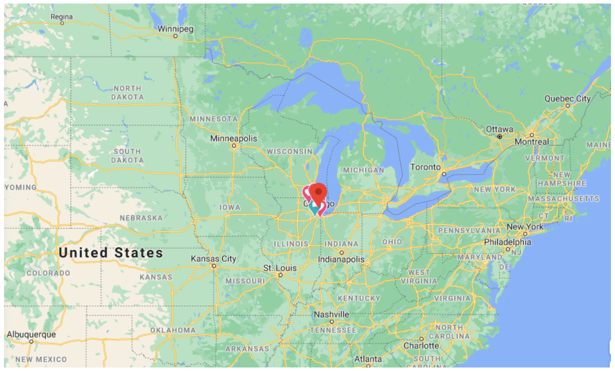
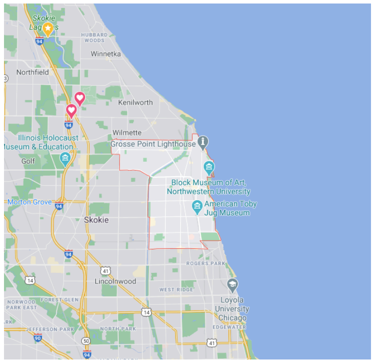
Northwestern

Seda Ogrenci-Memik
ECE & CS

"I'm a software engineer, so I can confirm it works by magic."

Northwestern





Computer Engineering Research @Northwestern

- Housed within the Electrical and Computer Engineering Department, co-owned by the Computer Science Department
  - Faculty have dual appointments
  - PhD programs in both departments have Computer Engineering tracks
- Design Automation / CAD and VLSI, Computer Architecture, Internet of Things, Embedded and Cyberphysical Systems, Big Data & AI

Computer Engineering Research @Northwestern

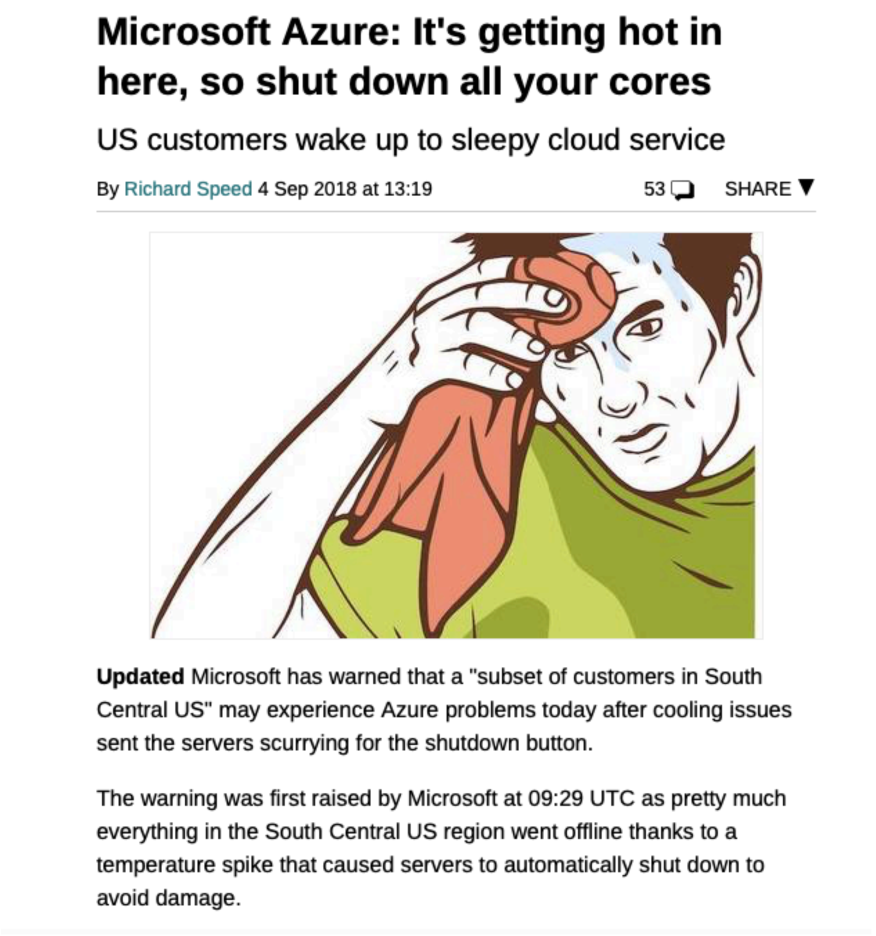
- Hardware assisted ML
- Neuromorphic computing
- Hardware security
- Photonic circuits in computer architectures
- Emphatic computing
- Batteryless computing
- Secure and verified software for automotive application
- Adaptive architecture – compiler in the loop

My Research Interests

- Electronic Design Automation (EDA)
- Circuits and Systems
- High Performance Computing (HPC) Systems

Research Activities

- Design Automation
  - High-Level Synthesis
    - Converting applications described with programming languages to hardware
- Circuits and Systems
  - Thermal and Power Management of ICs
    - Temperature Sensors, 3D Integration, Power and Performance Modeling using Machine Learning



Microsoft Azure: It's getting hot in here, so shut down all your cores

US customers wake up to sleepy cloud service

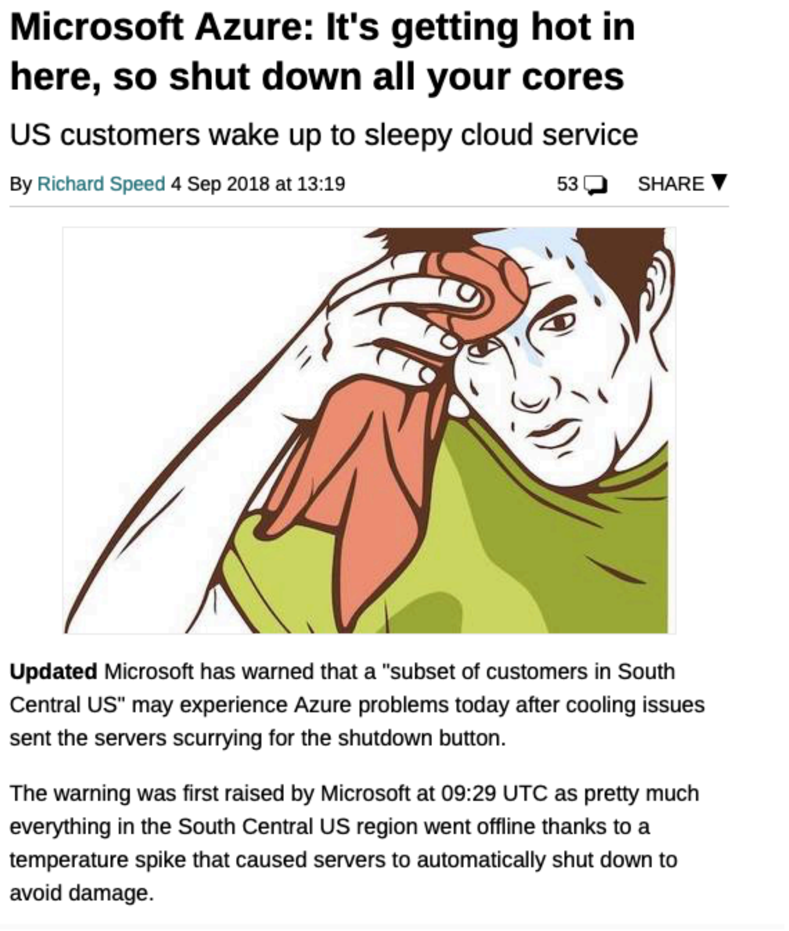
By Richard Speed 4 Sep 2018 at 13:19 53 SHARE ▼

Updated Microsoft has warned that a "subset of customers in South Central US" may experience Azure problems today after cooling issues sent the servers scurrying for the shutdown button.

The warning was first raised by Microsoft at 09:29 UTC as pretty much everything in the South Central US region went offline thanks to a temperature spike that caused servers to automatically shut down to avoid damage.

Research Activities

- High Performance Computing Systems
  - FPGA-based accelerators for HPC applications
  - Thermal and Power Management in large scale systems



Microsoft Azure: It's getting hot in here, so shut down all your cores

US customers wake up to sleepy cloud service

By Richard Speed 4 Sep 2018 at 13:19 53 SHARE ▼

Updated Microsoft has warned that a "subset of customers in South Central US" may experience Azure problems today after cooling issues sent the servers scurrying for the shutdown button.

The warning was first raised by Microsoft at 09:29 UTC as pretty much everything in the South Central US region went offline thanks to a temperature spike that caused servers to automatically shut down to avoid damage.

HW Assisted ML

- Design Automation for real-time AI Cyberinfrastructure for scientists who deploy HW systems for ML

Physics

Particle Tracking

Accelerator Control

Material Science

Image Reconstruction
Real-time control of microscopy

Astro-physics

Semantic compression

Data filtering, compression, reconstruction, feedback controllers

Handling Big Data

- Source of the computational challenge
  - Particle acceleration experiment in high-energy physics
    - Billions of "event" happen every ~25ns creating Pb/ sec data rates
    - "Interesting" events (~3% of all) need to be recognized and filtered for further processing
  - Sky survey data collected from observatories stream in at rates ~Gb/s (or more)
    - Limited energy (generators in South Pole) to move data to datacenters

Handling Big Data

- Source of the computational challenge
  - CPU-GPU systems perform streaming inference on images captured by an electron microscope within ~300ms latency
  - Desired goal is to perform inference on images captured at 10s of millions of frames per second under 50ms latency
  - ~50ms latency could allow real-time control of the EM ***during*** material synthesis
    - Think of making defects for a quantum material, deposition of nano layers, etc.

Projects – Complete/Underway

- READS – Accelerator Real-time Edge AI for Distributed Systems
- Design of a reconfigurable autoencoder algorithm for detector front-end ASICs

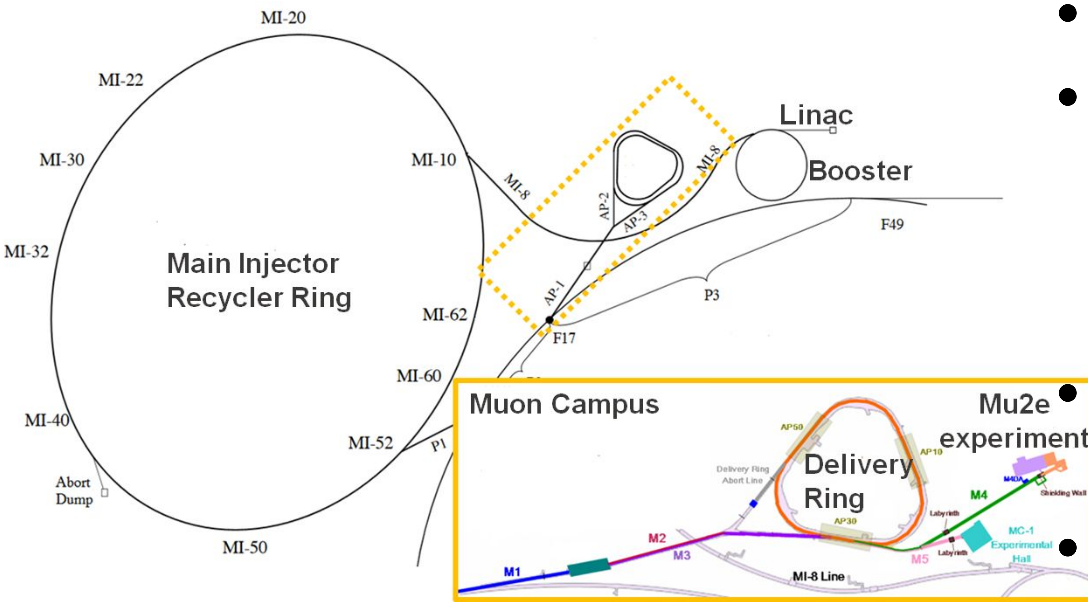
Projects – Preliminary

- CryoAI - 22nm testchip
- Adaptive ML accelerators

READS – Accelerator Real-time Edge AI for Distributed Systems

- Goal: integrate ML into accelerator operations
- Challenges: Beam Loss
  - High Energy Physics (HEP) experiment use proton beams
  - Particles get lost through interactions with the beam vacuum pipe
  - Intensity/pace of beam extraction affects radiation in the environment
  - Human operators tune parameters of hundreds to thousands of devices inside the accelerator complex
  - If beam loss (“leak”) is at extreme levels, system needs to shut-down – loss of access to facility

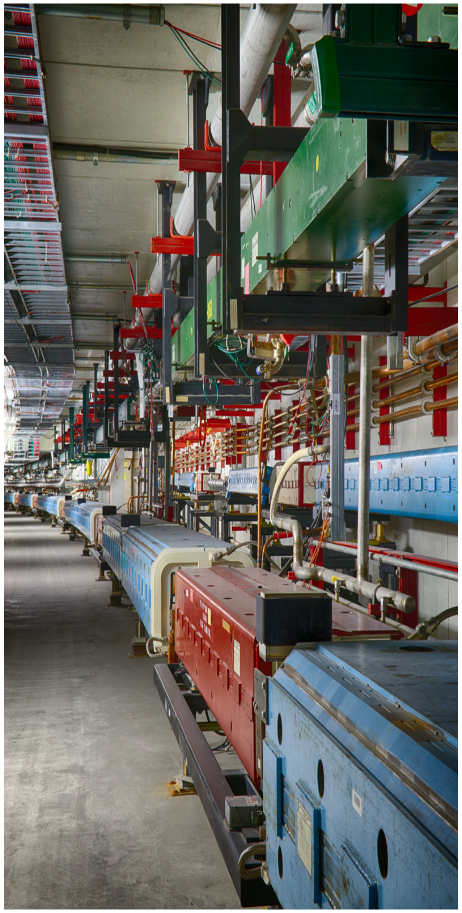
READS – Accelerator Real-time Edge AI for Distributed Systems



- Muon Complex
- Two rings are on top of each other: Main Injector Ring and the Recycler Ring
- Protons are accelerated within the Booster
- Injected into the Recycler Ring and Main Injector Ring

Project Overview

- Main Injector and Recycler share an enclosure
- Both machines can and do often have high intensity beam in them simultaneous
- Both machines can generate significant beam loss
- “Lost” portion of the beam escapes as dangerous radiation and if that is too much, it forces the facility to shot down.
- The machine origin of a beam loss is often hard to distinguish



Main Injector tunnel
Recycler (top) Main Injector (bottom)

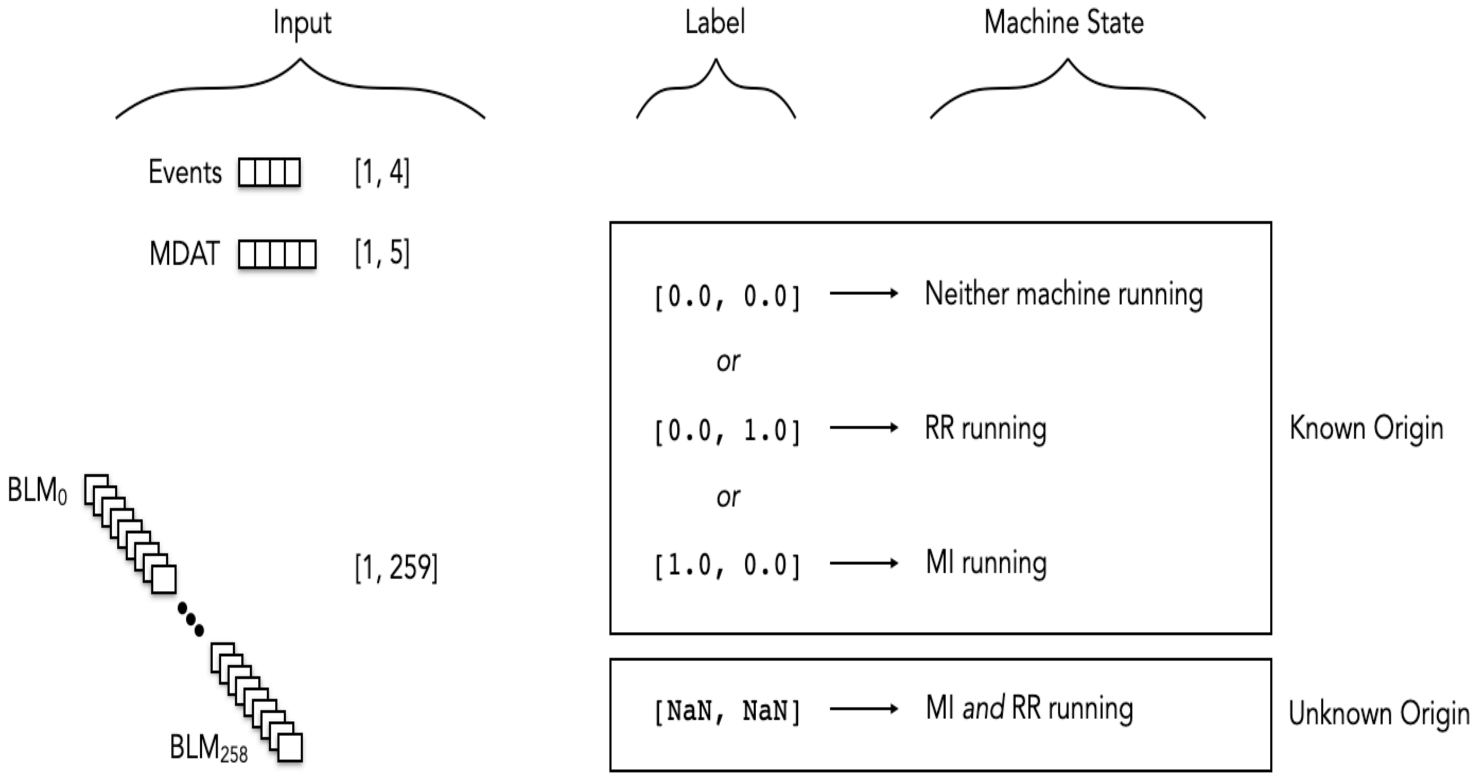
Project Overview

- Using time, location and state of the machine, machine experts can sometimes attribute loss to a particular machine
  - This suggests a Machine Learning (ML) model may be trainable to automatically attribute loss and replicate or improve upon the expert's ability
- Often losses from one machine end up tripping the machine permit of the other resulting in unnecessary beam downtime

The projects aims to deploy a machine learning model on a FPGA that when fed streamed beam loss readings from around the Main Injector complex, will infer in real-time the machine loss origin

ML Model Architecture: Overview

Objective: Assign BLM-wise probabilities for the loss originating in MI/RR



Input

Events [1, 4]

MDAT [1, 5]

BLM_0 … BLM_{258} [1, 259]

Label

Machine State

`[0.0, 0.0]` ⟶ Neither machine running

or

`[0.0, 1.0]` ⟶ RR running

or

`[1.0, 0.0]` ⟶ MI running

Known Origin

`[NaN, NaN]` ⟶ MI *and* RR running

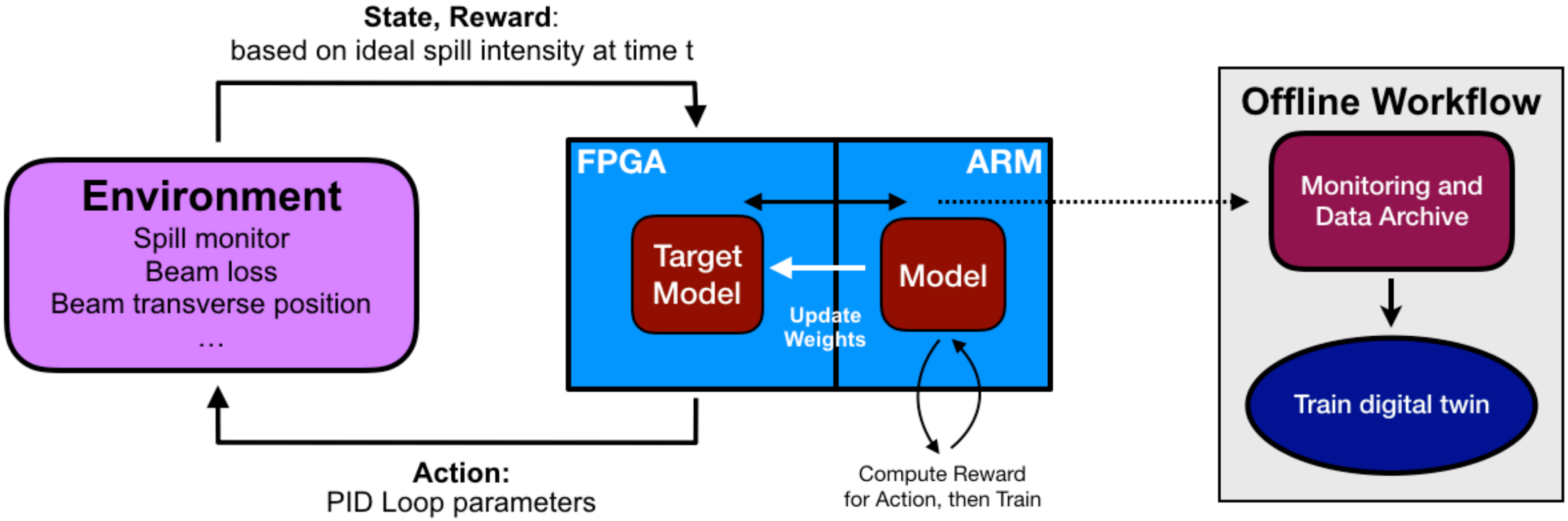
Unknown Origin

READS – Accelerator Real-time Edge AI for Distributed Systems

- System Design Goal: PID Controller gains need to be optimized in real-time ~ms
- The ML Processor receives inputs from sensors
  - – beam position monitor (BPM)
  - – beam loss monitor (BLM)

READS – Accelerator Real-time Edge AI for Distributed Systems

- System Architecture:

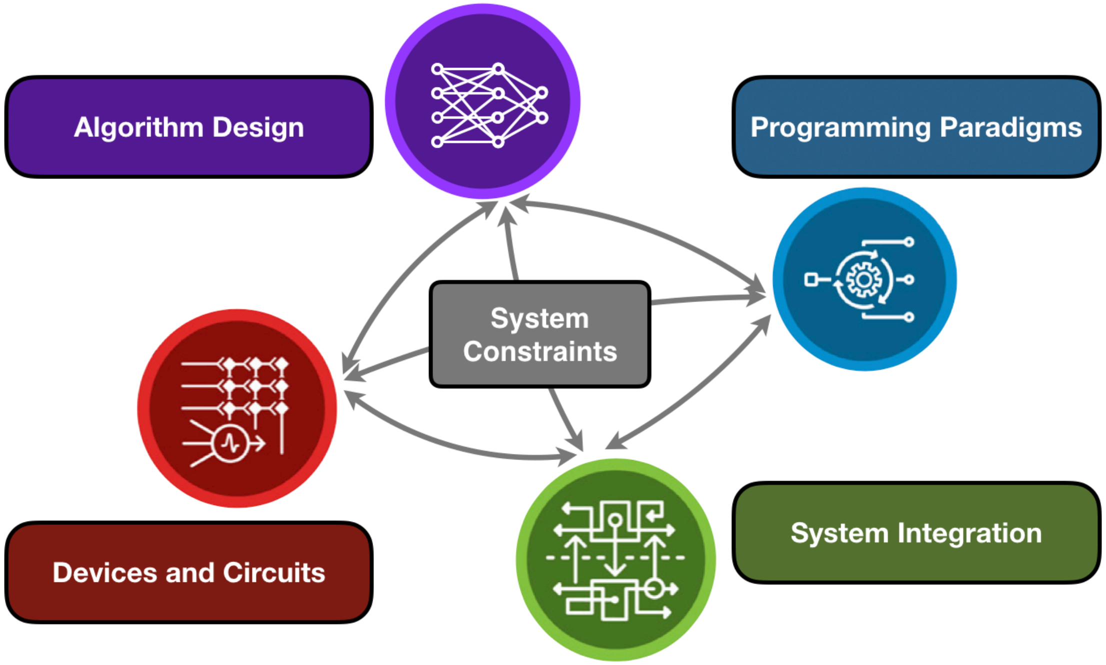


READS – Accelerator Real-time Edge AI for Distributed Systems

- System Architecture:
  - Edge device continuously optimizes the online control agent
  - Data streamed to a cloud system for large scale training

READS – Accelerator Real-time Edge AI for Distributed Systems

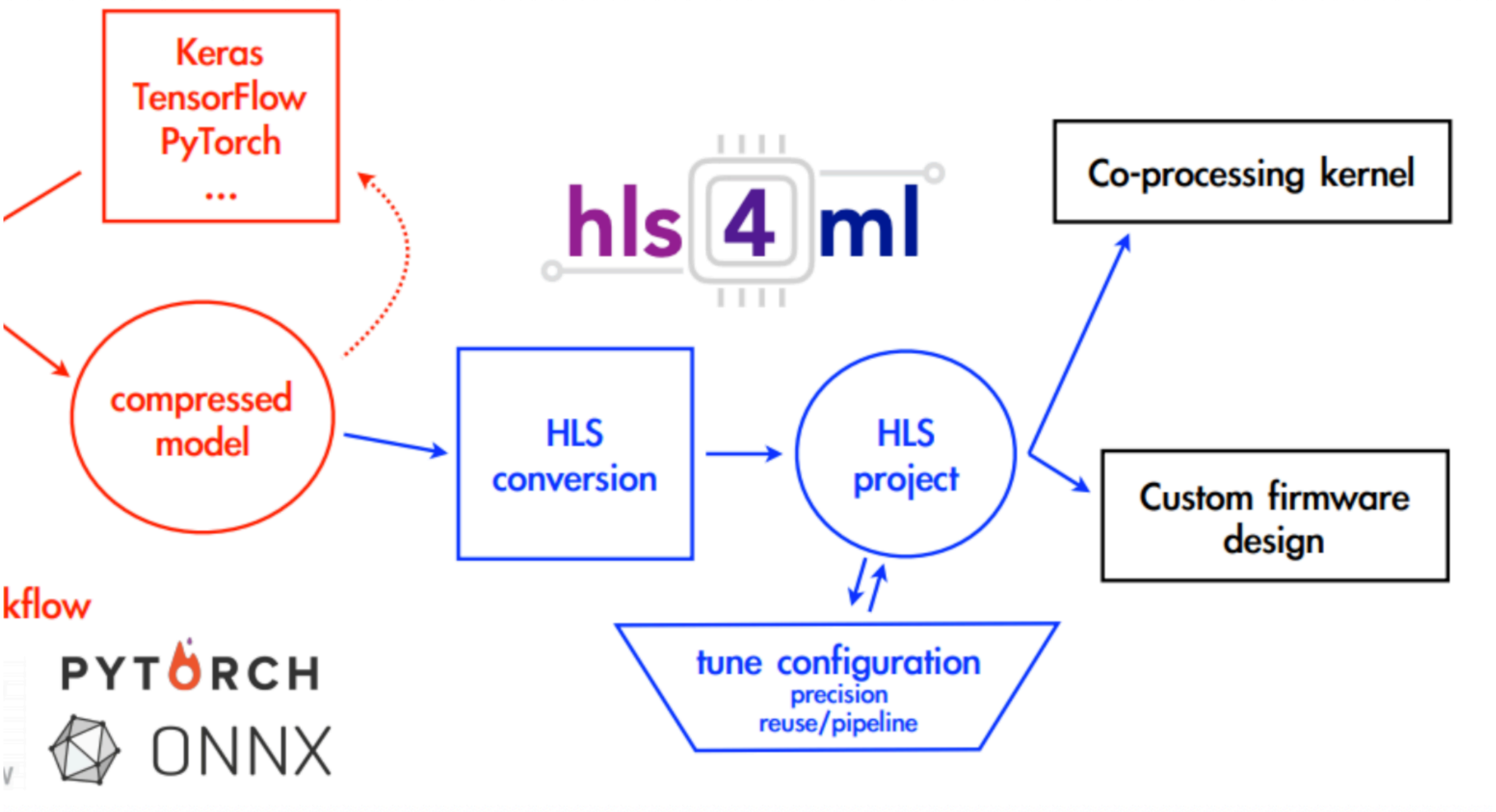
- Algorithm-Architecture Co-design
  - – A common toolchain to program FPGA devices as well as create interfaces



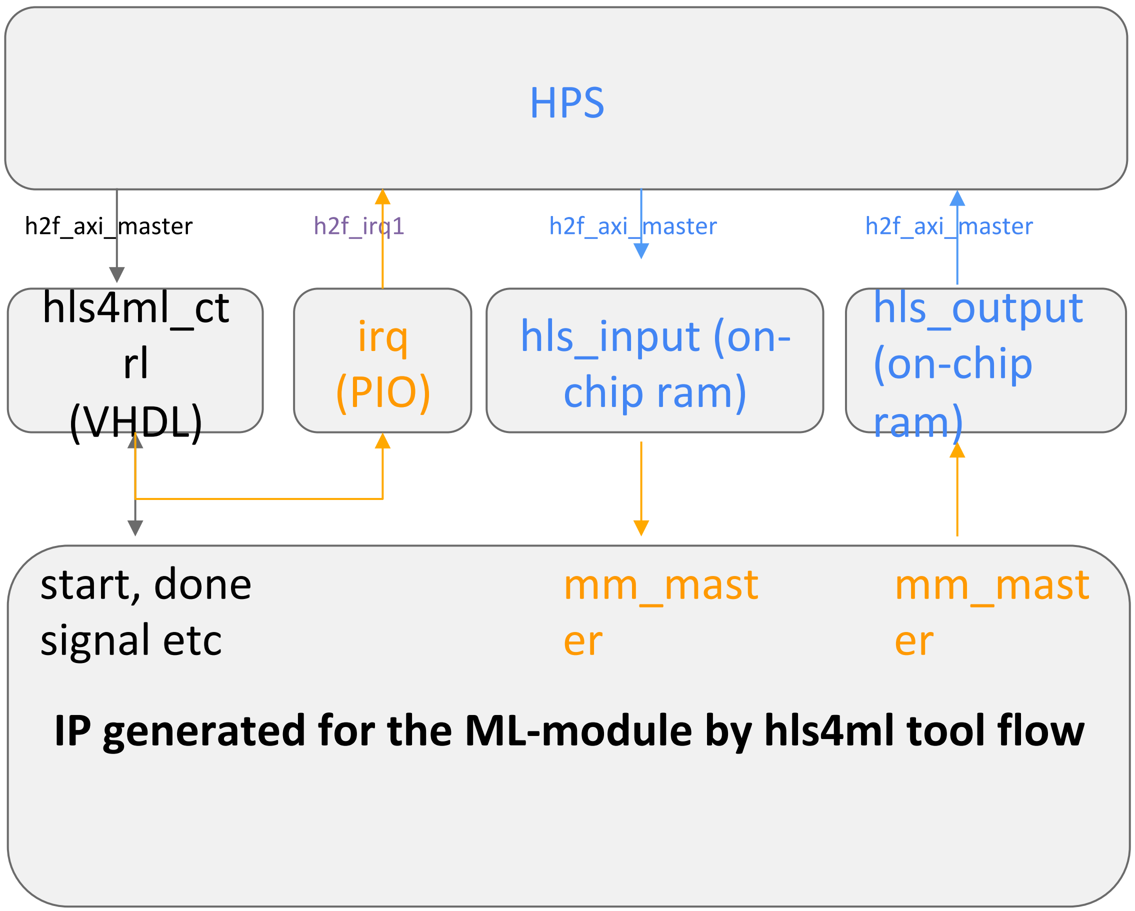
READS – Accelerator Real-time Edge AI for Distributed Systems

- Algorithm-Architecture Co-design
  - – Create modular neural network components
    - Regroup, recombine
  - – Establish physics/science-aware methodology
    - Hardware-aware quantization and pruning techniques
    - Homogeneous versus heterogeneous quantization
    - Quantization-aware training
    - Control resource re-use trade-offs of high level synthesis tools
    - Differentiate between the relative hardware cost of storage, DSP, interconnect

READS – Accelerator Real-time Edge AI for Distributed Systems



Target - Arria10 HPS + FPGA (on-chip ram)



HPS

h2f_axi_master

h2f_irq1

h2f_axi_master

h2f_axi_master

hls4ml_ctrl (VHDL)

irq (PIO)

hls_input (on-chip ram)

hls_output (on-chip ram)

start, done signal etc

mm_master

mm_master

IP generated for the ML-module by hls4ml tool flow

+ Testbench of hls4ml

Hardware Optimization for the ML-IP

- The current model implemented as IP consists of
  - Dense Layer – ReLU – Dense Layer - Sigmoid
- 259 inputs and 518 outputs
  - 16 bits ac_fixed values
    - Representation still under investigation
- HLS (hls4ml) tool was optimized to explore resource sharing for computation such as dense layer
- HLS tool was not equipped with features to optimize other layers such as Sigmoid
  - When we are dealing with a model with 100s of outputs situation changes
- Logic Synthesis stage (Quartus)
  - Large scale models introduce clock timing violations that were not observed in smaller models

Project #2: Reconfigurable autoencoder for detector front-end ASICs

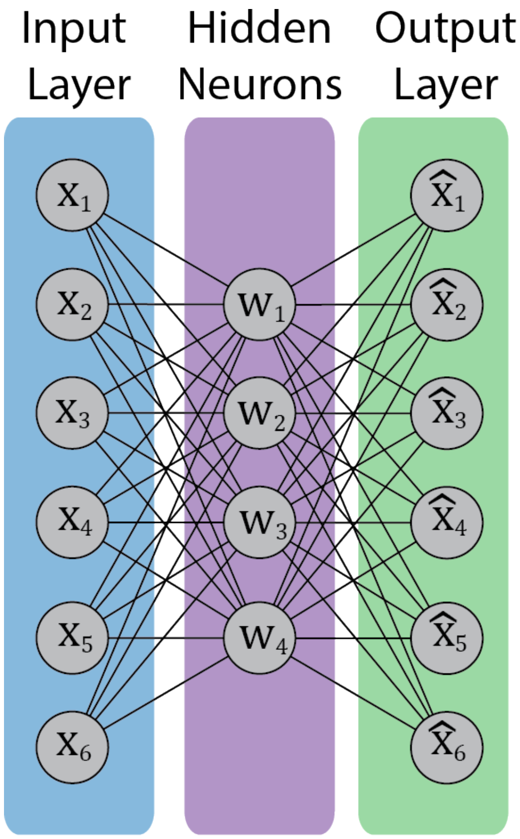
- Edge computing for particle collider experiments
- Data collected from a large number of photon detectors are compressed to representative information of the “shape”
  - Charge measurements from the detectors are compressed to a radiation pattern
  - 6 million detector channels sending data at 40MHz
  - The data from the original space is compressed to lower dimensionality by the edge ASIC, transmitted, then decoded on the receiving end

Reconfigurable autoencoder for detector front-end ASICs

- System constraints
  - Low power
    - Will be part of a larger system with power budget
  - Radiation tolerant
  - Reprogrammable weights through accessible registers to enable updates and customization

Reconfigurable autoencoder for detector front-end ASICs

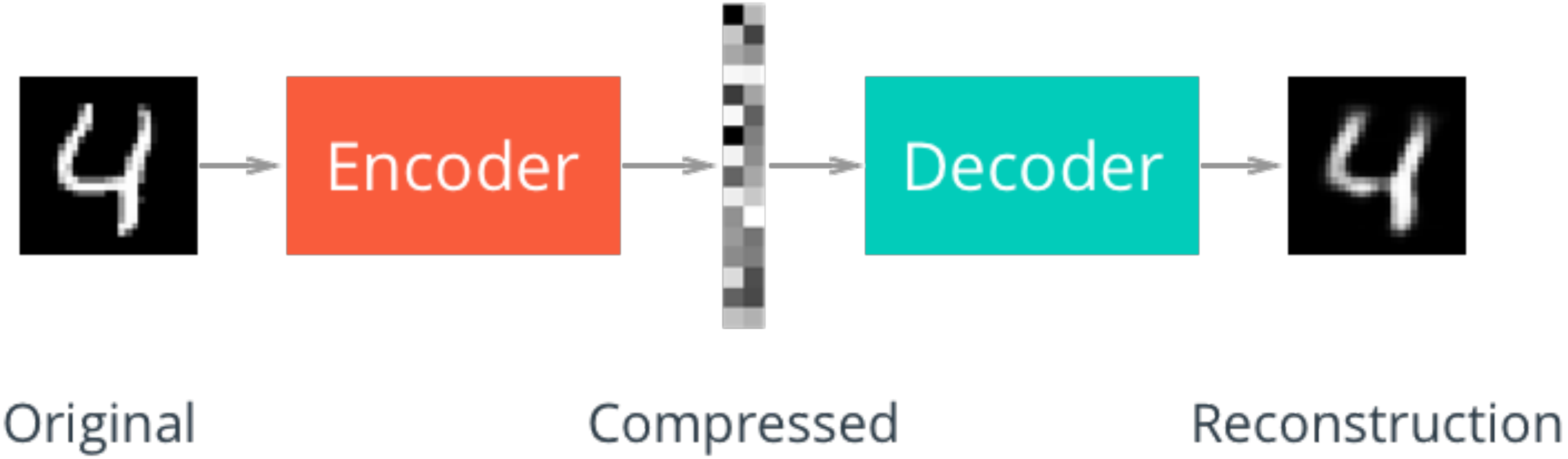
- Autoencoder: neural network with single convolutional layer followed by a dense layer



$X \in \mathbb{R}$

$$\hat{X} = \sum_{\forall N} w_i x_i + b_i$$

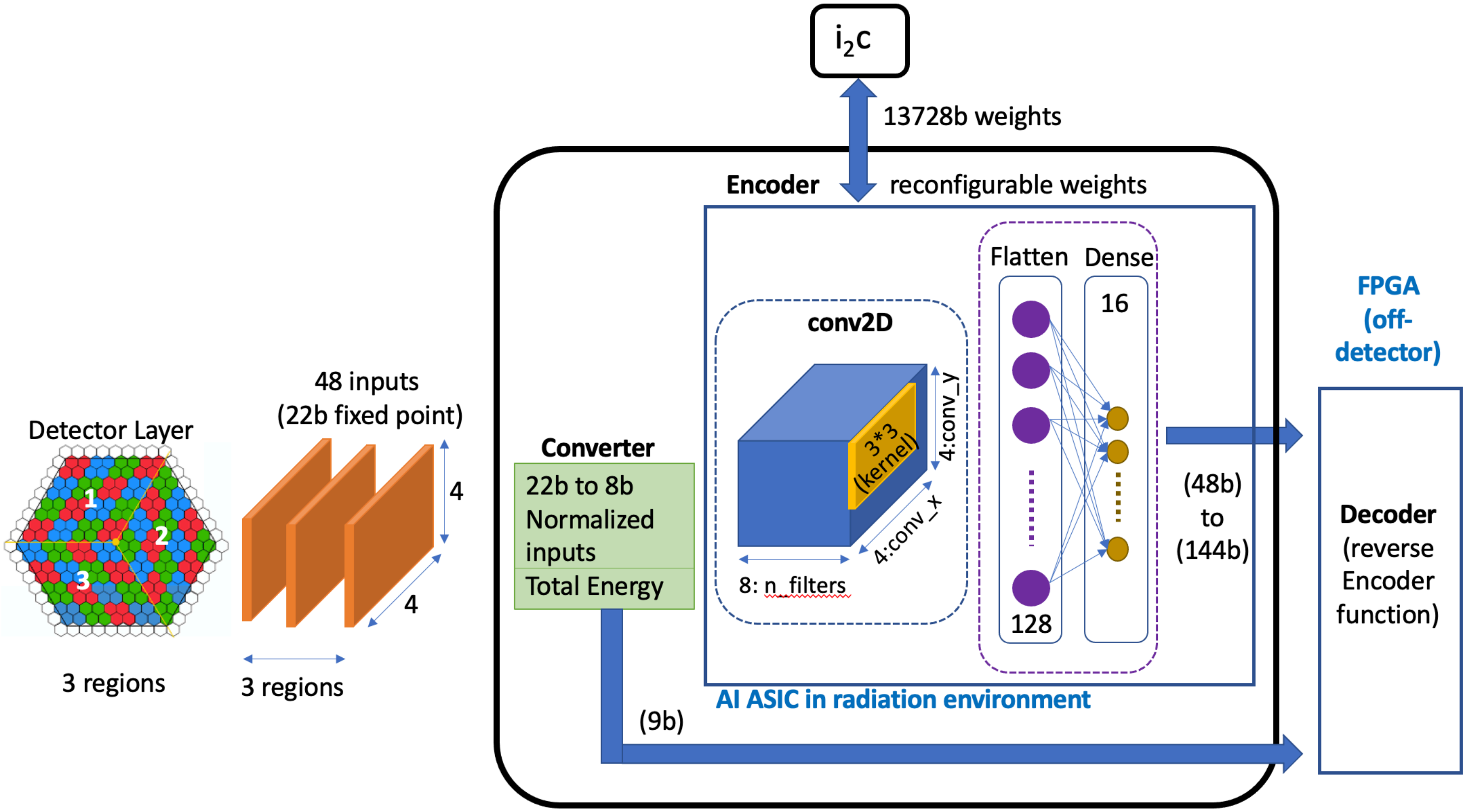
Reconfigurable autoencoder for detector front-end ASICs



Glossary of Deep Learning: Autoencoder, by Jaron Collins

Reconfigurable autoencoder for detector front-end ASICs

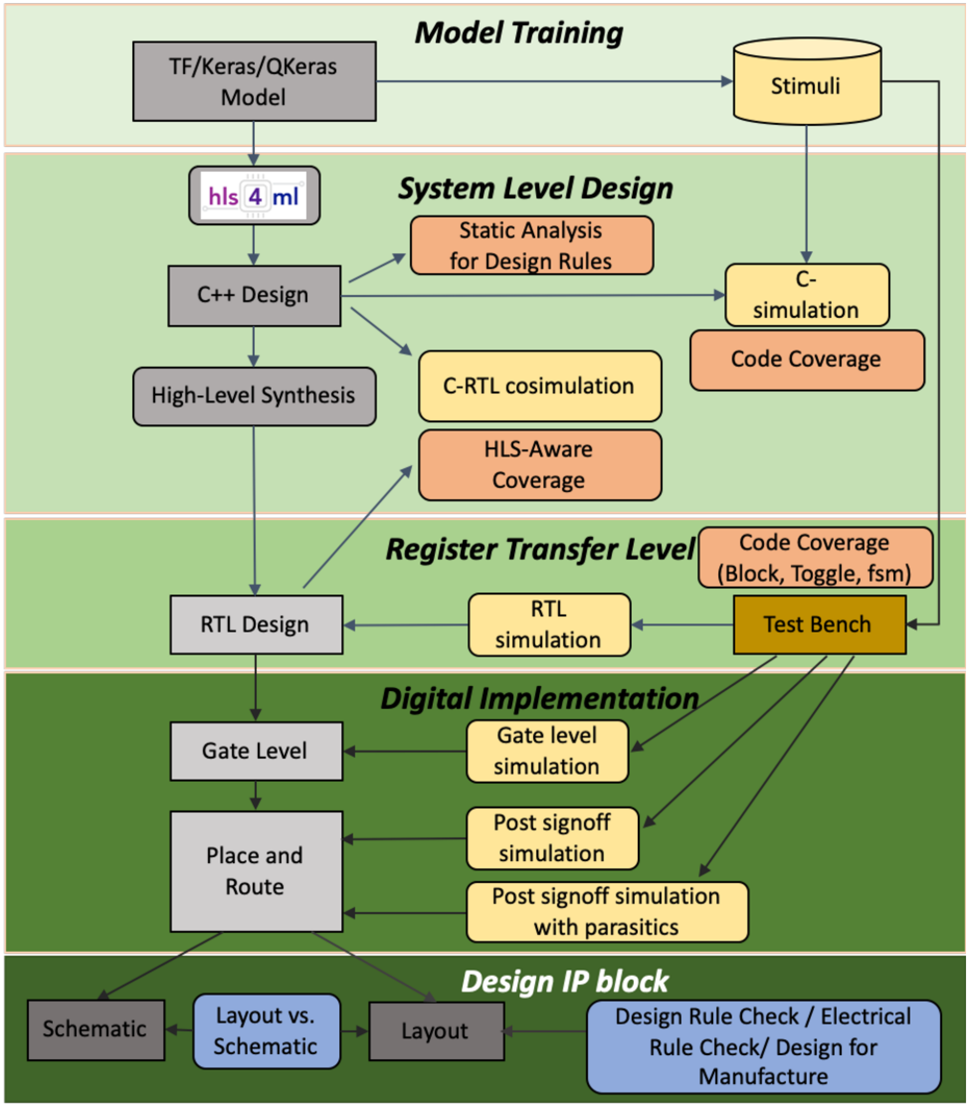
- Dataflow
  - 22-bit signals from 48 detector cells



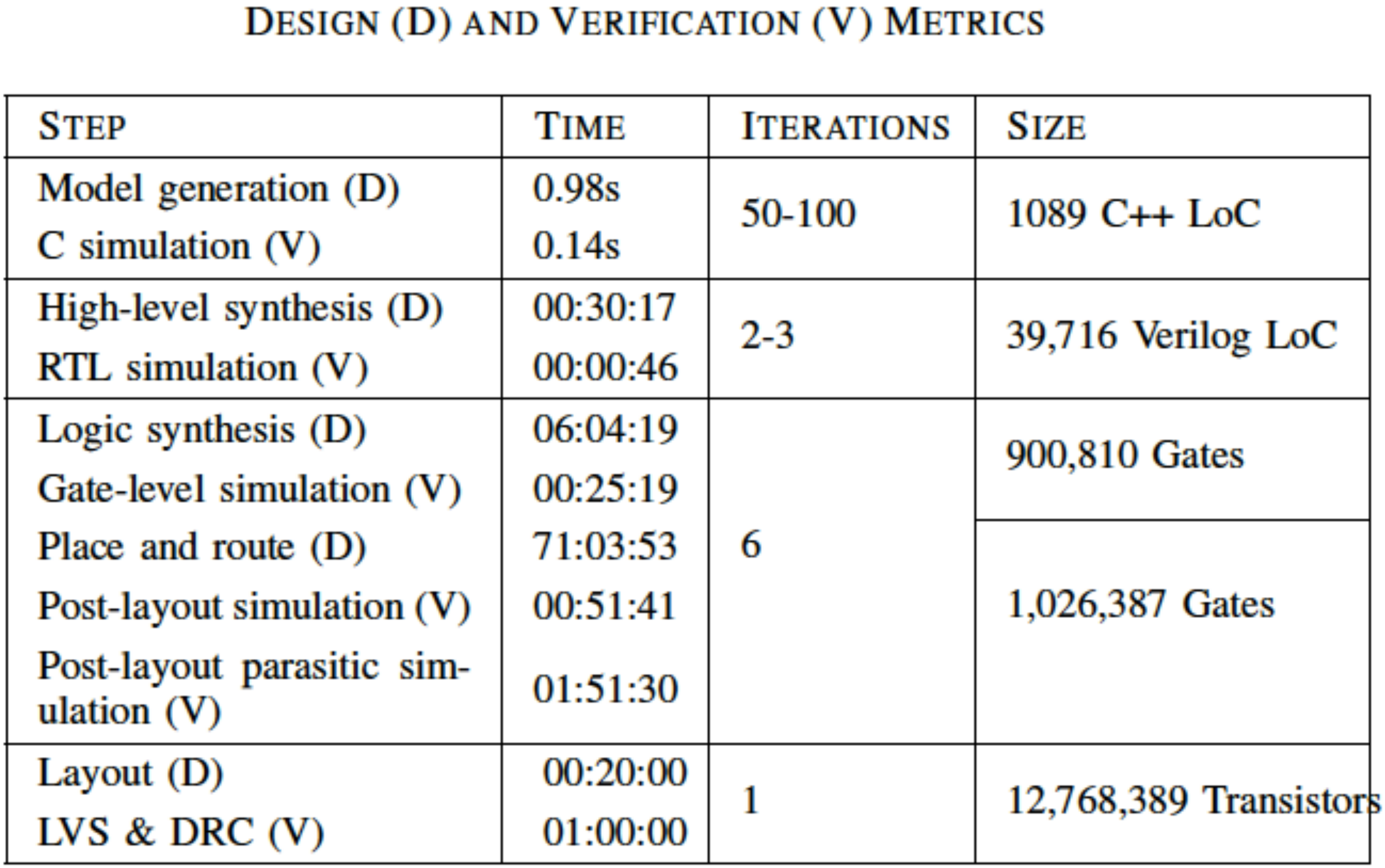
Reconfigurable autoencoder for detector front-end ASICs

- Network properties
  - CNN layer: eight 3x3x3 kernel matrices resulting in 128 outputs
  - ReLu activation after CNN and after final dense layer
  - 6-bits weights
  - Dense layer produces 16 10-bit outputs
  - Chip can be reconfigured to produce as low as total of 64 bits in output

Reconfigurable autoencoder for detector front-end ASICs



Reconfigurable autoencoder for detector



DESIGN (D) AND VERIFICATION (V) METRICS

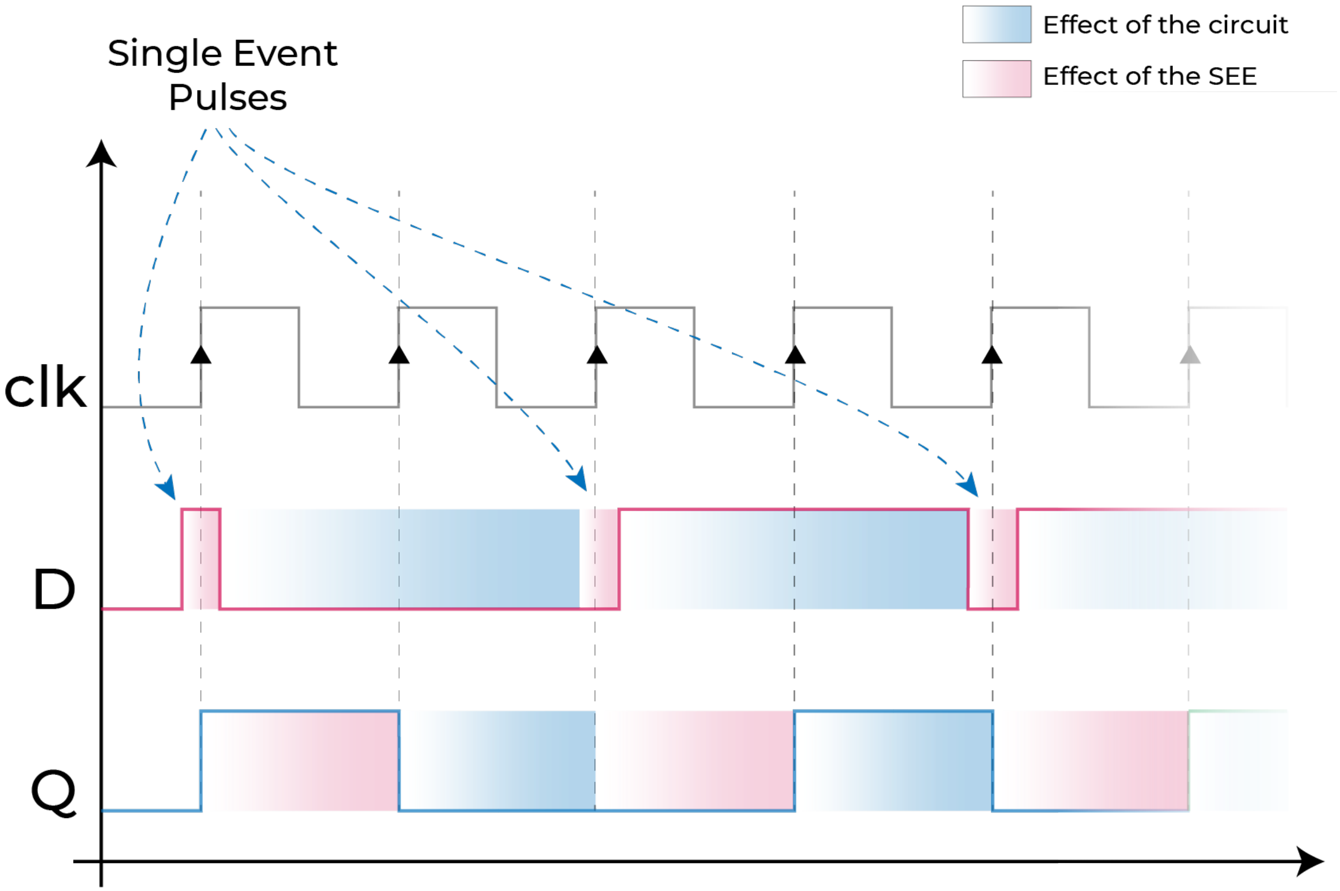
STEP	TIME	ITERATIONS	SIZE
Model generation (D)	0.98s	50-100	1089 C++ LoC
C simulation (V)	0.14s		
High-level synthesis (D)	00:30:17	2-3	39,716 Verilog LoC
RTL simulation (V)	00:00:46		
Logic synthesis (D)	06:04:19	6	900,810 Gates
Gate-level simulation (V)	00:25:19		
Place and route (D)	71:03:53		1,026,387 Gates
Post-layout simulation (V)	00:51:41		
Post-layout parasitic simulation (V)	01:51:30		
Layout (D)	00:20:00	1	12,768,389 Transistors
LVS & DRC (V)	01:00:00		

Reconfigurable autoencoder for detector front-end ASICs

- Chip specs
  - 6b weight and bias parameters
    - Total: 2,286 = 13,724 bits
  - Parameters loaded via I2C interface
  - Decoder component implemented off-detector on FPGA
  - Chip latency – 25ns
  - 7nJ per inference
  - 280mW
  - Can withstand 200MRad ionizing radiation
  - 2.5mm^2

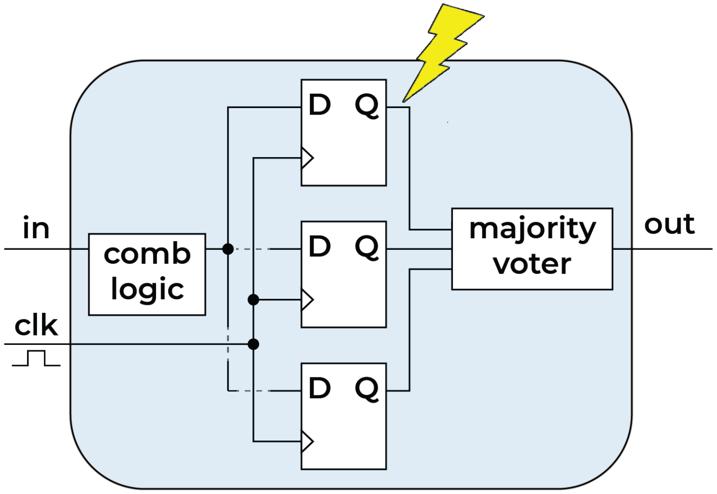
Reconfigurable autoencoder for detector front-end ASICs

- Protection against Single Event Upsets

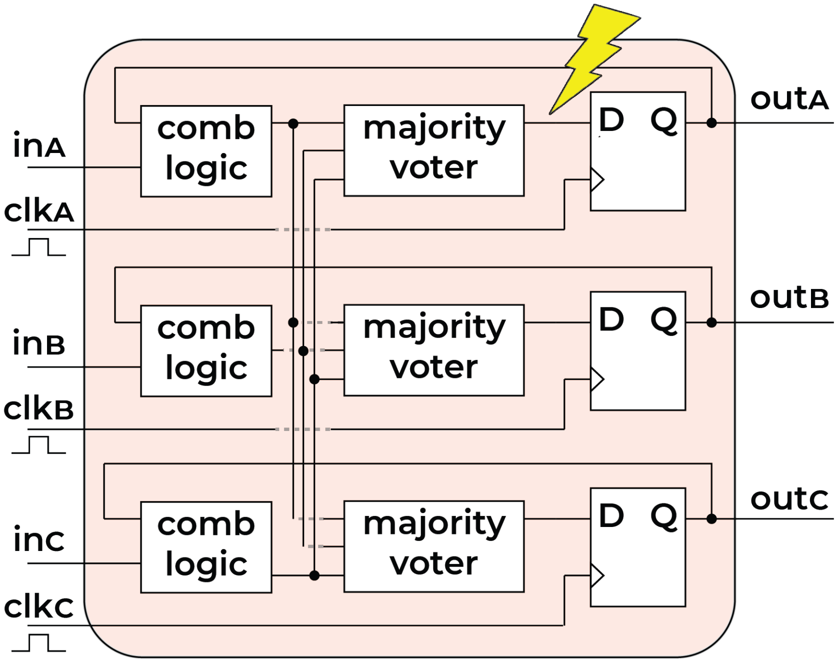


Reconfigurable autoencoder for detector front-end ASICs

- Protection against Single Event Upsets



Partial TMR: suppresses error without correction



Full TMR: both suppresses and corrects error

Reconfigurable autoencoder for detector front-end ASICs

- Partial TMR can also be implemented in a few different hybrid modes
  - Applied to only registers
  - Include clock lines
  - Apply to a sub-block

Reconfigurable autoencoder for detector front-end ASICs

- Exploring trade-offs in TMR
  - Area versus TMR coverage
  - Leakage Power versus TMR coverage
  - Fanout (routability) versus TMR coverage
  - Timing (impact on slack)

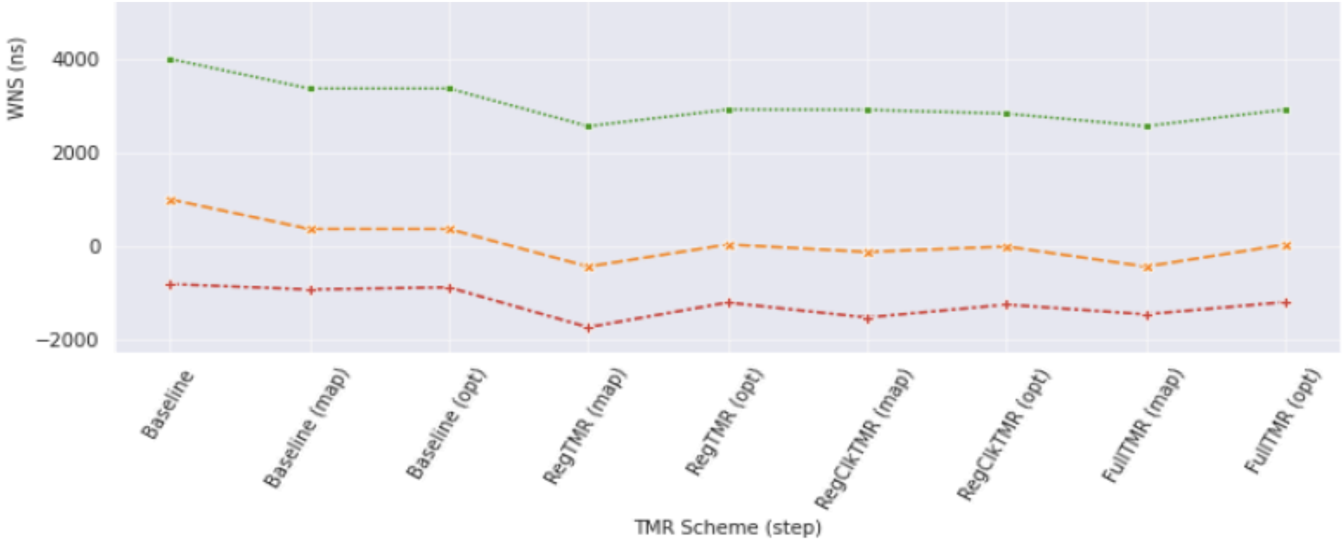


Fig. 6: Variation in Worst Negative Slack (WNS) for different triplication schemes applied to the *counter_state_machine_converter* design.

Projects – Preliminary

- Adaptive ML accelerators

Adaptive ML accelerators

- Why adaptation?
  - Varying energy constraints
  - Input variability
  - Translation to new platform/device
  - Transfer Learning

Adaptive ML accelerators

- Some directions
  - Emulate loss through drop out/connect
    - Techniques in ML literature equate these phenomena to forcing weights to zero
    - Loss in (because of) hardware may look very different
  - Continuous diagnostics
    - Evaluation of network certainty
    - Concept of surprise
  - Detection of temporal dominance of classes
    - Reconfigure optimized version for dominant class

Adaptive ML accelerators

- Some directions
  - Critical path based hardware-aware resource management
    - **Class-based CP:** using contributions of a neuron to a specific class
      - Mean Absolute Activation, first order Taylor Approximations
    - **Generalized CP:** relative participation of output channels at the routing of the output from a layer
  - Distribute resources (e.g., total number of bits allocated for weights) according to a criticality metric

Adaptive ML accelerators

- Some directions
  - Characterize circuits (e.g., SRAM) to create models
    - Associate voltage drop/power outage with cell decay
    - Create characteristic bit masks for weights
  - Neural network architecture search
    - Lessons learned from design space exploration in high-level synthesis
    - Search for a new cell from basic building blocks
      - Input: a set of convolutions and pooling of varying size
      - Think of it as your module library

Adaptive ML accelerators

- Some directions
  - Partition system into two part
    - Part1: Fixed ML architecture
    - Part2: ML architecture with capability to train within the edge device
  - Training consumes resources and energy on the limited edge device
    - Need to explore the trade-off carefully
    - Co-design ML architecture and training hardware

Future Directions

- Fluid implementations
  - Quickly re-targetable from software to hardware
- Scientific domains offer a vast space of computational challenges
  - They need interpretable systems
  - Laws of nature apparent in the system’s output
- Multiple paths need to converge
  - Photonic circuits – not much automated
  - FPGA
  - Neuromorphic – not much automated
  - Quantum

Collaborators

- Northwestern University
  - Han Liu (CS), Kristian Hahn (Physics)
  - Manuel Blanco Valentin, Rui Shi, Deniz Ulusel, Bincong Ye, Yingyi Luo (now at Google), Sid Joshi (now at Intel)
- Fermi National Laboratories
  - Nhan Tran, Farah Fahim, Christian Herwig, Kiyomi Seiya, et al.
- Columbia University
  - Giuseppe di Guglielmo
- Lehigh University
  - Josh Agar

THANK YOU!