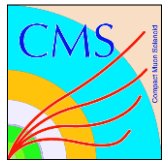


UF | UNIVERSITY of FLORIDA

Simplified Deep Learning Framework with DAAL and OpenCL: towards Hardware Applications

David Ojika

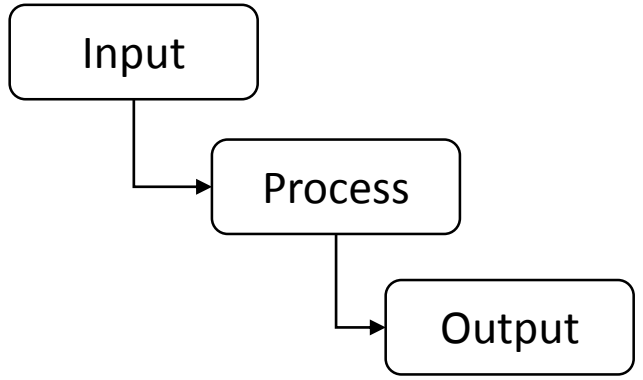
PhD Candidate, University of Florida; Intel Fellow

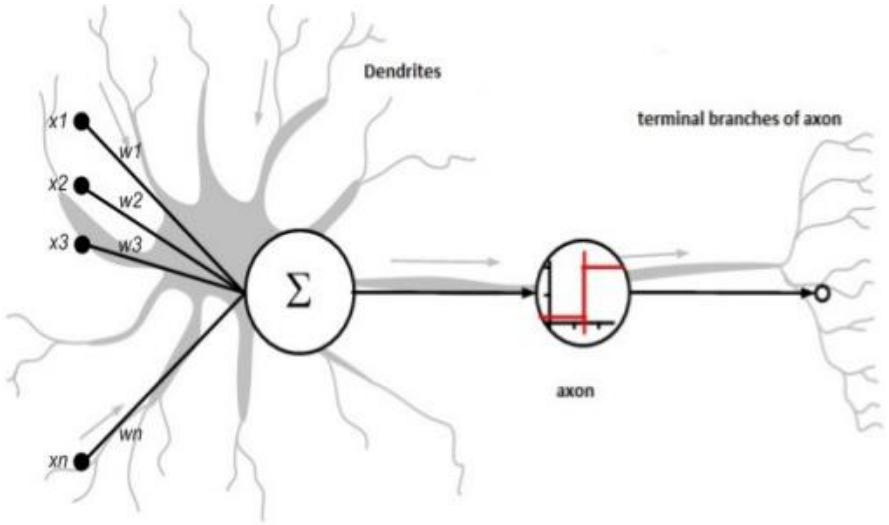
Outline

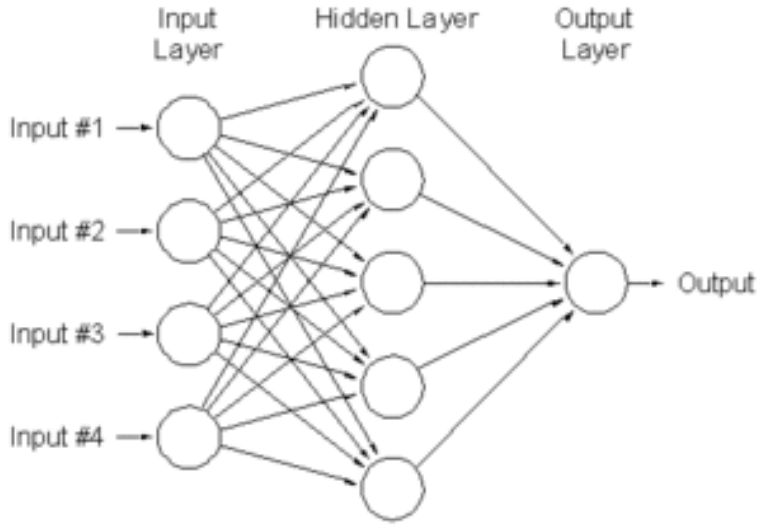
- General overview of deep learning
- Current application of ML in physics
  - Deep learning, ROOT
- Intel DAAL
  - Overview
- DL with DAAL for Higgs/background classification
  - Setup
- Towards HW implementation
  - HPC with OpenCL
  - Benchmarking
- Deep Learning SDK Demo

Introduction to Neural Networks

- Neural Network
  - Data processing system to approximate functions of large number of parameters
  - Consists of interconnected computational devices (neurons – inspired by biology)



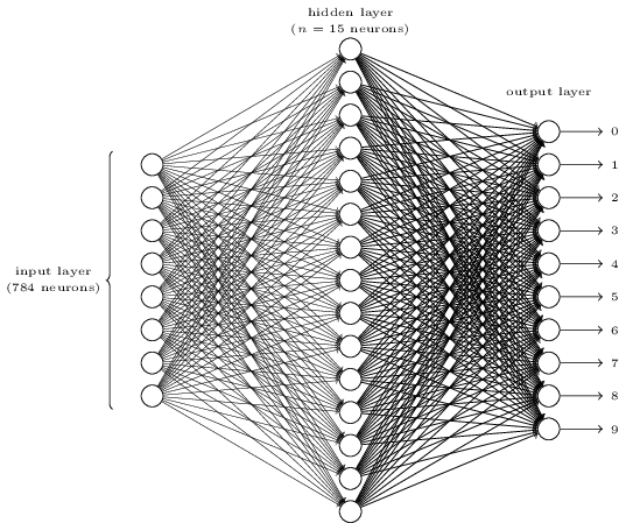




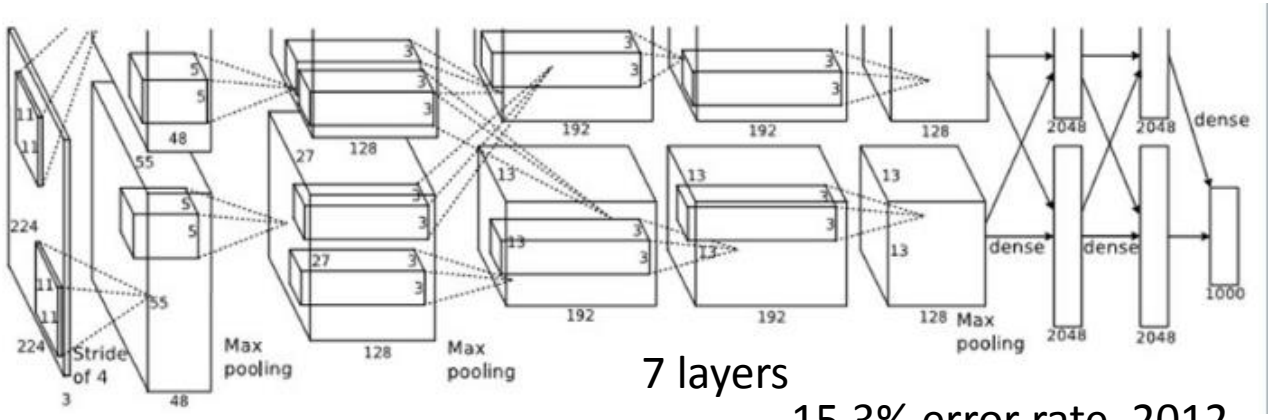
Network model

Deep Learning

- From 1 layer with few neurons to multiple layers with millions/billions of neurons

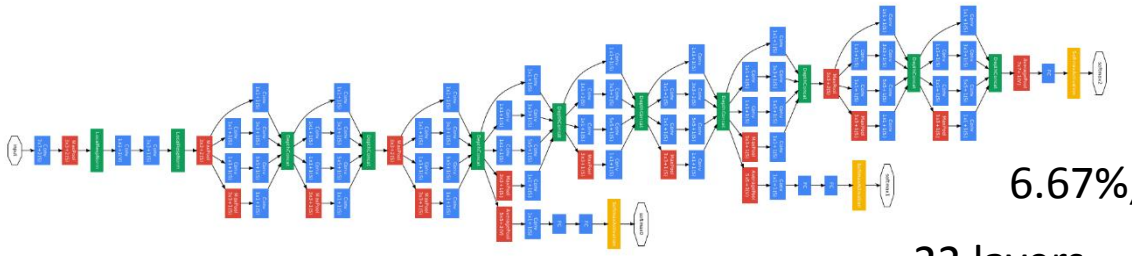


1 (hidden) layer



7 layers

15.3% error rate, 2012



6.67%, 2014

22 layers

Handwriting Recognition

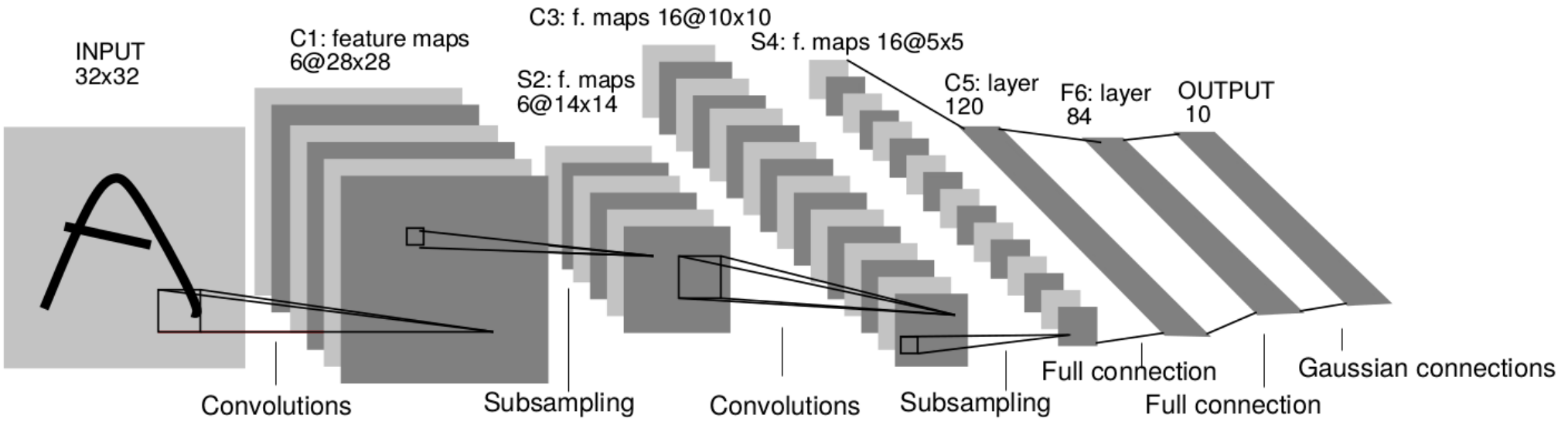
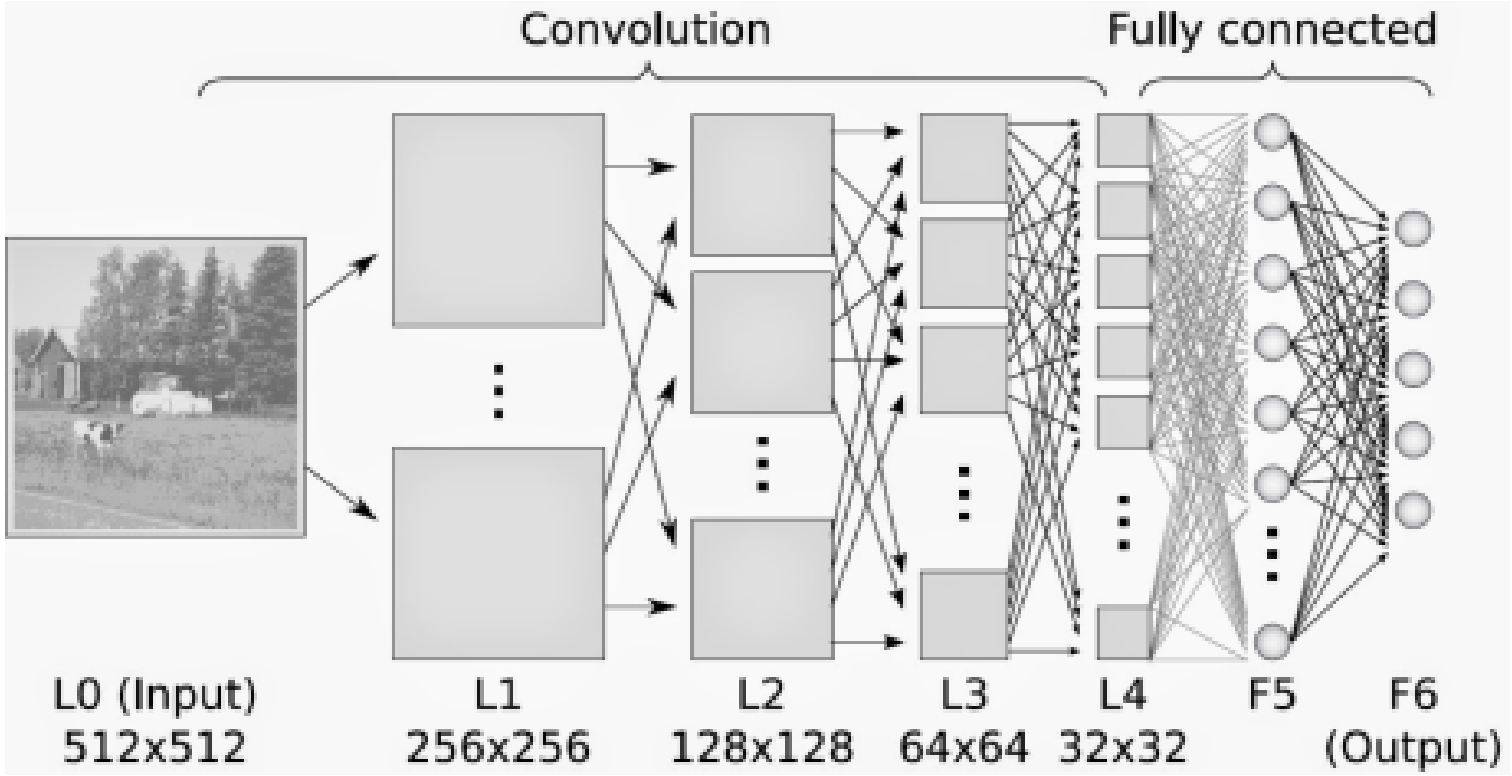


Image / Object Recognition



ImageNet Competition

www.gatorvision.org

Deep Learning in HEP

	AUC		
Technique	Low-level	High-level	Complete
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (< 0.001)	0.885 (0.002)

	AUC		
Technique	Low-level	High-level	Complete
NN 300-hidden	0.733	0.777	0.816
NN 1000-hidden	0.788	0.783	0.841
NN 2000-hidden	0.787	0.788	0.842
NN 10000-hidden	0.790	0.789	0.841
DN 3 layers	0.836	0.791	0.850
DN 4 layers	0.868	0.797	0.872
DN 5 layers	0.880	0.800	0.885
DN 6 layers	0.888	0.799	0.893

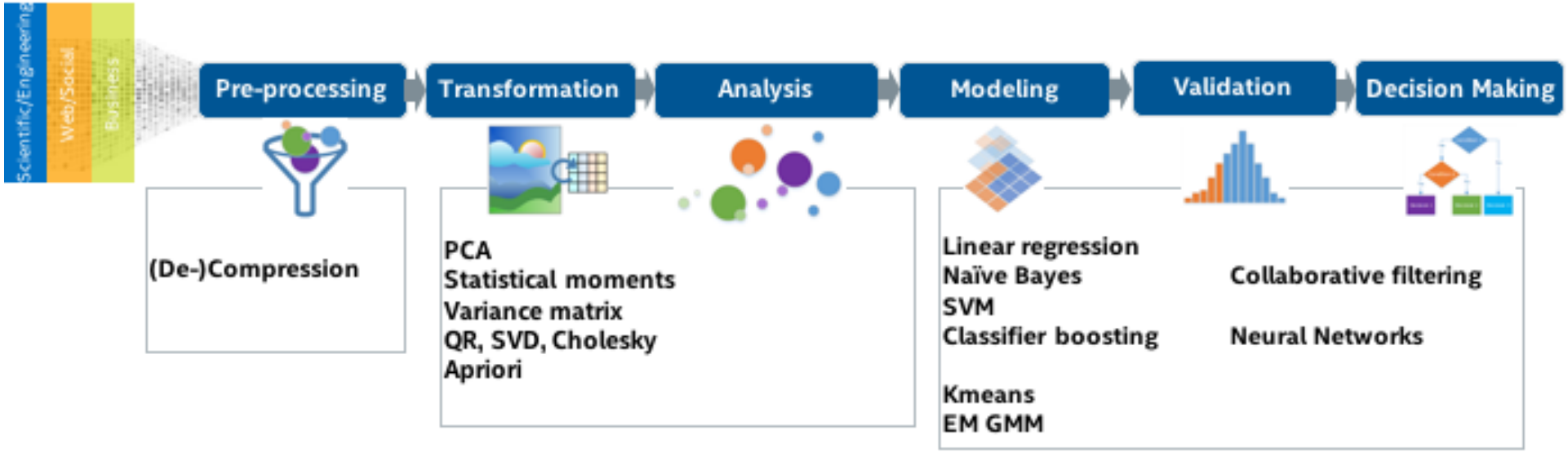
"Searching for Exotic Particles in High-Energy Physics with Deep Learning". P. Baldi, [2014] et al.

- 5-layer neural network with 300 hidden units in each layer with 2.6 million training examples
  - 8% improvement over best current approaches

Scholarly work on Deep Learning in HEP

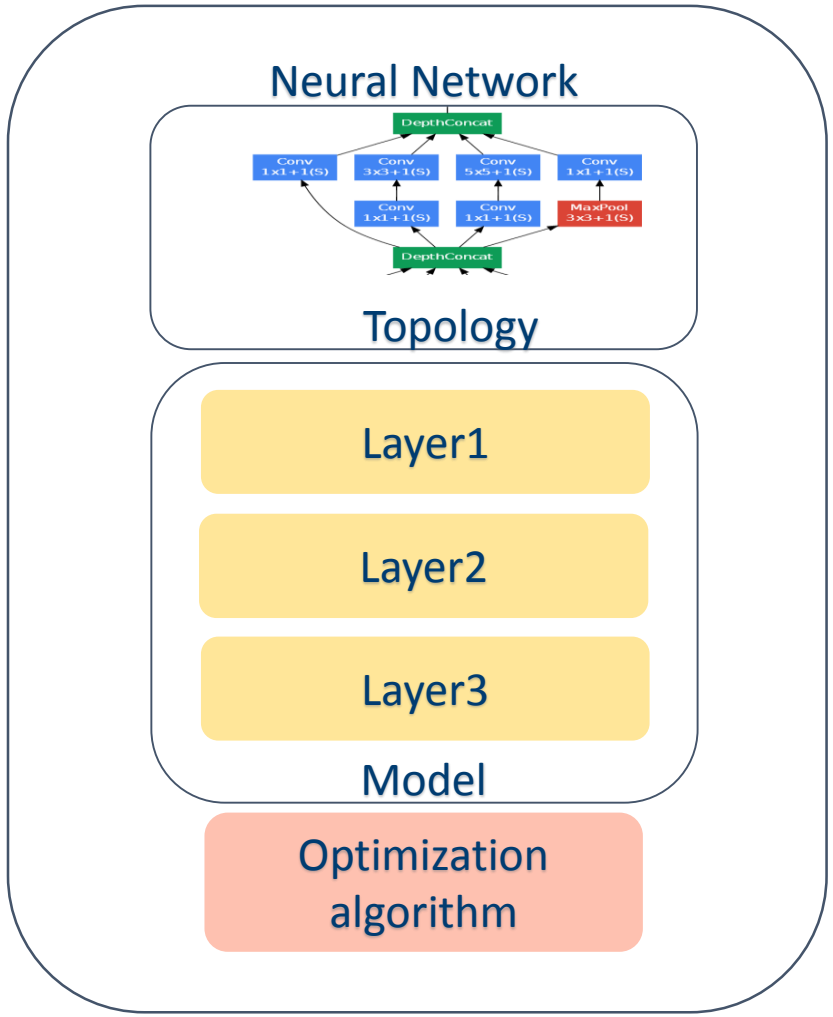
- Jet images and deep learning: https://arxiv.org/abs/1511.05190
- Jet substructure and deep learning: http://inspirehep.net/record/1437937/
- Parton shower uncertainties and jet substructure: http://inspirehep.net/record/1485081?ln=en
- Deep learning for ttH searches http://inspirehep.net/record/1491175?ln=en
- NOVa http://inspirehep.net/record/1444342
- Daya Bay https://arxiv.org/abs/1601.07621
- NEXT: http://inspirehep.net/record/1487439?ln=en
- Microboone: http://inspirehep.net/record/1498561?ln=en

Intel Data Analytics and Acceleration Library (DAAL)



- Optimized functions for deep learning and classical machine learning
- Language API for C++, Java and Python for Linux and Windows
- Support data ingress from Hadoop and Spark
- Free and open-source versions available

Intel DAAL



- **Layer**: NN building block
- **Model**: Set of layers
- **Optimization:** Objective function /solver
- **Topology:** NN description
- **NN:** Topology, model & optimization algorithm
- **Tensor:** Multidimensional data structure

Common layers	Activation	Normalization	Optimization / Solver
Convolutional	Logistic	Z-score	MSE
Pooling (max, average)	Hyperbolic tangent	Batch	Cross entropy
Fully connected	ReLU, pReLU, smooth ReLU	Local response	Mini batch SGD
	Softmax		Stochastic LBFGS
Dropout	Abs		

DAAL API Example

- Layer

```
SharedPtr<layers::fullyconnected::Batch<> > fcLayer1(new fullyconnected::Batch<>(20));
```

- Topology

```
SharedPtr<layers::fullyconnected::Batch<> > fcLayer1(new fullyconnected::Batch<>(20));
Collection<LayerDescriptor> configuration;
configuration.push_back(LayerDescriptor(0, fcLayer1, NextLayers(1)));
```

- Optimization Solver

```
services::SharedPtr<optimization_solver::mse::Batch<double> > mseObjectiveFunction(new
optimization_solver::mse::Batch<double>(nVectors));
optimization_solver::sgd::Batch<> sgdAlgorithm(mseObjectiveFunction);
```

- Model

```
trainingNet.compute();
… … ….
services::SharedPtr<training::Model> tModel = trainingNet.getResult()->get(model)
services::SharedPtr<prediction::Model> pModel = tModel->getPredictionModel();
```

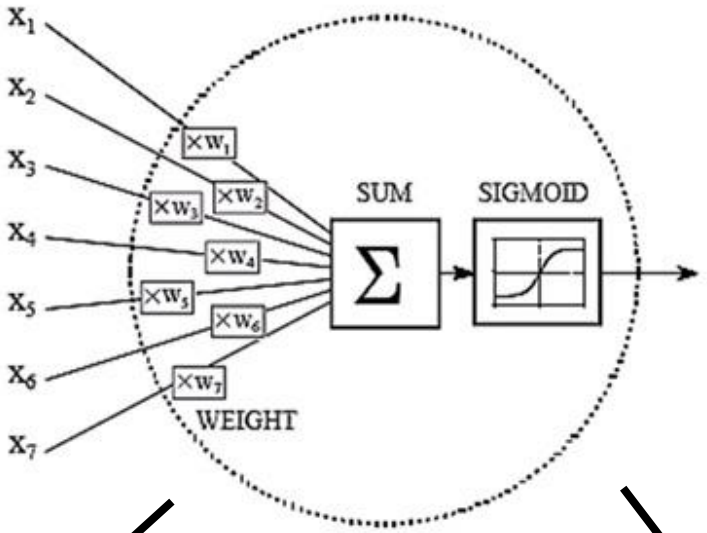
Experimental Setup with DAAL

	D. Ojika et. al	**P. Baldi, et. al**
Platform	Xeon Phi (KNL)	Tesla 2070 (GPU)
Host	Self-hosted (self-boot node)	Xeon CPU
Memory	64 GB (+ 2GB MCDRAM*)	64 GB
Library	MKL	CUDA + Pylearn2
Framework	DAAL	Theano
Dataset: Higgs	10.5M training; 500K validation	2.6M training; 500K validation

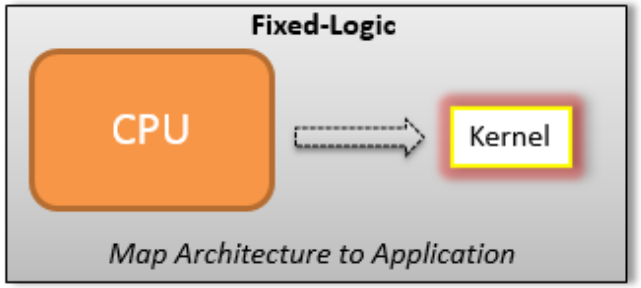
**MCDRAM (a high-bandwidth memory) could provide additional performance benefits when acting in cache mode.*

- Benchmark also to be performed against latest release of ROOT, for both single-node and multi-node configurations

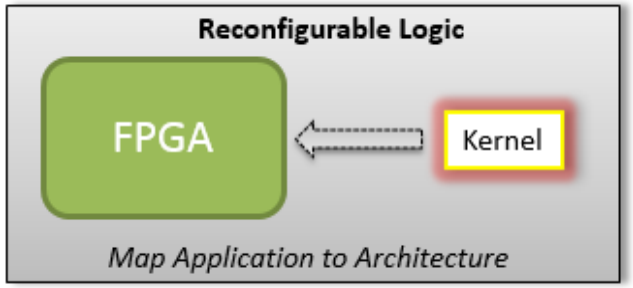
Towards H/W implementation



Neural network topology



Model exploration



GOAL

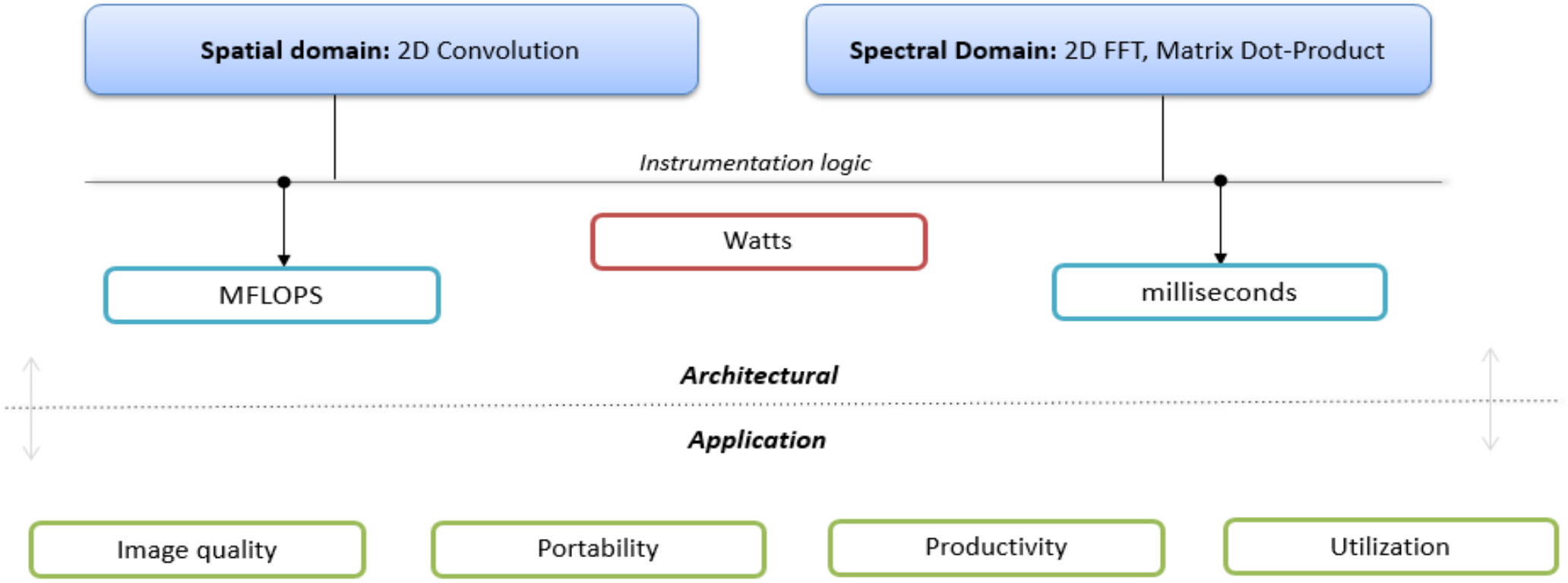
OpenCL-based DL Implementation

- Kernel Implementation
  - **Matrix-Multiplication**
  - 2-D Convolution
  - 2-D FFT
- Micro-benchmarking
  - Global Memory Bandwidth
  - Memory Latency
  - Latency Per Operation
  - Local Memory Bandwidth

All possibilities of cross-product of performance metrics with bandwidth required

{Glops/s, … } x {Cache, DRAM, PCIe, Network } → ***Micro-benchmarks***

Evaluation Metrics



- Functional portability
- Performance portability
- Normalized performance

- Improvement over native lang.

- FPGA resource utilization

Devices under Study

Device	OpenCL Device Type
Intel Xeon Phi	Accelerator
Altera Stratix-5	Accelerator
Intel Xeon E5-2620 (Host)	Multicore CPU

Table 1: List of OpenCL devices evaluated

Extensive memory subsystem

	Xeon Phi	Stratix-5
Memory Type	GDDR5	GDDR3
Memory Interconnect	Ring Bus	Proprietary
Memory Channels	16	2
Memory Capacity (GB)	8	8
Peak Memory B/W (GB/s)	320	10
SIMD (4-byte wide per lane)	16	4
PCI	2.0 x16 lanes	3.0 x8 lanes
Bus Bandwidth (GB/s)	8	7.7

Table 2: Architectural differences between two accelerators

Kernel	Domain
2-D Convolution	Time
2-D FFT	Frequency
Matrix-Multiplication	Frequency

Table 4: Kernel set used in Edge detection application

**Table 3: info. as returned by OpenCL API function calls.*

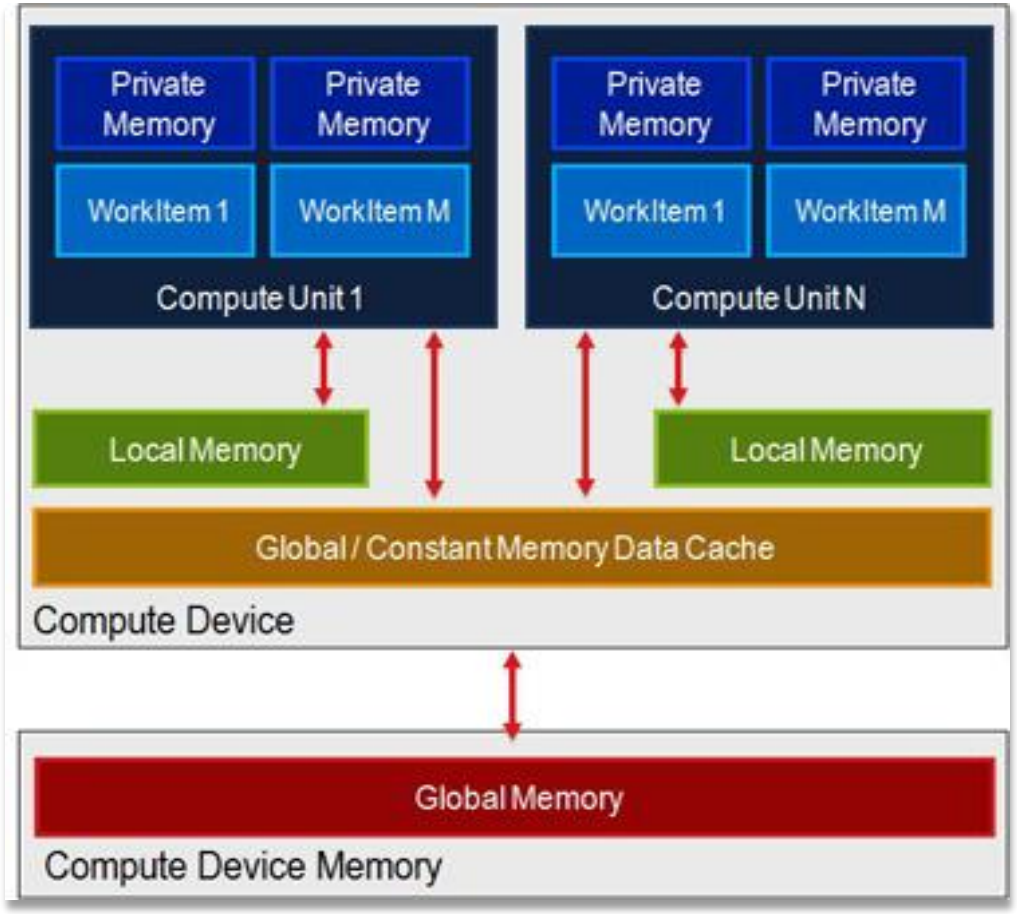
Susceptible to vectorization, but limited by PCI memory B/W

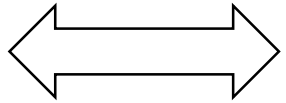
	Xeon Phi	Xeon E-5
Generic		
Type	Accelerator	CPU
Clock (GHZ)	1052	2000
Compute Units	236	12
Max Workgroup Size	8192	8192
Memory Alignment Size	1024	1024
Image Support	No	Yes
SIMD		
Vector Engine Width (Int)	4	16
Vector Engine Width (Float)	8	16
Vector Engine Width (Double)	4	8
Memory		
Global Memory Cache Line Size (B)	64	64
Global Memory Cache Size (KB)	256	256
Global Memory Size (GB)	5.64	31.33
Constant Memory Size (KB)	128	128
Local Memory Size (KB)	32	32

Table 3: Architectural differences between an accelerator and a CPU

Intel Xeon Phi is the Knights Corner (KNC) edition. Work in progress to port code to KNL;
As well as migrate code from Statix-5 to Arria-10 / Stratix-10

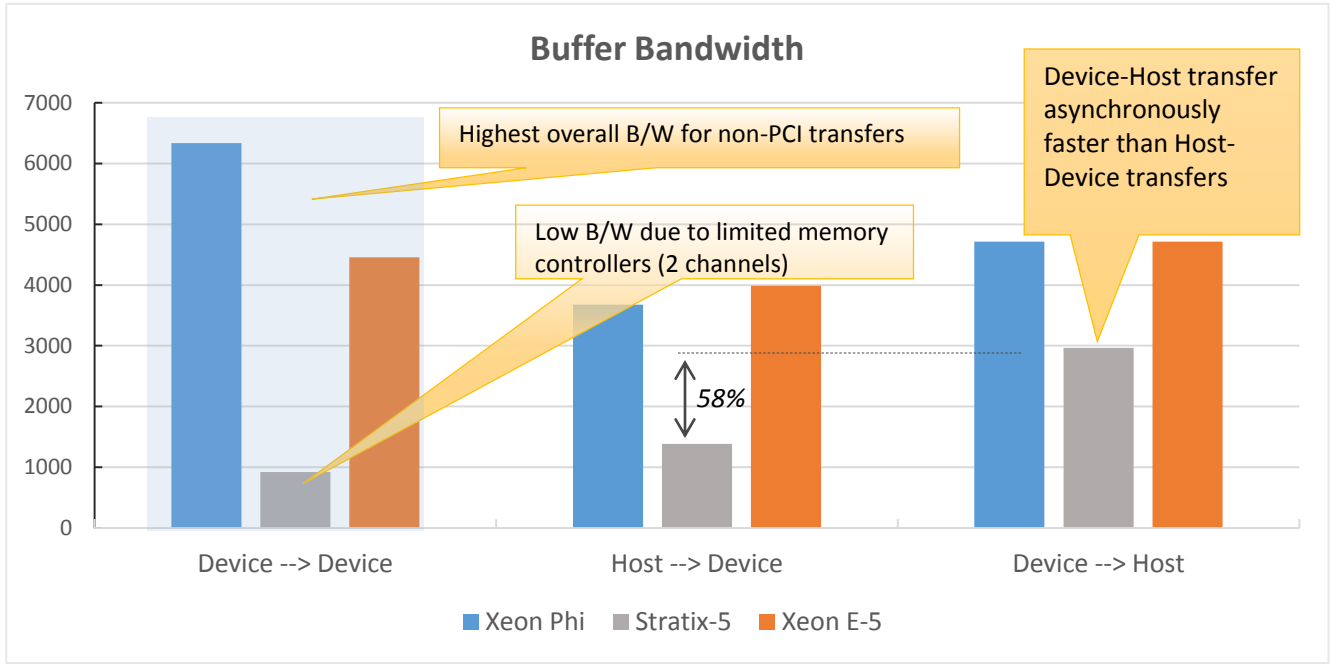
OpenCL Memory Model

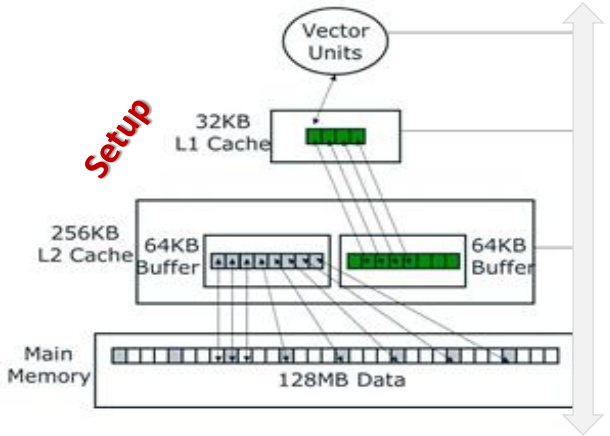


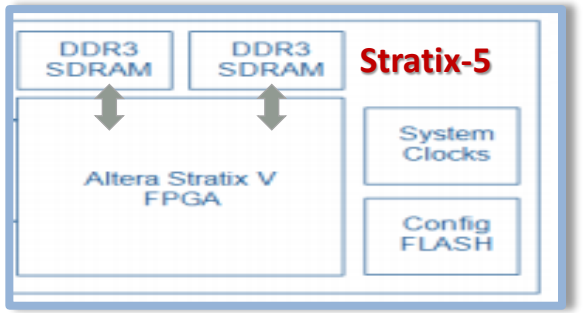


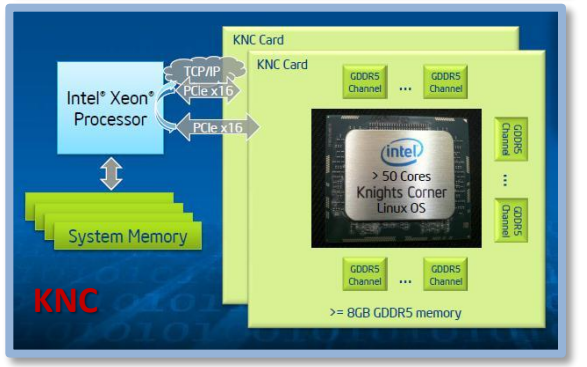
HOST

Global Memory Bandwidth









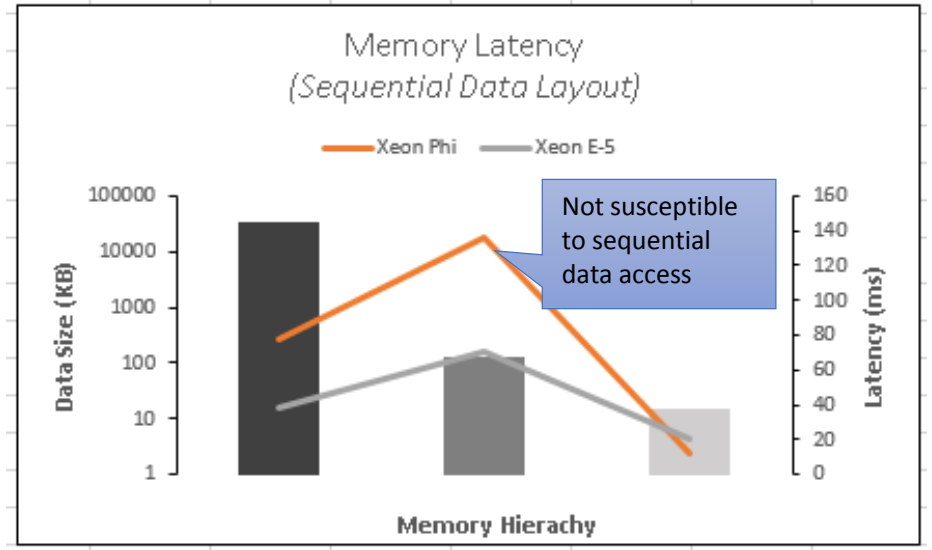
	Device --> Device	Host --> Device	Device --> Host
Xeon Phi	6336.63	3678.16	4712.81
Stratix-5	922.46	1382.29	2965.71
Xeon E-5	4456.82	3990.02	4712.81

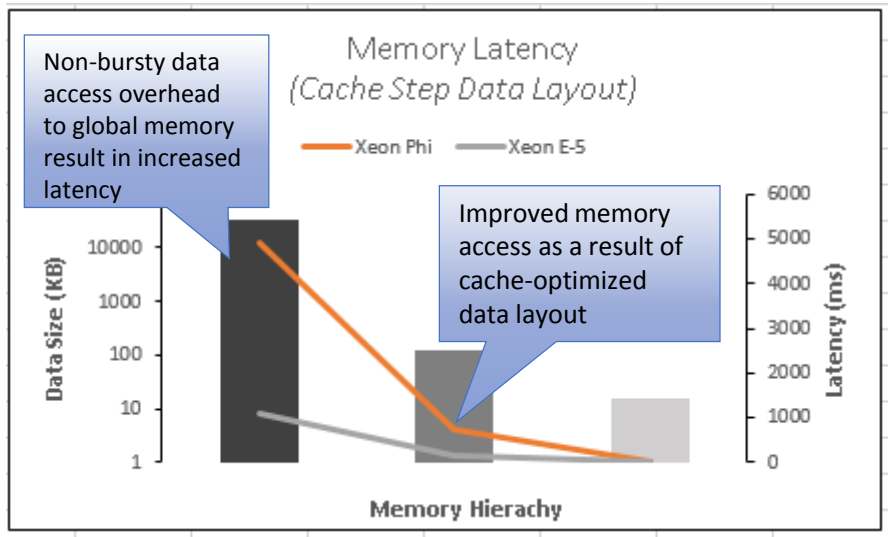
Achieved global memory bandwidth in MB/s

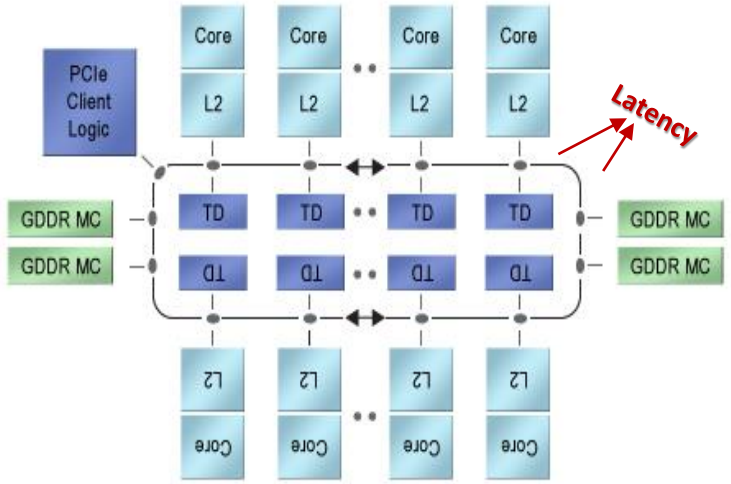
Memory Latency

Memory Layer	Data size
Global	32 MB
Constant	128 KB
Local	16KB/ 32 KB

Benchmark parameters



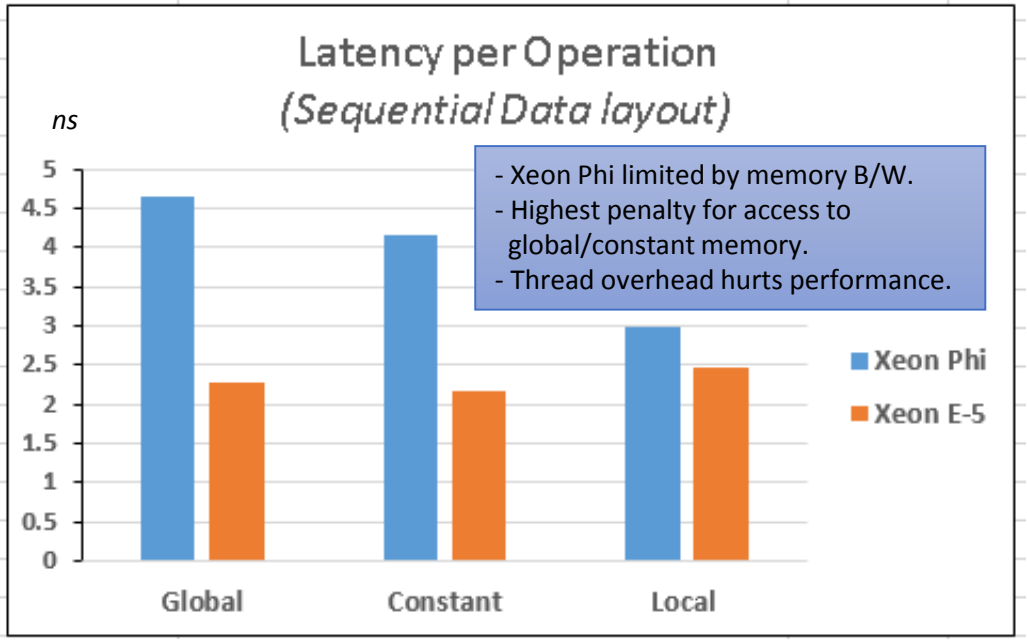


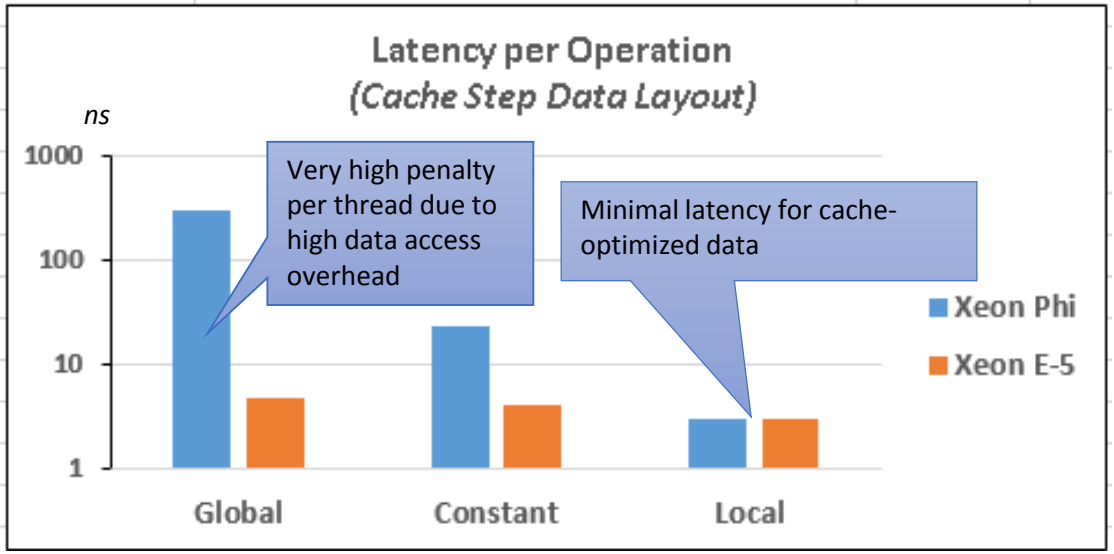


Legend: **Global** **Constant** **Local**

Certain kinds of OpenCL benchmarking not feasible on FPGA. Xeon E-5 used as baseline.

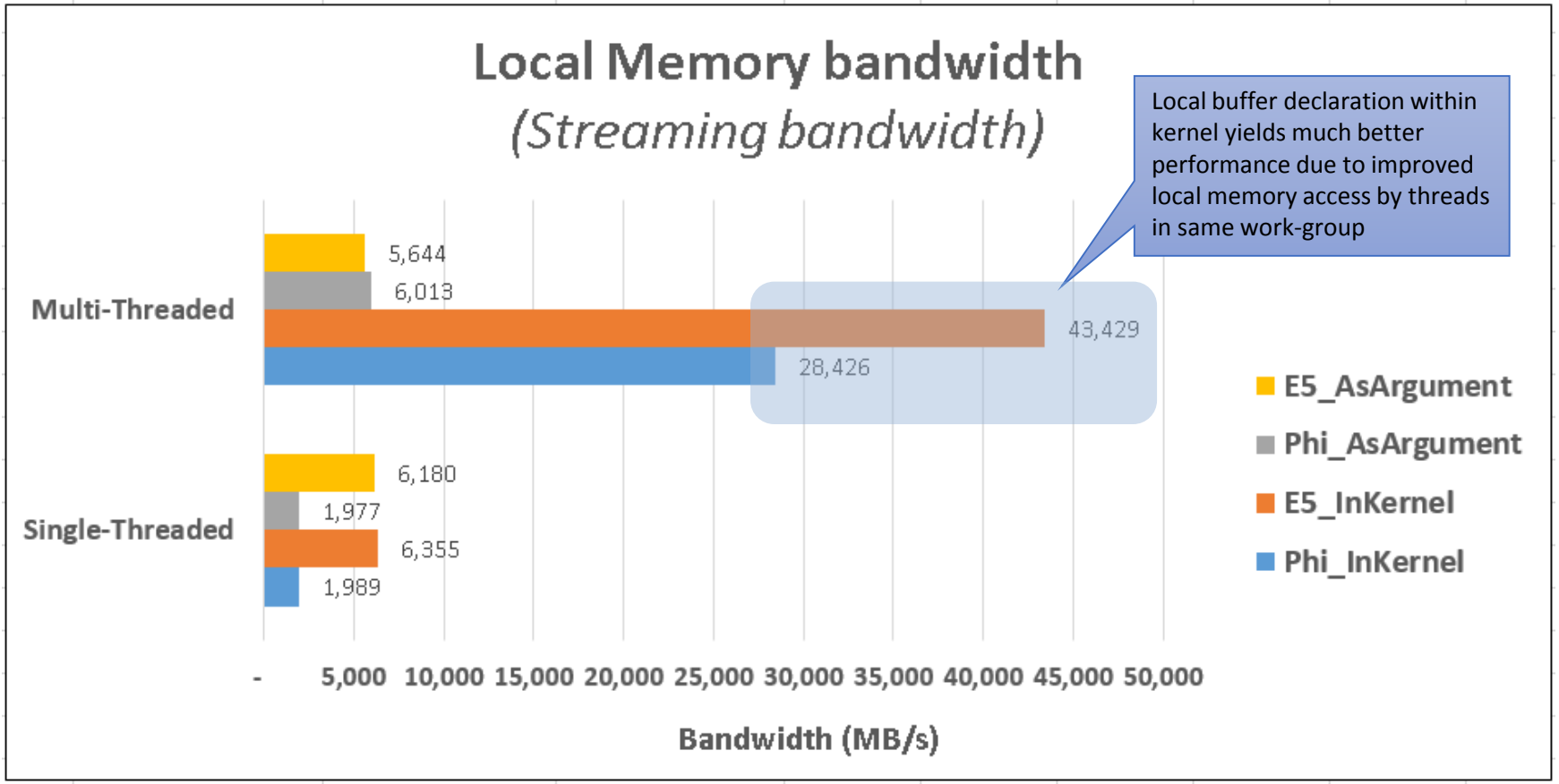
Memory Latency per Operation





Certain kinds of OpenCL benchmarking not feasible on FPGA. Xeon E-5 used as baseline.

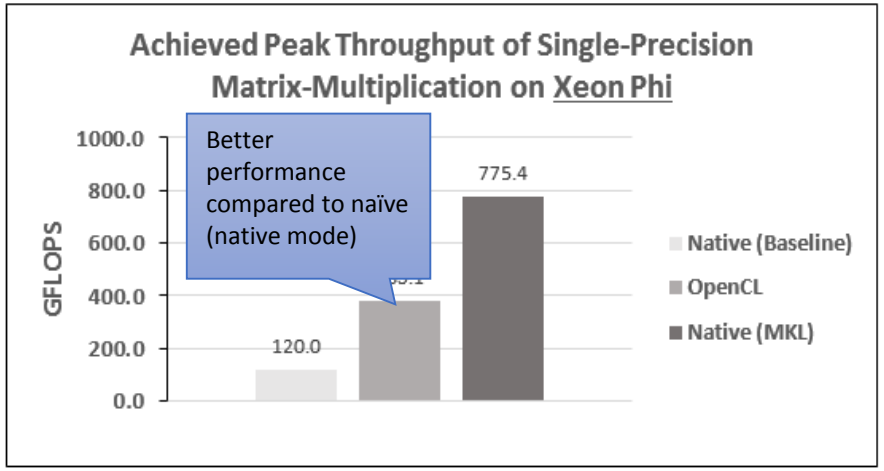
Local Memory Bandwidth

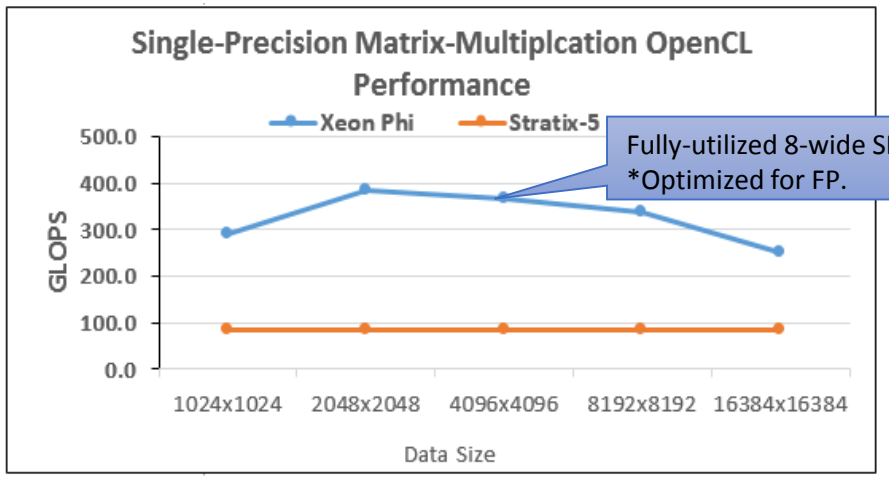


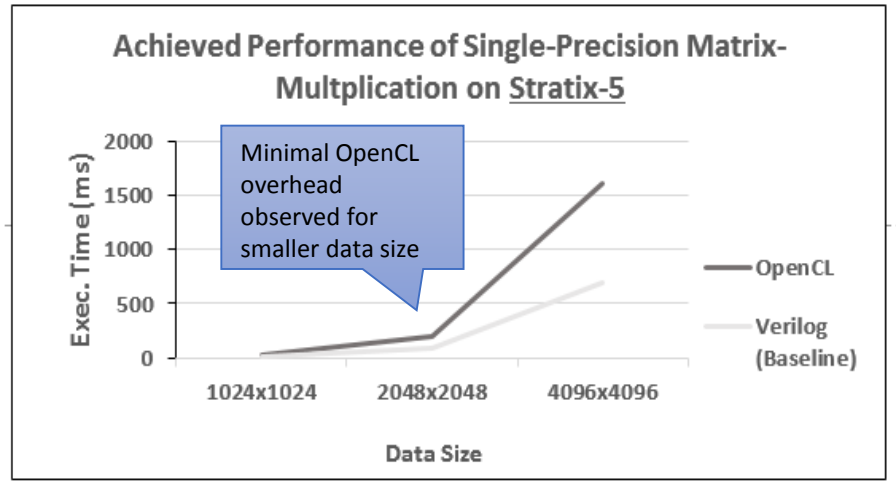
Total local memory bandwidth of device is obtained by the multiplying streaming bandwidth of compute unit by number of compute units on device.

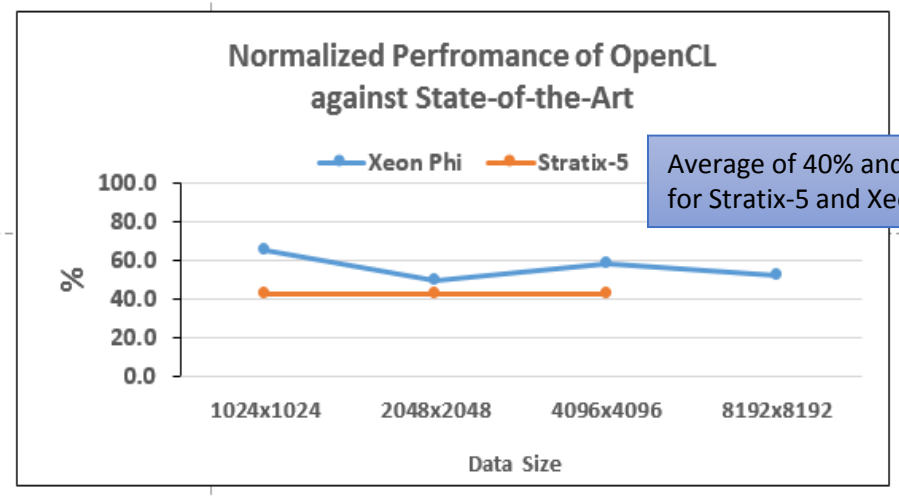
AsArgument: host buffer passed to kernel as argument
InKernel: buffer declared within kernel

Matrix-Multiplication







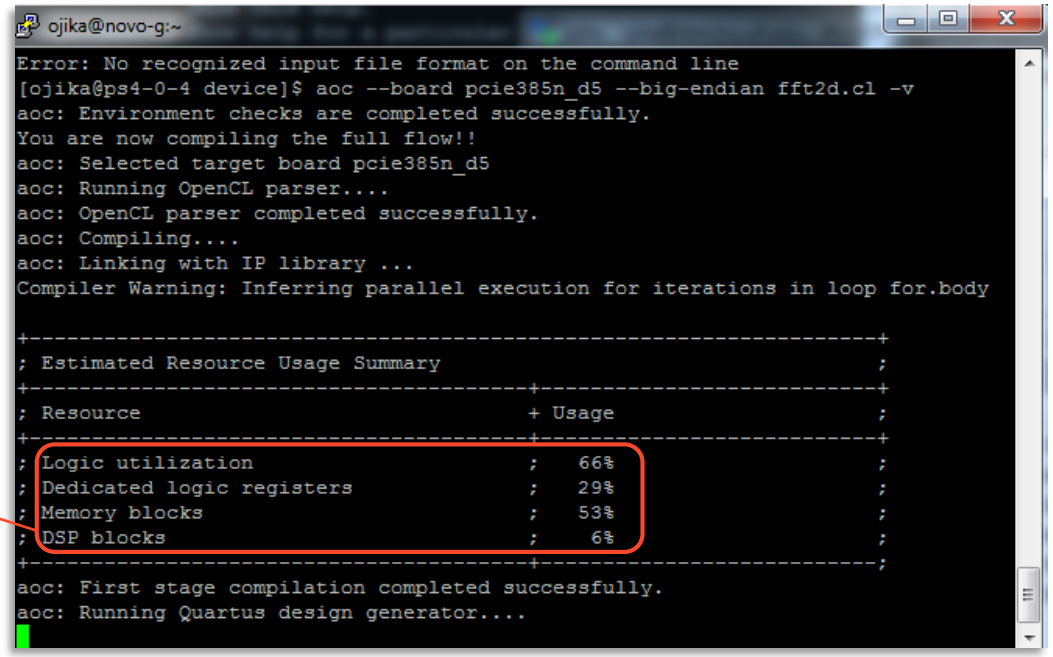


***Naïve**: OpenMP with no compiler-enabled vectorization efforts*

2D FFT

	Stratix-5 (OpenCL)	
Matrix Size	Exec. Time (ms)	Single-Precision Performance (GFLOPS)
1024x1024	2.44	42.9749

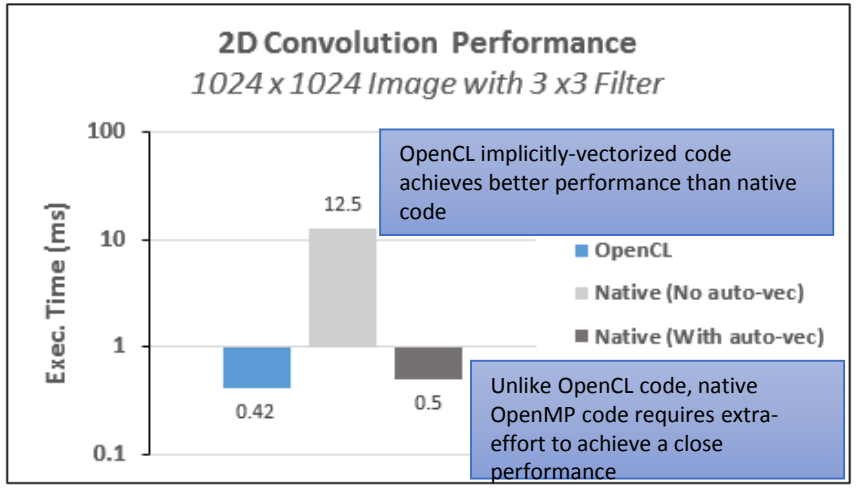
Resource Utilization

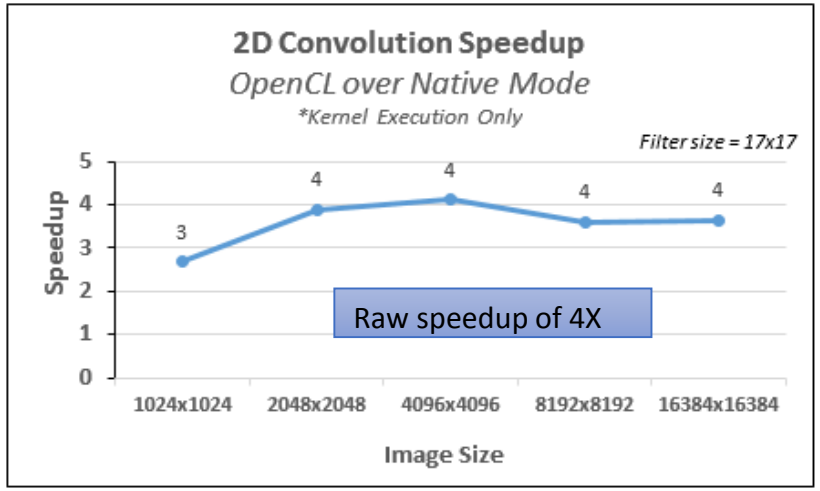


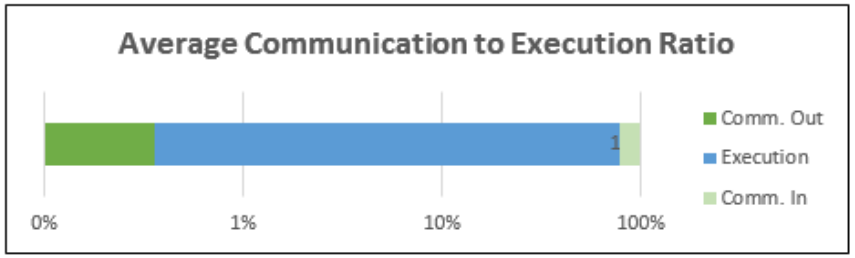
Offline compilation: ~ 2 hours

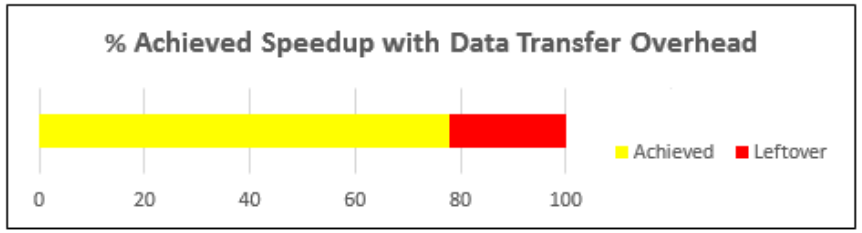
2D FFT = 2 x FFT, Dot-Product Matrix-Multiply, Inverse FFT

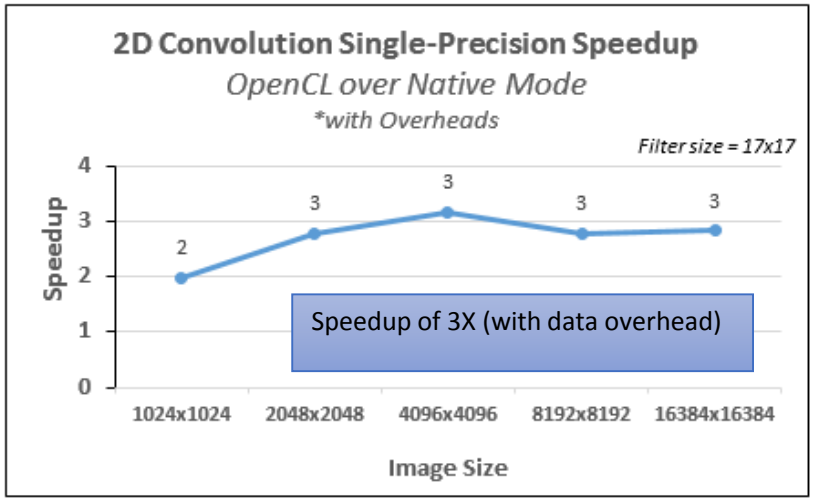
2D Convolution









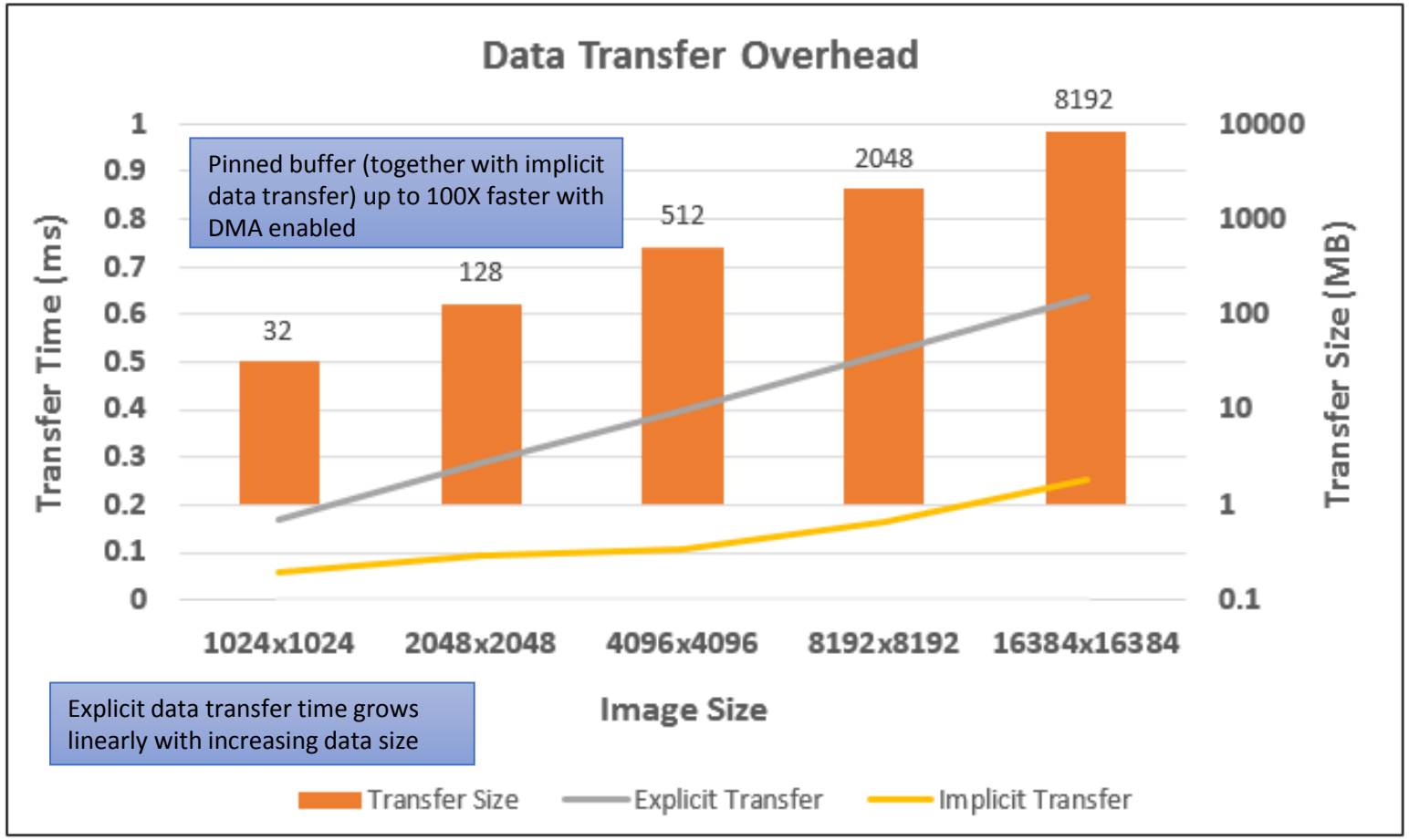


Comm. Out (data out) *and* ***Comm. In (data in)*** *are asynchronous in data transfer speed. Latter is faster.*

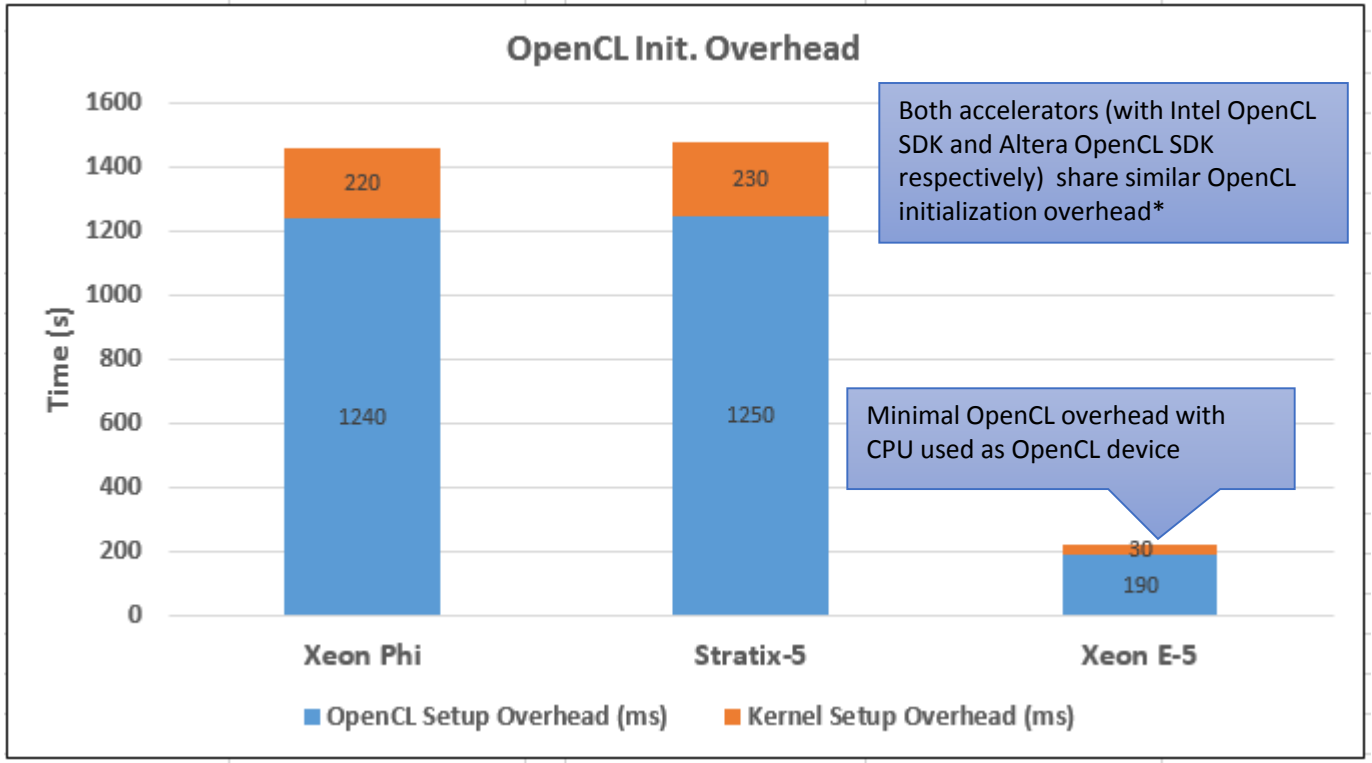
Leftover*: Useable execution period – used for data transfer*

Data Transfer (with 2-D Convolution)

Transfer size = Image Buffer + Filter Buffer	
Image Buffer:	1024 x 1024 x 32 bits
Filter Buffer:	17 x 17 x 32 bits
**32-bit integer or float*	



OpenCL Initialization Overhead



	OpenCL Setup Overhead (ms)	Kernel Setup Overhead (ms)	Total Overhead (ms)
Xeon Phi	1240	220	1460
Stratix-5	1250	230	1480
Xeon E-5	190	30	220

***SInitialization overhead** = OpenCL Setup Time + Kernel Setup Time*

Summary (DL Layers)

1. From software

DAAL : Improve / replicate accuracy of state of the art

(MKL, transparently)

KNL : Benchmark KNL against serial, existing; OpenCL benchmarks

---

2. To hardware:

Simplified-Framework

MKL-DNN / OpenCL

Xeon+FPGA, SmartMemory : OpenCL BSP; OpenCL benchmarks

---

3. User App: Supplemental / Demo with image classification on DL-SDK

DL-SDK, ROOT

Caffe

MKL

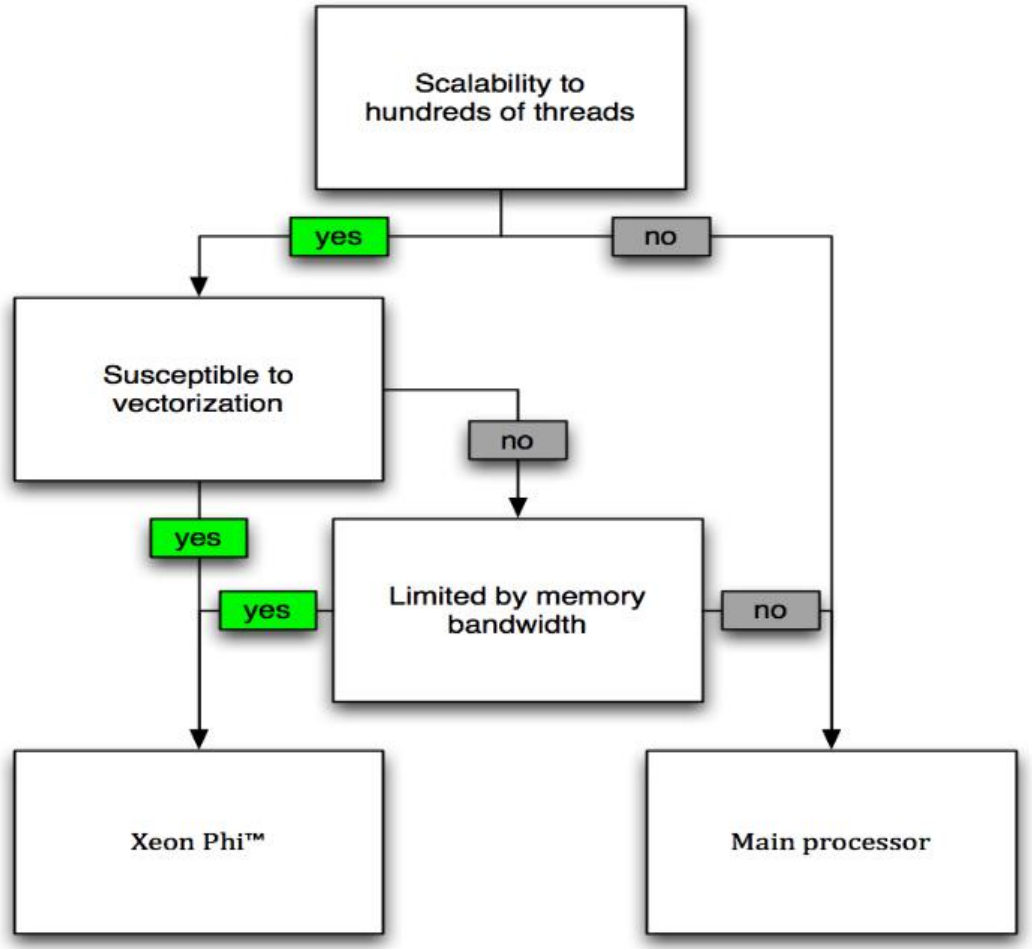
CPU

ROOT integration:
:https://indico.cern.ch/event/505613/contributions/2228344/attachments/1347106/2041567/oral-CHEP16-SergeiVGleyzer.pdf

Next Quarter and Meeting…

- KNL results
- R&D with SmartMemory (FPGA-based accelerator)

BACKUP


Scalability to hundreds of threads
yes
no
Susceptible to vectorization
no
yes
yes
Limited by memory bandwidth
no
Xeon Phi™
Main processor