Microsoft

(Cloud) Acceleration at Microsoft

Derek Chiou, Eric Chung, and Susan Carrie
Microsoft Azure
{dechiou, erchung, scarrie}@microsoft.com

Microsoft's Cloud

- Microsoft's Cloud consists of much more than Azure
  - Bing, Office 365/Exchange, SQL Server, Xbox, OneDrive, Teams/Skype, etc.
  - Very broad collection of workloads
- Share infrastructure to increase engineering and business efficiency
  - Motherboards, firmware, BIOS
  - Server configurations
  - Actual machines
  - Networking
  - Storage
  - Etc.
- How did/do/will we efficiently accelerate Microsoft's Cloud?
  - Share acceleration infrastructure and engineering, rapid iteration, roadmaps/requirements
  - Two examples

Microsoft Is Accelerator Agnostic: E.g., in AI

Benefits of Cloud Acceleration @ Microsoft

- 1st party (internal) development very helpful for acceleration
  - No legal issues, see code
  - Shared detailed requirements enable shared infrastructure, development platforms, ideas
  - Can share internal success stories
  - Draw from internal development resources
  - Partner directly with developers / users
- Example
  - Bing accelerated ranking with FPGAs
  - Azure Networking saw the work as an existence proof, leveraged it
    - Borrowed developers, architect
    - Leveraged most code, ideas, infrastructure
  - Re-achitected board/system enabled Networking and much higher scale for Bing

Cloud-Specific Considerations for Accelerators (I)

- Optimized servers
  - Space: In-server or appliance
  - Limited IO slots/bandwidth (e.g., PCIe)
- Power
  - Chassis/rack/row have fixed power limits, must tradeoff
  - More heat → more cooling → more fan → more power
    - Fans take much more power than you might expect
- Deployment is king
  - Working in the lab is just the start
  - Many deployments have challenges (e.g., interactions with other components, very low probability silicon bugs)
  - Livesite support

Cloud-Specific Considerations for Accelerators (II)

- Scale
  - Cloud is one giant machine with millions of nodes/customers
  - Smaller number of sockets over which NRE is amortized
  - Sharing between applications can increase volumes, reduce per unit costs
- Reliability/availability
  - Varies depending on application, but generally very high
  - Ability to diagnose/repair without physical access
- Multi-tenant support
- Security
  - Hardened against external, internal, and physical attacks
  - Isolation between customers

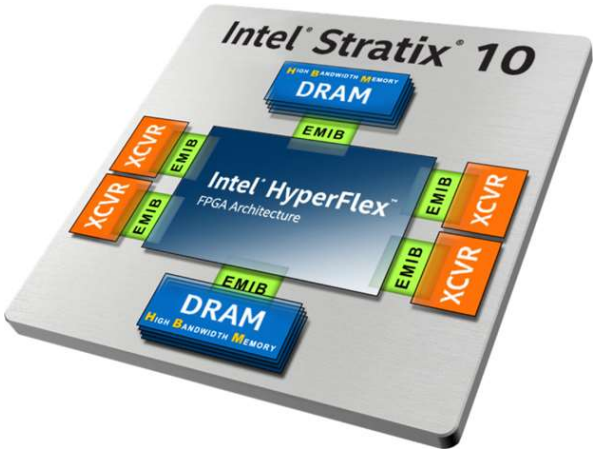
Acceleration

- Traditional accelerator categories are FPGAs, ASICs, GPUs
  - Microsoft uses all (but not all in the same numbers)
  - Can be either discrete package or an IP block in an SoC
- All things being equal, specialization is the root of efficiency
  - Eliminate overheads of unnecessary functionality
  - e.g., if 4b numbers sufficient for DNN inference, why do I need 32b data types?

- Accelerators are more efficient because they are specialized
- However, specialization increases costs
  - Engineering costs: silicon, application/system integration, agility/time-to-market, etc.
  - Business costs: additional spares, reduced volumes for each component, etc.
- A fundamental tradeoff

Issue with Traditional Accelerator Categorization

- Traditional accelerator categorization is product-based rather than architectural, structural, or capability based
  - FPGA == Intel/Xilinx/Lattice/Microsemi/Achronix/etc.
  - ASIC is nebulous, very few truly application specific chips (e.g., floating point unit)
  - GPU == Nvidia/AMD/etc.



- All have changed significantly over time
  - FPGAs have hard PCIe, DRAM, Ethernet, memories, ALUs, cores
  - GPUs went from graphics-specific to data parallel cores

Alternative Hardware Accelerator Categorization

- ***Fixed-function:*** (mostly) single function
  - e.g., PCIe controller, Ethernet controller, DRAM controller
  - Some configurability (e.g., PCIe Gen3/Gen4), not programmable (e.g, PCIe controller != Ethernet MAC)
  - Very unlikely/impossible to change requirements (e.g., attached to something else)
  - Implemented most efficiently in ***fixed logic*** (specialized hard circuit)
- ***Programmable:***
  - ***Circuit programmable***: e.g., FPGAs lookup tables + routing
  - ***Instruction/micro-instruction programmable***: e.g., network processors, GPUs
  - Needed when functionality changes over time, enables optimization over time

- Circuit programmable (FPGA) is often compared to fixed logic
  - Programmability costs, fixed logic will (all things being equal) be more efficient
  - FPGA generally an order of magnitude larger, slower, and less energy efficient than fixed logic
- Better comparison is between programmable components

Programmable Tradeoffs

- Circuit programmable device: inefficient gates, efficient circuits
  - Can highly specialize implementation, simplifies, easier to verify
  - Cannot deal with rapid differences
- Instruction programmable: efficient gates, processing overheads
  - More generality means larger total addressable market, but the less efficient
  - Can deal with rapidly changing requirements

- For example, crypto
  - AES: (CBC, GCM, CTR, XTS) * (128, 192, 256, 512), SHA, etc.
  - Often the case that one crypto standard supported at a specific time
    - Optimized circuit works well
  - If must switch every cycle across a range, instruction programmable may be preferred

Microsoft's Hybrid Approach: Use Best Tool for Job

- Balance efficiency gains with costs of specialization at the solution level, accounting for business concerns and leveraging others' work and hardware

- "Best" depends on many factors, such as
  - Solution performance, power, cost/unit, time to market
  - Stability of algorithms, size of opportunity, NRE
  - Opportunity costs, delta engineering costs, iteration speed, when does engineering need to be done
  - Reuse of deployed infrastructure
  - Business issues (volumes, partnerships, legal, export)
  - Competitive position

Some Examples

- Circuit Programmable
  - Accelerated Networking (> 1M deployed) on FPGA
    - provides low latency, flexibility to adapt to rapidly changing network needs
  - https://www.usenix.org/conference/nsdi18/presentation/firestone
- Circuit/Instruction Programmable
  - DNN Inference (Brainwave) soft cores implemented on FPGA
  - https://www.microsoft.com/en-us/research/uploads/prod/2018/06/ISCA18-Brainwave-CameraReady.pdf
- Fixed Function
  - Corsica compression, encryption, data integrity
  - https://github.com/opencomputeproject/Project-Zipline

Microsoft

Project Brainwave

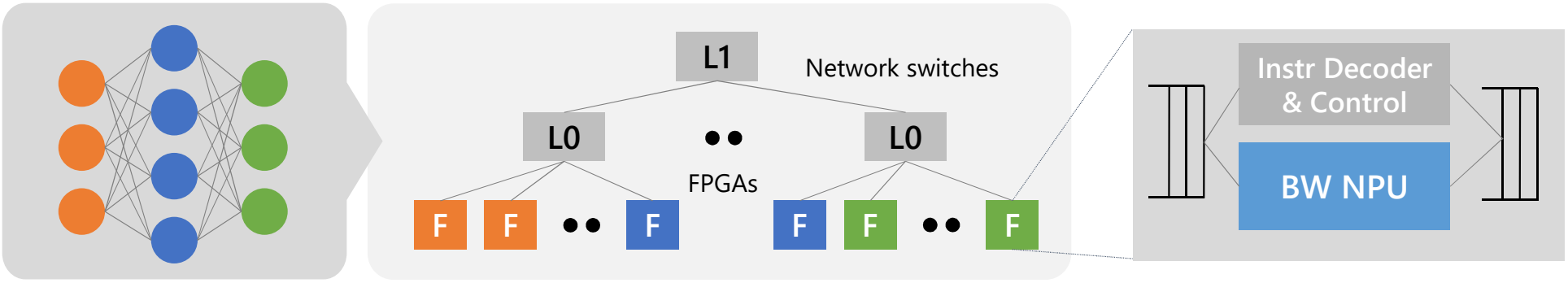
Project Brainwave

A Scalable DNN Serving Platform

Fast: Latency, high-throughput serving of DNN models without requiring batching

Flexible: Deployment on FPGA can adapt to fast-moving AI algorithms

Friendly: Turnkey deployment of pretrained models in popular frameworks

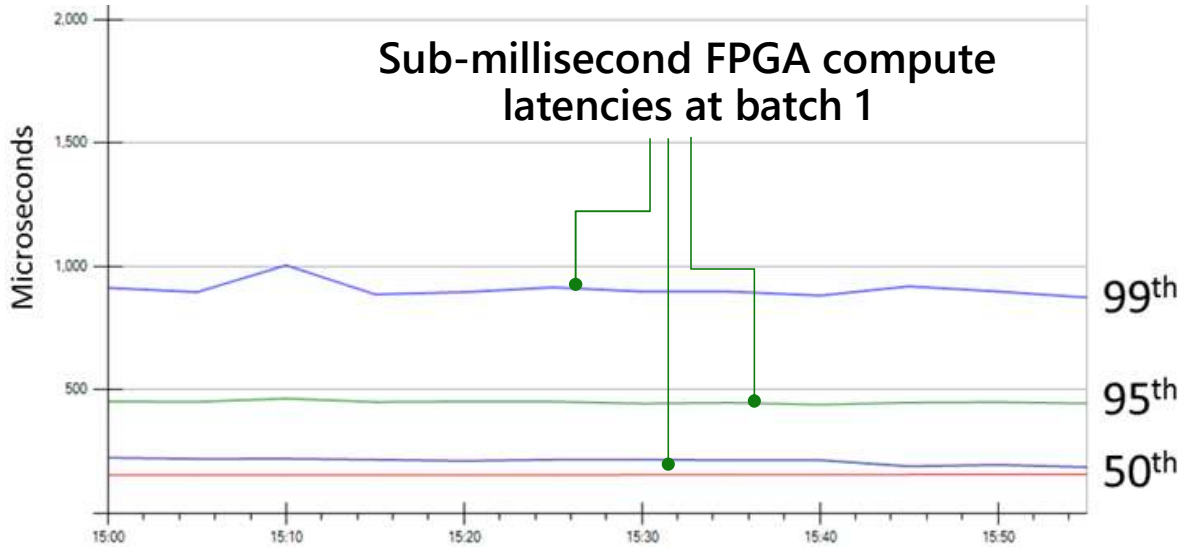


Pretrained DNN Model in ONNX, etc.

Scalable DNN Hardware Microservice

Brainwave Soft NPU

In Production since 2017



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 negligible in E2E software pipelines

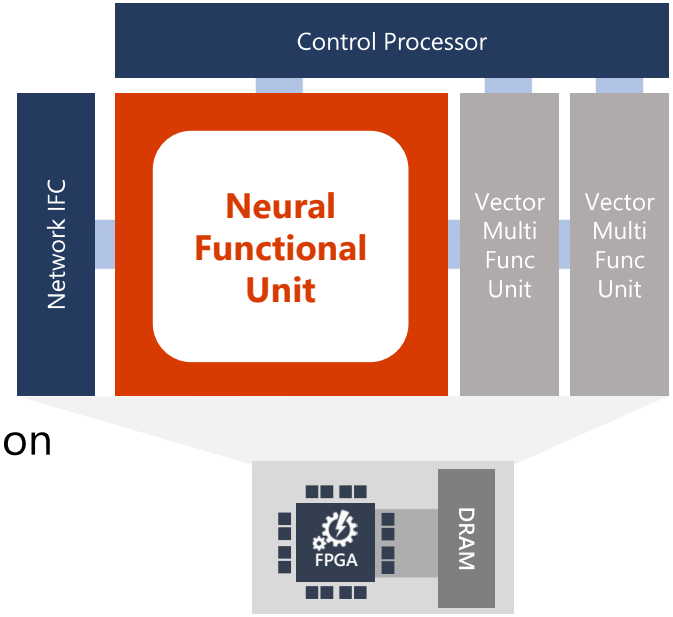
Brainwave Soft NPU

Objectives

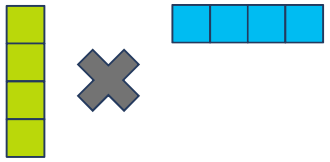
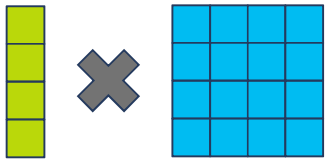
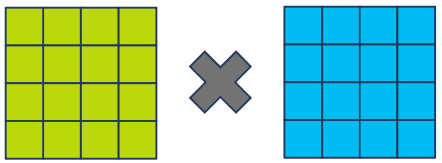
- Optimized for no-batch inferencing
- Simplify programmability and targetability with a single thread of control

- Balance instruction granularity and flexibility
  - Abstracted to M-V and V-V operations of native dimension N
  - Composable into diverse models: RNN, 1D+2D CNNs, etc.

- Instruction chaining
  - Results of one instruction results forwarded to the next instruction
  - Enables long chains without dependency analysis
    or multi-ported register files

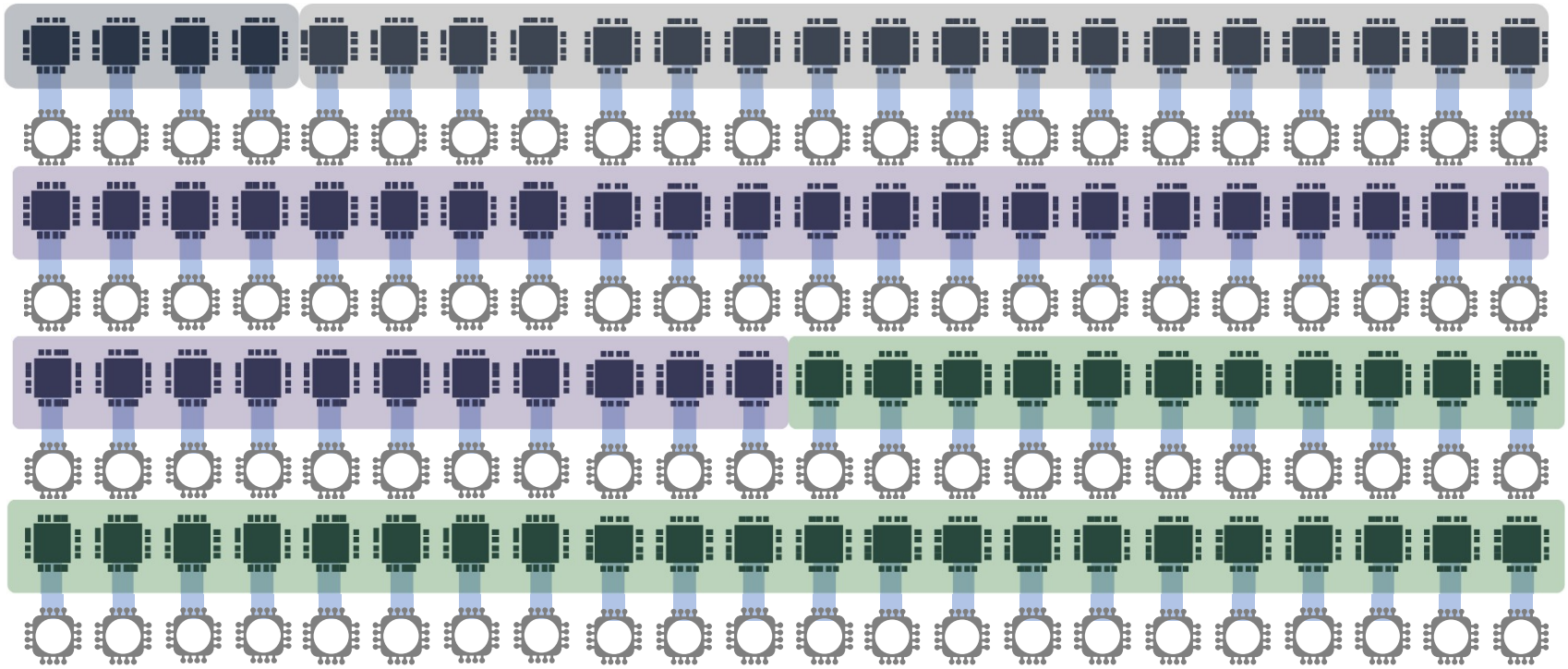


Dominant Compute Pattern in ML/DL: Matrix Multiply

Primitive	Good for	Data Reuse	
Vector * Vector	CNNs RNNs FCs	*IN0: O(1)* *IN1: O(1)*	
Matrix * Vector	CNNs RNNs FCs	*IN0: O(N)* *IN1: O(1)*	
Matrix * Matrix	CNNs Batched FCs Batched RNNs	*IN0: O(N)* *IN1: O(N)*	

Isn’t Matrix Vector Memory Bound?

Distributed Weight Pinning on HW Microservices



Weight pinning across multiple on-chip scratchpads enables distributed serving of challenging AI models with minimal or no batching

Example LSTM Program (single threaded)

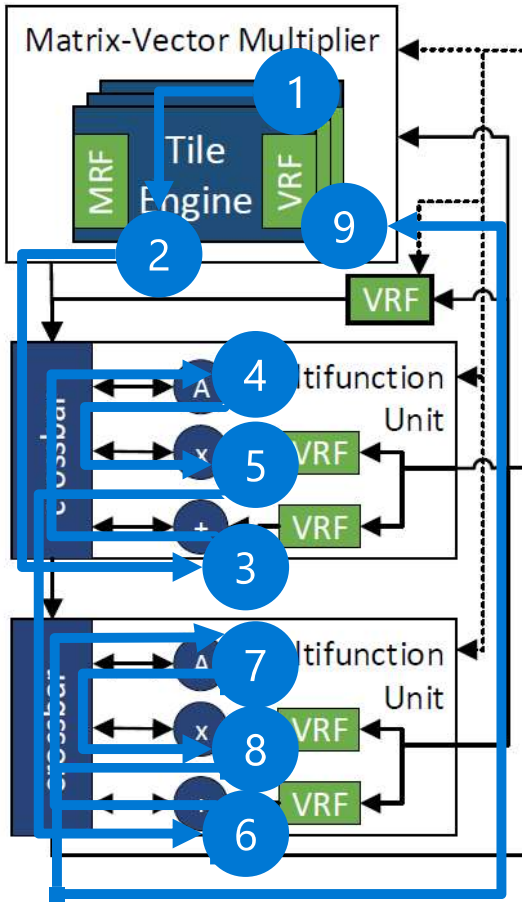
```
1.  void LSTM(int steps) {
2.      for (int t = 0; t < steps; t++) {
3.          v_rd(NetQ);
4.          v_wr(InitialVrf, xt);
5.          v_rd(InitialVrf, xt);
6.          mv_mul(Wf);
7.          vv_add(bf);
8.          v_wr(AddSubVrf, xWf);
9.          v_rd(InitialVrf, h_prev);
10.         mv_mul(Uf);
11.         vv_add(xWf);
12.         v_sigm();
13.         vv_mul(c_prev);
14.         v_wr(AddSubVrf, ft_mod);
15.         v_rd(InitialVrf, h_prev);
16.         mv_mul(Uc);
17.         vv_add(xWc);
18.         v_tanh();
19.         vv_mul(it);
20.         vv_add(ft_mod);
21.         v_wr(MultiplyVrf, c_prev);
22.         v_wr(InitialVrf, ct);
23.         v_rd(InitialVrf, ct);
24.         v_tanh();
25.         vv_mul(ot);
26.         v_wr(InitialVrf, h_prev);
27.         v_wr(NetQ);
28.     }
29. }
```

Microarchitecture

Executes a continuous stream of instruction chains

- Function units arranged to mirror chain structure
- Chaining reduces latency of critical path and register pressure

- Scaling M*V
  - Memory bandwidth
  - Tile engines
  - Data types
  - Scheduling



1. Read Vector x
2. M*V by W
3. Add Vector by b
4. Vector Tanh
5. Multiply Vector by i
6. Add Vector by f
7. Vector Tanh
8. Multiply by o
9. Write Vector h

Scaling M*V: Narrow Precision Data Types

MSFP8 - MSFP11 sufficient

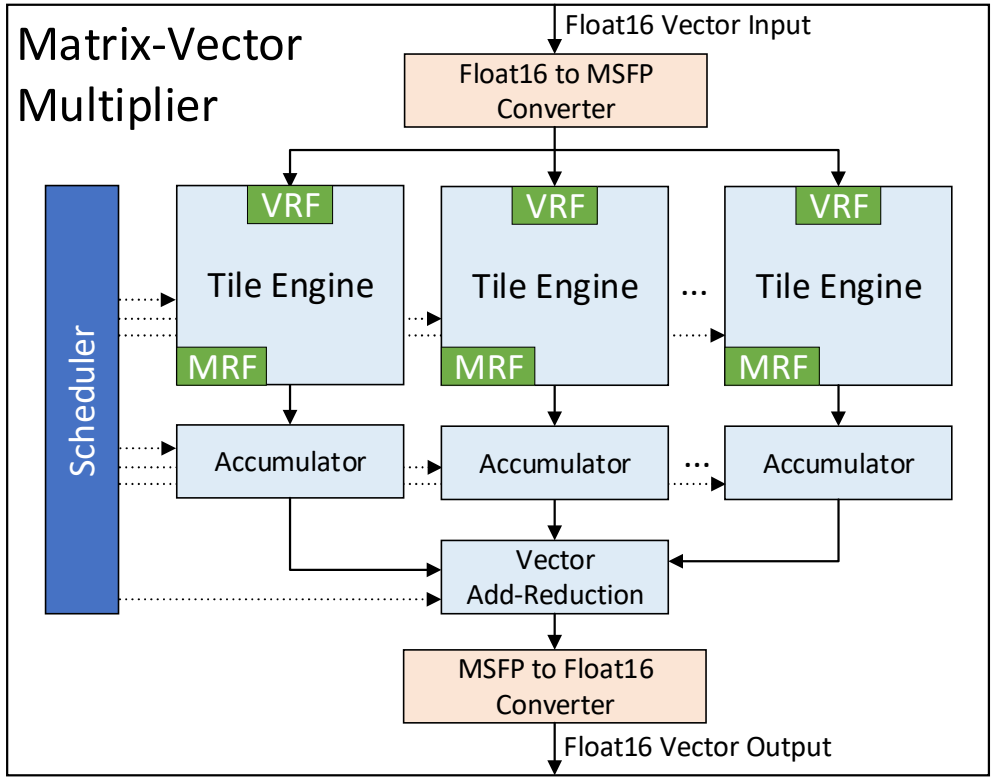
e.g., 1-bit sign, 5-bit exponent, 2- to 5-bit mantissas

96,000 MSFP8 soft logic MACs

versus

5760 FP32 hard logic MACs

➔ 16X more efficient (area & energy)



https://www.microsoft.com/en-us/research/uploads/prod/2018/06/ISCA18-Brainwave-CameraReady.pdf

Brainwave NPU Evaluation (2018)

Device	Logic	SRAM	DSPs	MHz	Peak TFLOPs
Stratix V D5	87%	59%	66%	200	2.4
Arria 10 1150	51%	80%	100%	300	9.8
Stratix 10 280	91%	69%	91%	250	48

Current Peak perf per FPGA family

	Titan Xp	BW_S10
Numerical Type	Float32	MSFP8
Peak TFLOPs	12.1	48
TDP (W)	250	125
Process (nm)	TSMC 16	Intel 14

Benchmark	Parameters	Device	Latency (ms)	TFLOPs	% Utilization
GRU	h=2816, t=750	BW	1.987	35.92	74.8
GRU	h=2816, t=750	Titan Xp	178.6	0.4	3.3
GRU	h=2560, t=375	BW	0.993	29.69	61.8
GRU	h=2560, t=375	Titan Xp	74.62	0.4	3.3
GRU	h=2048,t=375	BW	0.954	19.79	41.2
GRU	h=2048,t=375	Titan Xp	51.59	0.37	3
GRU	h=1536, t=375	BW	0.951	11.17	23.3
GRU	h=1536, t=375	Titan Xp	31.73	0.33	2.8
GRU	h=1024, t=1500	BW	3.792	4.98	10.4
GRU	h=1024, t=1500	Titan Xp	59.51	0.32	2.6
GRU	h=512, t=1	BW	0.013	0.25	0.5
GRU	h=512, t=1	Titan Xp	0.06	0.05	0.4

Bing Intelligent Search Backed By Project Brainwave

DECEMBER 13 2017

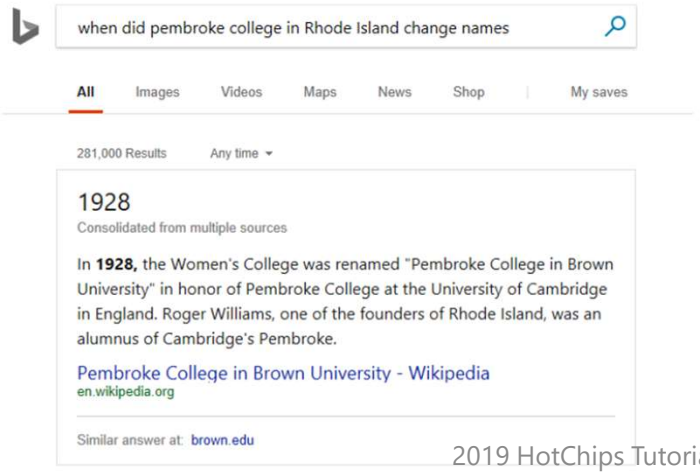
Bing launches new intelligent search features, powered by AI

Today we announced new Intelligent Search features for Bing, powered by AI, to give you answers faster, give you more comprehensive and complete information, and enable you to interact more naturally with your search engine.

Intelligent answers:

Intelligent answers leverage the latest state of the art machine reading comprehension, backed by Project Brainwave running on Intel's FPGAs, to read and analyze billions of documents to understand the web and help you more quickly and confidently get the answers you need.

Bing now uses deep neural networks to validate answers by aggregating across multiple reputable sources, rather than just one, so you can feel more confident about the answer you're getting.



Stratix V RNN-optimized NPUs in Scale Production

Bing TP1			
	CPU-only	**Brainwave-accelerated**	**Improvement**
Model details	GRU 128x200 (x2) + W2Vec	LSTM 500x200 (x8) + W2Vec	Brainwave-accelerated model is > 10X larger and > 10X lower latency
End-to-end latency per Batch 1 request at 95%	9 ms	0.850 ms	
Bing DeepScan			
	CPU-only	**Brainwave-accelerated**	**Improvement**
Model details	1D CNN + W2Vec *(RNNs removed)*	1D CNN + W2Vec + GRU 500x500 (x4)	Brainwave-accelerated model is > 10X larger and 3X lower latency
End-to-end latency per Batch 1 request at 95%	15 ms	5 ms	

https://www.microsoft.com/en-us/research/uploads/prod/2018/03/mi0218_Chung-2018Mar25.pdf

https://blogs.bing.com/search/2017-12/search-2017-12-december-ai-update

Project Brainwave Optimized for Microsoft Cloud

- Inference-optimized, SW-programmable NPU deployed on circuit programmable FPGA
  - Initially targeted for Bing => batch size of 1
  - Rapid iteration enabled quick innovation (e.g., data word types) and deployment
  - Other than data type specializations, architecture is FPGA-agnostic
- Network-attached FPGAs affected design (large numbers of FPGAs)
  - Enabled large models, high performance by distributing across FPGAs
- Leveraged existing FPGA work
  - Deployed FPGAs
  - Infrastructure
  - Tooling
- Contributed back to FPGA effort
- Currently deployed in scale first party services such as Bing
- Available to third party customers through AzureML

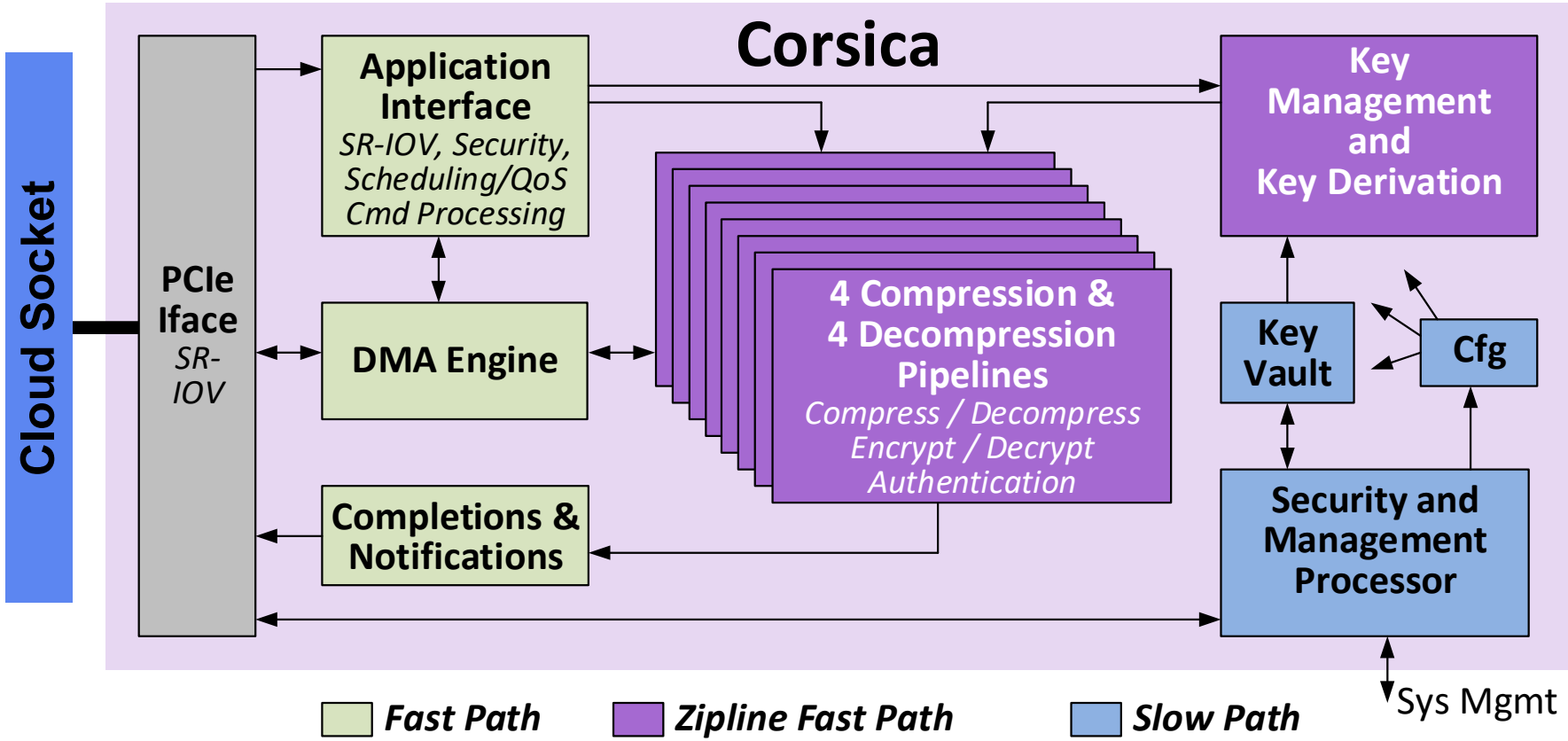
Microsoft

Corsica/Zipline

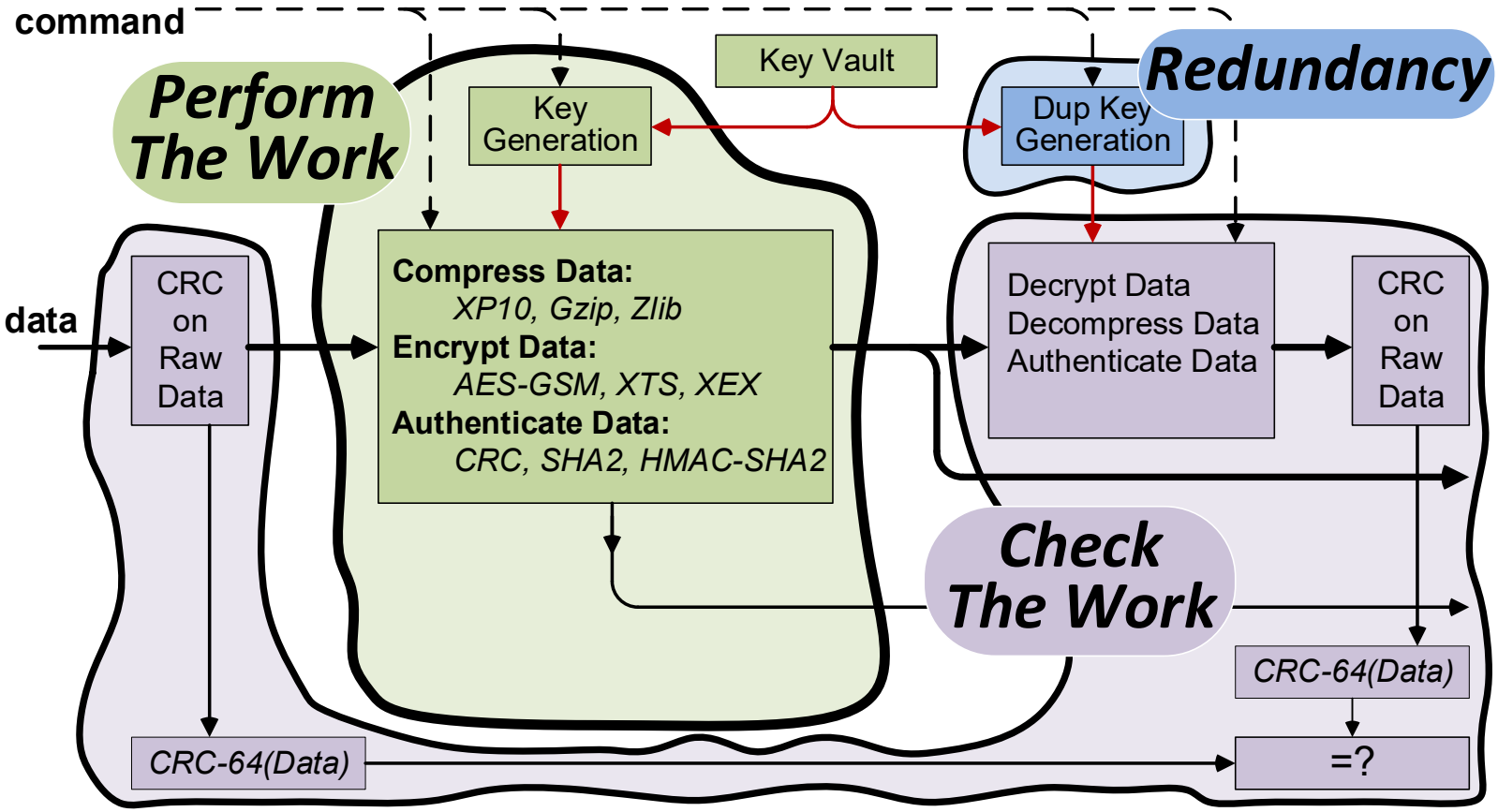
ASIC Acceleration: Corsica

- Targeted toward data at rest: storage
- Two types of work:
  - Compression, encryption, authentication
  - Decryption, decompression, authentication
- Partnership with Key First Party Customer (Storage) critical to defining requirements
  - Which encryption algorithm, which KDF, key usage
  - Full pipeline of function elements
  - Very low failure rate (think alpha particles)
  - low latency (25Gbps for a single command)
  - high throughput (max out 100Gbps PCIe)
  - tight integration with existing SW stack
- Simple, inexpensive fixed function solution (no external DRAM)

Corsica – High Level View



Corsica – Compression Pipeline Highlights



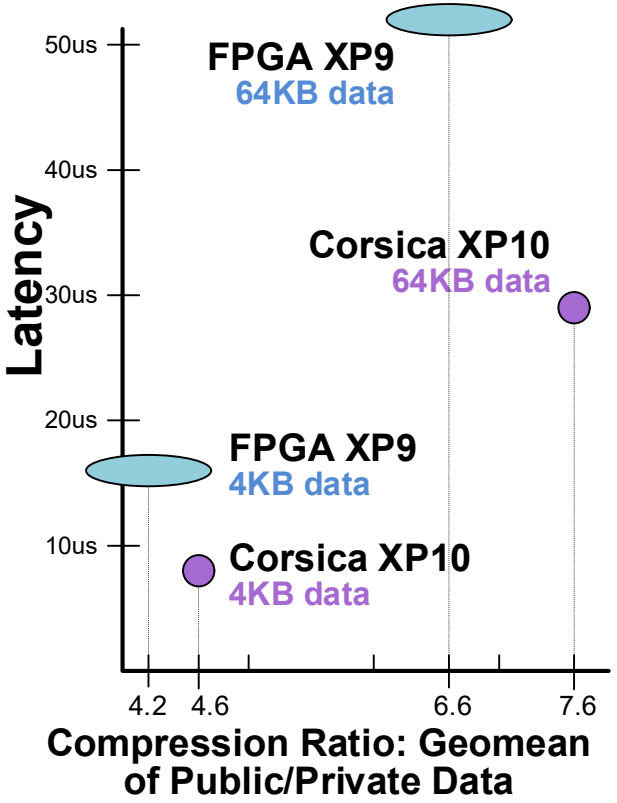
Corsica Support of Cloud Requirements

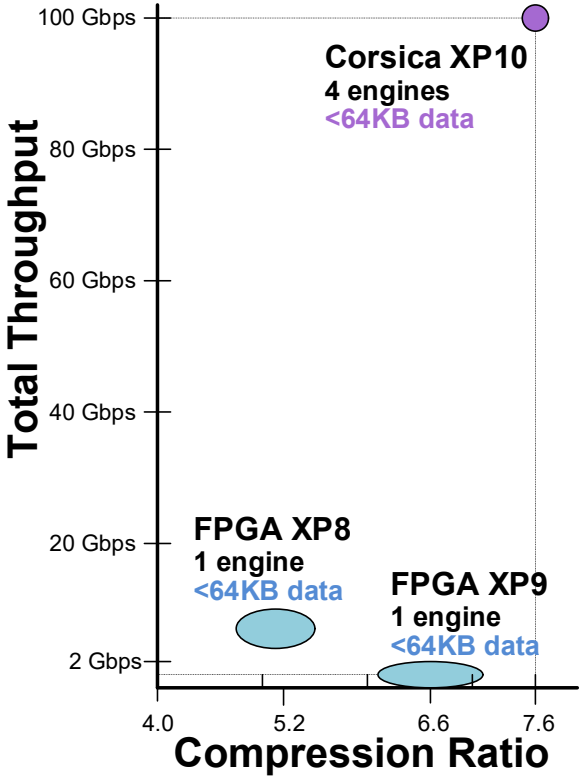
- Multi-tenant support
  - Support for sharing pipelines → QoS, scheduling, SR-IOV
- Reliability
  - Support "check the work" and "redundancy"
  - FW support for diagnosis and debug
- Security
  - Cerberus support
  - Xbox level of security
  - Secure transfer of information across PCIe (public/private and symmetric keys)
  - Using SR-IOV to provide isolation between customers

Comparing Corsica to FPGA Implementations

·At similar technology node (14nm/!6nm), single FPGA does not support required compression ratio/speed

·Compression / encryption algorithms are stable





Another Consideration: Open Sourcing

- Lower volumes from a single cloud increase NRE per chip
- Two paths
  - Be in the business of making hardware
    - Continuous development of differentiating hardware
    - Derive ROI from differentiation
  - Enable vendor eco-system to share NRE
    - Enable vendors to provide best HW
    - Derive ROI from selling services

Corsica Is Open Source

- The Corsica compression RTL was open-sourced
  - Project Zipline contributed to OCP (Open Compute Platform)
  - Shares the key requirements for Microsoft
  - Open sourced XP10 compression standard suitable for cloud data sets
- Goal is to enable the eco-system
- https://github.com/opencomputeproject/Project-Zipline

Conclusions

- Cloud introduces new set of requirements
- Each acceleration opportunity should be evaluated in context
  - Every situation is different
  - Situations where fixed function, circuit programmable, instruction programmable make sense
- Microsoft leverages many kinds of accelerators
- Open sourcing hardware may make sense in cloud
- Microsoft is interested in enabling eco-systems, contact us

- We're hiring!