Acceleration of Scientific Deep Learning Models
On Heterogeneous Computing Platform
With Intel FPGAs

Herman Lam, *Site Director*

* **SHREC:** NSF Center for Space, High-Performance, and Resilient Computing
University of Florida

Mission-Critical Computing
NSF CENTER FOR SPACE, HIGH-PERFORMANCE, AND RESILIENT COMPUTING (SHREC)

University of Pittsburgh
BYU Brigham Young University
Virginia Tech
UF University of Florida

C. Jiang, D. Ojika: *SHREC @ UF**
T. Kurth, Prabhat: *NERSC Berkeley Lab***
S. Vallecorsa: *CERN openlab*
B. Patel: *Dell EMC*
H. Lam: *SHREC @ UF**

** NERSC: National Energy Research Scientific Computing Center, Lawrence Berkeley National Lab

Outline

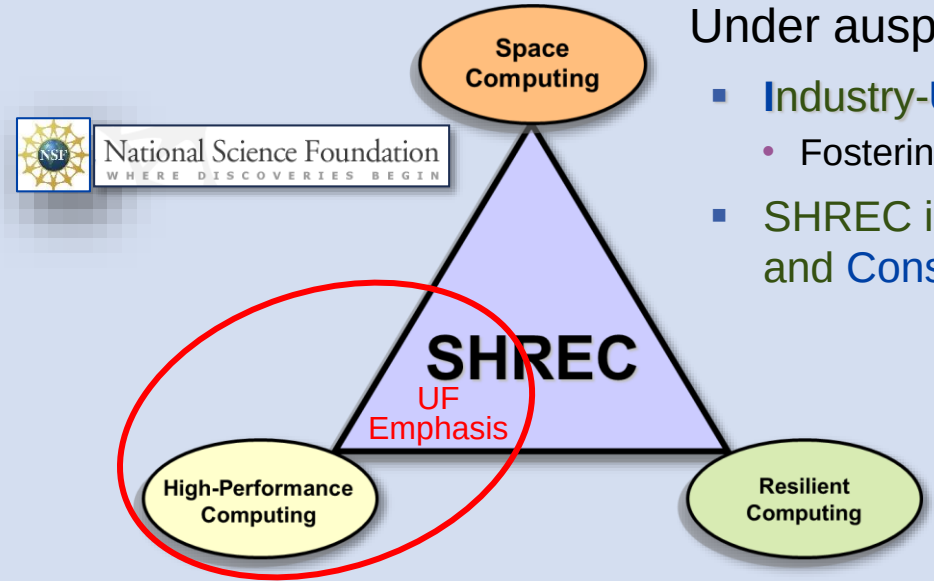
- Overview of SHREC
- Hetergeneous computing for deep learning
  - Data pre-processing; model training; model inference
  - Focus on FPGA-acceleration of inference stage
- Experimental platforms & tools
  - Intel PAC10 card; OpenVINO; DLA* design suite
- Case studies
  - HepCNN: initial & optimized results
  - CosmoGAN: initial & optimized results
- Conclusions & going forward

* DLA: Deep Learning Accelerator

Introduction to SHREC@UF*

Fermilab · Ball Aerospace · DELL · Sensors, Space Vehicles · NASA ARC, GSFC, IV&V, JSC, LaRC · National Security Agency · Lockheed Martin · LPS The Laboratory for Physical Sciences · ONR · Harris · Los Alamos National Laboratory · Honeywell · MIT Lincoln Laboratory · Boeing · Walt Disney Animation Studios · Raytheon · U.S. Naval Research Laboratory · Satlantis · BAE Systems Inspired Work · Sandia National Laboratories · Space Micro · Draper Laboratory · intel · ARL · Collins Aerospace · L3 Space & Sensors · Emergent Space Technologies

***** NSF Center for Space, High-Performance, & Resilient Computing



Under auspices of **IUCRC** Program at ***NSF***

- **I**ndustry-**U**niversity **C**ooperative **R**esearch **C**enters
  - Fostering university, agency, & industry R&D collaborations
- SHREC is both National Research Center (universities) and Consortium (member organizations)

- University of Pittsburgh (lead site)
- Brigham Young University
- ***** University of Florida (UF)
- Virginia Tech

Heterogeneous Computing[1] for Deep Learning

Motivation

- Deep learning becoming *pervasive* for mission-critical computing
  - *Heterogeneous computing** offers unique capabilities to accelerate DNNs[2]

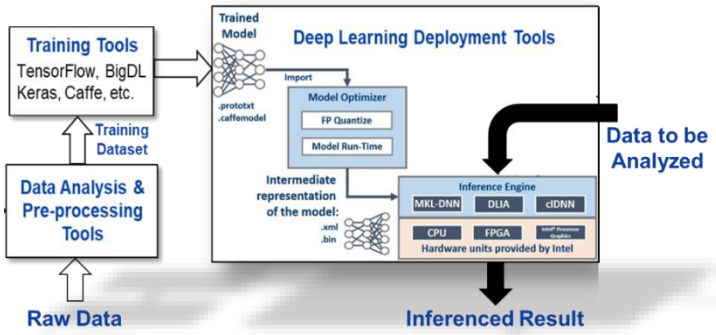
Goal

Perform design-space exploration:

- Of *emerging HGC[1] archs/tools* and *DNN models*
- For *acceleration* of selected mission-critical apps

Approach

Focus on use of *FPGAs* to *accelerate inference stage* of the HGC workflow



Stages of HGC workflow

- Data analysis & pre-processing
- Model training
- DNN inference

Collaborating partners

- ***NERSC**:*** HepCNN, CosmoGAN model support
- ***CERN openlab:*** 3D GAN model support
- ***Dell:*** SHREC membership support, equipment
- ***Intel:*** Deep-learning tools; engineering support

DNN Models *from NERSC & CERN Openlab*

- HEP-CNN
- CosmoGAN
- 3D GAN

*HGC: Heterogeneous Computing (CPU+GPU+FPGA)

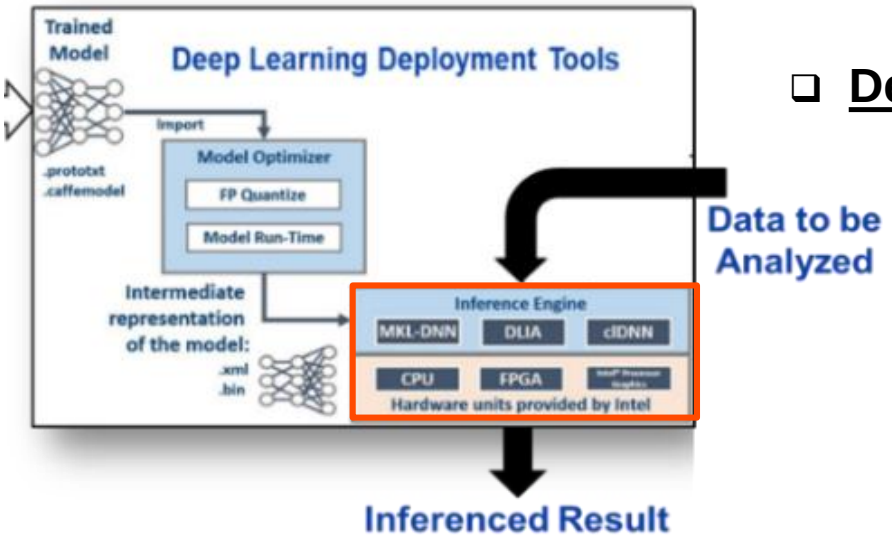
** NERSC: National Energy Research Scientific Computing Center, Lawrence Berkeley National Lab

FPGA Acceleration for DNN Inference

Experimental Setup & Tools

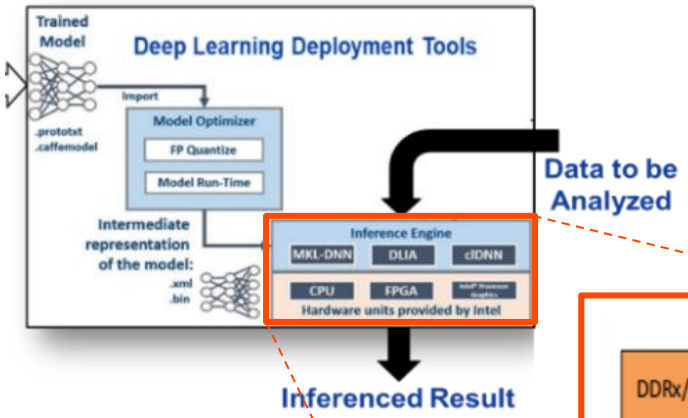
- **Experimental platforms**
  - **Dell EMC server:** 2x Intel Xeon Gold 6130 CPU (up to 32 cores, 64 threads)
  - **Intel PAC:** Arria 10 GX FPGA

- **Intel OpenVINO Toolkit**
  - Model Optimizer
    - Convert mainstream deep learning framework model *(TensorFlow, Caffe, etc.)* into unified *intermediate representations (IR)*
  - Inference Engine
    - API library for mapping IR onto *Intel hardware platforms (CPU, GPU, FPGA, etc.)*
    - Integrated with *Deep Learning Accelerator suite* for *FPGA acceleration*



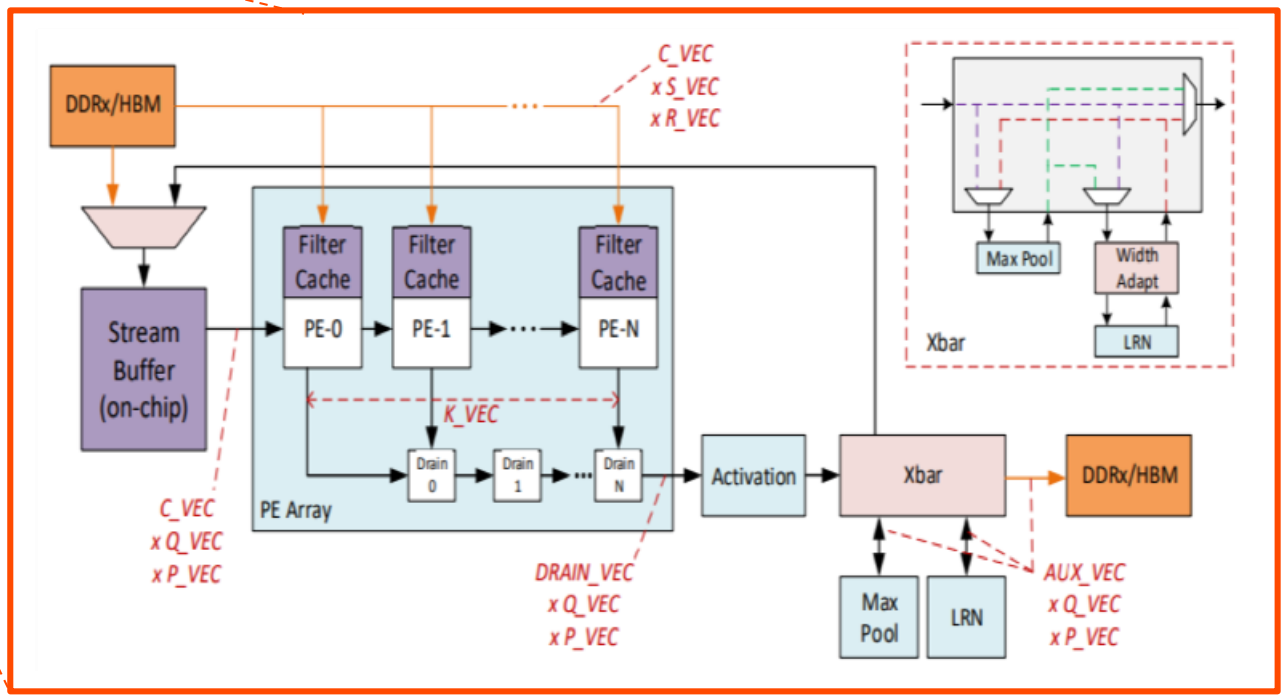
- **Deep Learning Accelerator suite (DLA)**
  - *OpenCL-based* implementation of DNN inferencing hardware architecture
  - Source code acquired through NDA with Intel to be *optimized for various applications*

Deep Learning Accelerator suite (DLA [1])



- *OpenCL-based* implementation of DNN inferencing hardware architecture
- Source code acquired through NDA with Intel to be *optimized for various applications*

- DDR/HBM
- Stream Buffer
- PEs: processing elements
- Activation module
- Xbar
- Max Pool module
- LRN: Normalization



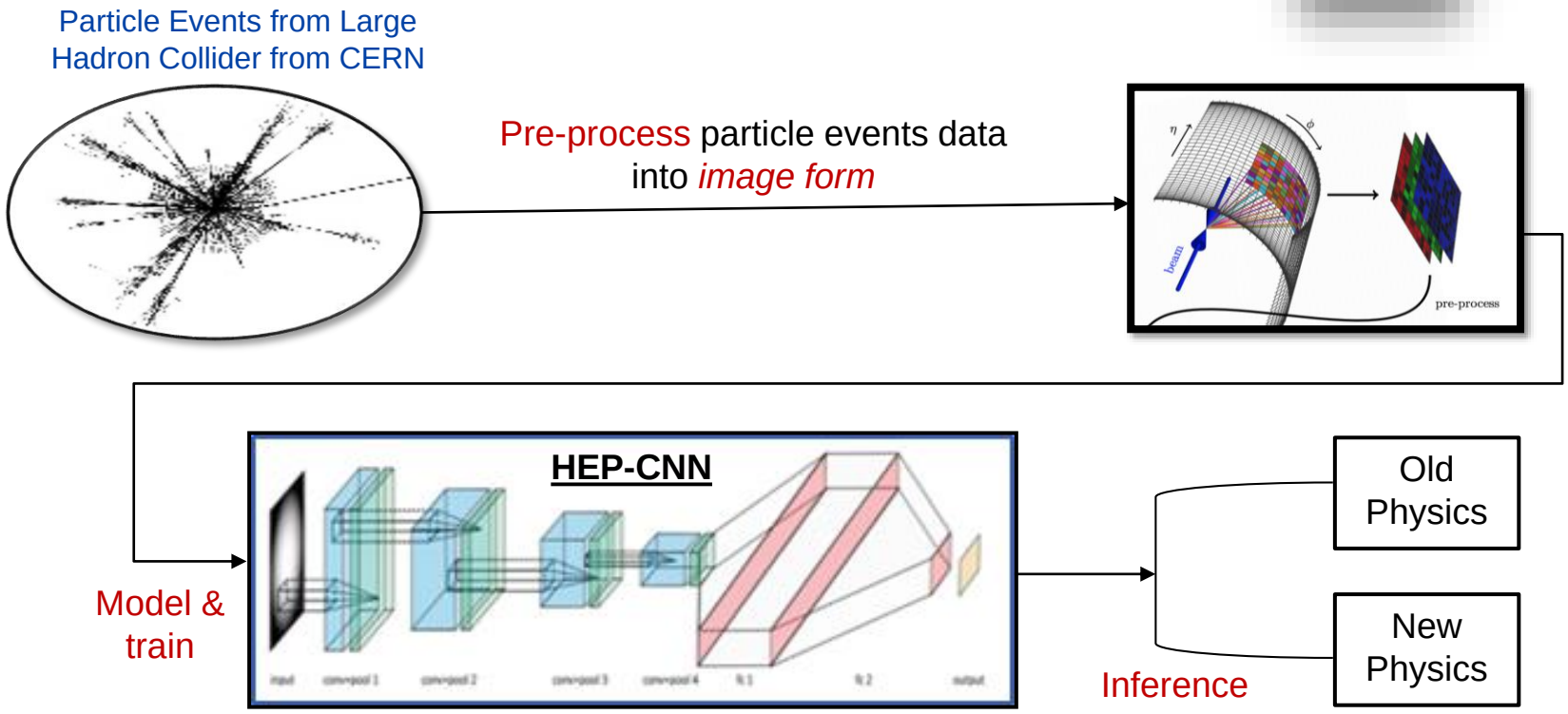
[1] Abdelfattah, M.S. et al., DLA: Compiler and FPGA Overlay for Neural Network Inference Acceleration. arXiv e-prints arXiv:1807.06434 (Jul 2018)

Outline

- Overview of SHREC
- Hetergeneous computing for deep learning
  - Data pre-processing; model training; model inference
  - Focus on FPGA-acceleration of inference stage
- Experimental platforms & tools
  - Intel PAC10 card; OpenVINO; DLA* design suite
- Case studies
  - HepCNN: initial & optimized results
  - CosmoGAN: initial & optimized results
- Conclusions & going forward

* DLA: Deep Learning Accelerator

Case Study: HEP-CNN Model [2] from



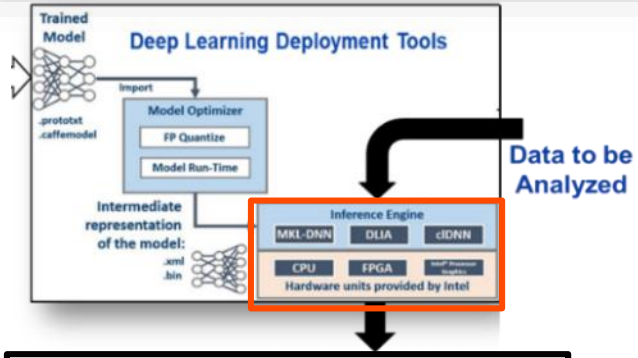
- Classifies *particle events* between
  - "ones which *can be fully described by standard model physics*"
  - "ones which *contain new physics*".
- Developed and trained by *NERSC at Lawrence Berkeley Lab* using CNN (Convolutional Neural Network) topology

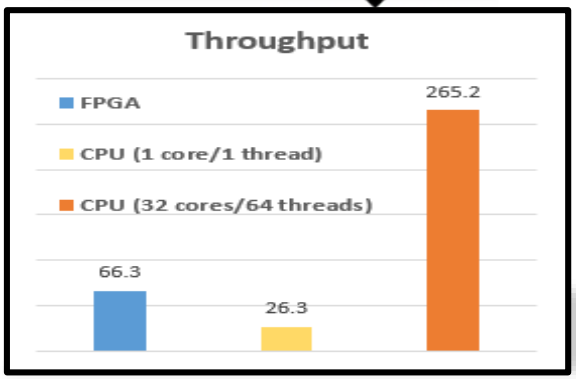
[2] T. Kurth et al. 2017. Deep Learning at 15PF: Supervised and Semi-supervised Classification for Scientific Data. High Performance Computing, Networking, Storage and Analysis (SC '17). 7:1–7:11.

HEP-CNN: Initial Experimental Results

FPGA vs. CPU performance *w/ Native DLA* Implementation

Bit precision	FPGA* Throughput	CPU (1 core/1 thread)** Throughput	CPU (32 core/64 thread)** Throughput	FPGA Speedup vs. 1 core CPU
FP16	66.3 images/sec	26.3 images/sec	265.2 images/sec	2.52





* Arria 10 at 20 nm process
** Intel Xeon Gold 6130 CPU at 14 nm process

Observations:

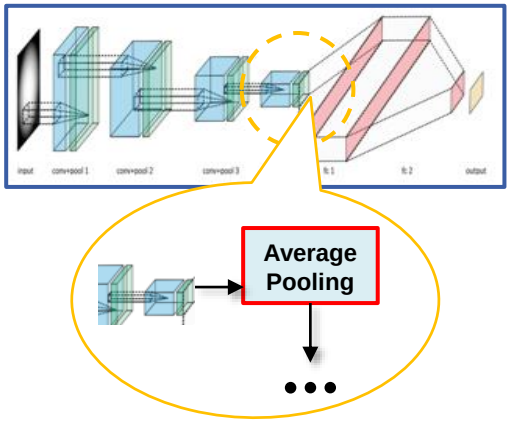
FPGA performance is *unoptimized* due to unsupported average pooling layers by native DLA

- Although FPGA mode, OpenVINO executes unsupported layer on CPU as a fallback device
- Thus introduces communication latencies during data transferring process
- Still *2.52x* speedup vs. 1-core/1-thread Xeon Gold CPU

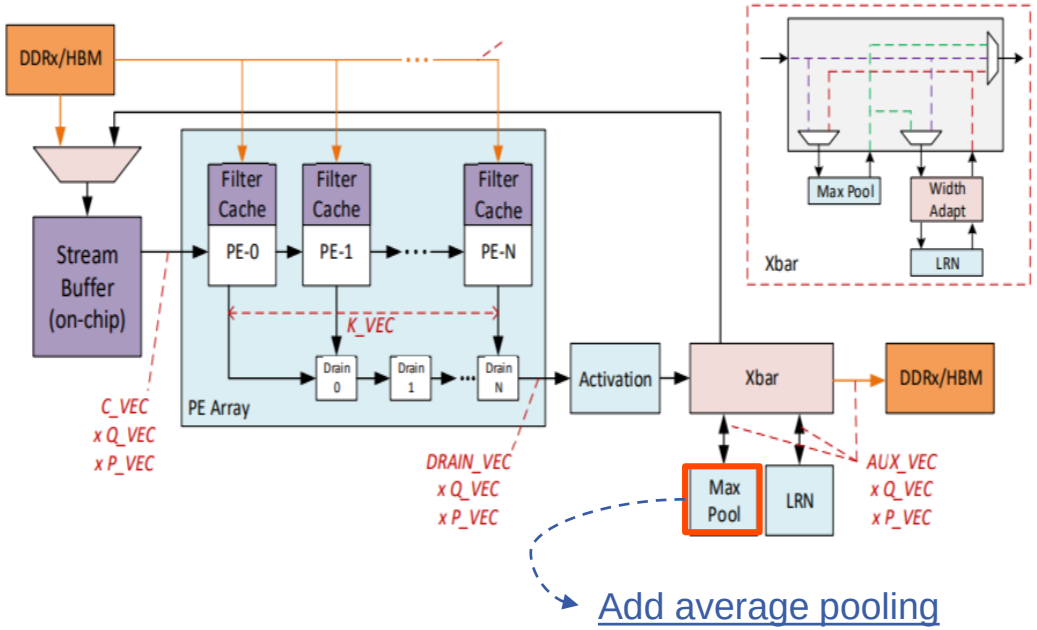
Performance Optimization through DLA Customization

Problem:

- Native DLA architecture does not have FPGA primitive to process *Average Pooling* layer



HEP-CNN model structure



System-level architecture of DLA [1]

Solution:

- Customize DLA source code to add Average Pooling FPGA primitive in Max Pool module

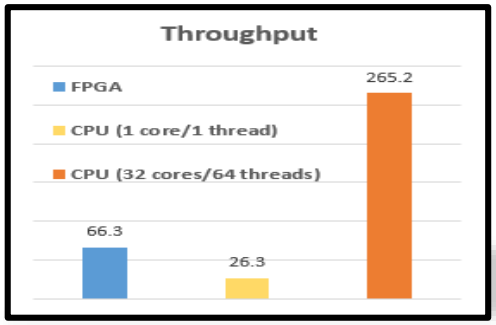
[1] Abdelfattah, M.S. et al., DLA: Compiler and FPGA Overlay for Neural Network Inference Acceleration arXiv e-prints arXiv:1807.06434 (Jul 2018)

HEP-CNN: Improved Results

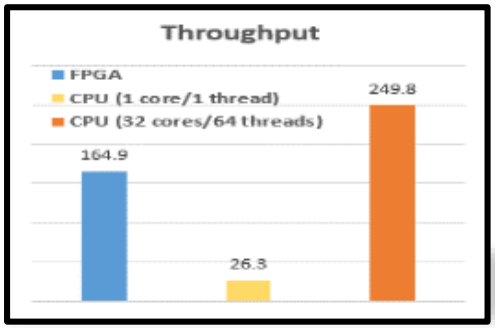
FPGA vs. CPU performance *w/ Custom DLA* Implementation

Bit precision	FPGA * Throughput	CPU (1 core/1 thread) ** Throughput	CPU (32 core/64 thread) ** Throughput	FPGA Speedup vs. 1 core CPU
FP16	**164.9 images/sec**	**26.3 images/sec**	**265.2 images/sec**	**6.27**

* Arria 10 at 20 nm process
** Intel Xeon Gold 6130 CPU at 14 nm process



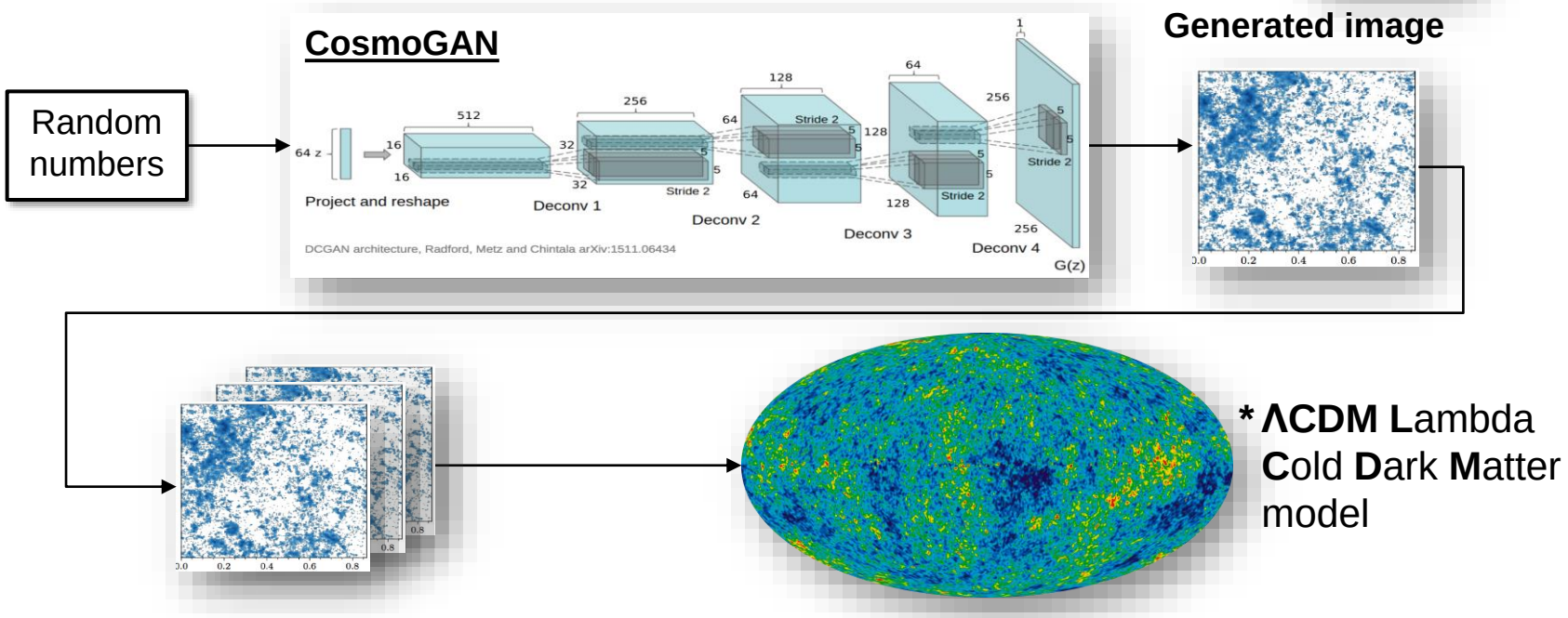
Initial results (2.52X)



Improved results (6.27X)

- All processing performed in FPGA
- Improved from *2.52x to 6.27x* vs 1-core/1-thread Intel Xeon CPU

Case Study: CosmoGAN[3] Model from



- Generates ΛCDM* weak lensing convergence maps for cosmological studies
  - Each output image is a *measure of the density of the universe* observed from a particular direction
- Developed and trained by NERSC at Lawrence Berkeley Lab using *DCGAN (Deep Convolutional Generative Adversarial Network)* topology
- **Objective:** study how to improve FPGA acceleration for complex scientific DNNs

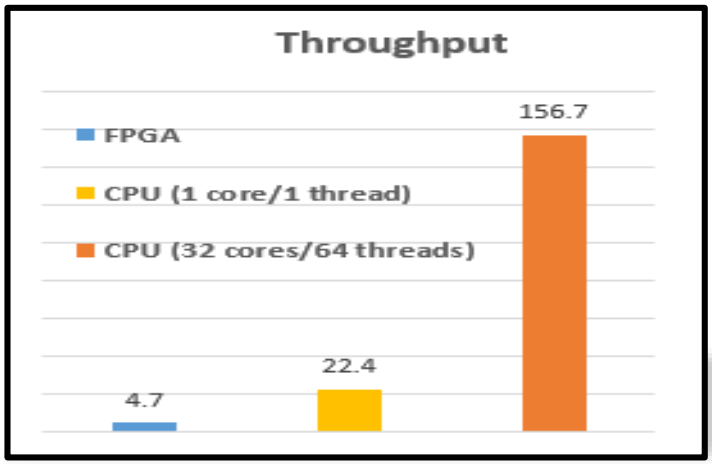
[3] M. Mustafa, D. Bard, W. Bhimji, R. Al-Rfou, and Z. Lukić,.Creating Virtual Universes Using Generative Adversarial Networks. 2017

CosmoGAN: Initial Experimental Results

FPGA vs. CPU performance *w/ Native DLA* Implementation

Bit precisionz	FPGA* Throughput	CPU (1 core/1 thread)** Throughput	CPU (32 core/64 thread)** Throughput	FPGA Speedup vs. 1 core CPU
FP16	**4.7 images/sec**	**22.4 images/sec**	**156 images/sec**	**0.21**

* Arria 10 at 20 nm process
** Intel Xeon Gold 6130 CPU at 14 nm process



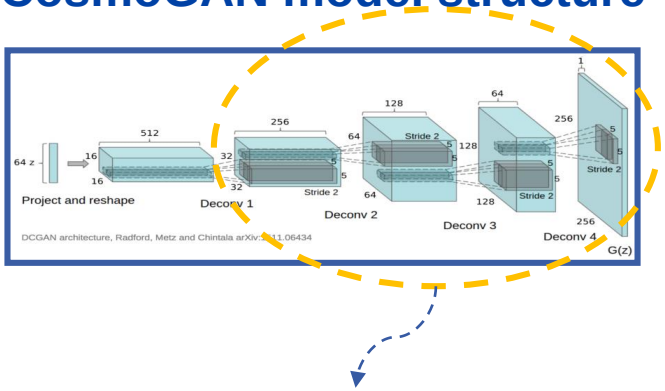
Observations:

Identified two problems which caused *FPGA* performance to be extremely *poor*:

- **P1:** Deconvolutional layers not support in FPGA mode
- **P2:** Inefficiency of DLA architecture to process the Normalization and Activation layers

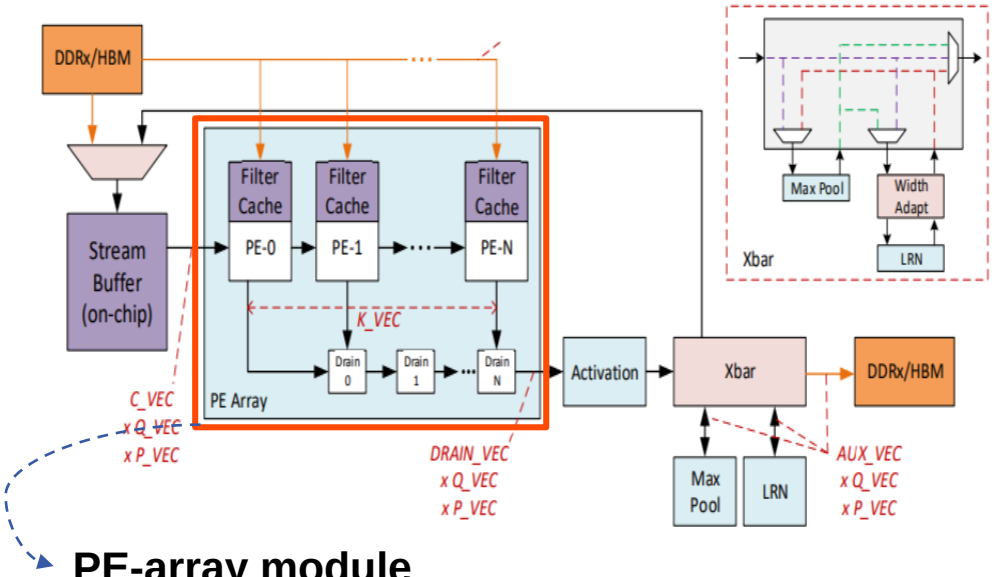
P1: FPGA Primitive to Support Deconvolutional Layer

CosmoGAN model structure



Deconvolutional layers

System-level architecture of DLA



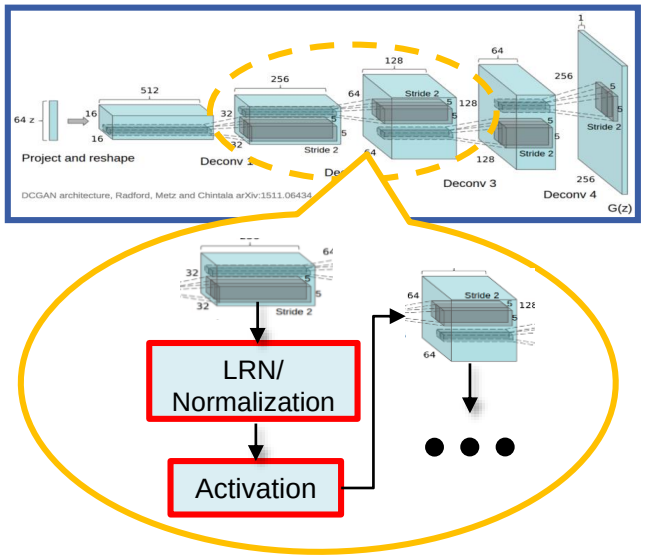
PE-array module

Solution:

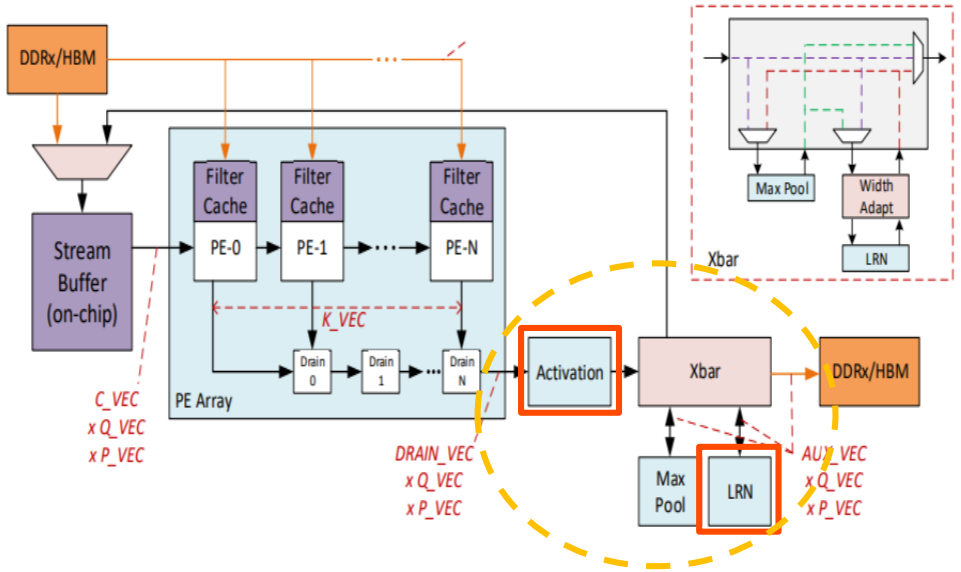
- Customize DLA source code to enable deconvolutional operation in PE-array module

P2: Inefficient Processing of LRN* and Activation Layers

CosmoGAN model structure



System-level architecture of DLA



- CosmoGAN requires Normalization (LRN) before Activation

- *Activation* layer is hardwired in DLA to be processed *before* *LRN* layer

  Requiring *extra iteration* for each deconvolutional + LRN + Activation subgraph

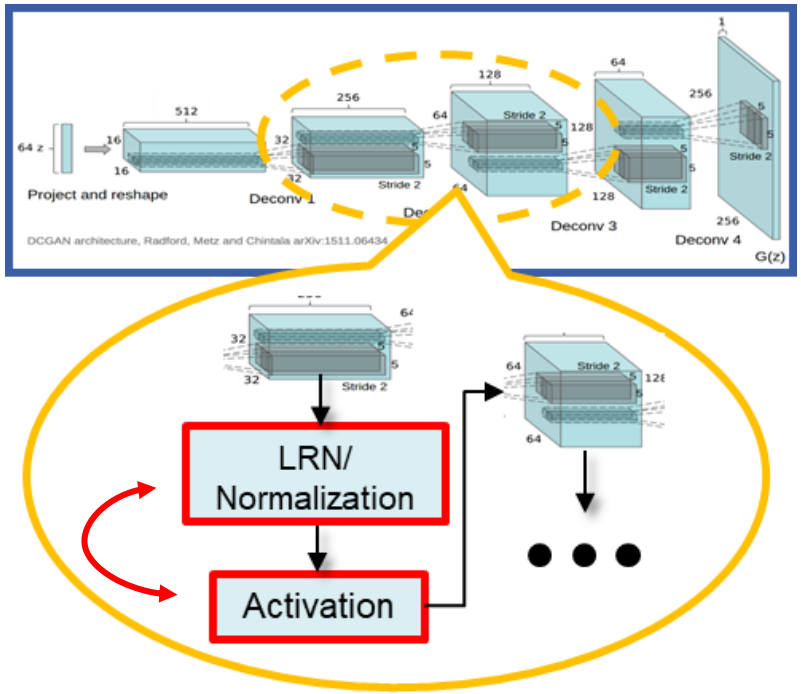
Solution:

- Currently working with Intel engineering to make DLA architecture more flexible

* LRN: Local Response Normalization layer

P2: Experiment to Predict CosmoGAN Performance

CosmoGAN model structure



- CosmoGAN requires Normalization (LRN) before Activation
- *Activation* layer is hardwired in DLA to be processed *before LRN* layer

  Requiring *extra iteration* for each deconvolutional + LRN + Activation subgraph

Current situation:

- Cannot easily modify DLA architecture to make it more flexible
- Can easily change the order of processing (Activation before LRN)

Hypothesis: By switching the processing order of LRN and Activation in CosmoGAN:

- Inference results will not be correct
- But the computational complexity is equivalent to original model

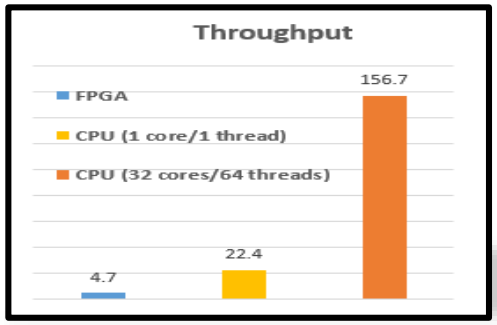
* LRN: Local Response Normalization layer

Prediction of Optimized CosmoGAN Results

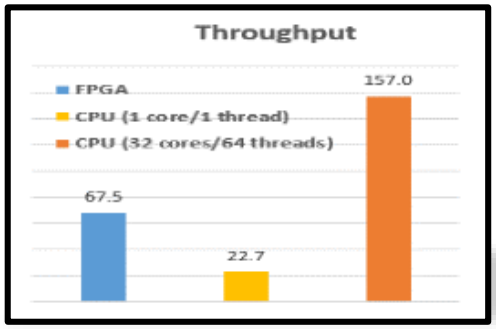
FPGA vs. CPU performance *w/ re-ordered* CosmoGAN model

Bit precision	FPGA* Throughput	CPU (1 core/1 thread)** Throughput	CPU (32 core/64 thread)** Throughput	FPGA Speedup vs. 1 core CPU
FP16	67.5 images/sec	22.4 images/sec	156 images/sec	**3.01**

* Arria 10 at 20 nm process
** Intel Xeon Gold 6130 CPU at 14 nm process



Initial results (0.21x)

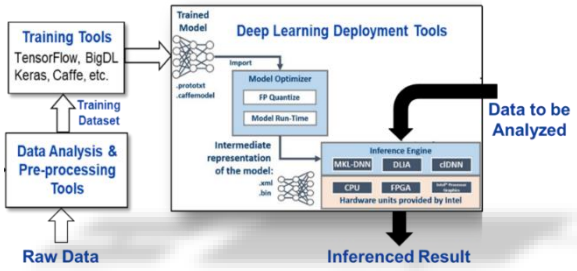


Expected results (3.01x)

- By processing LRN before Activation, requires only single *iteration* for each deconvolutional + LRN + Activation subgraph
- Improved from *0.21 x to 3.01x* vs 1-core/1-thread Intel Xeon CPU

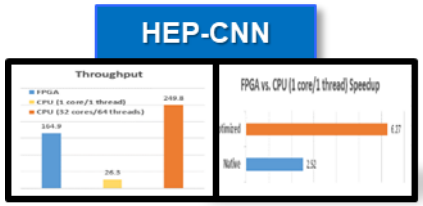
Summary &Conclusions

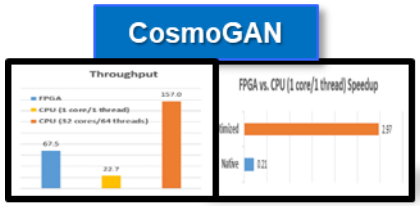
- Heterogeneous computing for deep learning

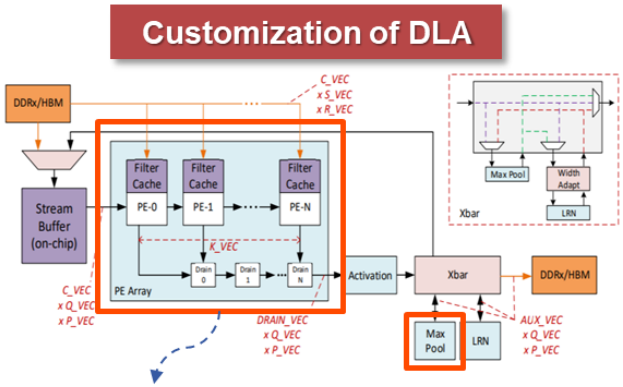


- Collaboration with NERSC & CERN openlab on scientifically relevant DNNs
- SHREC: Focused on FPGA-acceleration of inference stage

- Exploration of FPGA-based platforms & tools
  - Intel PAC10 card; OpenVINO; DLA design suite
  - Explore the use and improvement of state-of-art tools







Going Forward

- Continue to explore & improve FPGA-based DNN platforms & tools
  - E.g., Extend DLA's function to support "3D convolutional layer" for 3D GAN (CERN openlab)
  - Scale up to multiple FPGAs for faster inference
  - Explore FPGA (+ emerging technologies) for efficient DNN model training
- Appropriate use of FPGA-based DNN platforms
  - Compare FPGA-based platform vs. CPU, GPU, & other emerging devices (energy, size, weight, cost, etc.)
  - Determine appropriate missions for FPGA-based systems

QUESTIONS