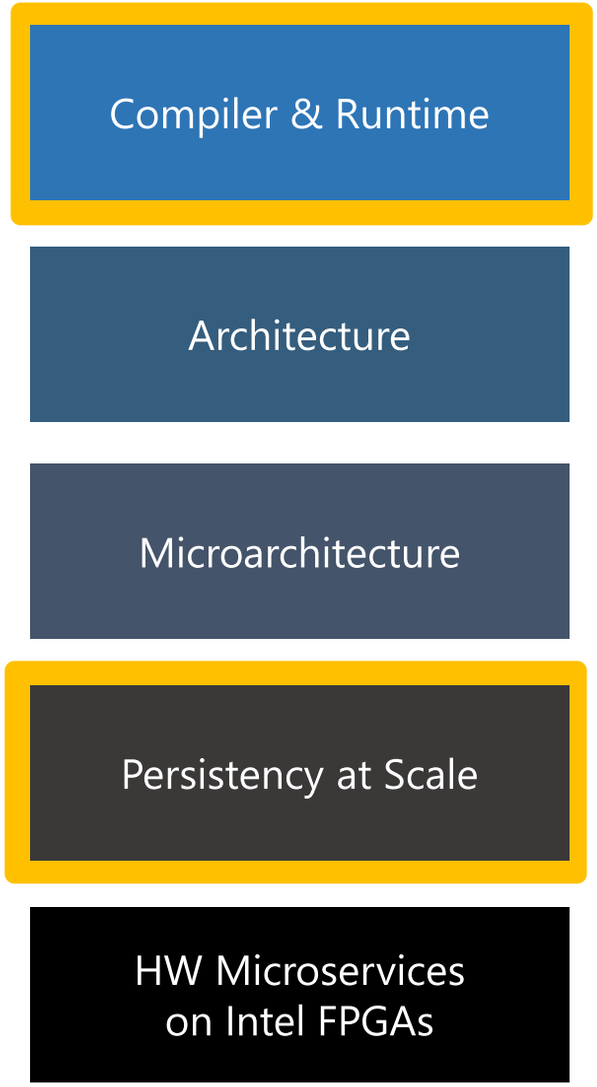
Interconnected FPGAs form a separate plane of computation

Can be managed and used independently from the CPU

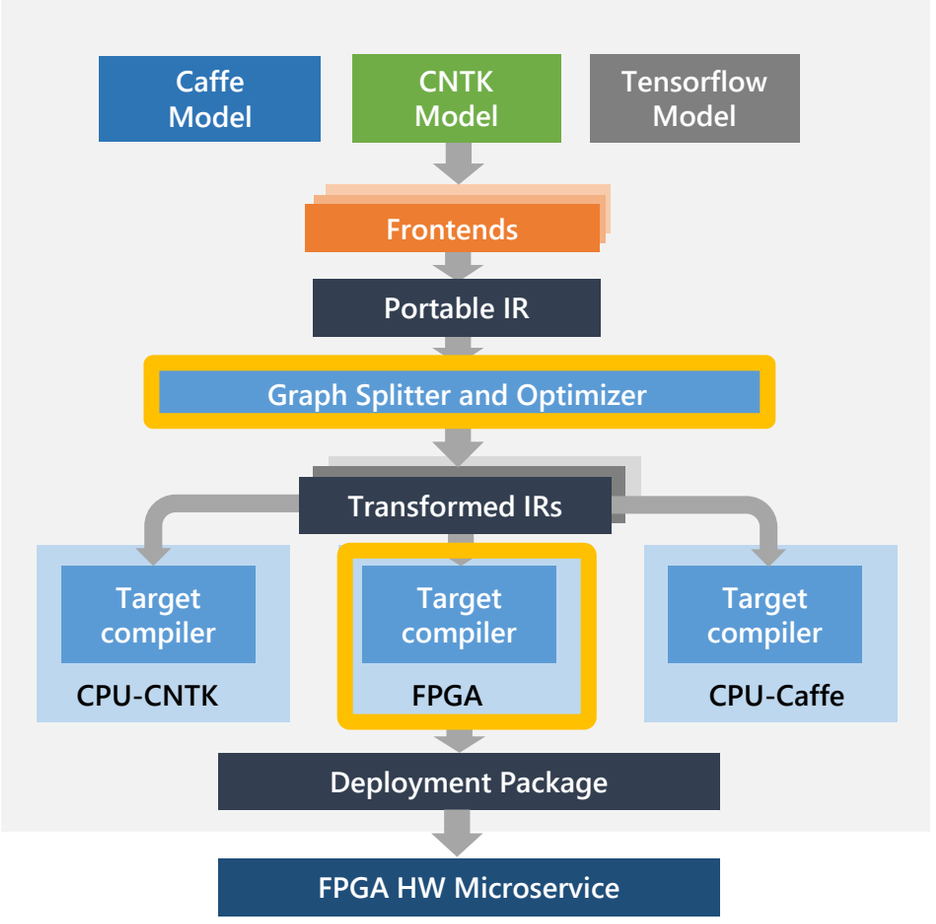


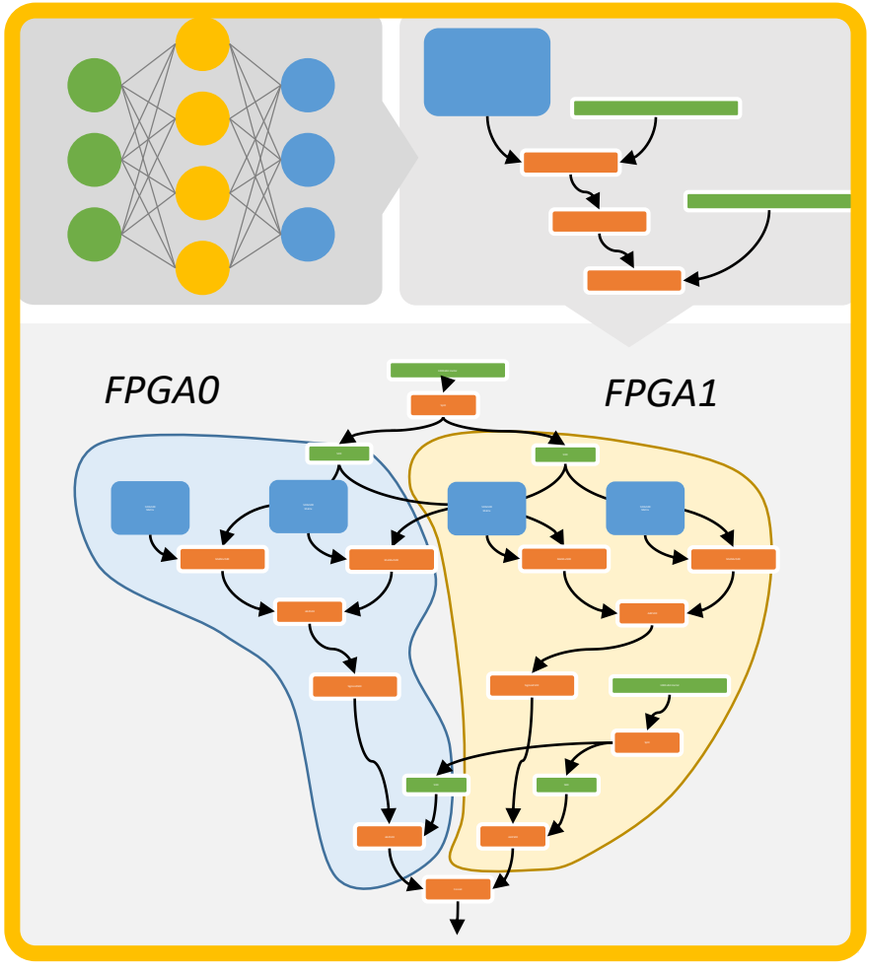


The BrainWave Stack

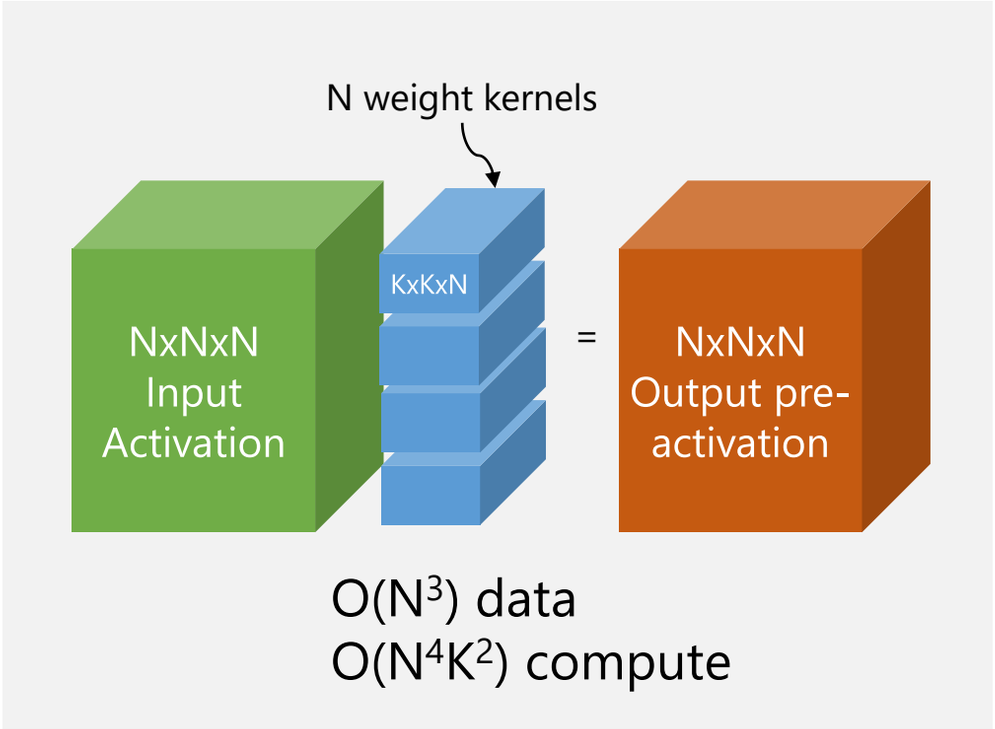


BrainWave Compiler & Runtime



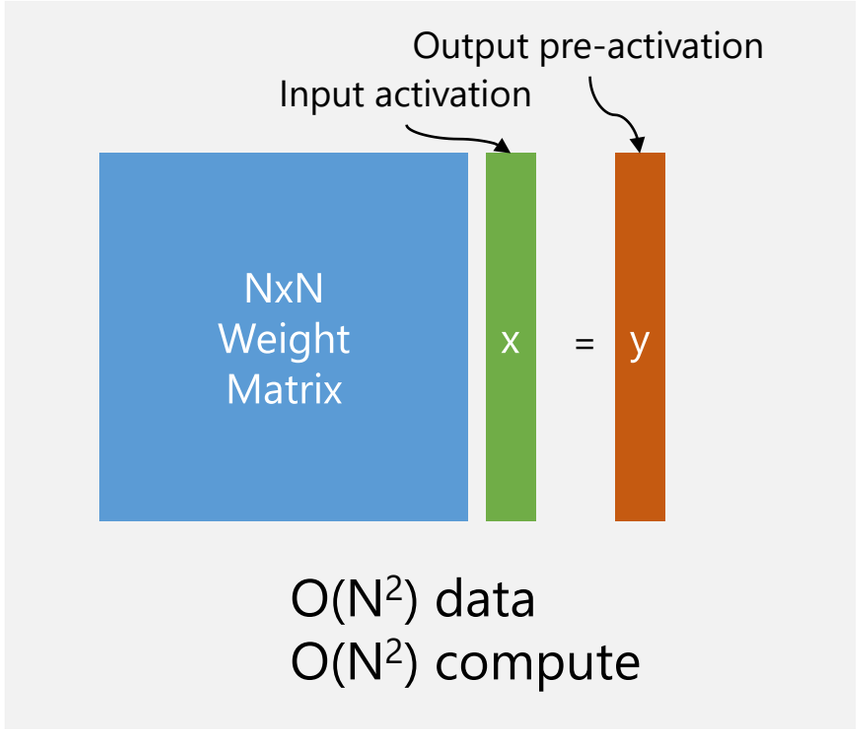


Common Scenarios



$O(N^3)$ data
$O(N^4K^2)$ compute

Convolutional Neural Network (CNN)
High Compute-to-Data Ratio



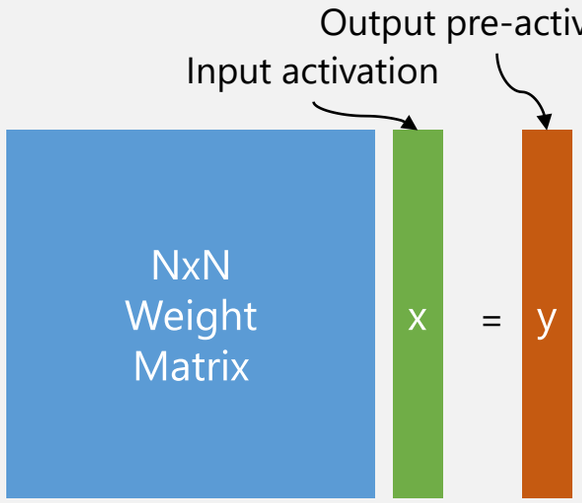
$O(N^2)$ data
$O(N^2)$ compute

MLPs, LSTMs, GRUs
Low compute-to-data ratio

Common Scenarios

$O(N^3)$ data
$O(N^4K^2)$ compute

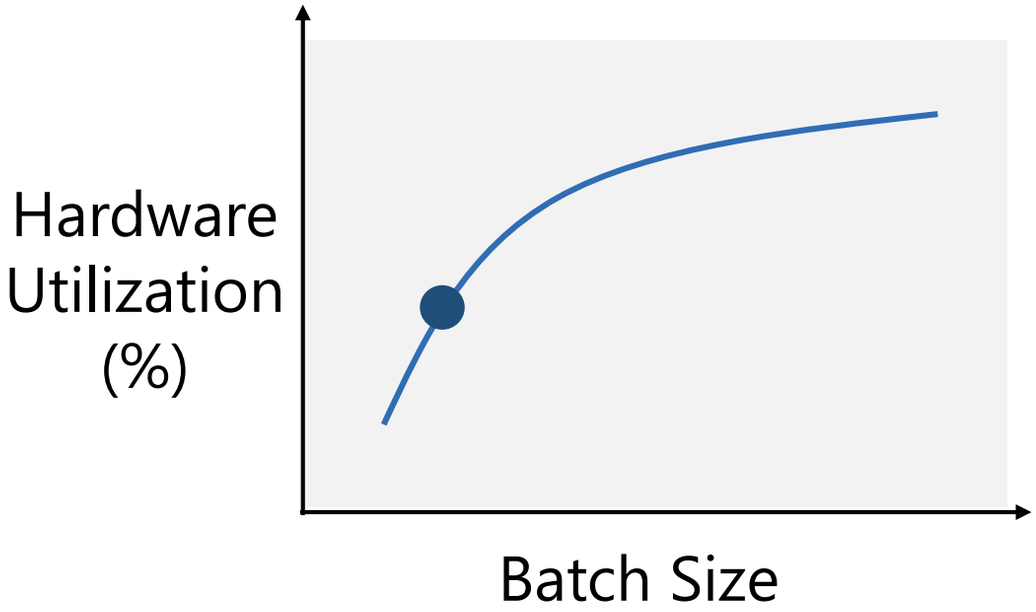
Convolutional Neural Network (CNN)
High Compute-to-Data Ratio

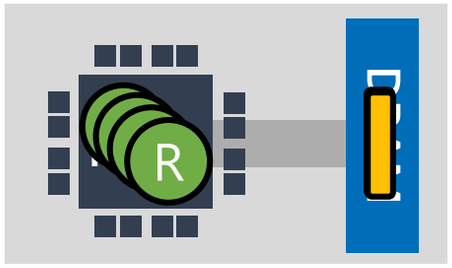


$O(N^2)$ data
$O(N^2)$ compute

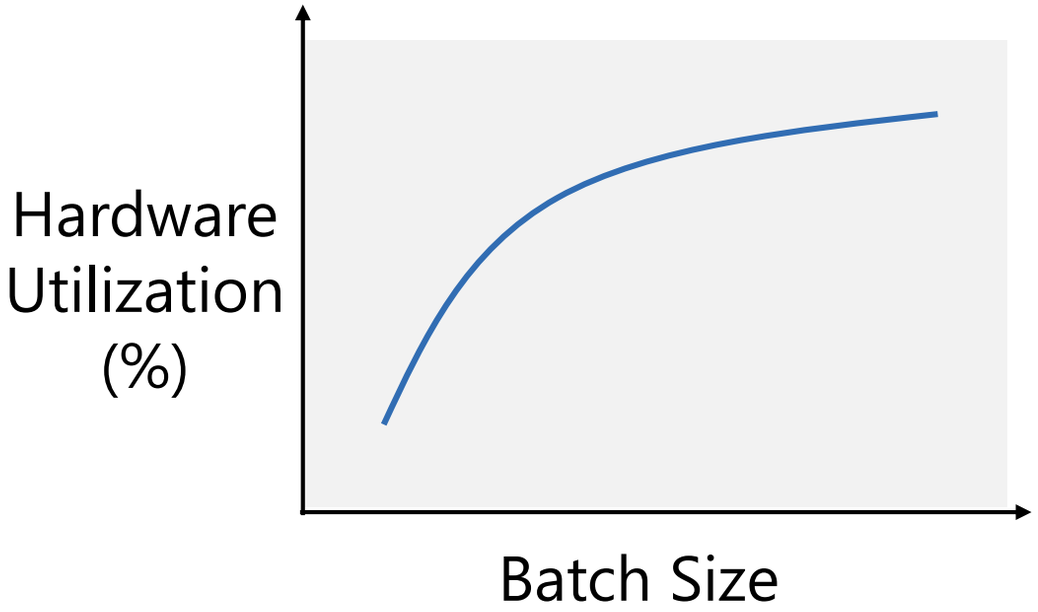
MLPs, LSTMs, GRUs
Low compute-to-data ratio

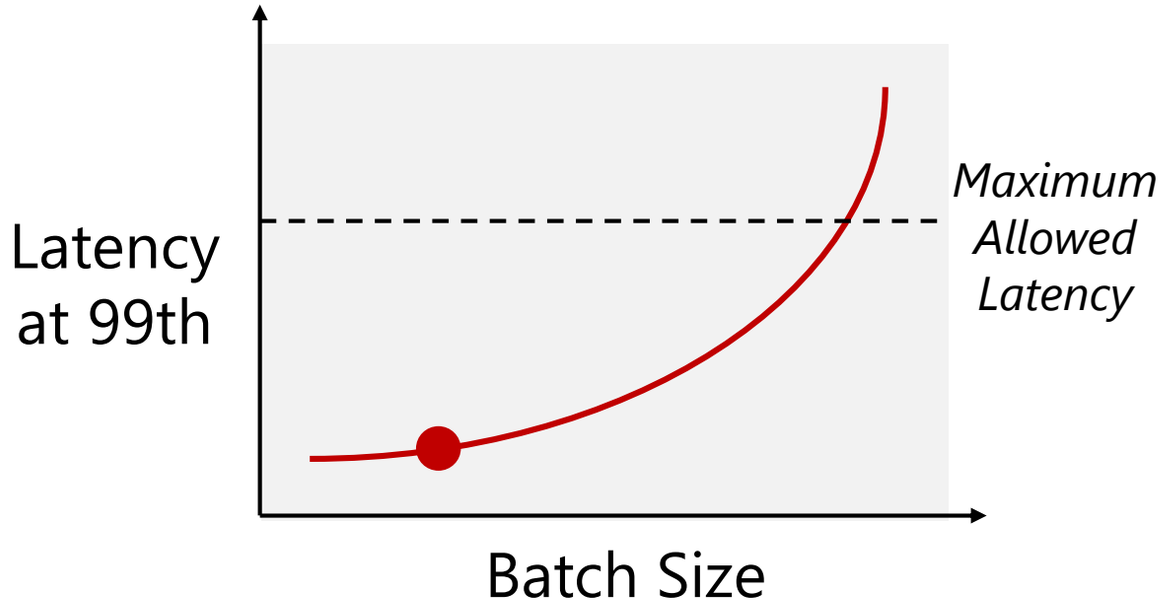
Improving HW utilization with batching





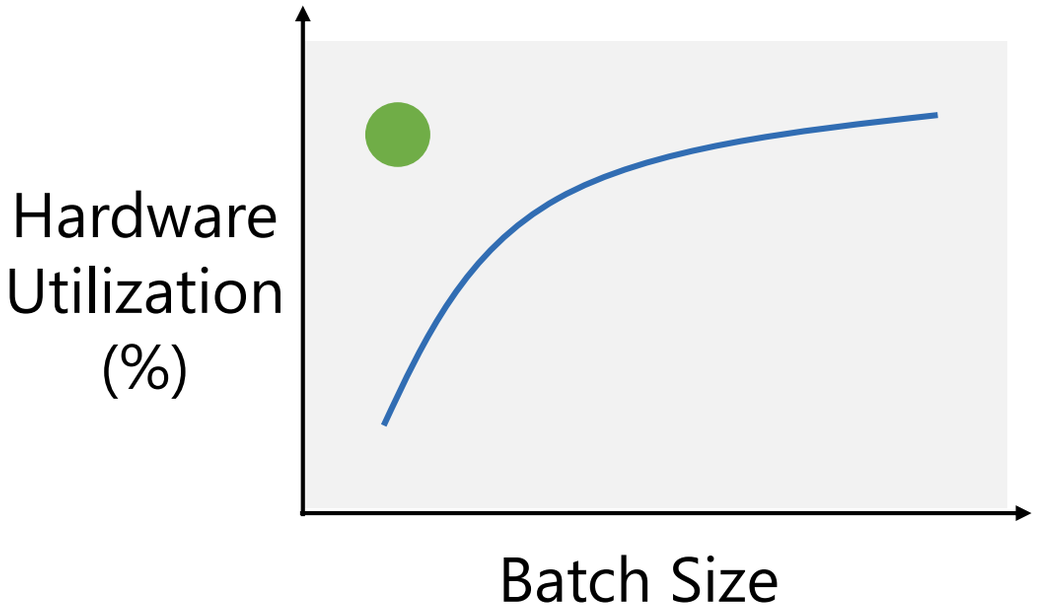
Improving HW utilization with batching

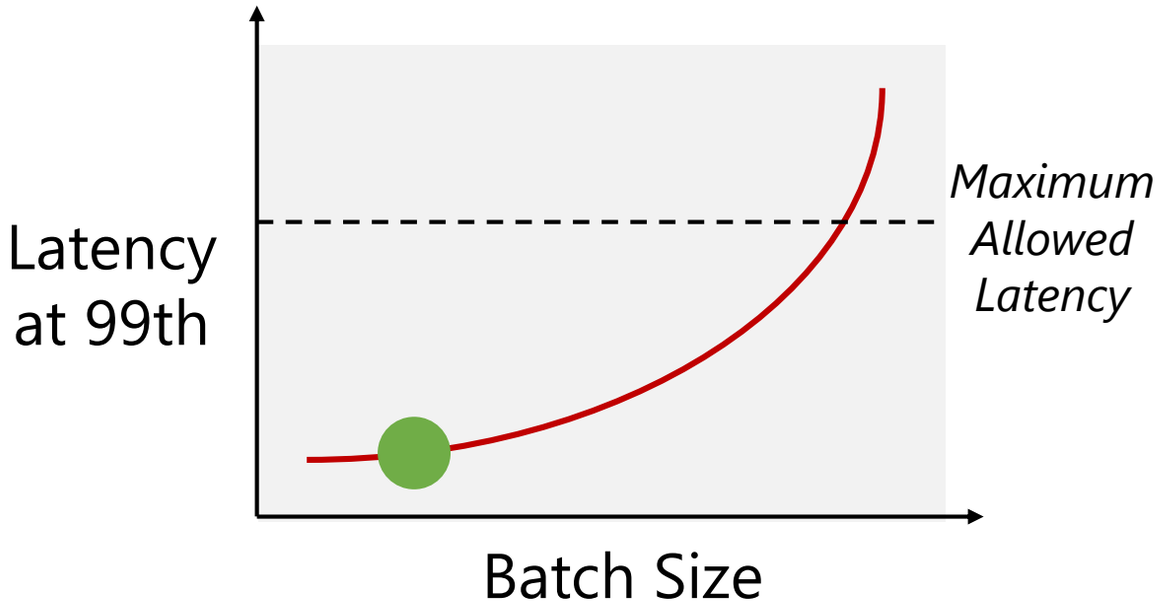




Batching improves HW utilization but increases latency

Improving HW utilization with batching

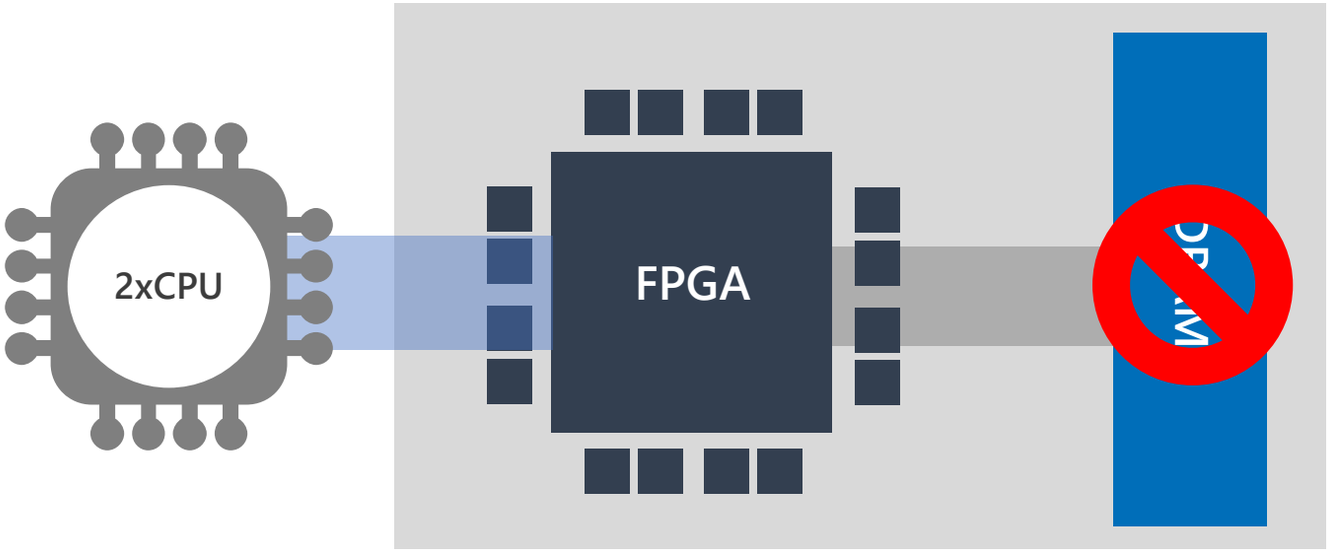




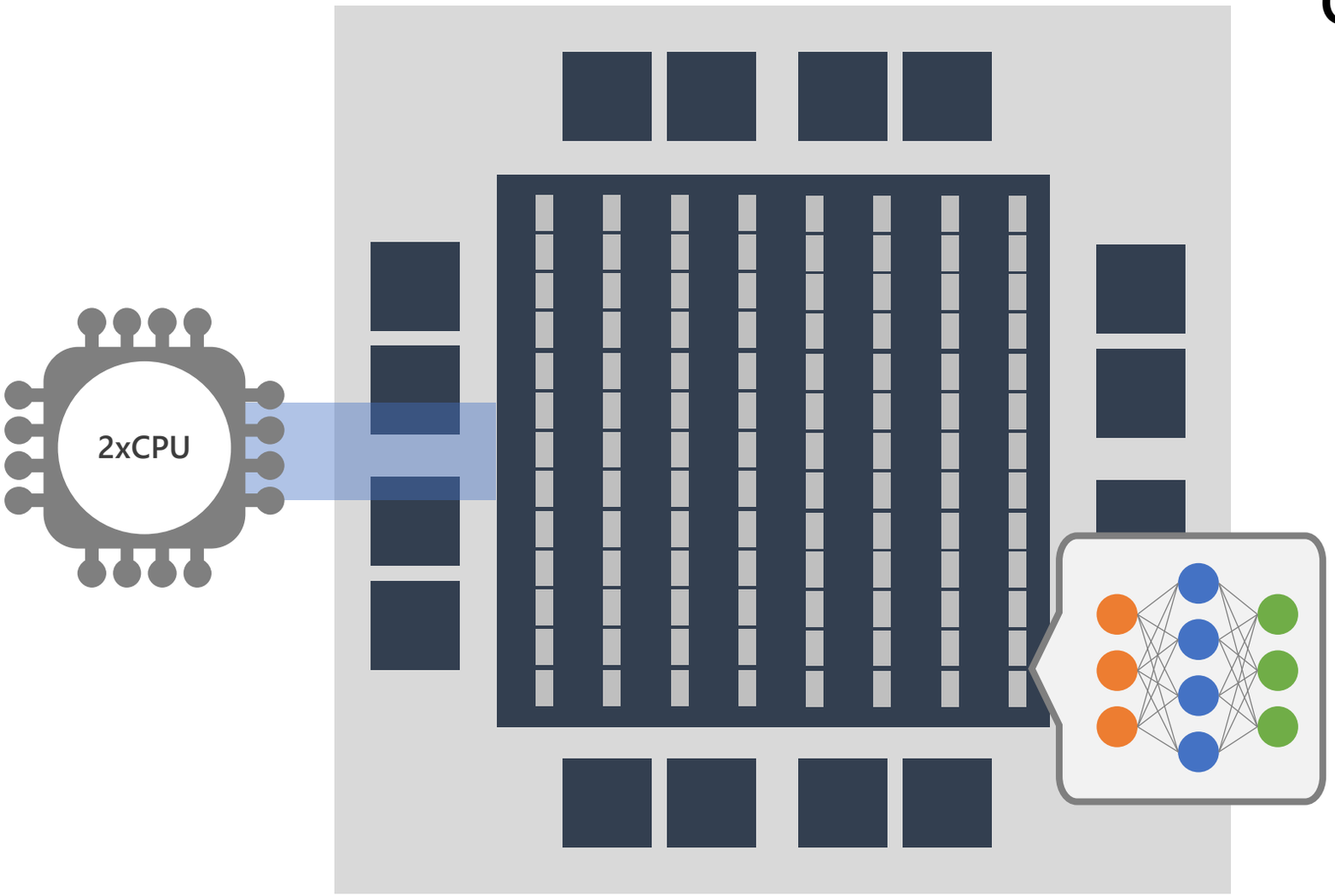
Batching improves HW utilization but increases latency

Ideally want high HW utilization at low batch sizes

Alternative: “Persistent” Neural Nets



Alternative: "Persistent" Neural Nets



Observations

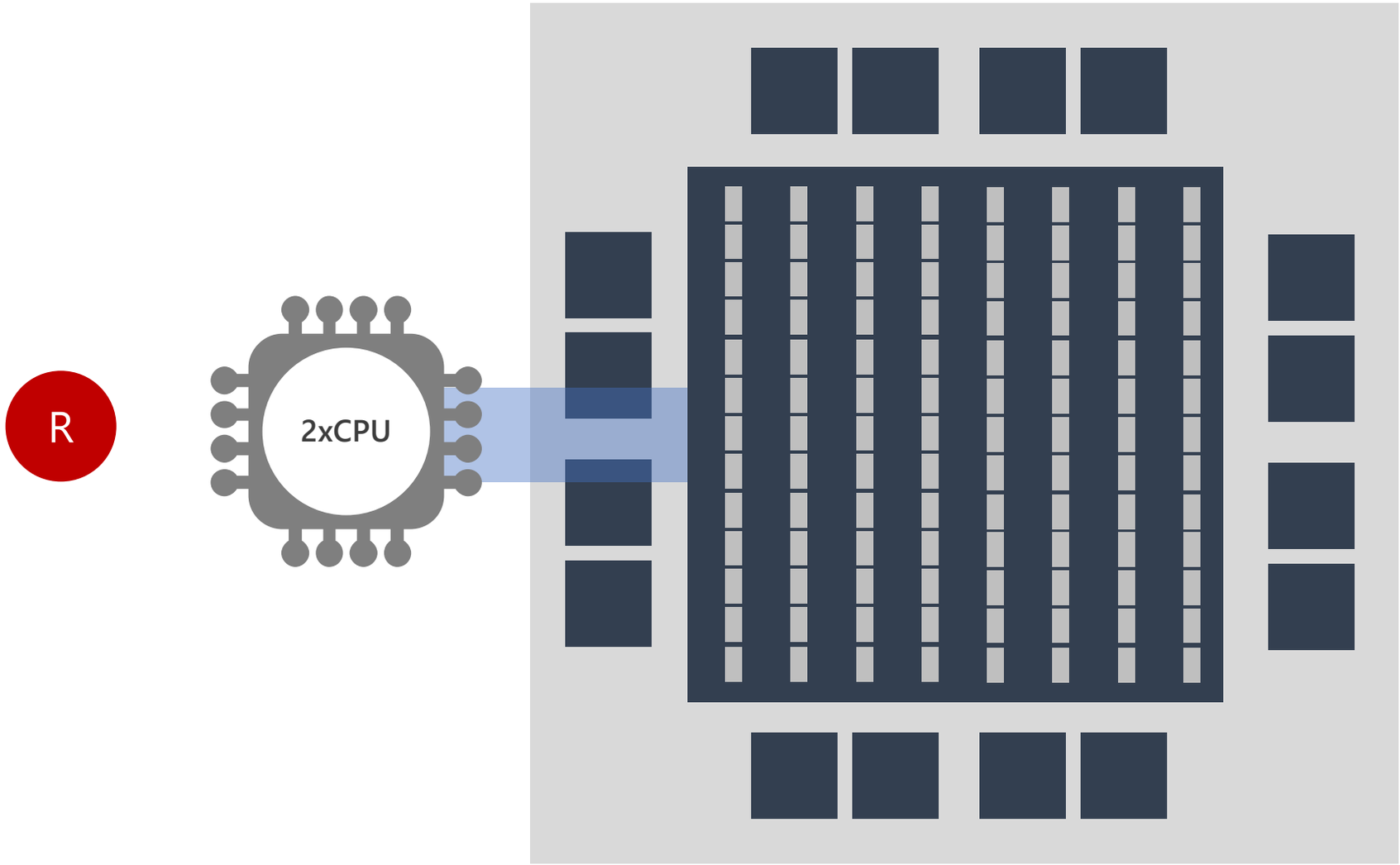
State-of-art FPGAs have O(10K) distributed Block RAMs O(10MB)

➔ Tens of TB/sec of memory BW

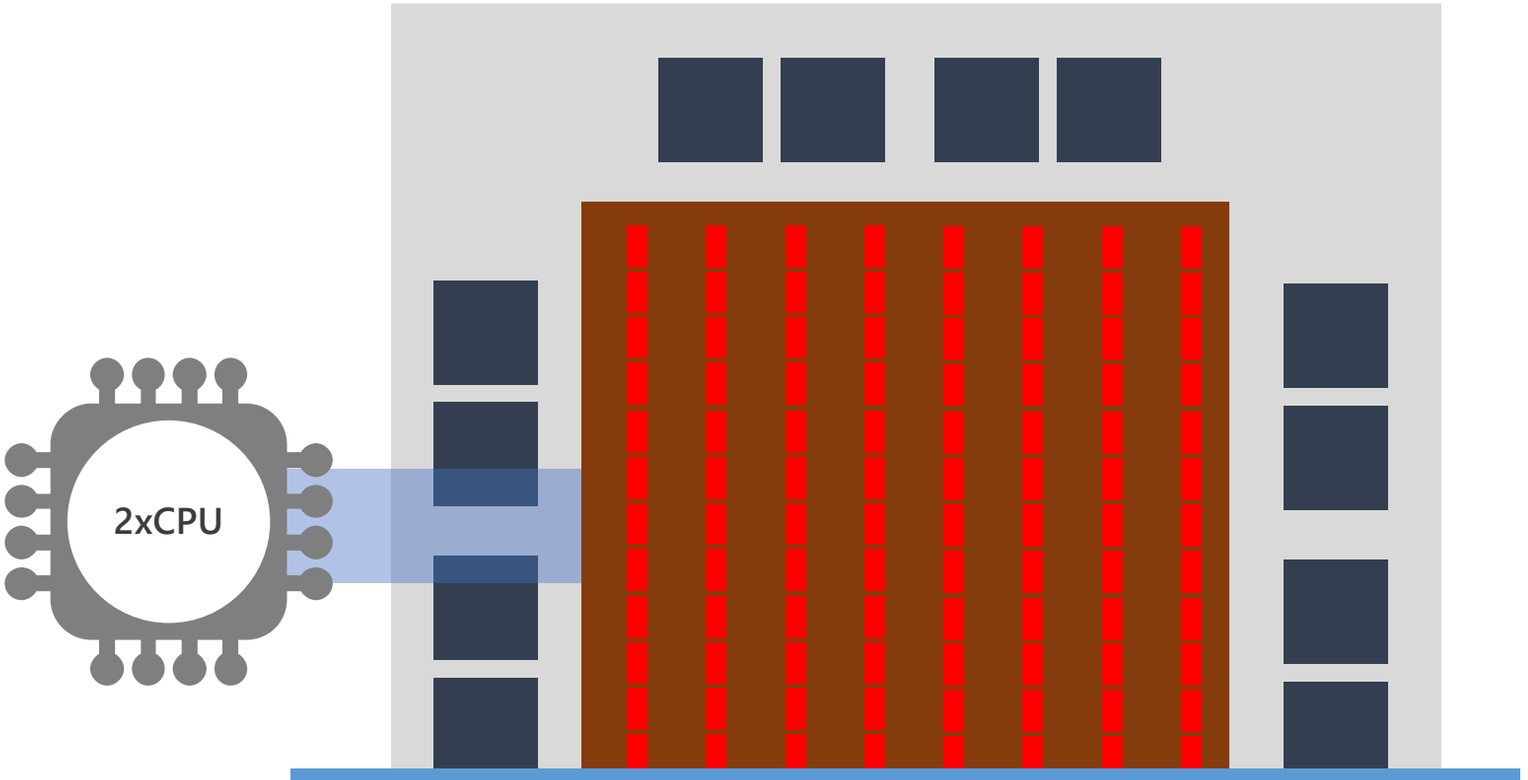
Large-scale cloud services and DNN models run persistently

Solution: persist all model parameters in FPGA on-chip memory during service lifetime

Alternative: “Persistent” Neural Nets



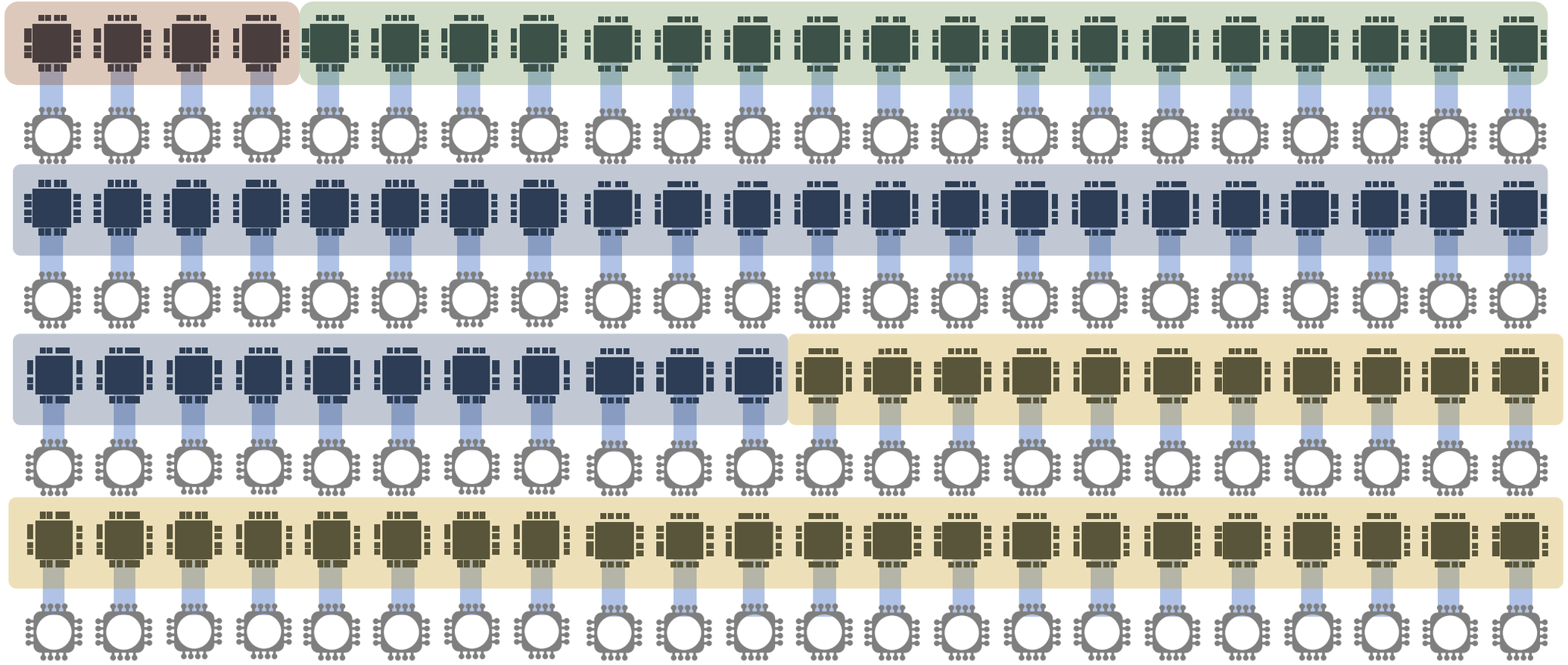
Alternative: “Persistent” Neural Nets



When single request arrives, all chip resources (on-chip memories and compute units) are used to process a single query (no batching required)

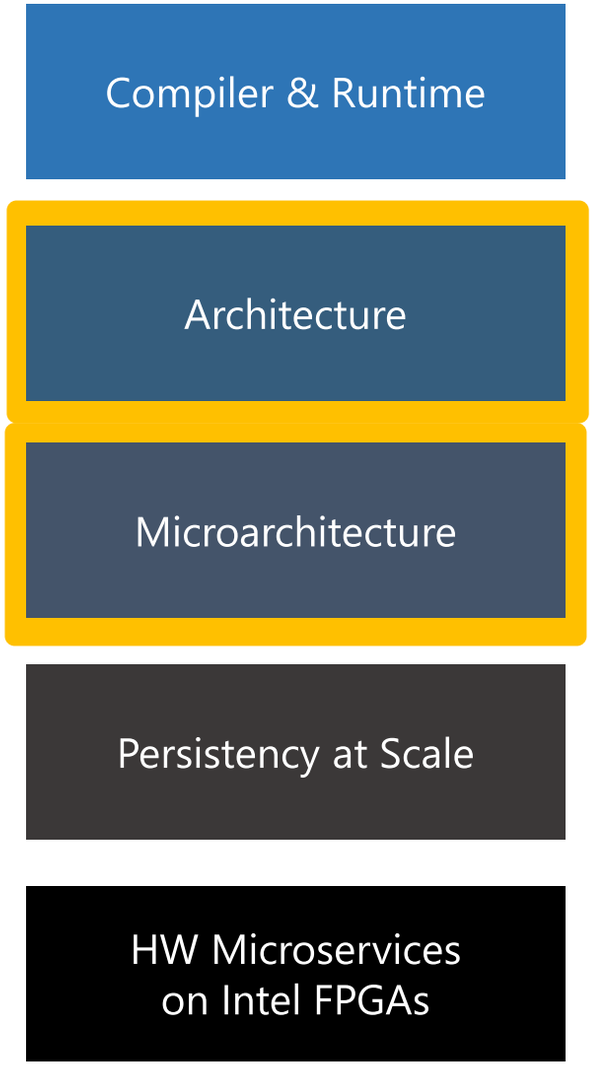
What if model doesn’t fit in single FPGA?

Solution: Persistency at Datacenter Scale



Multiple FPGAs at datacenter scale can form a persistent DNN HW microservice, enabling scale-out of models at ultra-low latencies

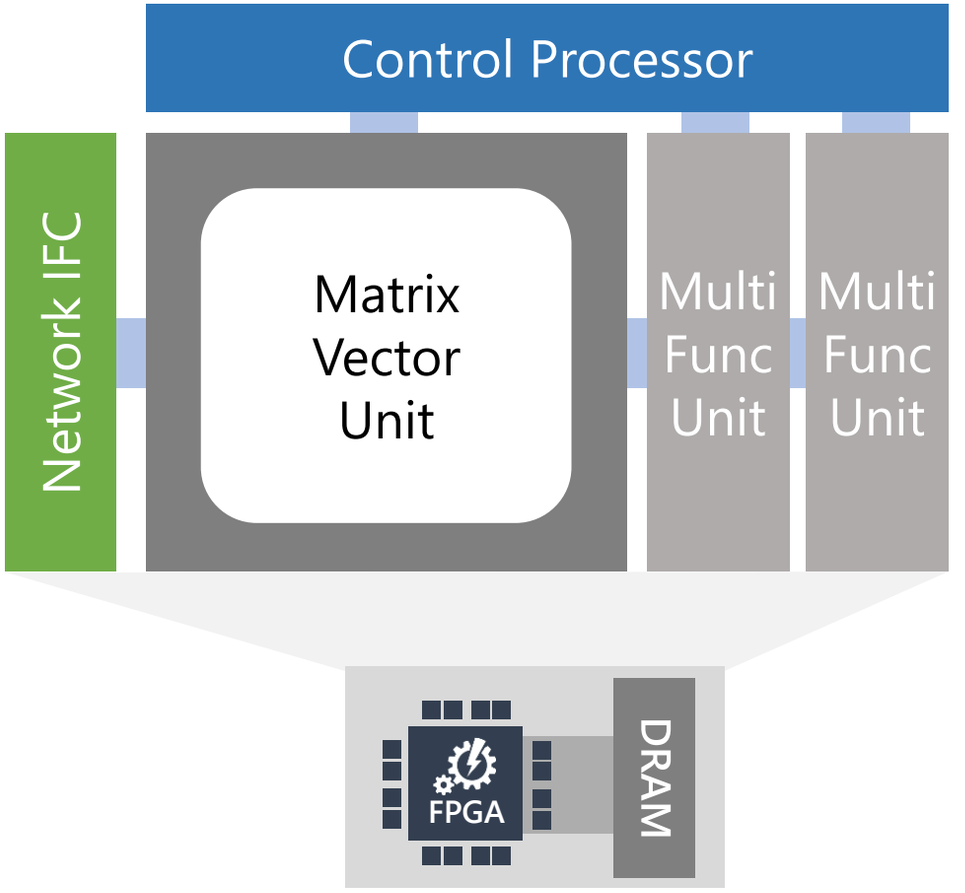
The BrainWave Stack



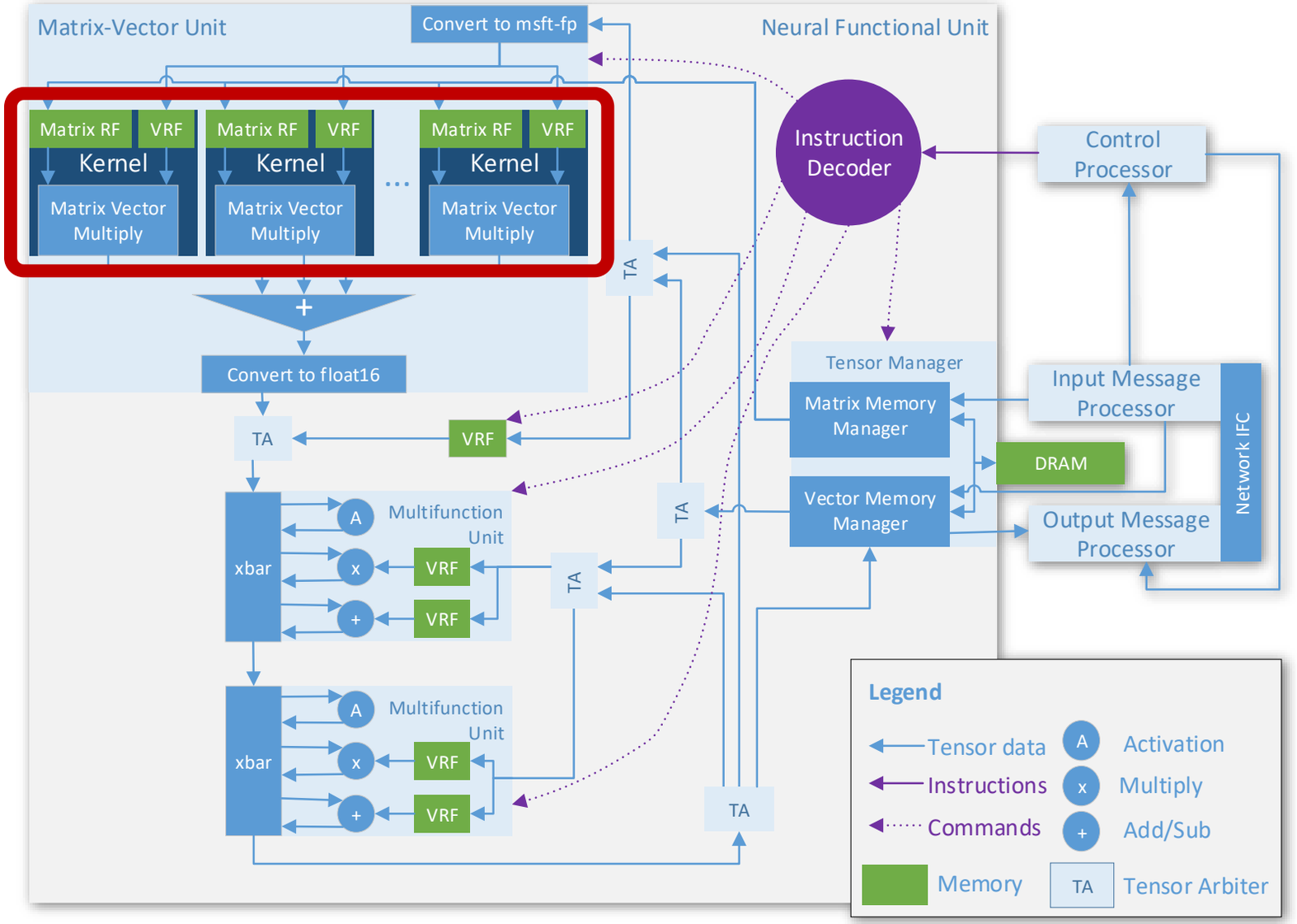
BrainWave Soft NPU Architecture

Core Features

- Single-threaded C programming model (no RTL)
- ISA with specialized instructions: dense matmul, convolutions, non-linear activations, vector operations, embeddings
- Proprietary parameterizable narrow precision format wrapped in float16 interfaces
- Parameterizable microarchitecture and scalable to large FPGAs (~1M ALMs)
- Fully integrated with HW microservices (network-attached)
- P2P protocol to CPU hosts and FPGAs
- Easy to extend ISA with custom operators



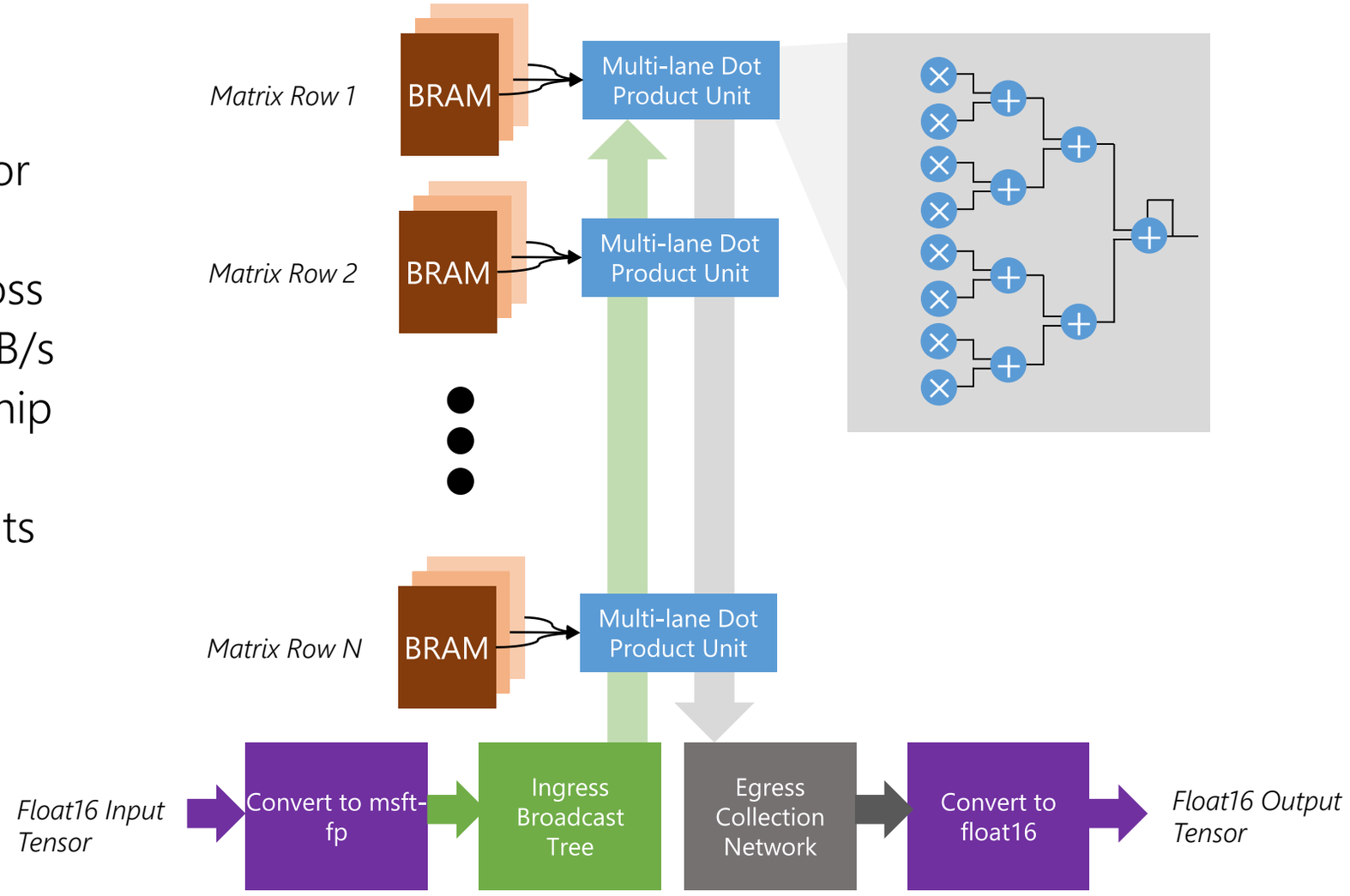
BrainWave Soft NPU Microarchitecture



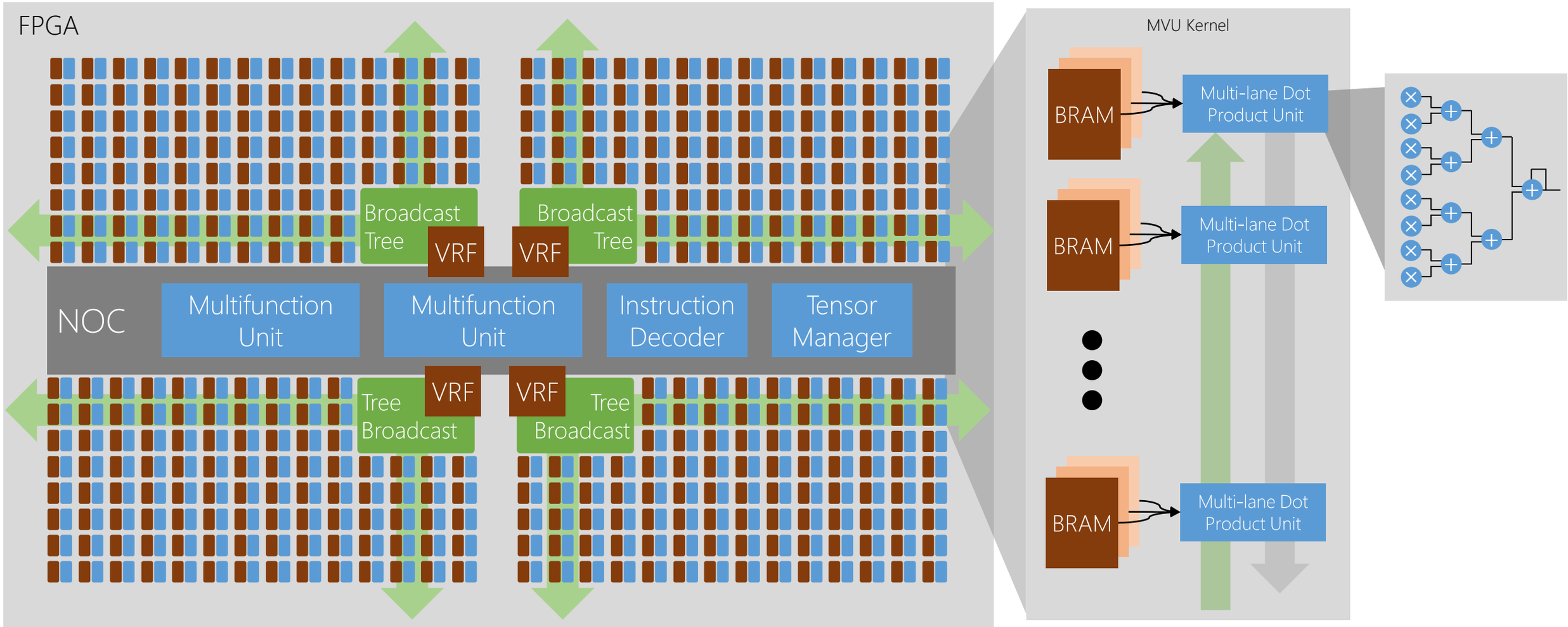
Matrix Vector Unit

Features

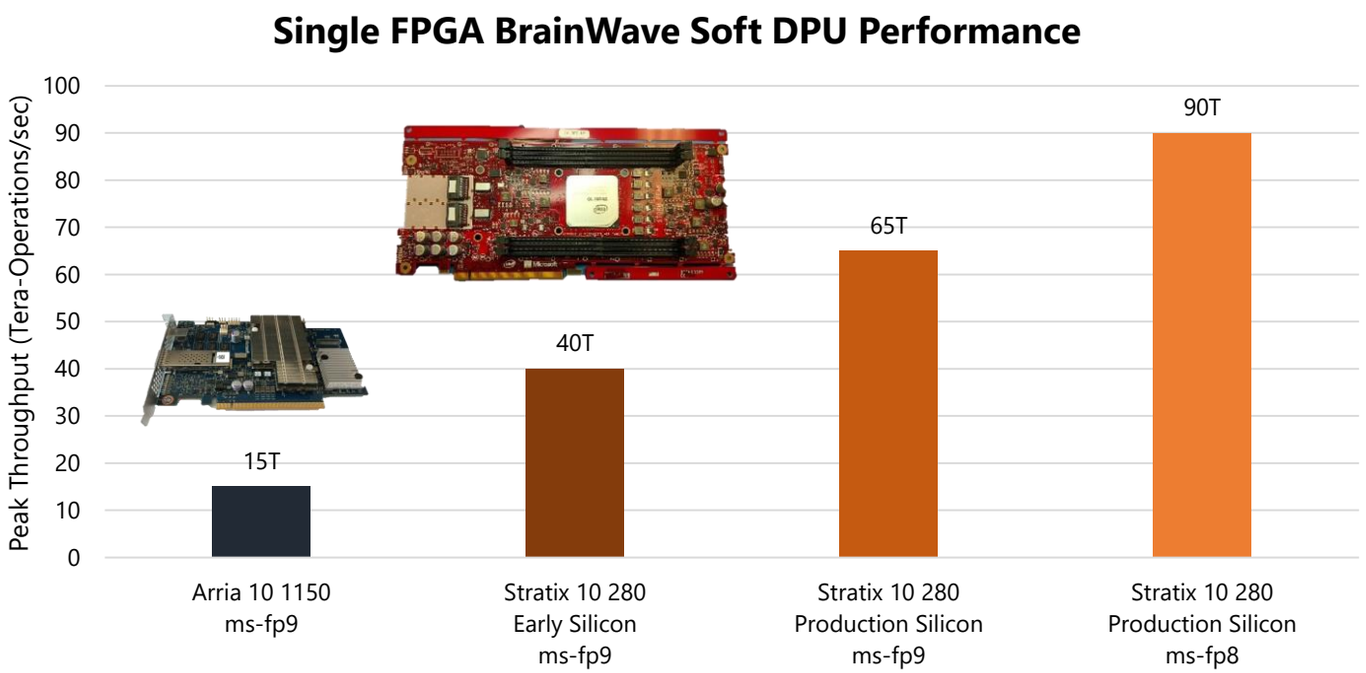
- Optimized for batch 1 matrix-vector multiplication
- Matrices distributed row-wise across 1K-10K banks of BRAM, up to 20 TB/s
- Can scale to use all available on-chip BRAMs, DSPs, and soft logic
- In-situ conversion of float16 weights and activations to internal format
- Dense dot product units map efficiently to soft logic and DSPs



Matrix Vector Unit

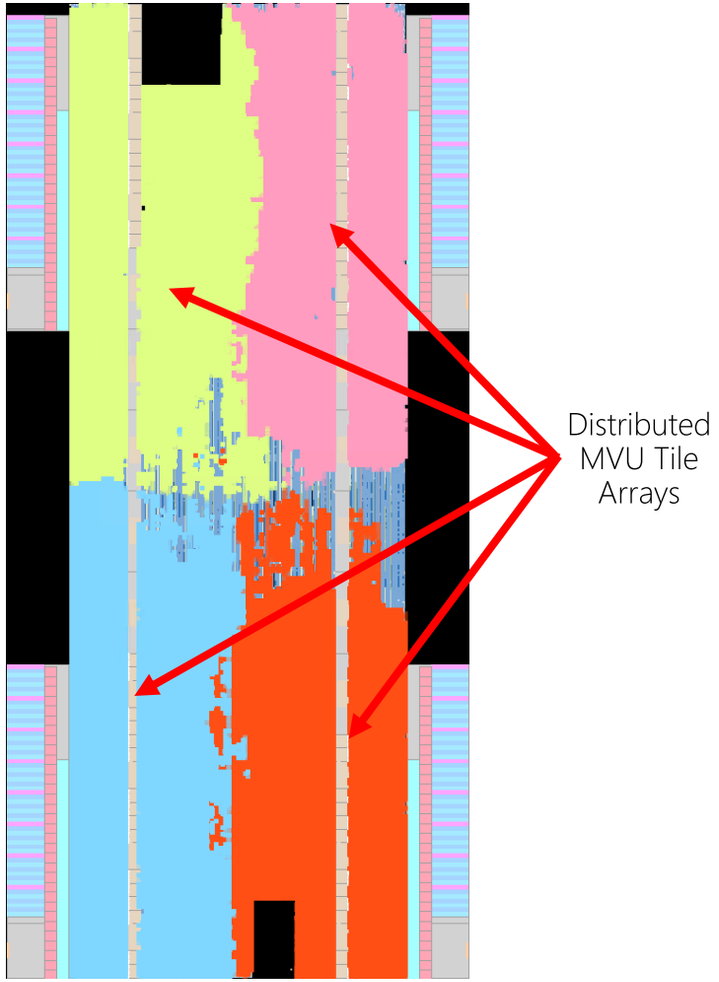


BrainWave Soft DPU Performance



Arria 10 1150 (20nm)
ms-fp9
316K ALMs (74%)
1442 DSPs (95%)
2,564 M20Ks (95%)
160 GOPS/W

Stratix 10 280 Early Silicon (14nm)
ms-fp9
858K ALMs (92%)
5,760 DSPs (100%)
8,151 M20Ks (70%)
320 GOPS/W → 720 GOPS/W (production)



BrainWave Soft DPU
Floorplan on Stratix 10 280

Conclusion

Project BrainWave is a powerful platform for an accelerated AI cloud

- Runs on Microsoft's hyperscale infrastructure with FPGAs
- Achieves excellent performance at low batch sizes via persistency and narrow precision
- Adaptable to precision and changes in future AI algorithms

BrainWave running on Hardware Microservices will push the boundary of what is possible to deploy in the cloud

- Deeper/larger CNNs for more accurate computer vision
- Higher dimensional RNNs toward human-like natural language processing
- State-of-the-art speech
- And much more…

Stay tuned for announcements about external availability.

Thank you!