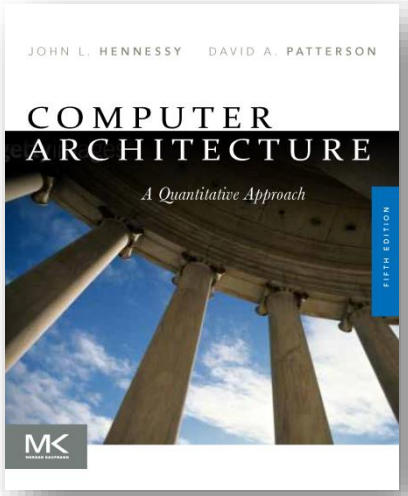


Chapter 7

Domain-Specific Architectures

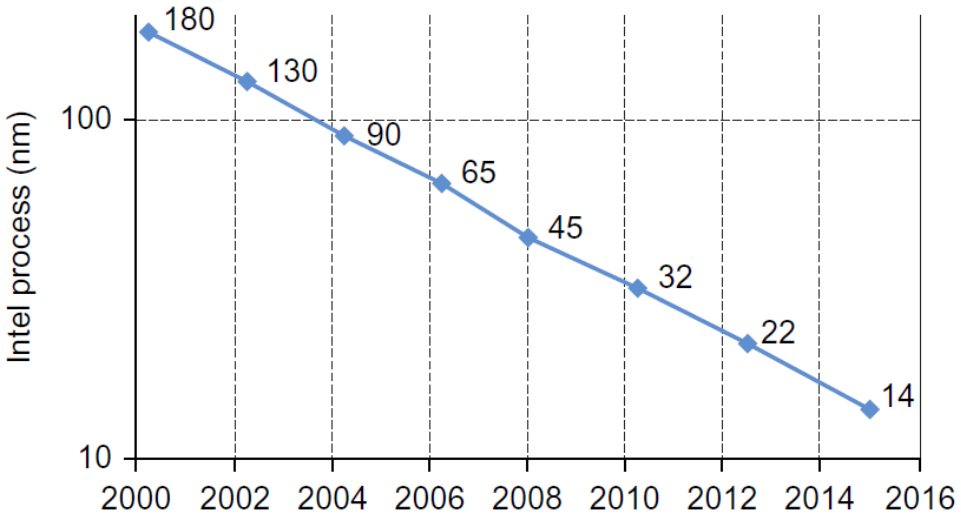
Adapted by Prof. Gheith Abandah

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Introduction

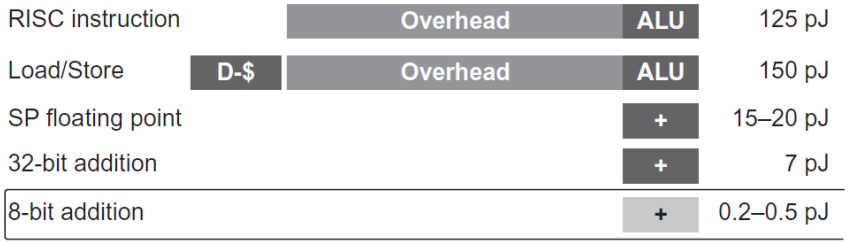
- Moore's Law enabled:
  - Deep memory hierarchy
  - Wide SIMD units
  - Deep pipelines
  - Branch prediction
  - Out-of-order execution
  - Speculative prefetching
  - Multithreading
  - Multiprocessing



- Objective:
  - Extract performance from large applications that is oblivious to architecture

Introduction

- Moore's Law is slowing
- Dennard scaling ended
  - More transistors switching now means more power
- Conventional architectures suffer from high energy overheads for doing arithmetic ops.
- To improve efficiency, need factor of 100 improvements in number of operations per instruction
  - Requires domain specific architectures (DSA)
  - For ASICs, NRE cannot be amortized over large volumes
  - FPGAs are less efficient than ASICs



Contents

Guidelines for DSAs

1. Use dedicated memories to minimize data movement
2. Invest resources into more arithmetic units or bigger memories
3. Use the easiest form of parallelism that matches the domain
4. Reduce data size and type to the simplest needed for the domain
5. Use a domain-specific programming language

Guidelines for DSAs

The four DSAs in Chapter 7 and how closely they followed the five 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

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Example: Deep Neural Networks

- Inpired by neuron of the brain
- Computes non-linear “activiation” function of the weighted sum of input values
- Neurons arranged in layers

Name	DNN layers	Weights	Operations/Weight
MLP0	5	20M	200
MLP1	4	5M	168
LSTM0	58	52M	64
LSTM1	56	34M	96
CNN0	16	8M	2888
CNN1	89	100M	1750

Example: Deep Neural Networks

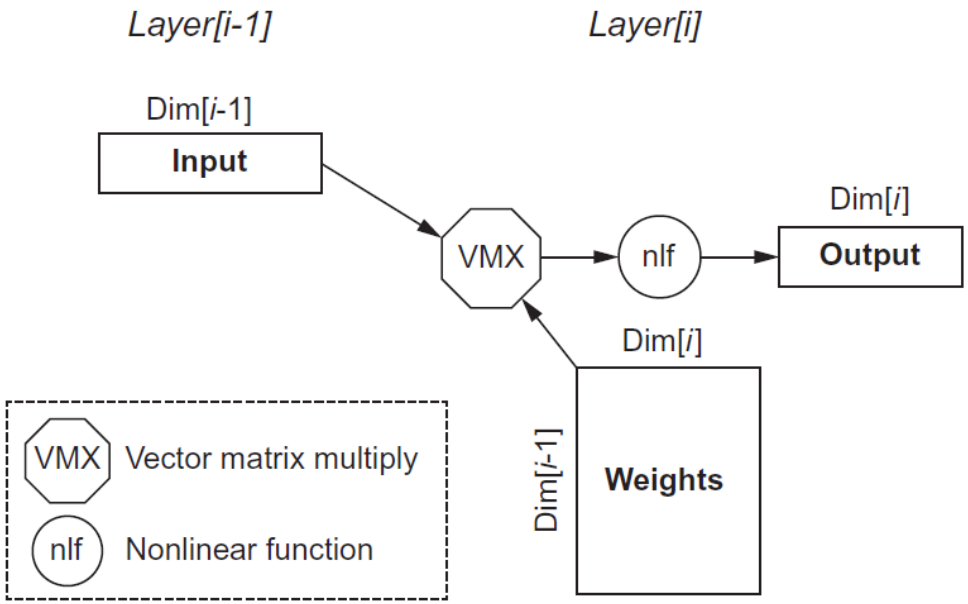
- Most practitioners will choose an existing design
  - Topology
  - Data type
- **Training** (learning):
  - Calculate weights using backpropagation algorithm
  - Supervised learning: stochastic gradient descent

Type of data	Problem area	Size of benchmark's training set	DNN architecture	Hardware	Training time
text [1]	Word prediction (word2vec)	100 billion words (Wikipedia)	2-layer skip gram	1 NVIDIA Titan X GPU	6.2 hours
audio [2]	Speech recognition	2000 hours (Fisher Corpus)	11-layer RNN	1 NVIDIA K1200 GPU	3.5 days
images [3]	Image classification	1 million images (ImageNet)	22-layer CNN	1 NVIDIA K20 GPU	3 weeks
video [4]	activity recognition	1 million videos (Sports-1M)	8-layer CNN	10 NVIDIA GPUs	1 month

- **Inference**: use neural network for classification

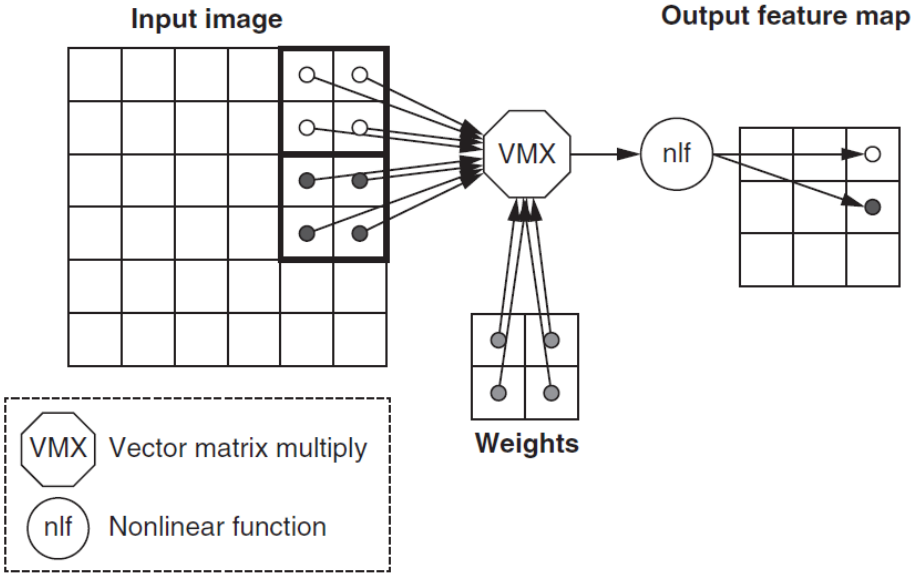
Multi-Layer Perceptrons

- Parameters:
  - Dim[i]: number of neurons
  - Dim[i-1]: dimension of input vector
  - Number of weights: Dim[i-1] x Dim[i]
  - Operations: 2 x Dim[i-1] x Dim[i]
  - Operations/weight: 2

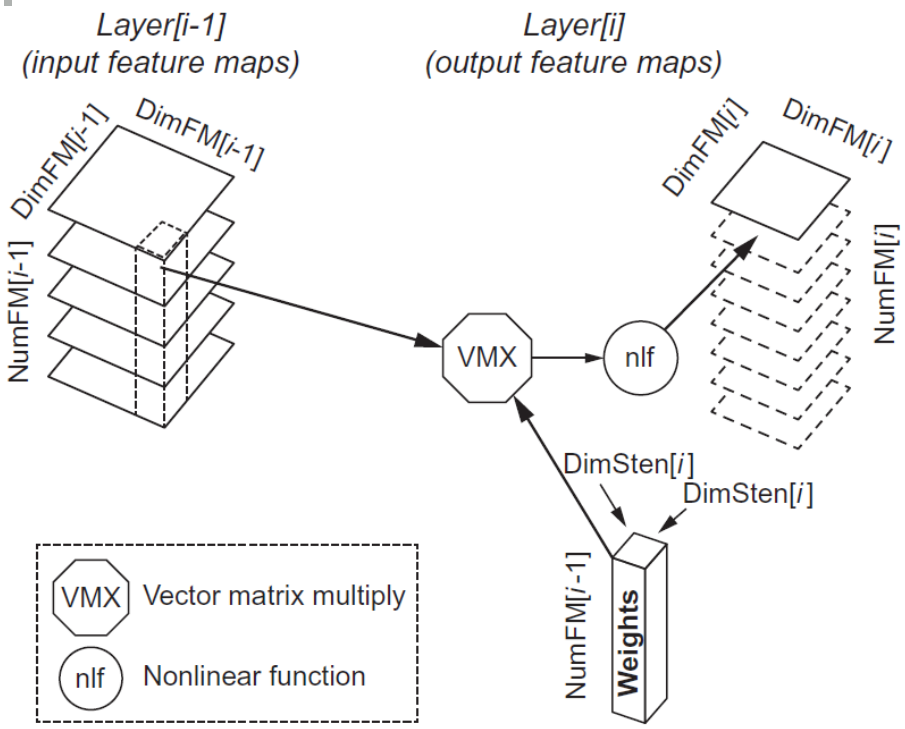


Convolutional Neural Network

- Computer vision
- Each layer raises the level of abstraction
  - First layer recognizes horizontal and vertical lines
  - Second layer recognizes corners
  - Third layer recognizes shapes
  - Fourth layer recognizes features, such as ears of a dog
  - Higher layers recognizes different breeds of dogs



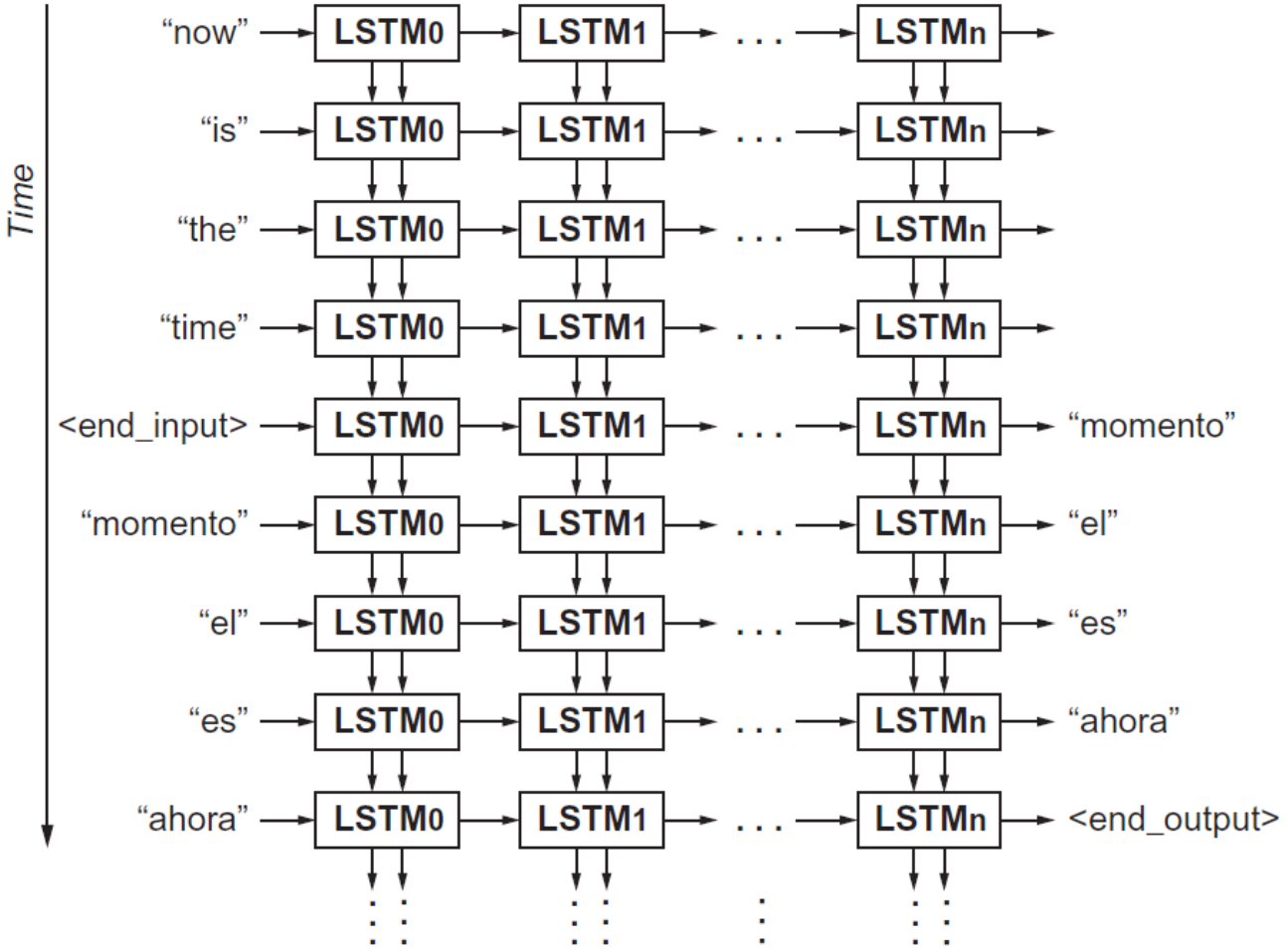
Convolutional Neural Network



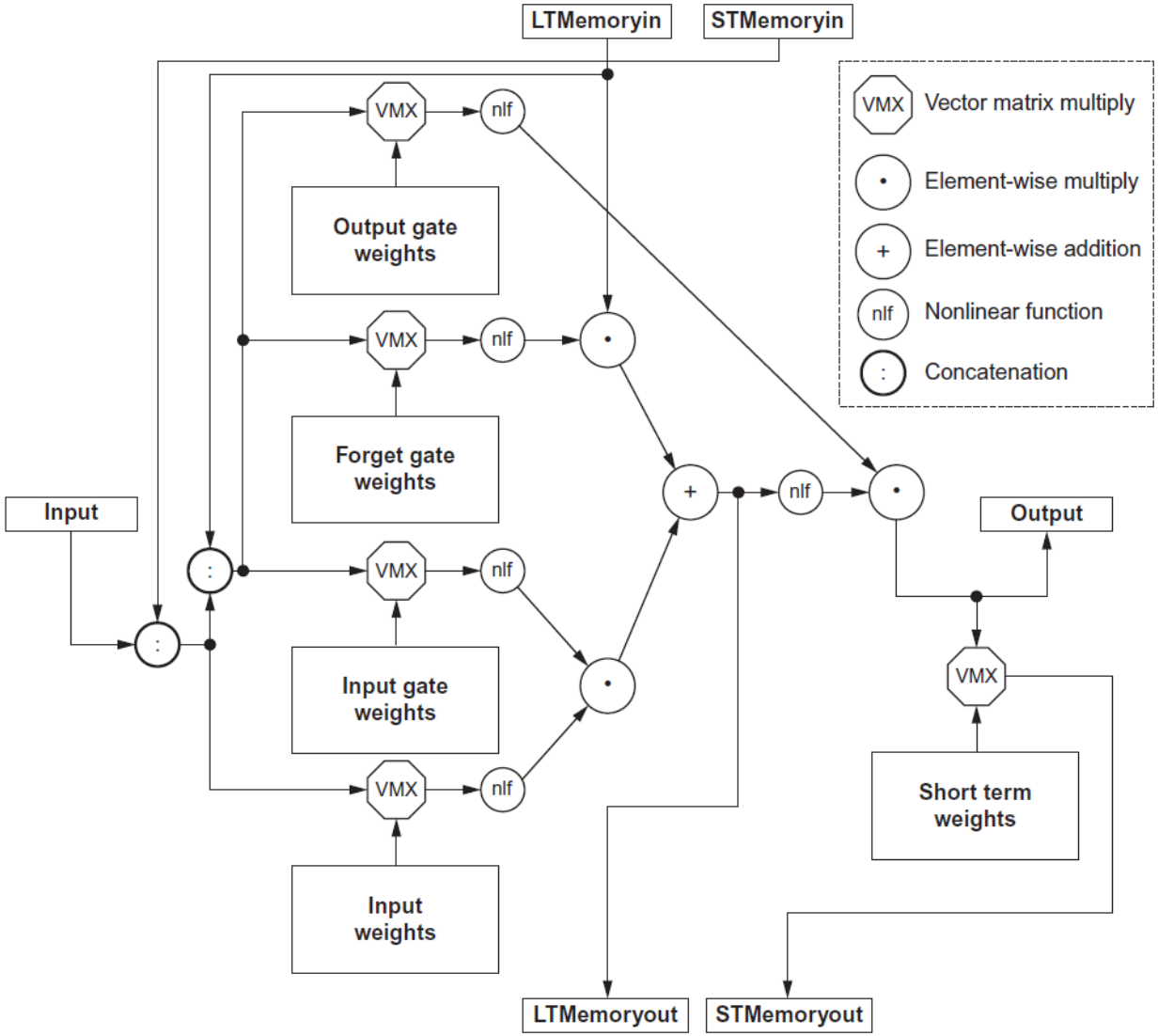
- Parameters:
  - DimFM[i-1]: Dimension of the (square) input Feature Map
  - DimFM[i]: Dimension of the (square) output Feature Map
  - DimSten[i]: Dimension of the (square) stencil
  - NumFM[i-1]: Number of input Feature Maps
  - NumFM[i]: Number of output Feature Maps
  - Number of neurons: NumFM[i] x DimFM[i]^2
  - Number of weights per output Feature Map: NumFM[i-1] x DimSten[i]^2
  - Total number of weights per layer: NumFM[i] x Number of weights per output Feature Map
  - Number of operations per output Feature Map: 2 x DimFM[i]^2 x Number of weights per output Feature Map
  - Total number of operations per layer: NumFM[i] x Number of operations per output Feature Map = 2 x DimFM[i]^2 x NumFM[i] x Number of weights per output Feature Map = 2 x DimFM[i]^2 x Total number of weights per layer
  - Operations/Weight: 2 x DimFM[i]^2

Recurrent Neural Network

- Used for speech recognition and language translation
- Long short-term memory (LSTM) network



Recurrent Neural Network



- Parameters:
  - Number of weights per cell: 3 x (3 x Dim x Dim)+(2 x Dim x Dim) + (1 x Dim x Dim) = 12 x Dim^2
  - Number of operations for the 5 vector-matrix multiplies per cell: 2 x Number of weights per cell = 24 x Dim^2
  - Number of operations for the 3 element-wise multiplies and 1 addition (vectors are all the size of the output): 4 x Dim
  - Total number of operations per cell (5 vector-matrix multiplies and the 4 element-wise operations): 24 x Dim^2 + 4 x Dim
  - Operations/Weight: ~2

Example: Deep Neural Networks

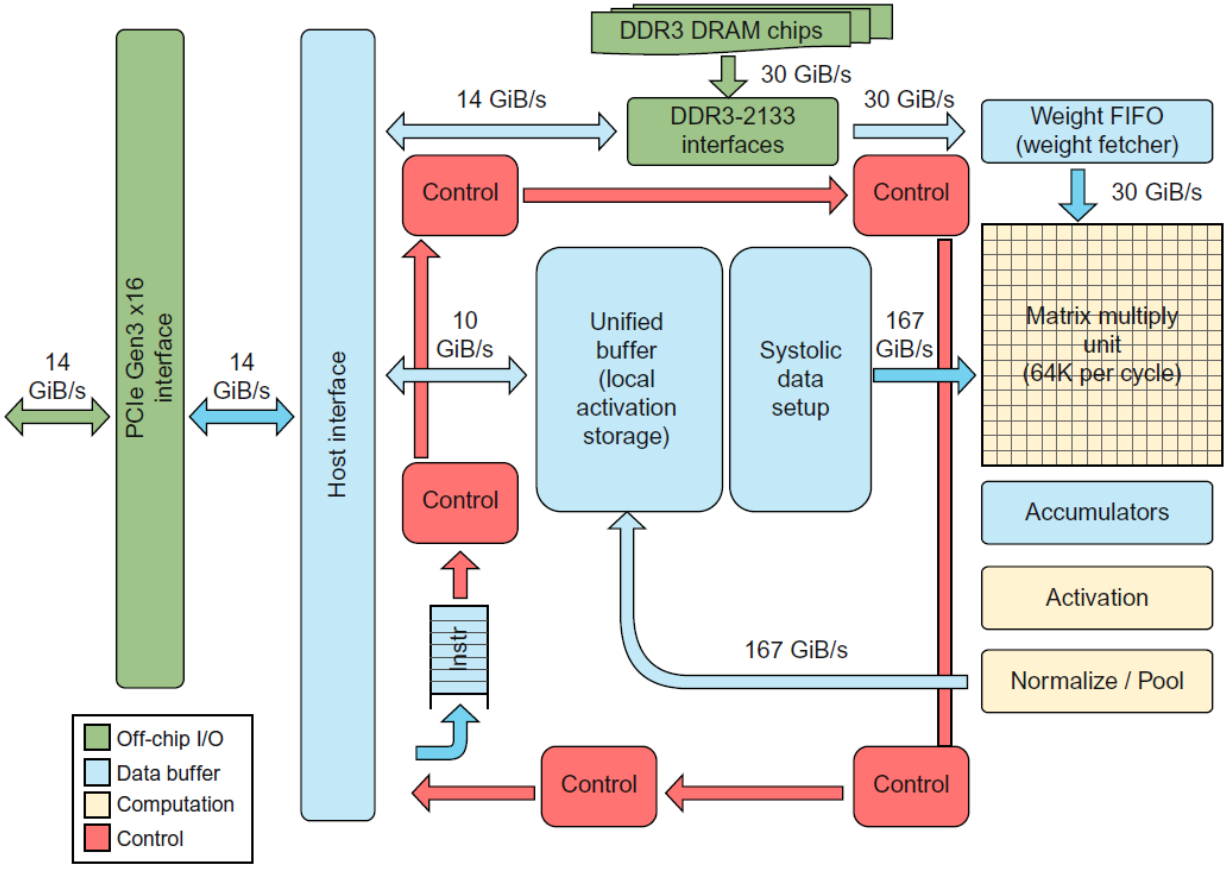
- Batches:
  - Reuse weights once fetched from memory across multiple inputs
  - Increases operational intensity
- Quantization
  - Sufficient to use 8- or 16-bit fixed point
- Summary:
  - Need the following kernels:
    - Matrix-vector multiply
    - Matrix-matrix multiply
    - Stencil computations
    - ReLU activation
    - Sigmoid activation
    - Hyperbolic tangent (tanh) activation

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Tensor Processing Unit

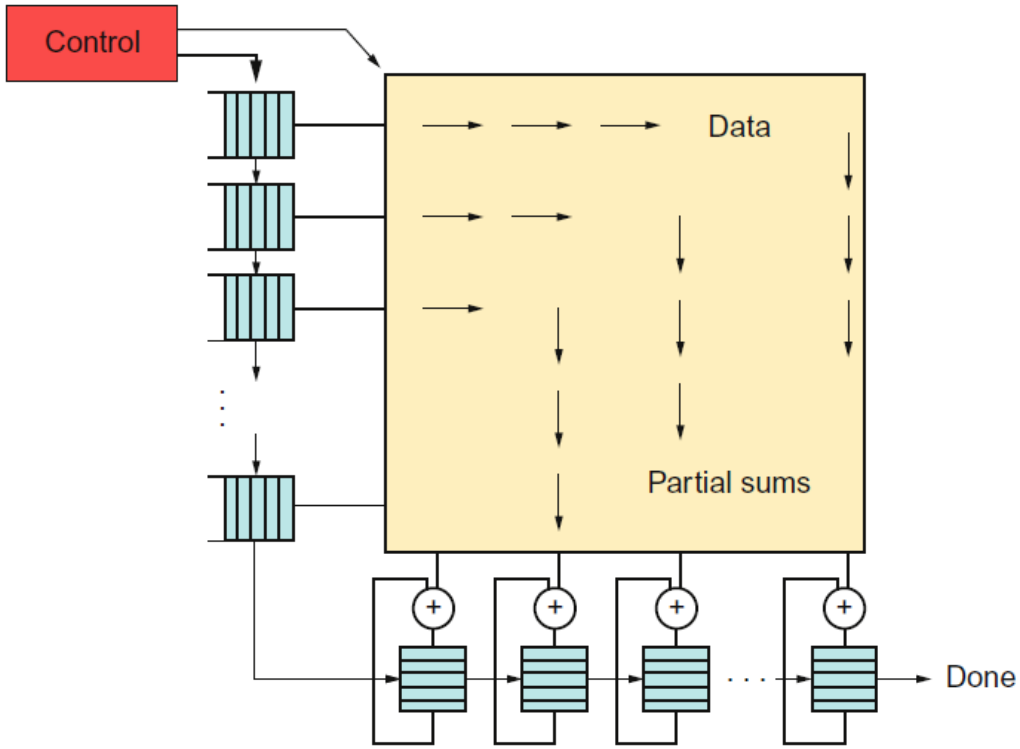
- Google's DNN ASIC
- 256 x 256 8-bit matrix multiply unit
- Large software-managed scratchpad
- Coprocessor on the PCIe bus

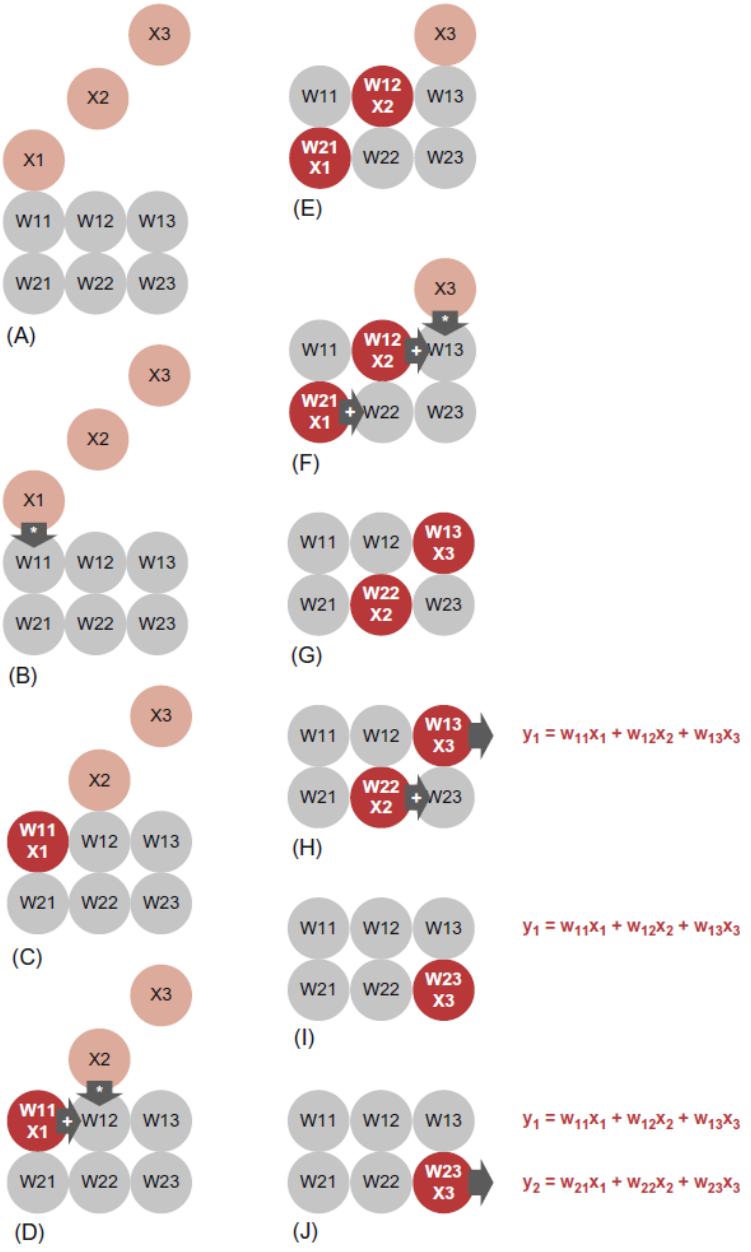


TPU ISA

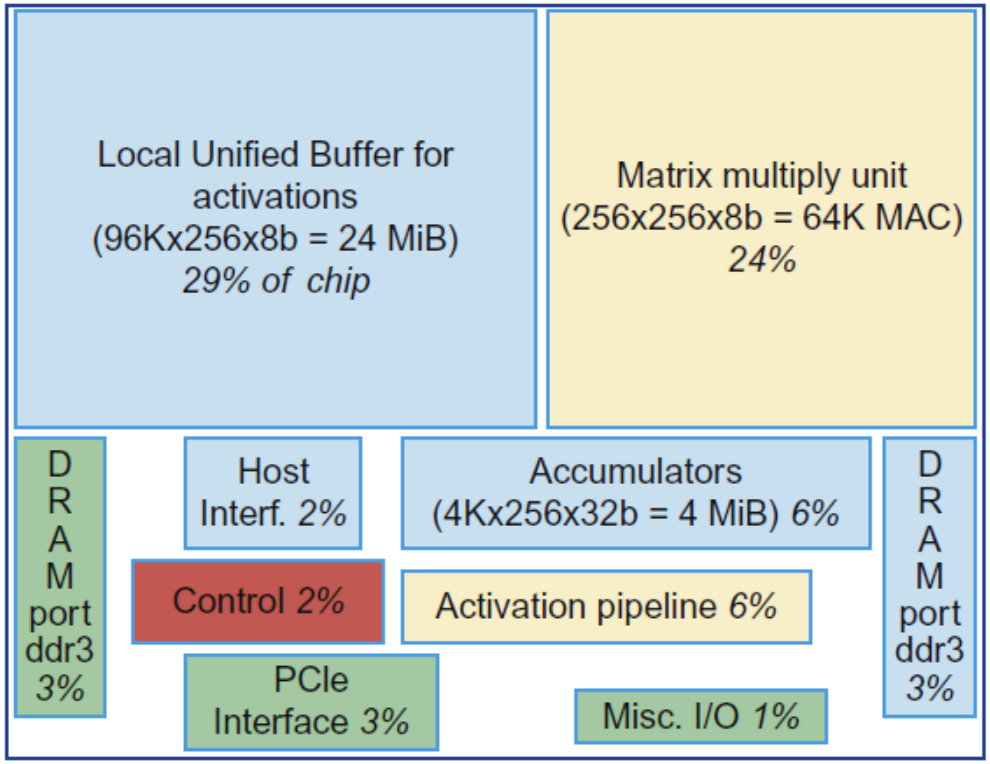
- Read_Host_Memory
  - Reads memory from the CPU memory into the unified buffer
- Read_Weights
  - Reads weights from the Weight Memory into the Weight FIFO as input to the Matrix Unit
- MatrixMatrixMultiply/Convolve
  - Perform a matrix-matrix multiply, a vector-matrix multiply, an element-wise matrix multiply, an element-wise vector multiply, or a convolution from the Unified Buffer into the accumulators
  - takes a variable-sized B*256 input, multiplies it by a 256x256 constant input, and produces a B*256 output, taking B pipelined cycles to complete
- Activate
  - Computes activation function
- Write_Host_Memory
  - Writes data from unified buffer into host memory

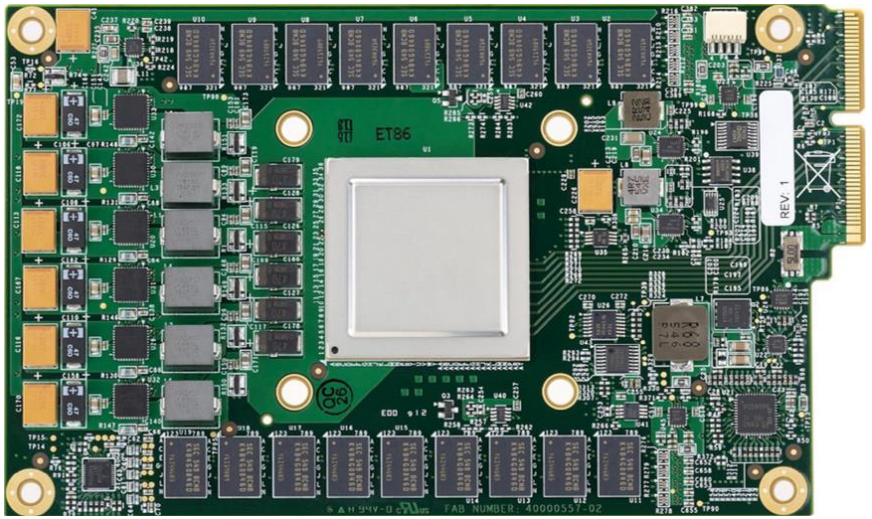
TPU Microarchitecture





TPU Implementation





The TPU and the Guidelines

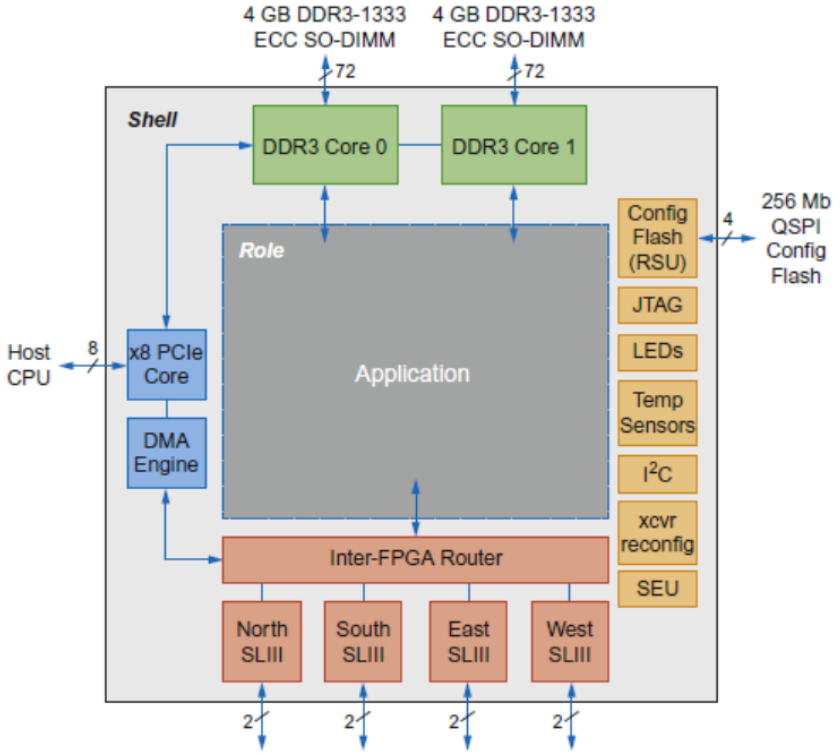
- Use dedicated memories
  - 24 MiB dedicated buffer, 4 MiB accumulator buffers
- Invest resources in arithmetic units and dedicated memories
  - 60% of the memory and 250X the arithmetic units of a server-class CPU
- Use the easiest form of parallelism that matches the domain
  - Exploits 2D SIMD parallelism
- Reduce the data size and type needed for the domain
  - Primarily uses 8-bit integers
- Use a domain-specific programming language
  - Uses TensorFlow

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

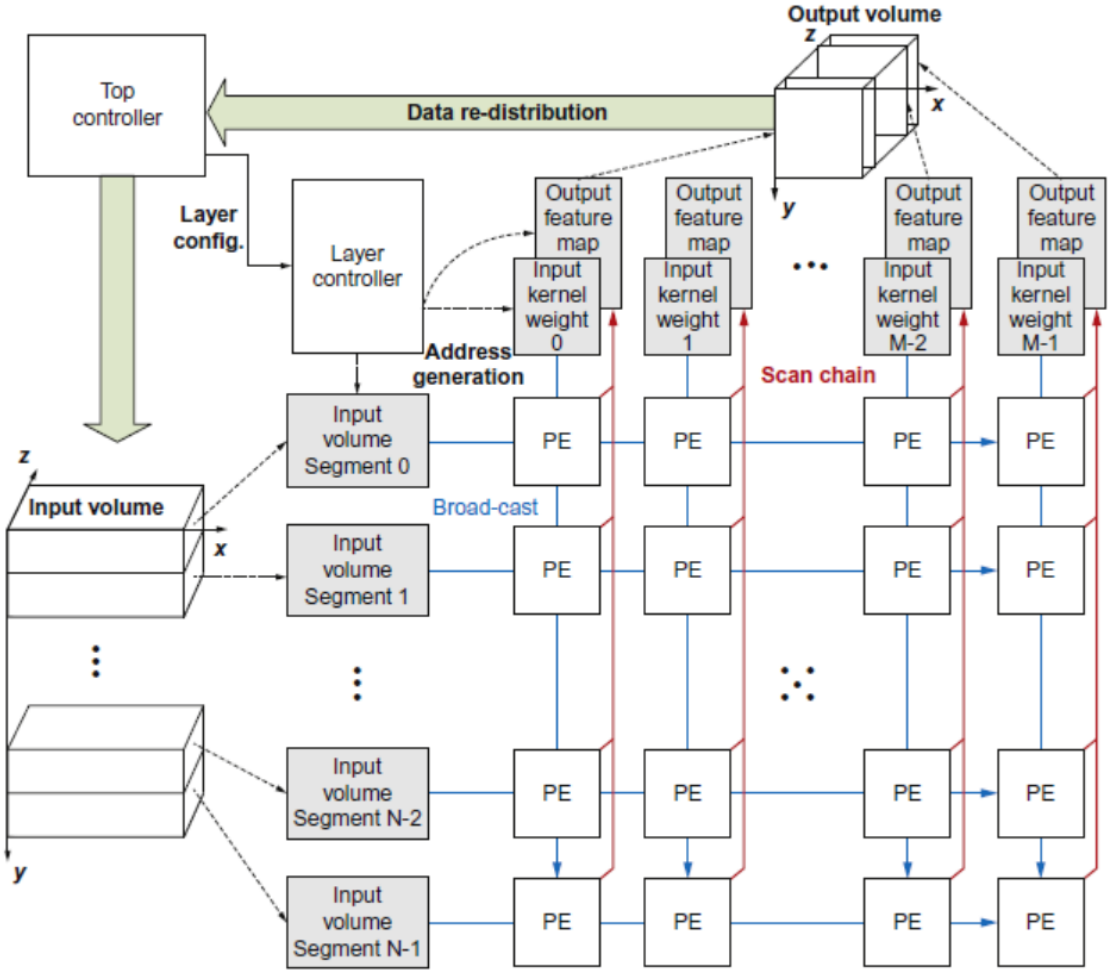
Microsoft Catapult

- Needed to be general purpose and power efficient
  - Uses FPGA PCIe board with dedicated 20 Gbps network in 6 x 8 torus
  - Each of the 48 servers in half the rack has a Catapult board
  - Limited to 25 watts
  - 32 MiB Flash memory
  - Two banks of DDR3-1600 (11 GB/s) and 8 GiB DRAM
  - FPGA (unconfigured) has 3962 18-bit ALUs and 5 MiB of on-chip memory
  - Programmed in Verilog RTL
  - Shell is 23% of the FPGA

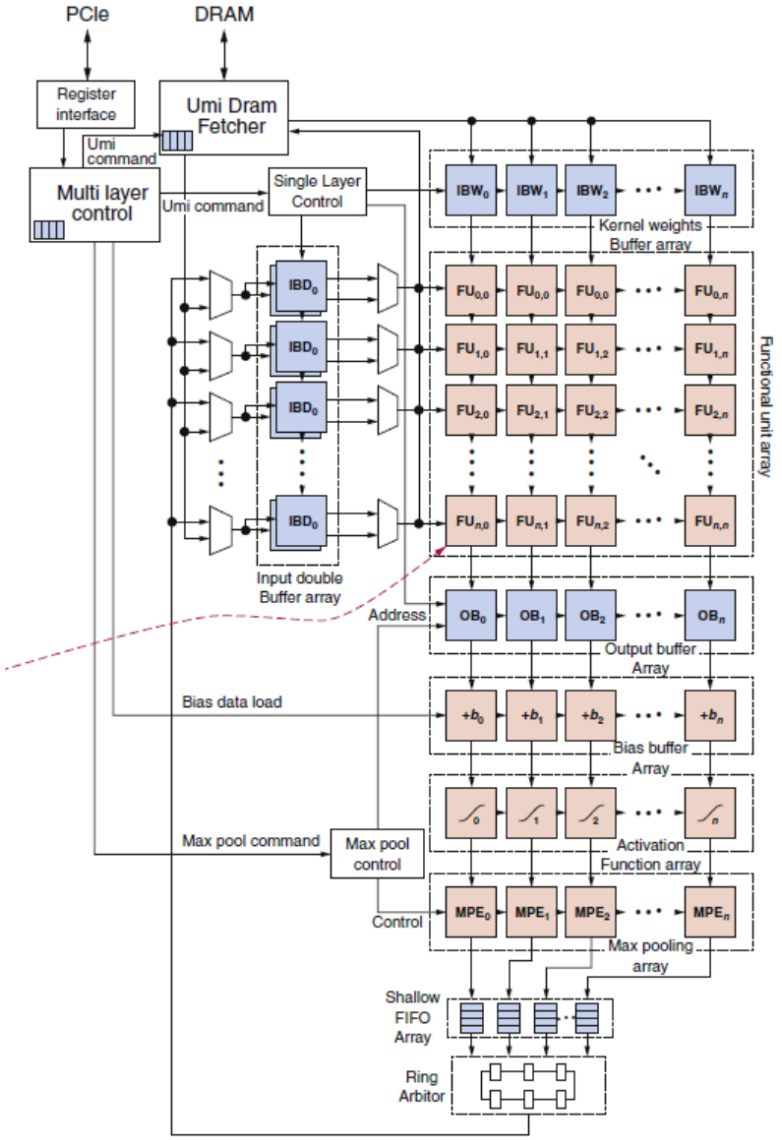


Microsoft Catapult: CNN

- CNN accelerator, mapped across multiple FPGAs

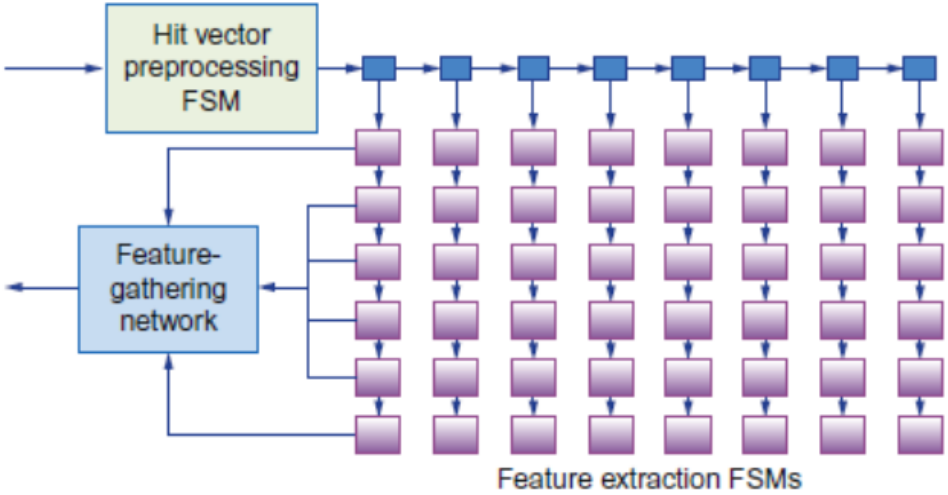


Microsoft Catapult: CNN



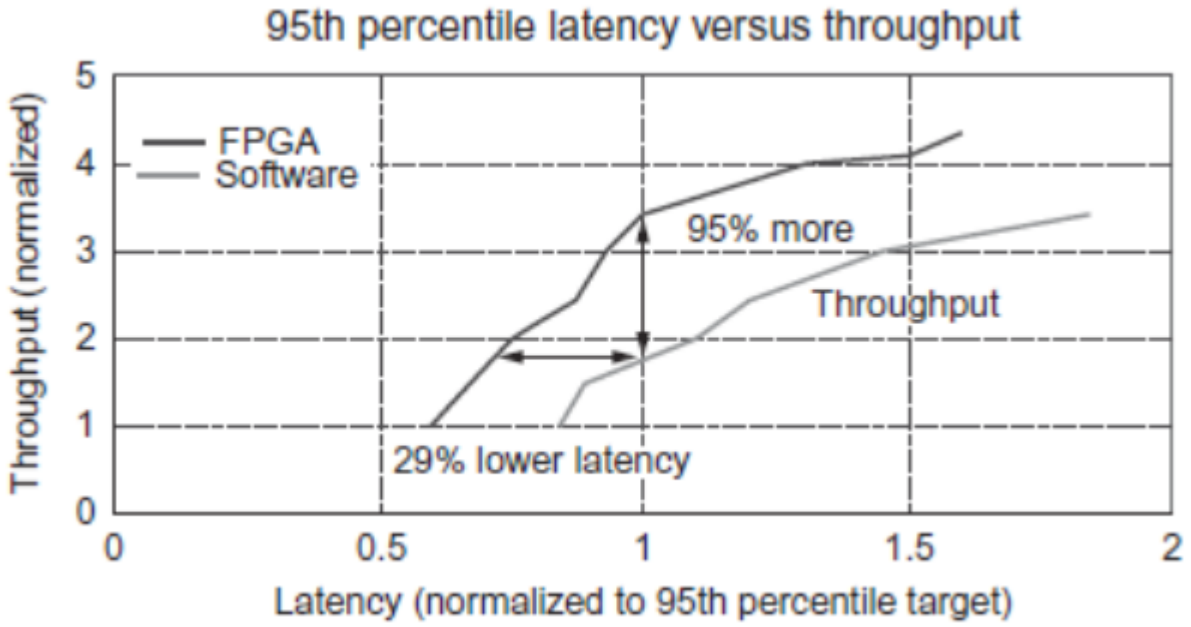
Microsoft Catapult: Search Ranking

- Feature extraction (1 FPGA)
  - Extracts 4500 features for every document-query pair, e.g. frequency in which the query appears in the page
  - Systolic array of FSMs
- Free-form expressions (2 FPGAs)
  - Calculates feature combinations
- Machine-learned Scoring (1 FPGA for compression, 3 FPGAs calculate score)
  - Uses results of previous two stages to calculate floating-point score
- One FPGA allocated as a hot-spare



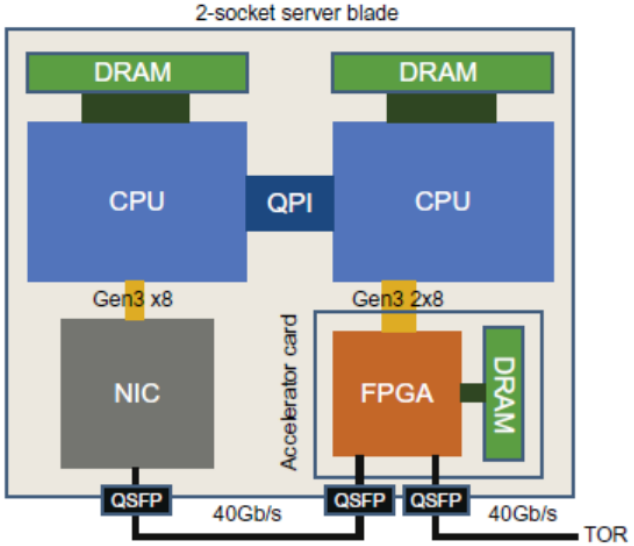
Microsoft Catapult: Search Ranking

- Free-form expression evaluation
  - 60 core processor
  - Pipelined cores
  - Each core supports four threads that can hide each other's latency
  - Threads are statically prioritized according to thread latency



Microsoft Catapult: Search Ranking

- Version 2 of Catapult
  - Placed the FPGA between the CPU and NIC
  - Increased network from 10 Gb/s to 40 Gb/s
  - Also performs network acceleration
  - Shell now consumes 44% of the FPGA
  - Now FPGA performs only feature extraction



Catapult and the Guidelines

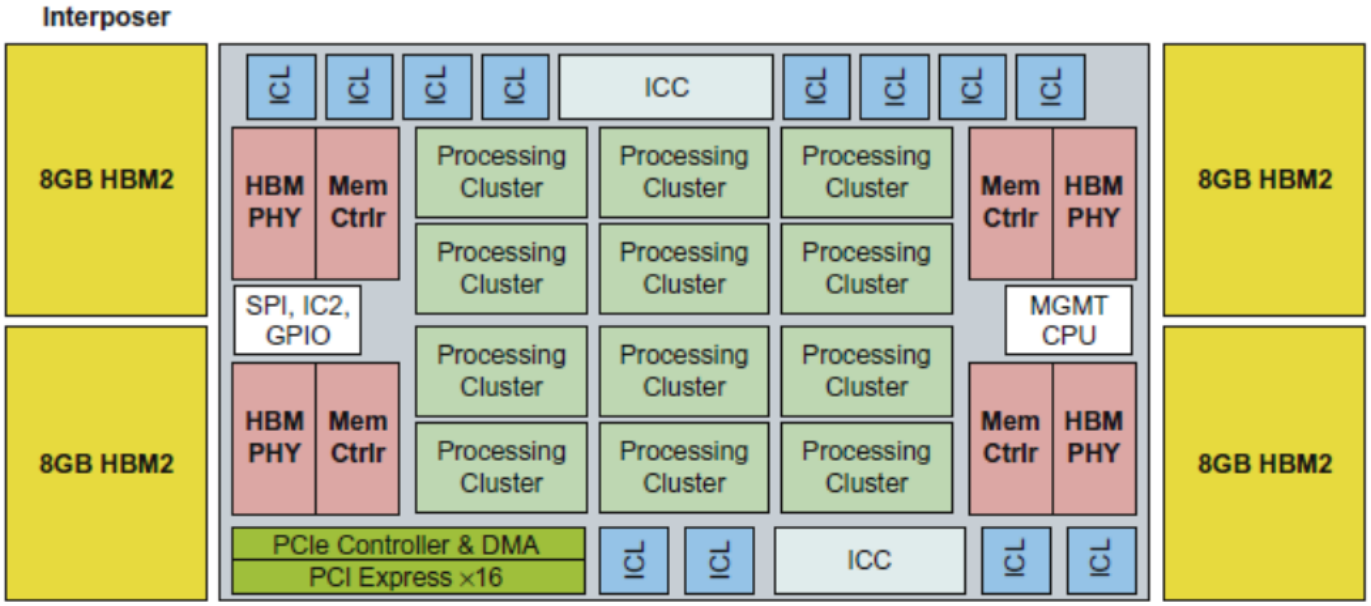
- Use dedicated memories
  - 5 MiB dedicated memory
- Invest resources in arithmetic units and dedicated memories
  - 3926 ALUs
- Use the easiest form of parallelism that matches the domain
  - 2D SIMD for CNN, MISD parallelism for search scoring
- Reduce the data size and type needed for the domain
  - Uses mixture of 8-bit integers and 64-bit floating-point
- Use a domain-specific programming language
  - Uses Verilog RTL; Microsoft did not follow this guideline

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Intel Crest

- DNN training
- 16-bit fixed point
- Operates on blocks of 32x32 matrices
- SRAM + HBM2

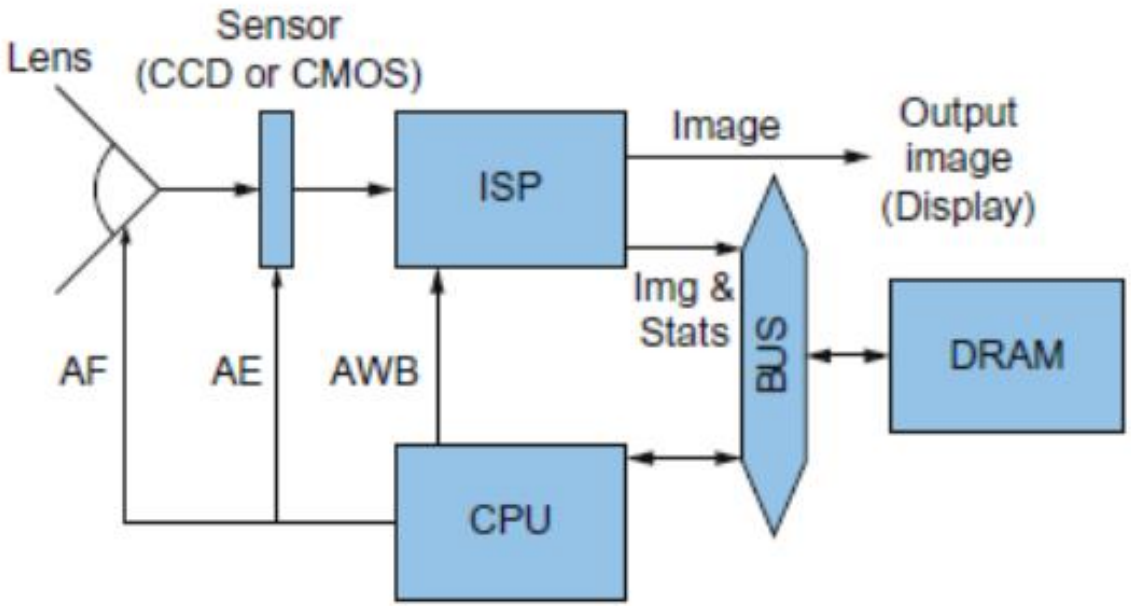


Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Pixel Visual Core

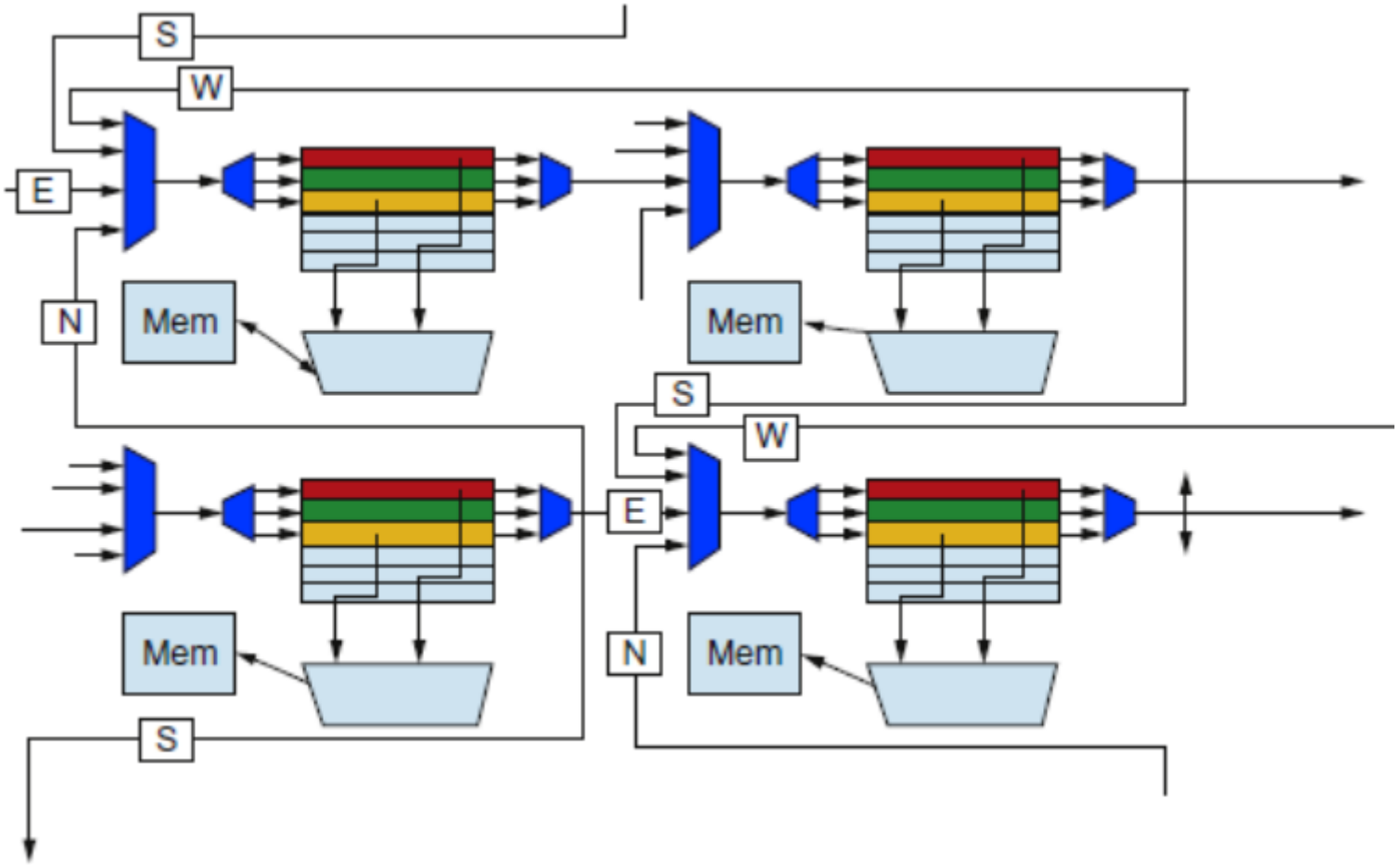
- Pixel Visual Core
  - Image Processing Unit
  - Performs stencil operations
  - Decended from Image Signal processor



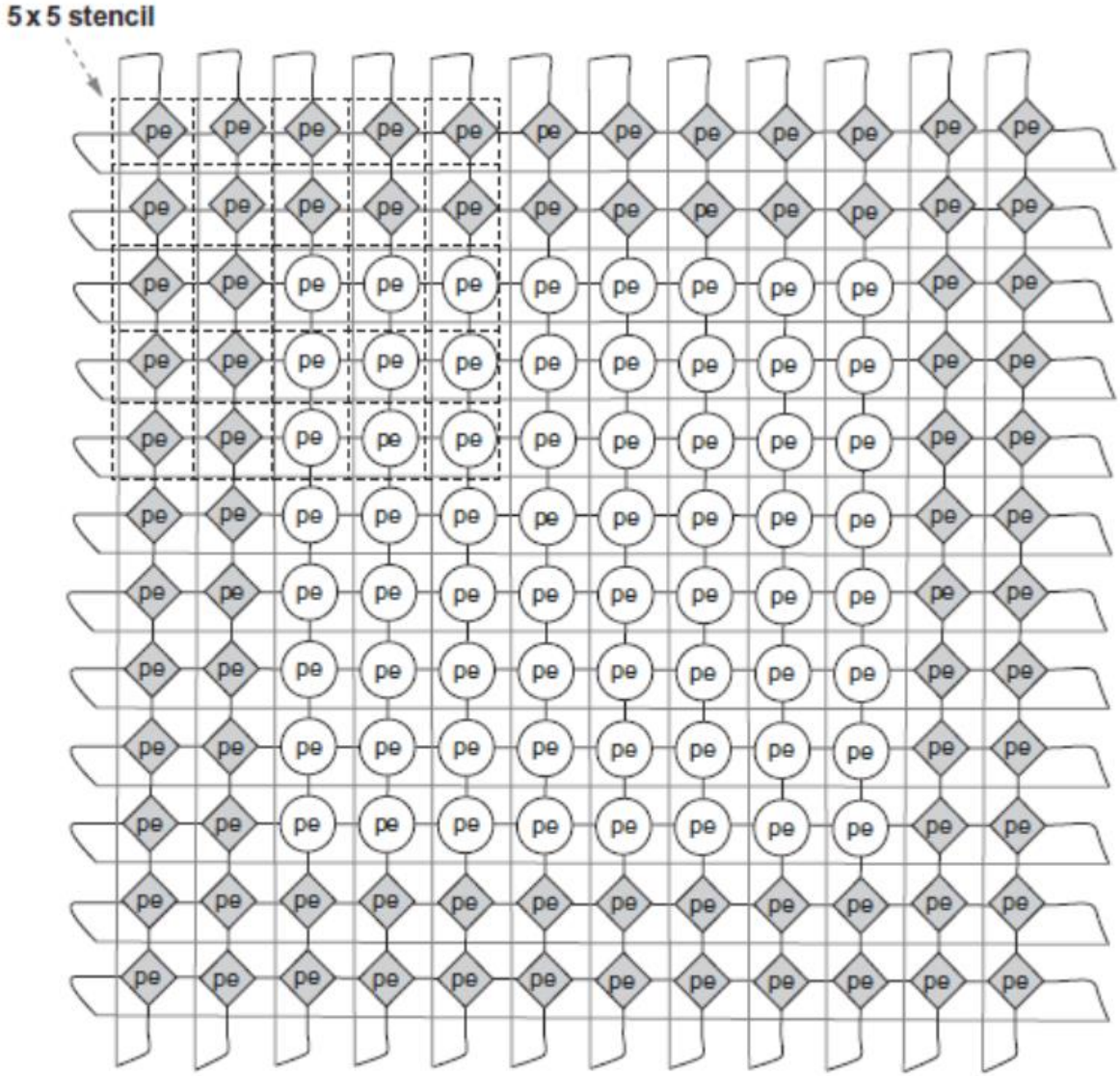
Pixel Visual Core

- Software written in Halide, a DSL
  - Compiled to virtual ISA
  - vISA is lowered to physical ISA using application-specific parameters
  - pISA is VLSI
- Optimized for energy
  - Power Budget is 6 to 8 W for bursts of 10-20 seconds, dropping to tens of milliwatts when not in use
  - 8-bit DRAM access equivalent energy as 12,500 8-bit integer operations or 7 to 100 8-bit SRAM accesses
  - IEEE 754 operations require 22X to 150X of the cost of 8-bit integer operations
- Optimized for 2D access
  - 2D SIMD unit
  - On-chip SRAM structured using a square geometry

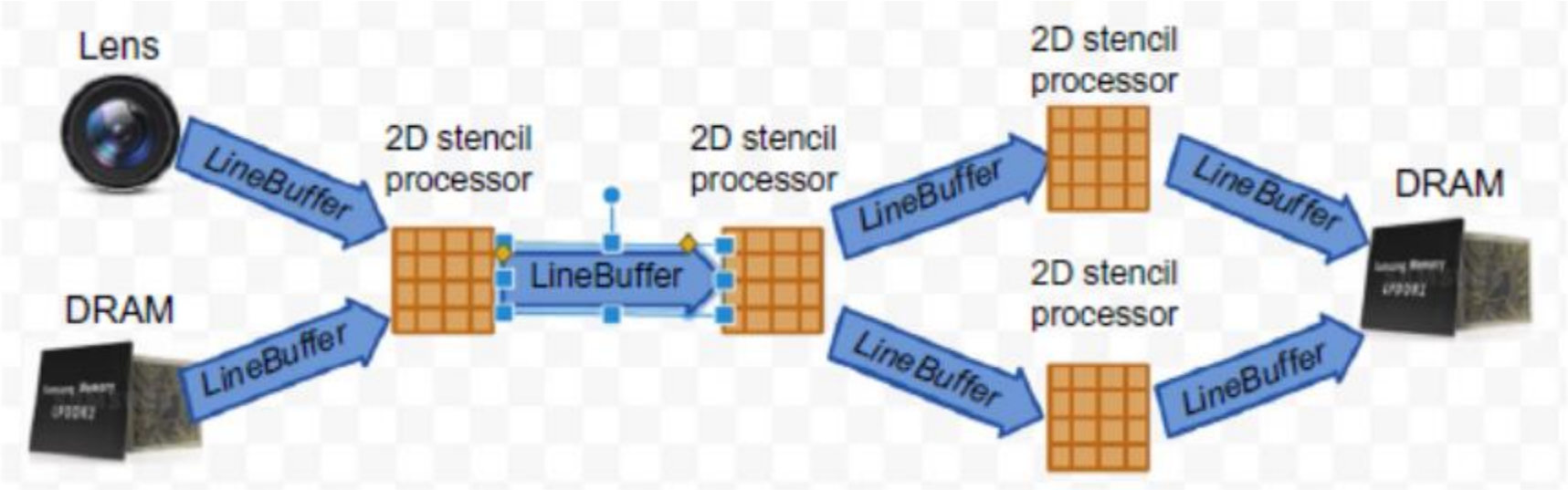
Pixel Visual Core



Pixel Visual Core



Pixel Visual Core



Visual Core and the Guidelines

- Use dedicated memories
  - 128 + 64 MiB dedicated memory per core
- Invest resources in arithmetic units and dedicated memories
  - 16x16 2D array of processing elements per core and 2D shifting network per core
- Use the easiest form of parallelism that matches the domain
  - 2D SIMD and VLIW
- Reduce the data size and type needed for the domain
  - Uses mixture of 8-bit and 16-bit integers
- Use a domain-specific programming language
  - Halide for image processing and TensorFlow for CNNs

Contents

- Introduction
- Guidelines for DSAs
- Example Domain: Deep Neural Networks
- Google's Tensor Processing Unit, an Inference Data Center Accelerator
- Microsoft Catapult, a Flexible Data Center Accelerator
- Intel Crest, a Data Center Accelerator for Training
- Pixel Visual Core, a Personal Mobile Device Image Processing Unit
- Fallacies and Pitfalls

Fallacies and Pitfalls

- F: It costs $100 million to design a custom chip
- P: Performance counters added as an afterthought
- F: Architects are tackling the right DNN tasks
- F: For DNN hardware, inferences per second (IPS) is a fair summary performance metric
- P: Being ignorant of architecture history when designing an DSA