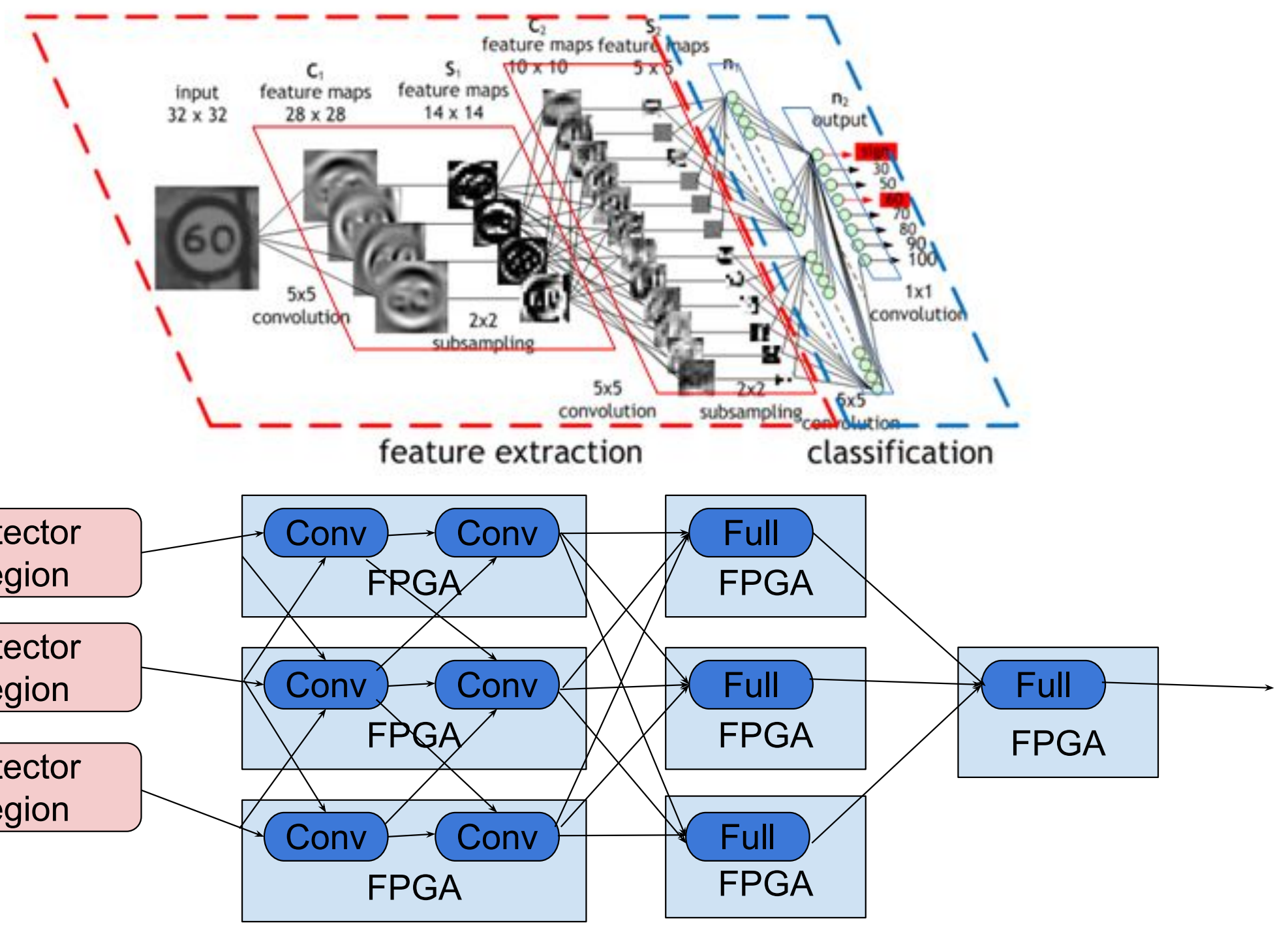
Deploying ML In Hardware
FPGAs & ASICs

SLAC NATIONAL ACCELERATOR LABORATORY

SLAC TID-AIR
Technology Innovation Directorate
Advanced Instrumentation for Research Division

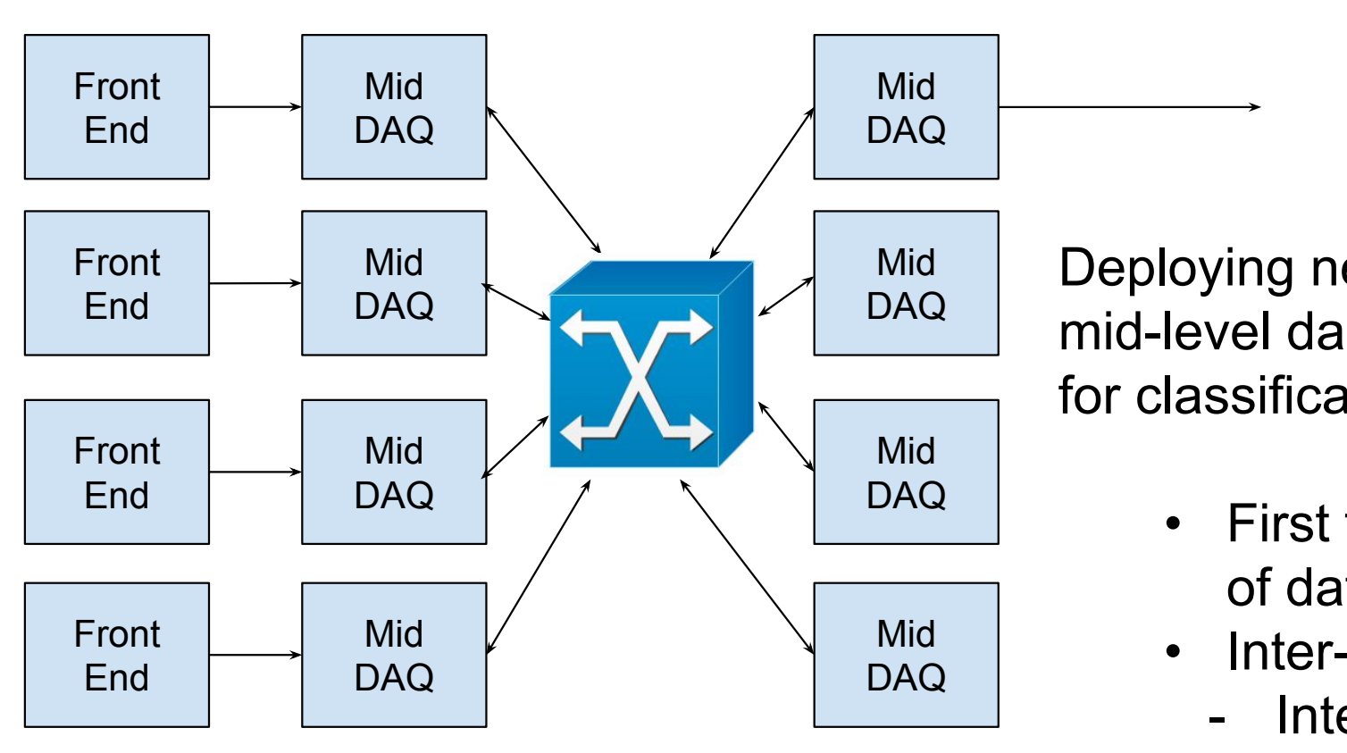
R. Herbst

Matching To Common DAQ Structures



Machine Learning In Hardware

- Simple to deploy FPGA firmware for classification
  - Leave out back propagation
  - Each layer is pipelined, allowing higher frame rate
  - Layers are flexible, can exist in different FPGAs
    - Take advantage of 8-bit quantization for DSP density



Classification
Results To
Back End DAQ/Trigger

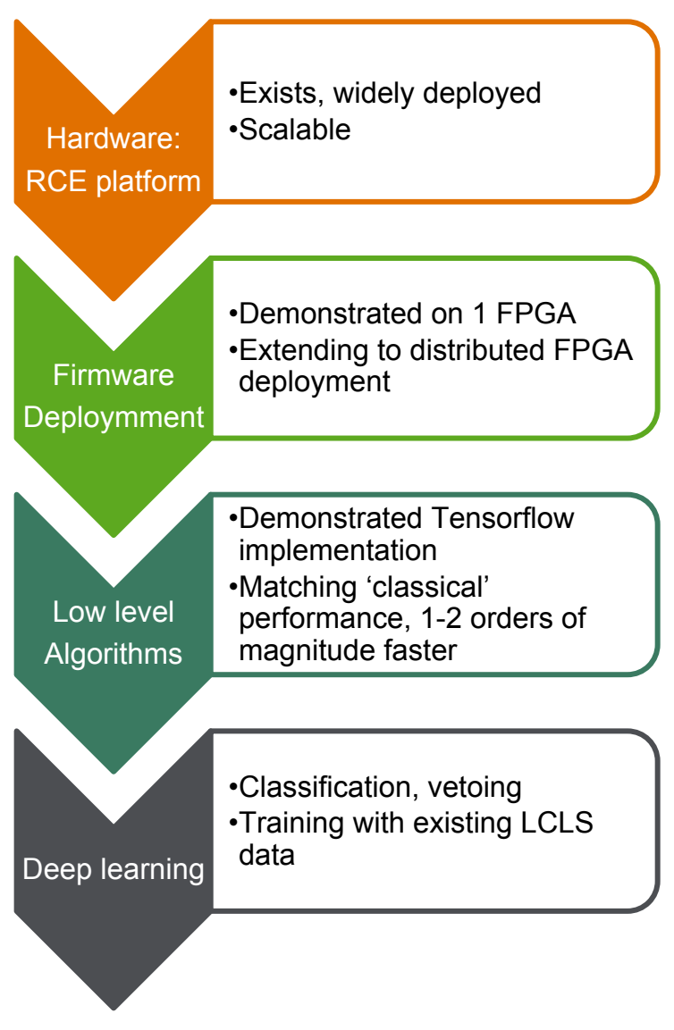
Deploying neural network structures in mid-level daq eliminates need to event build for classification purposes

- First few layers have minimal overlap of data (convolution)
- Inter-FPGA connections overlap data
  - Inter-firmware RUDP
  - Shared hybrid memory cube (HMC)
- Possible to design networks which minimize or eliminate overlaps

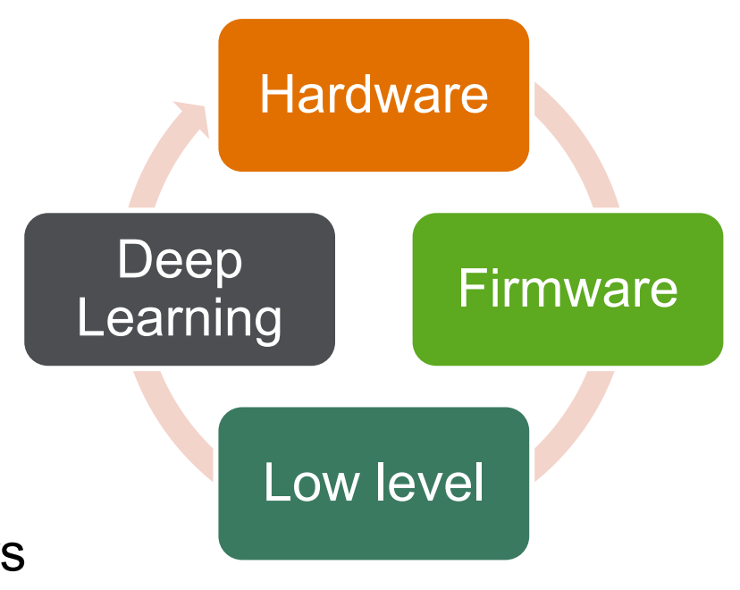
LDRD Proposal: Approach

G. Blaj, C.E. Chang, R. Herbst, J. Thayer

- Framework for deploying ML models on distributed FPGA systems (R. Herbst):
  - Preliminary version demonstrated for single FPGA systems
  - Using the existing RCE platform hardware (developed at SLAC, widely deployed: LCLS, LCLS-II, CERN-Atlas, LSST, Fermi Lab, Jefferson Lab, LSST)
- ML based algorithms for reducing and summarizing data from 2D detectors (G. Blaj):
  - Preliminary version matches performance of 'classical' algorithms with 1-2 orders of magnitude speed up
  - Sequence of hand-crafted filters (based on convolutional networks with optimized architectures)
    - Bonus: each layer, each node have clear physical meaning
    - Co-development with firmware framework and optimization (conversion to integers, pruning)
- Training deep learning models for data summarization, event vetoing (C.-E. Chang, scientists) using existing data and standard GPU training
- Performance optimization and validation with existing application-specific LCLS data sets (J. Thayer, scientists):
  - Applications: single particle imaging, diffuse scattering, protein crystallography, etc.
- Will enable:
  - Real time summarization (online AMI), compression
  - Automatic event vetoing, tuning of experiment parameters
- Risks: limited
  - Existing hardware: PCI-Express based FPGAs, scalable RCE platform, natively supporting LCLS and LCLS-II DAQ
  - Preliminary version of firmware framework for ML model deployment demonstrated for single FPGAs
  - Preliminary version of low-level algorithms demonstrated
  - Application-specific LCLS data sets already exist

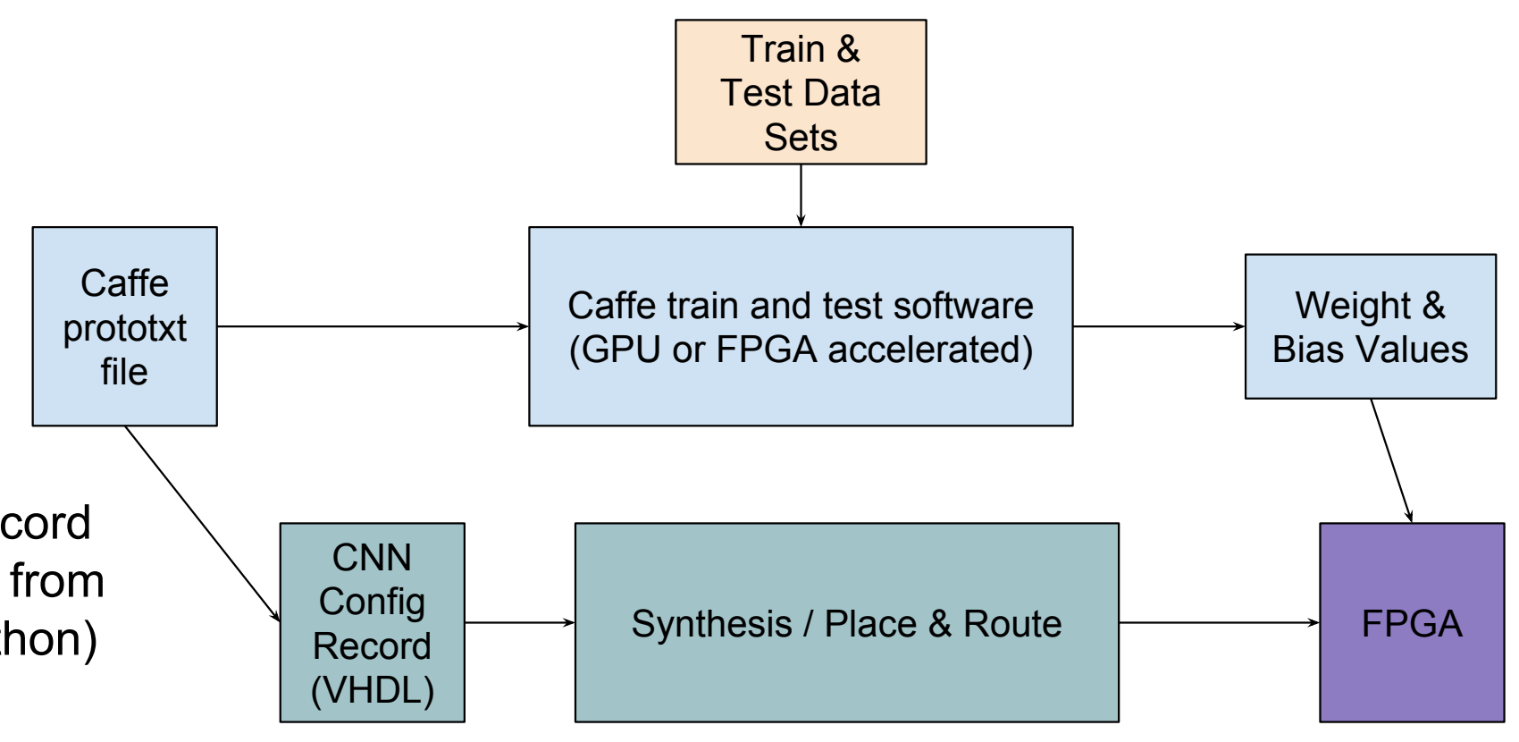

Hardware: RCE platform
•Exists, widely deployed
•Scalable
Firmware Deploymment
•Demonstrated on 1 FPGA
•Extending to distributed FPGA deployment
Low level Algorithms
•Demonstrated Tensorflow implementation
•Matching 'classical' performance, 1-2 orders of magnitude faster
Deep learning
•Classification, vetoing
•Training with existing LCLS data

System optimization



Proposed ML Framework

- Xilinx tool flow is geared towards co-processor based machine learning
  - Possible some openCl design flows allow self contained classification system
  - Proper solution allows the deployed networks to be integrated into a layered DAQ system
- Working design flow for deploying neural networks in FPGA auto generated from Caffe model:



VHDL config record derived directly from prototxt file (python)

ML Framework Proof Of Concept

- VHDL record driven generation of CNN synthesized into a pipelined classification engine]
- Deployed in TID-AIRs Firmware Library SURF
  - https://github.com/slaclab/surf
- LeNet (single digital written decimal recognition) Example:

```
constant CNN_LENET_C : CnnLayerConfigArray(5 downto 0) := (

  0 => genCnnConvLayer (strideX   => 1,  strideY   => 1,
                        kernSizeX => 5,  kernSizeY => 5,
                        filterCnt => 20,
                        padX      => 0,  padY      => 0,
                        chanCnt   => 10, rectEn    => false),

  1 => genCnnPoolLayer (strideX   => 2, strideY   => 2, kernSizeX => 2, kernSizeY => 2),

  2 => genCnnConvLayer (strideX   => 1,  strideY   => 1,
                        kernSizeX => 5,  kernSizeY => 5,
                        filterCnt => 50,
                        padX      => 0,  padY      => 0,
                        chanCnt   => 50, rectEn    => false),

  3 => genCnnPoolLayer (strideX   => 2, strideY   => 2, kernSizeX => 2, kernSizeY => 2),

  4 => genCnnFullLayer ( numOutputs => 500, chanCnt => 50, rectEn => true ),

  5 => genCnnFullLayer ( numOutputs => 10, chanCnt => 1, rectEn => false ));
```

Approaches To ML In Hardware

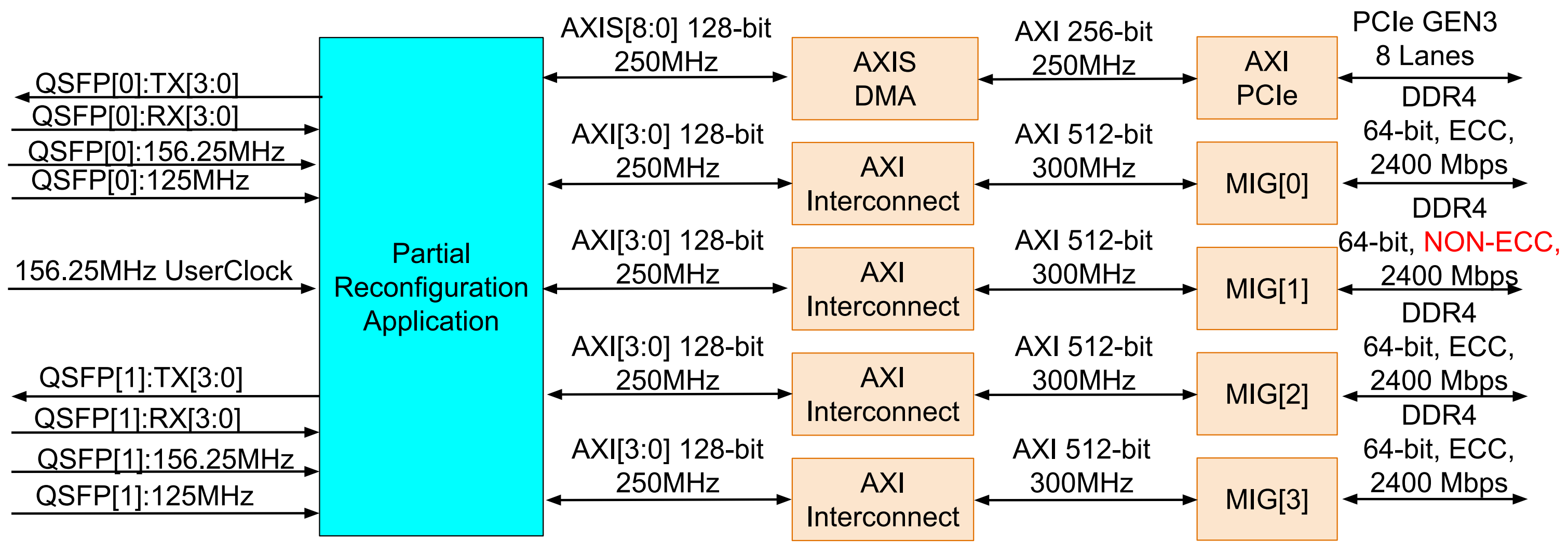
- Four categories of approaches to ML in FPGAs *
  - Single processing engine
    - Systolic array, processing each layer sequentially
    - Software based processing with FPGA coprocessor
  - Streaming architecture
    - One processing engine per network layer
    - Synchronous dataflow (SDF) model for mapping CNNs to FPGAs
    - Often involving software coordination, but not necessary
  - Vector processor
    - Instructions specific to accelerating the operations of convolutions
    - Software driven processing with FPGA coprocessor
  - Neurosynaptic processor
    - Map digital neurons and their interconnecting weights
    - ASIC based processing engines
- FPGA based solutions tend to fall into the "Streaming architecture" or "vector processor" categories

* arXiv:1612.07119 [cs.CV]: "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference"

Existing Frameworks

- The following frameworks are being studied to see if they can be used or serve as a guide for a SLAC framework
- Xilinx: **FINN**
  - Open Source
  - Xilinx Vivado HLS
  - Binarized Neural Networks (BNNs)
  - https://github.com/Xilinx/FINN
- CERN, Columbia, Fermilab, MIT, UI, etc: **hls4ml**
  - Open Source
  - Xilinx Vivado HLS
  - Demonstrated streaming inference with small networks
  - https://hls-fpga-machine-learning.github.io/hls4ml/
- Imperial College London: **fpgaConvNet**
  - Source Not Available
  - Xilinx Vivado HLS
  - http://cas.ee.ic.ac.uk/people/sv1310/fpgaConvNet.html

TID-AIR ES PCI-Express Application Framework

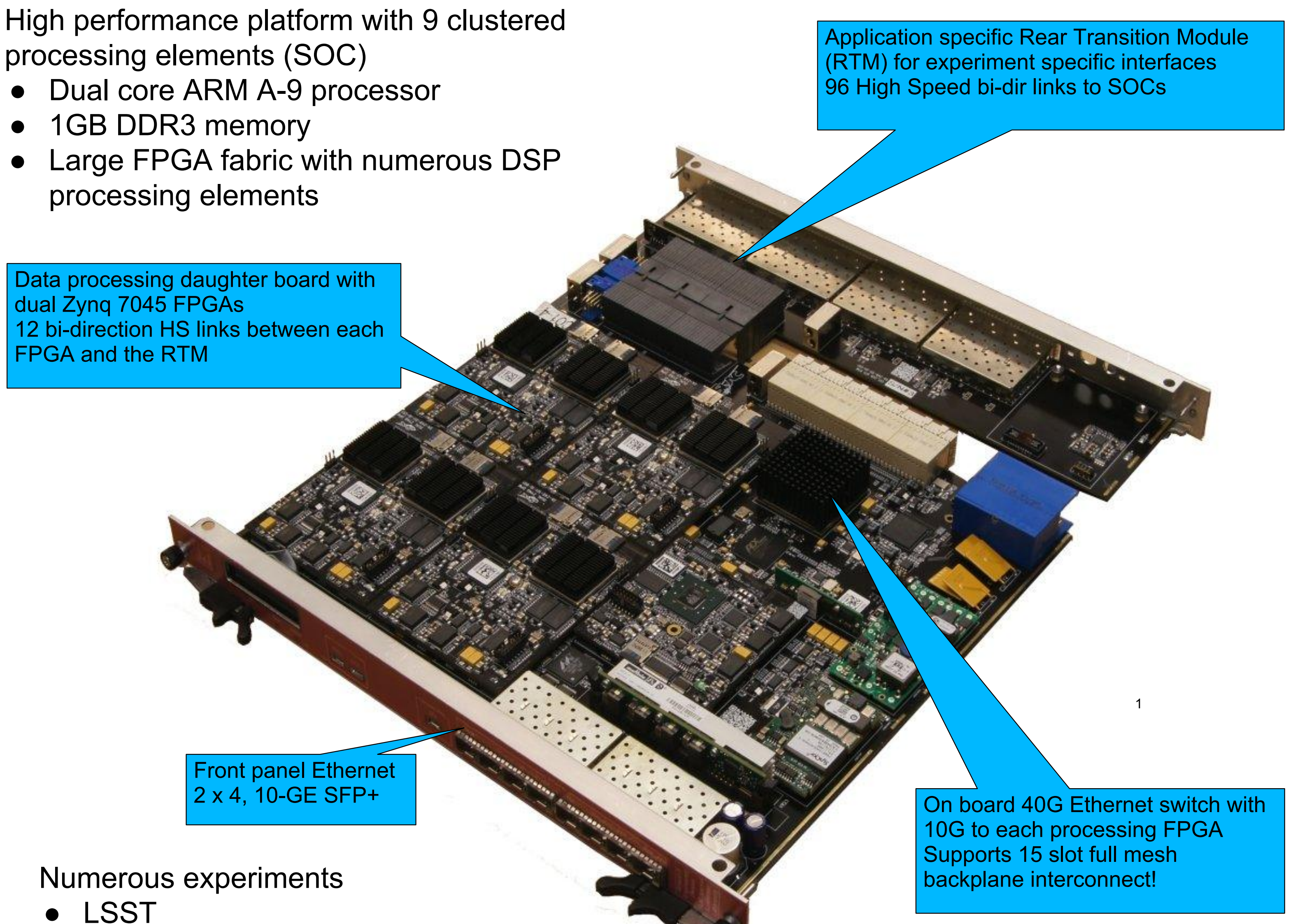


- Provides standard application interfaces which are portable between hardware platforms
- Also provides LCLS1 and LCLS2 timing cores along with timing/data event builder blocks
- Open source version of Amazon Cloud Computing node, static support blocks with user defined partial reconfiguration core

The RCE Platform

High performance platform with 9 clustered processing elements (SOC)

- Dual core ARM A-9 processor
- 1GB DDR3 memory
- Large FPGA fabric with numerous DSP processing elements



Application specific Rear Transition Module (RTM) for experiment specific interfaces
96 High Speed bi-dir links to SOCs

Data processing daughter board with dual Zynq 7045 FPGAs
12 bi-direction HS links between each FPGA and the RTM

Front panel Ethernet
2 x 4, 10-GE SFP+

On board 40G Ethernet switch with 10G to each processing FPGA
Supports 15 slot full mesh backplane interconnect!

Numerous experiments

- LSST
- Heavy Photon Search, LDMX
- DUNE 35Ton / ProtoDUNE
- ATLAS Muon
- ITK Development
- nEXO (Baseline)

SOC platform combines stable base firmware / software with application specific cores

- HLS for C++ based algorithms & compression
- Matlab for RF processing

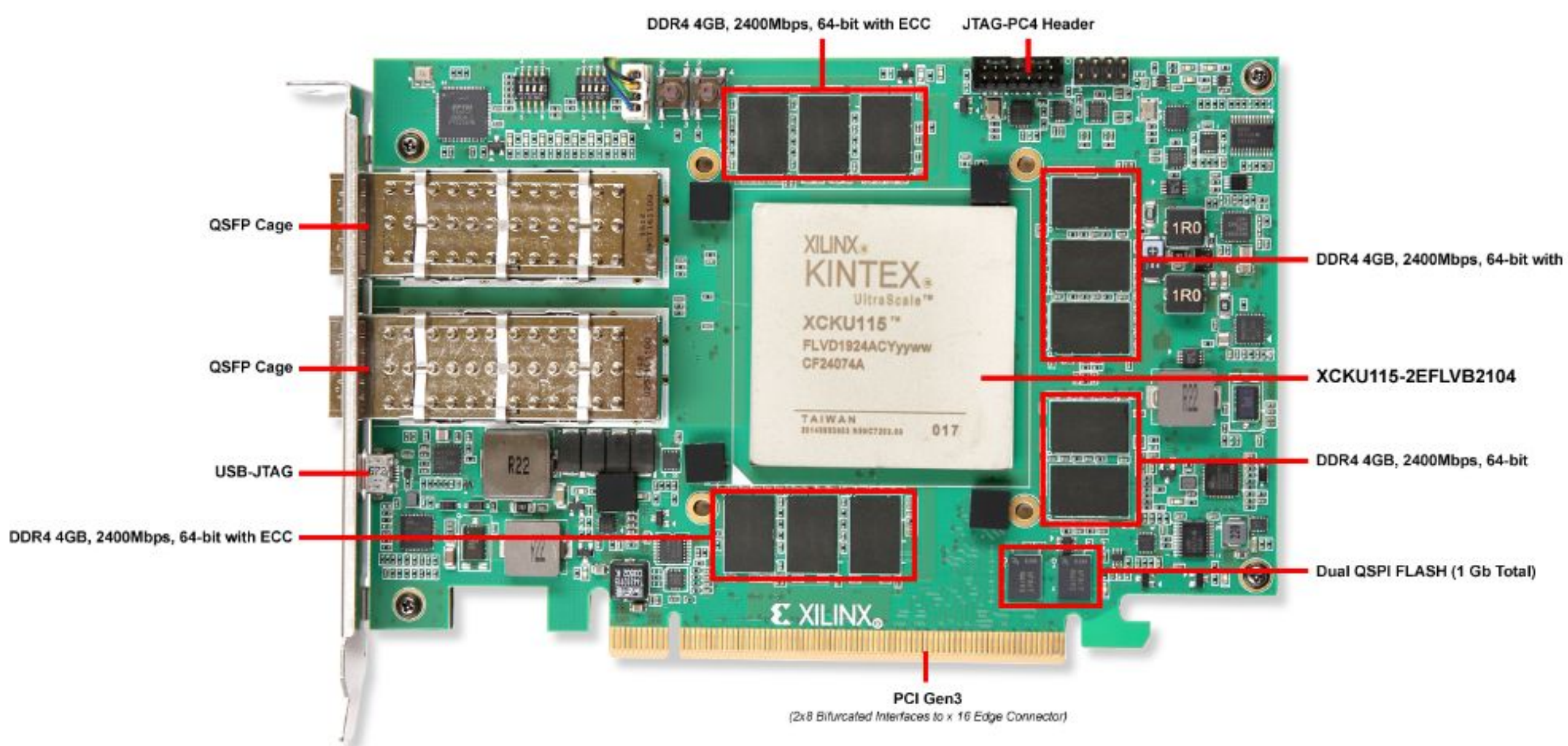
Commercial Xilinx Hardware

Xilinx KCU1500 co-processor
- XCKU115 FPGA
- 2 QSFP optical modules
- 16GB DDR
- Amazon AWS

Xilinx Virtex UltraScale+ VCU1525
- XCVU9P FPGA
- 2 QSFP optical modules
- 64GB DDR

Xilinx Alveo U200
- 2 QSFP28 optical modules
- 64GB DDR

Xilinx Alveo U250
- 2 QSFP28 optical modules
- 64GB DDR


DDR4 4GB, 2400Mbps, 64-bit with ECC
JTAG-PC4 Header
QSFP Cage
QSFP Cage
USB-JTAG
DDR4 4GB, 2400Mbps, 64-bit with ECC
DDR4 4GB, 2400Mbps, 64-bit with ECC
XCKU115-2EFLVB2104
DDR4 4GB, 2400Mbps, 64-bit
Dual QSPI FLASH (1 Gb Total)
PCI Gen3
(2x8 Bifurcated Interfaces to x 16 Edge Connector)