Concepts Introduced in Chapter 7

- guidelines for domain specific architectures (DSAs)
- example DSAs
  - Google's Tensor Processing Unit
  - Microsoft's Catapult
  - Google's Pixel Visual Core
- crosscutting issues

Reasons for Adoption of Domain Specific Architectures

- End of Dennard scaling and slowing of Moore's Law means we need to lower the energy per operation to improve performance.
- If order-of-magnitude performance improvements are to be obtained, we need to increase the number of arithmetic operations per instruction from one to hundreds.
- Conventional architecture innovations may not be a good match to some domains.
- Many believe that future computers will consist of standard processors that can run conventional programs along with domain-specific processors that can very efficiently perform only a narrow range of tasks.

Factors That Can Make a DSA Cost Effective

- Find a domain whose demand for applications is for enough chips to justify the nonrecurring engineering (NRE) costs.
- DSAs are best applied for small compute-intensive kernels of larger systems, where a significant fraction of the computation is done for some applications.
- This means that future computers may be much more heterogeneous than current homogeneous multicore chips.
- Architects must learn the application domain and algorithms.
- There must also be support for porting software to exploit these DSAs.

DSA Guidelines

- Use dedicated memories to minimize the distance over which data is moved.
  - Multi-level caches are expensive in area and energy for moving data.
  - Many domains have predictable memory access patterns with little data reuse.
  - DSA programmers understand their domain.
  - Data movement can be more efficient with software controlled memories.
- Invest resources saved from dropping advanced microarchitectural optimizations into more arithmetic units and/or larger on-chip memories.

DSA Guidelines (cont.)

- Use the easiest form of parallelism that matches the domain.
  - Target domains for DSAs will have inherent parallelism.
  - Need to exploit that parallelism and expose it to the software so it does not need to be automatically found by the hardware (no OoO execution).
- Reduce data size and type to the simplest needed for the domain.
  - Many domains are memory bound.
  - Can increase memory bandwidth and utilization by using narrower data types.
- Use a domain-specific programming language to port code to the DSA.
  - Allow for easier exploitation of parallelism.
  - Simplify porting of code to a DSA.

How Example DSAs Follow the Guidelines

Guideline	TPU	Catapult	Crest	Pixel Visual Core
Design target	Data center ASIC	Data center FPGA	Data center ASIC	PMD ASIC/SOC IP
1. Dedicated memories	24 MiB Unified Buffer, 4 MiB Accumulators	Varies	N.A.	Per core: 128 KiB line buffer, 64 KiB P.E. memory
2. Larger arithmetic unit	65,536 Multiply-accumulators	Varies	N.A.	Per core: 256 Multiply-accumulators (512 ALUs)
3. Easy parallelism	Single-threaded, SIMD, in-order	SIMD, MISD	N.A.	MPMD, SIMD, VLIW
4. Smaller data size	8-Bit, 16-bit integer	8-Bit, 16-bit integer 32-bit Fl. Pt.	21-bit Fl. Pt.	8-bit, 16-bit, 32-bit integer
5. Domain-specific lang.	TensorFlow	Verilog	TensorFlow	Halide/TensorFlow

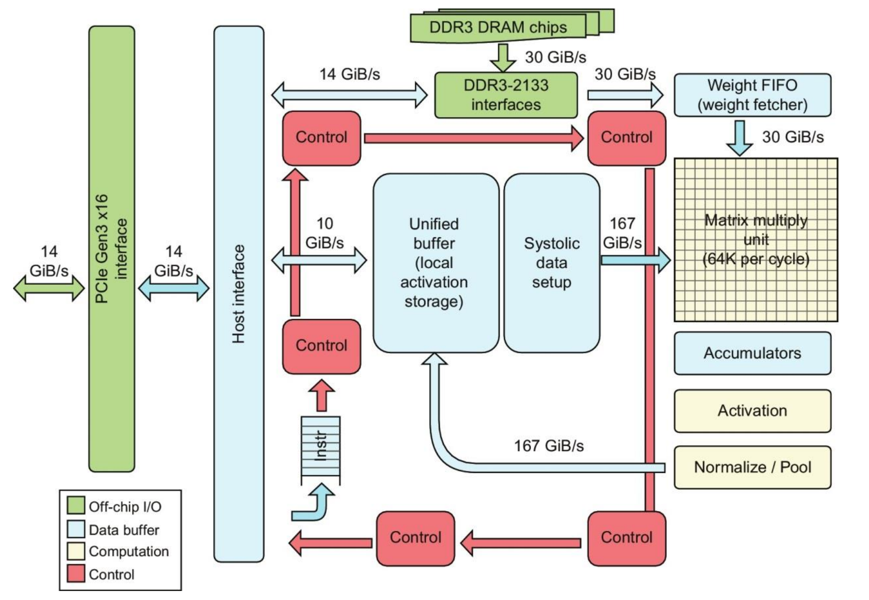
Google's Tensor Processing Unit (TPU)

- Google's first ASIC DSA for its WSCs.
- Domain is for deep neural networks (DNNs).
- Programmed using the TensorFlow language.
- Goal is to improve cost-performance by a factor of 10 over GPUs.
- Deployed in Google data centers since 2015.

TPU Architecture

- TPU is a coprocessor on the PCIe I/O bus from which it receives instructions.
- Has a large software managed on-chip memory, which consists of a 24 MiB Unified Buffer.
- Has off-chip 8 GiB DRAM for Weight Memory.
- Matrix Multiply Unit (MMU) contains 256x256 (65,536) ALUs that can perform 8-bit multiply-and-adds on integers.
  - Reads and writes 256 values per clock cycle.
  - Used for matrix multiplications and convolutions.

TPU Block Diagram

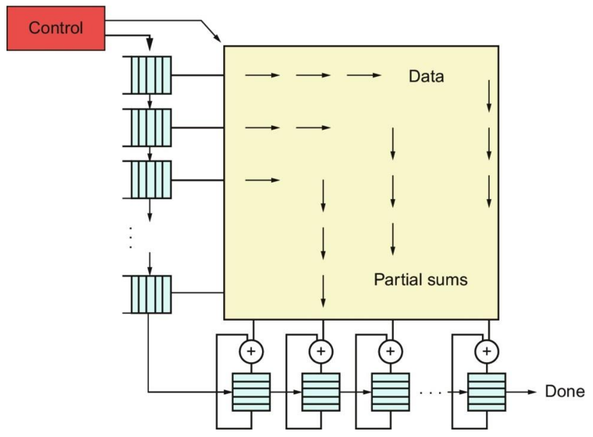


TPU ISA

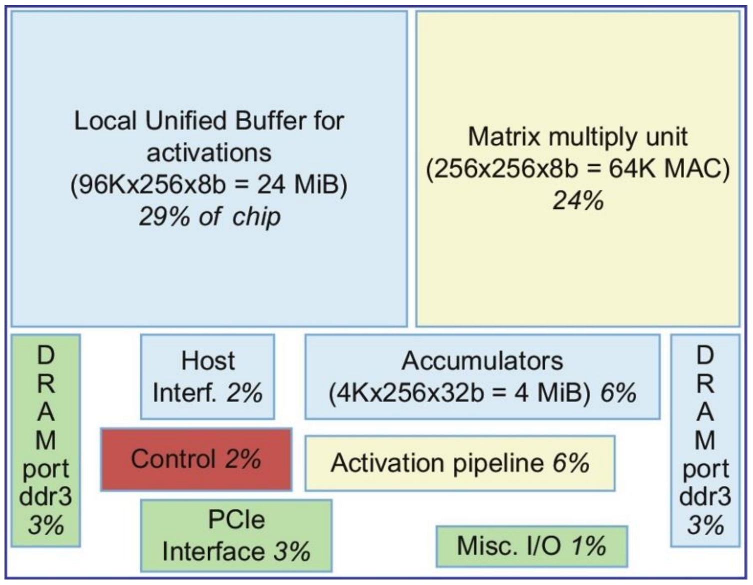
- Has no PC as instructions are sent from the host CPU.
- TPU ISA contains CISC-like instructions, including a repeat field to reduce interactions with the host CPU.
- example instructions
  - *Read_Host_Memory* - reads data from CPU host memory into unified buffer.
  - *Read_Weights* - reads weights from weight memory into weight FIFO.
  - *MatrixMultiply/Convolve* - causes MMU to perform operations on data in unified buffer and produce its output to the accumulators.
  - *Activate* - performs nonlinear function of the neuron from data in accumulators and stores results in unified buffer.
  - *Write_Host_Memory* - writes data from the unified buffer into the CPU host memory.

TPU MMU

- The MMU uses a systolic array, which is a 2D collection of arithmetic units.
- Values flow from one unit to another.
- Data is only read once from memory and only written once to memory.



TPU Die Floorplan


Local Unified Buffer for activations (96Kx256x8b = 24 MiB) *29% of chip*
Matrix multiply unit (256x256x8b = 64K MAC) *24%*
DRAM port ddr3 *3%*
Host Interf. *2%*
Accumulators (4Kx256x32b = 4 MiB) *6%*
DRAM port ddr3 *3%*
Control *2%*
Activation pipeline *6%*
PCIe Interface *3%*
Misc. I/O *1%*

TPU Summary

- *Use dedicated memories to minimize the distance over which data is moved.* Has on-chip a 24 MiB unified buffer that allows accessing 256 bytes each cycle, a weight FIFO, and a 4 MiB accumulators.
- *Invest resources saved from dropping advanced microarchitectural optimizations into more arithmetic units or bigger memories.* Has 28 MiB of on-chip memory and 64K 8-bit ALUs.
- *Use the easiest form of parallelism that matches the domain.* Exploits 2D SIMD parallelism with the 256x256 MMU.
- *Reduce data size and type to the simplest needed for the domain.* Primarily does computation on 8-bit integers.
- *Use a domain-specific programming language to port code to the DSA.* Programs to control the TPU are written in the TensorFlow language.

Microsoft's Catapult

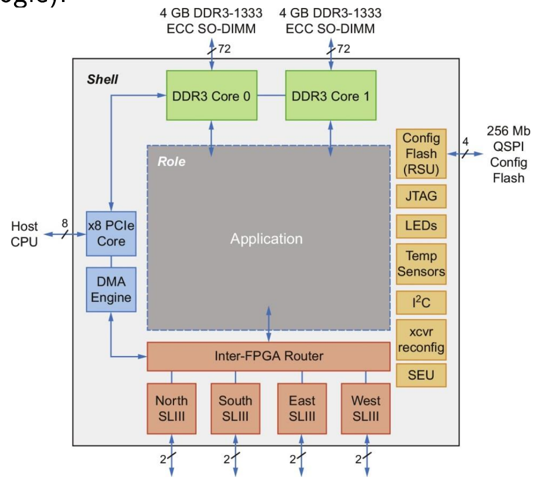
- Microsoft placed an FPGA on a PCIe bus board in data center servers.
- Used FPGA flexibility to tailor use for varying applications.
- FPGAs have lower NRE costs than ASICs.
- FPGAs are slower than ASICs.
- Key applications were to provide a CNN accelerator and to improve the performance of the Microsoft Bing search engine.

Catapult Architecture

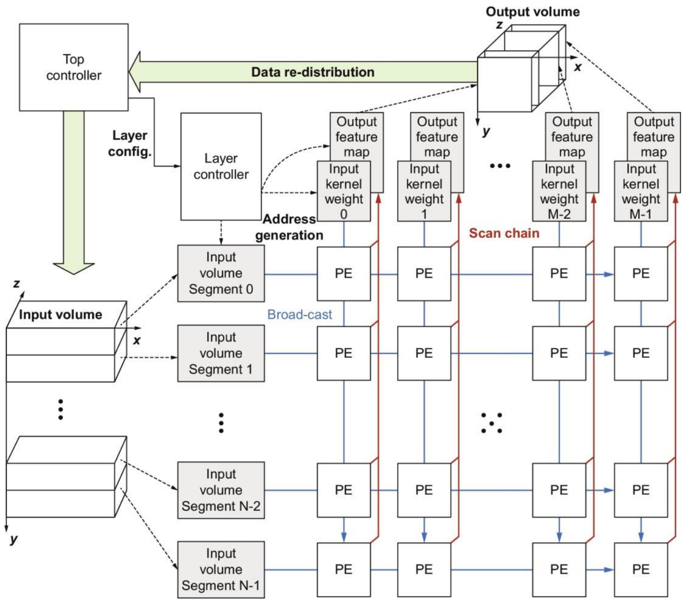
- Board has 32 MiB of flash memory and 8 GiB of DRAM.
- FPGA has 3926 18-bit ALUs, 5 MiB of on-chip memory, 8 GiB of DRAM on the board, and 11 GB/s bandwidth to the board DRAM.
- Designed to run at 25 W.

Catapult Software

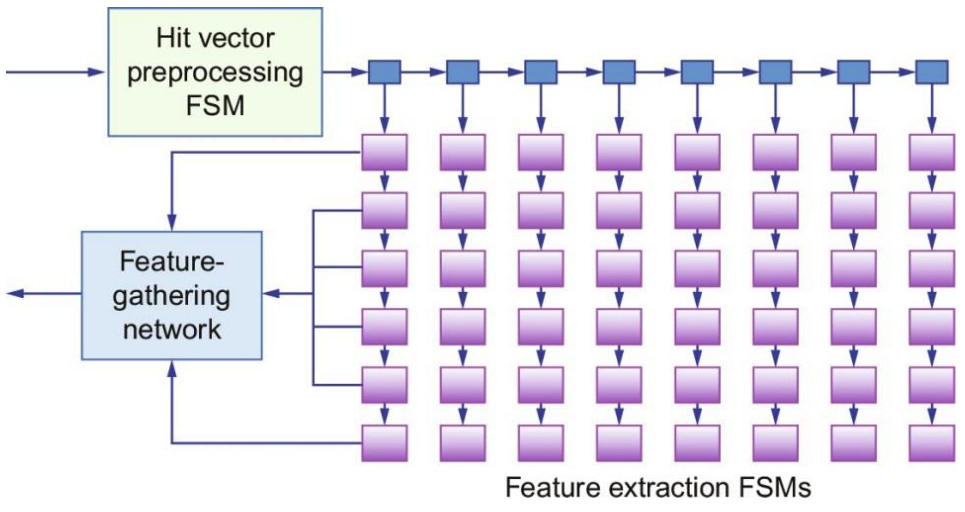
- Programming is done in a hardware description language (e.g. Verilog).
- RTL code divided into the shell (used across applications) and the role (application logic).



Catapult CNN Accelerator



Catapult Feature Extraction Accelerator



Catapult Summary

- *Use dedicated memories to minimize the distance over which data is moved.* Has 5 MiB of on-chip memory.
- *Invest resources saved from dropping advanced microarchitectural optimizations into more arithmetic units or bigger memories.* Has 3926 18-bit ALUs.
- *Use the easiest form of parallelism that matches the domain.* Exploits 2D SIMD parallelism for the CNN application and MISD parallelism for search ranking.
- *Reduce data size and type to the simplest needed for the domain.* Does computation on 8-bit integer to 64-bit FP values.
- *Use a domain-specific programming language to port code to the DSA.* Programming is done in Verilog.

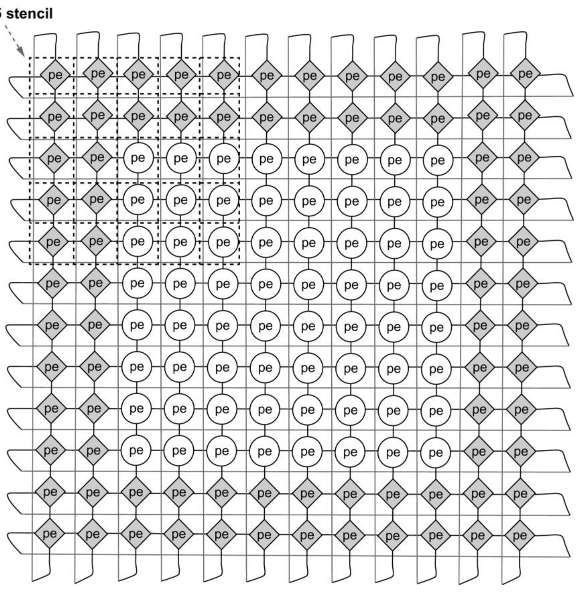
Google's Pixel Visual Core

- Google multicore design has 2 to 16 cores.
- Uses much smaller area and less energy than the TPU.
- Domain area is vision processing.
- Is an example of a class of DSAs called image processing units (IPUs) that analyze and modify an input image.

Pixel Visual Core Architecture

- Uses a 2D SIMD achitecture of independent processing elements (PEs), each containing 2 16-bit ALUs, 1 16-bit MAC unit, 10 16-bit registers, and 10 1-bit predicate registers.
- PE memory is a compiler managed scratchpad containing 128 16-bit entries (256 bytes).
- Each PE collects inputs from nearest neighbors.

Pixel Visual Core 2D Array of PEs

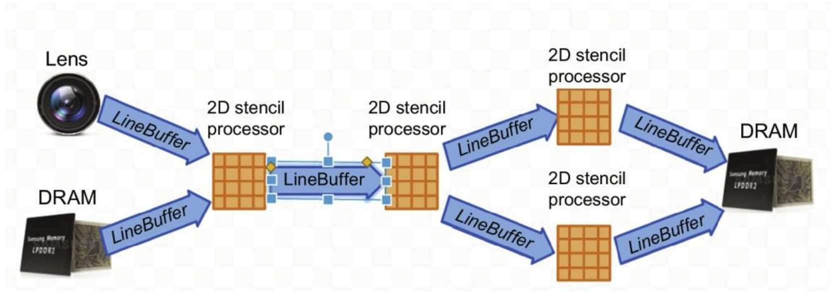


Pixel Visual Core ISA

- Programming is done in Halide, which is a domain specific functional programming language for image processing.
- Compiler first produces vISA (virtual ISA) instructions and then converts them to pISA (physical ISA) instructions.
- vISA is mostly independent of pISA instruction set changes.
- pISA is a VLIW instruction set with 119-bit instructions.
- Instruction memory of a Pixel Visual Core holds at most 2048 pISA instructions.

Pixel Visual Core Line Buffers

- Uses 2D line buffers to minimize DRAM accesses.
- Holds portions of the image between kernels.
- Programmer view is a DAG of kernels.
- Each line buffer has one FIFO producer and up to eight FIFO consumers.



Pixel Visual Core Summary

- *Use dedicated memories to minimize the distance over which data is moved.* Has 128 KiB of line buffers per core. Also has 64 KiB of software controlled PE memory.
- *Invest resources saved from dropping advanced microarchitectural optimizations into more arithmetic units or bigger memories.* Has 16x16 2D array of PEs per core.
- *Use the easiest form of parallelism that matches the domain.* Exploits 2D SIMD parallelism in its PE array, VLIW instructions for ILP, and MPMD for utilizing multiple cores.
- *Reduce data size and type to the simplest needed for the domain.* Does computation mostly on 8-bit and 16-bit integers.
- *Use a domain-specific programming language to port code to the DSA.* Programming is done in Halide for image processing and Tensorflow for CNNs.

Cross-Cutting Issues

- DSAs are often incorporated into a system over the I/O bus.
- DSA accelerators have local DRAM.
- Many DSA designs either use a custom RISC processor or RISC-V, which is a free and open instruction set and IP blocks.

Fallacies and Pitfalls

- Pitfall: Performance counters added as an afterthought for DSA hardware.
- Pitfall: Being ignorant of architecture history when designing a DSA.