Implementation of a High-Performance Pattern Recognition Associative Memory in an FPGA

J. Olsen[1], J. Hoff[1], Z. Hu[1], S. Jindariani[1], T. Liu[1], J. Wu[1], Z. Xu[2]

[1] *Fermi National Accelerator Laboratory, Batavia, Illinois, U.S.A*

[2] *SLAC National Accelerator Laboratory, Menlo Park, California USA*

Outline

- RAM and CAM
- Pattern Recognition Associative Memory (PRAM)
- Using PRAM
- PRAM implementation in an FPGA
- Performance and resource utilization
- Pattern density optimization
- Demonstration hardware
- Conclusion

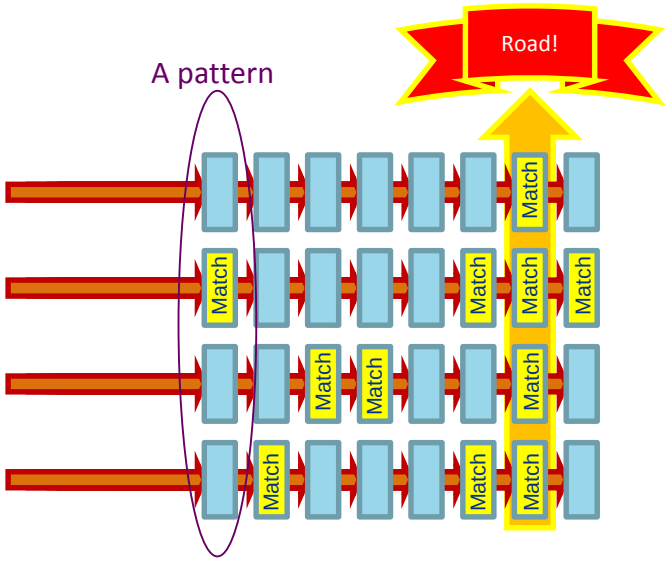
Intro: RAM vs CAM

- Random Access Memory
  - Address in, data out
  - Simple architecture: registers + output mux

- Content Addressable Memory (CAM)
  - Inverse of RAM: data in, address out
  - High performance, parallel lookup operation
  - Complex + resource intensive
  - Width and depth of memory array
  - Matching options complicate design
  - Output options complicate design

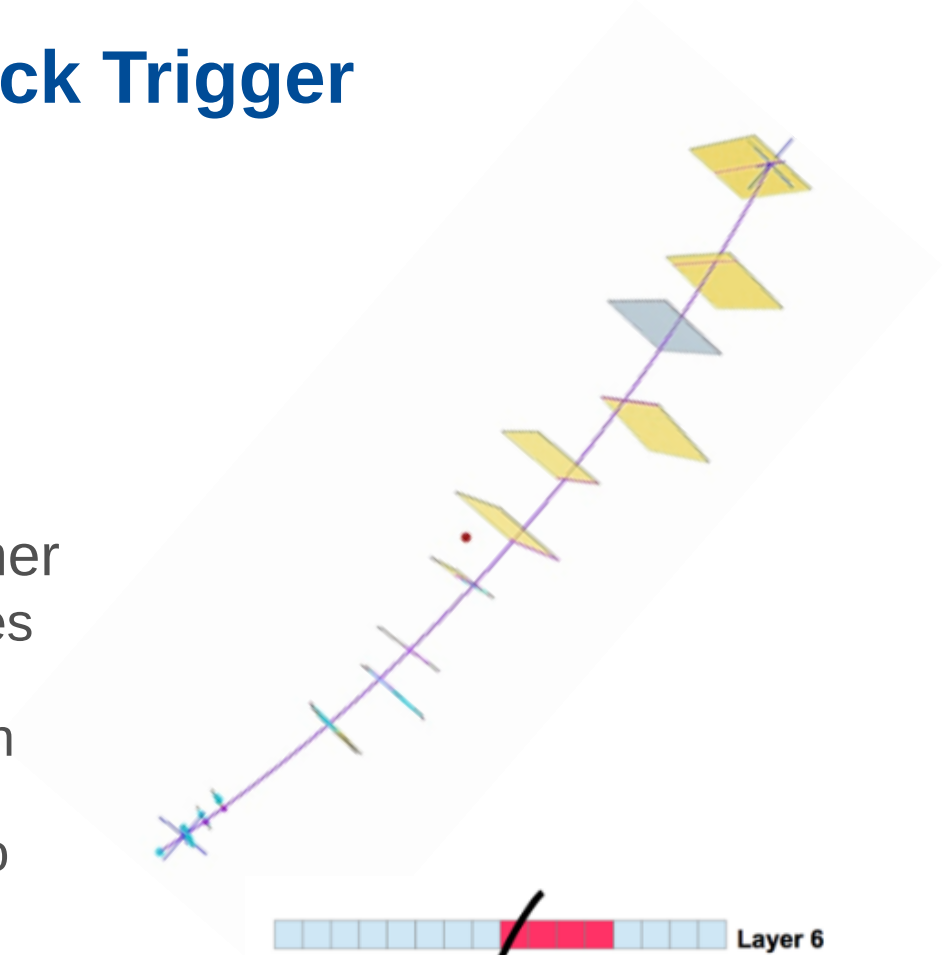
Intro: PRAM

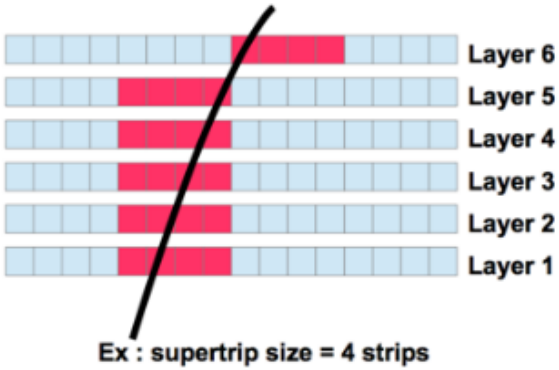
- Described as a “CAM of CAMs”
- Match on a multi-word pattern
  - Multiple input buses (layers)
  - Majority logic (perfect match, missing 1, missing 2, etc.)
- Words may include ternary ‘X’ bits
- Output the address of the matching pattern (road)
- PRAM is well suited for high bandwidth, low latency pattern recognition systems
  - Brute force parallel
  - No time for CPU iteration



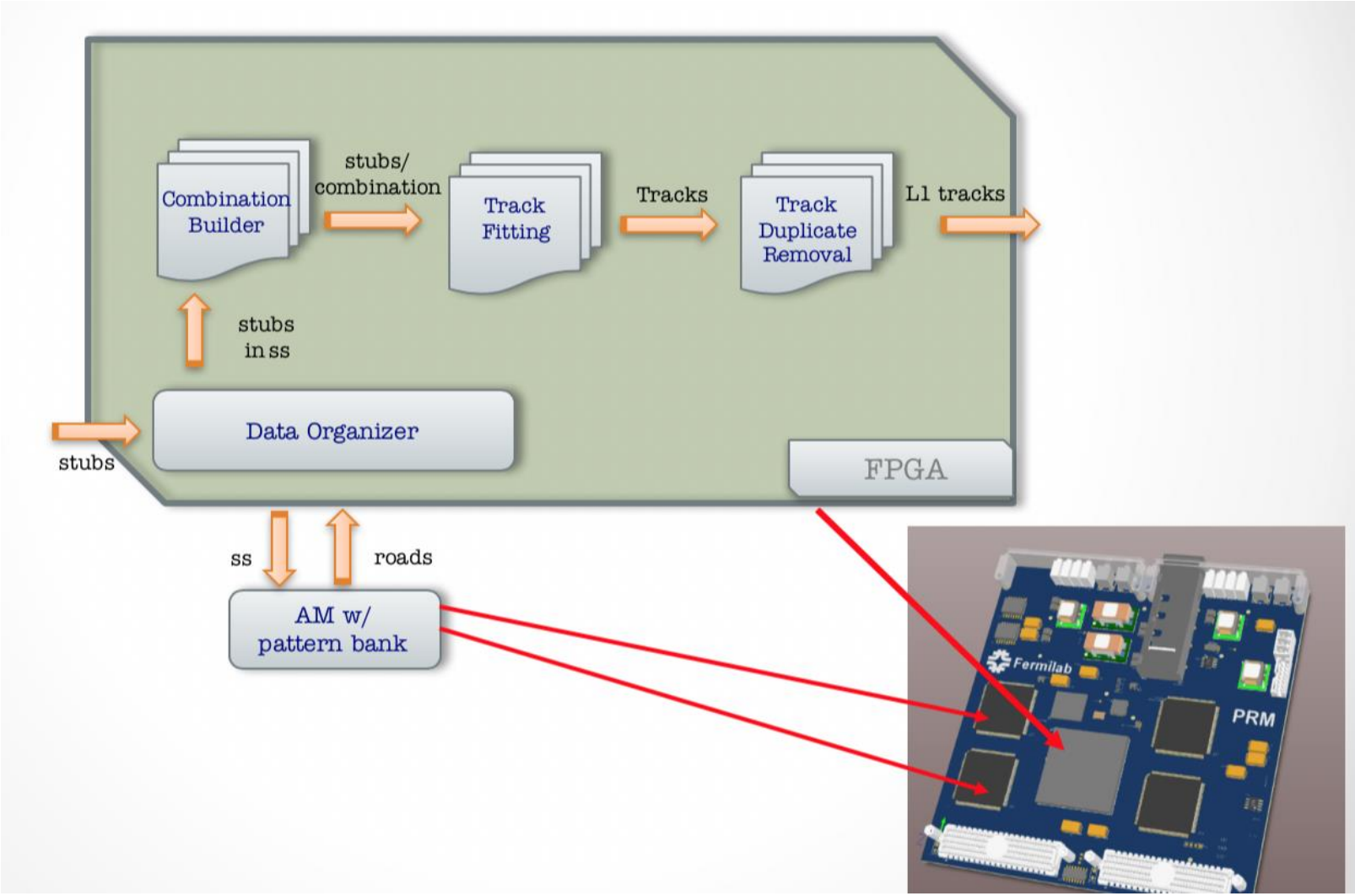
Example: L1 Silicon Track Trigger

- Silicon Tracker
  - ~10k modules
  - ~10M pixels/strips
  - ~100 Tbps data rate
  - 40MHz event rate
  - ~4us latency
- PRAM and FPGA work together
  - Modules report hit coordinates (stubs)
  - Store the full resolution hits in FPGA RAM (data organizer)
  - Form superstrips and send to PRAM
  - PRAM finds the coarse roads
  - Roads are used to quickly recall full resolution hits (from data organizer)
  - Full resolution hits used by the track fitter logic
  - Output track parameters

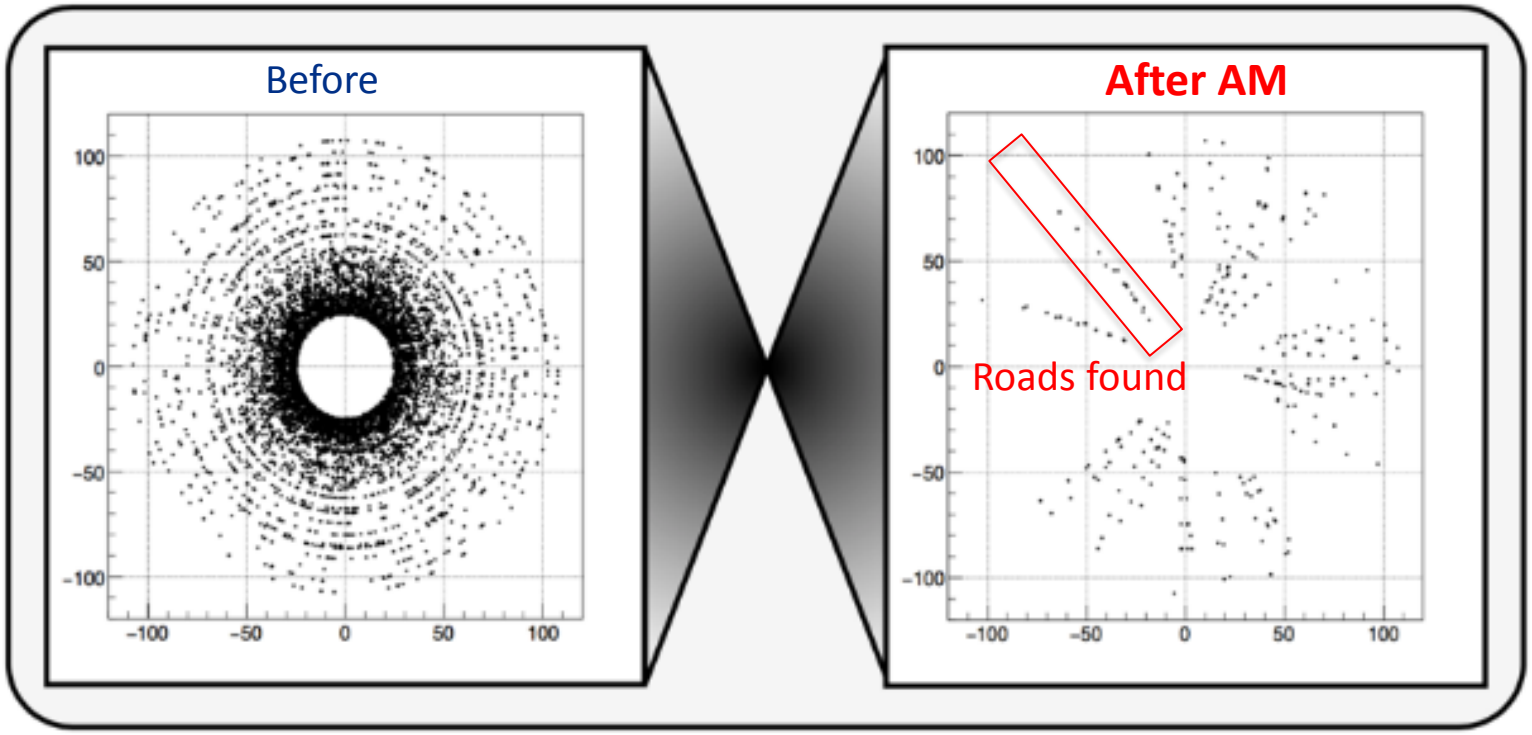




L1 Silicon Track Trigger



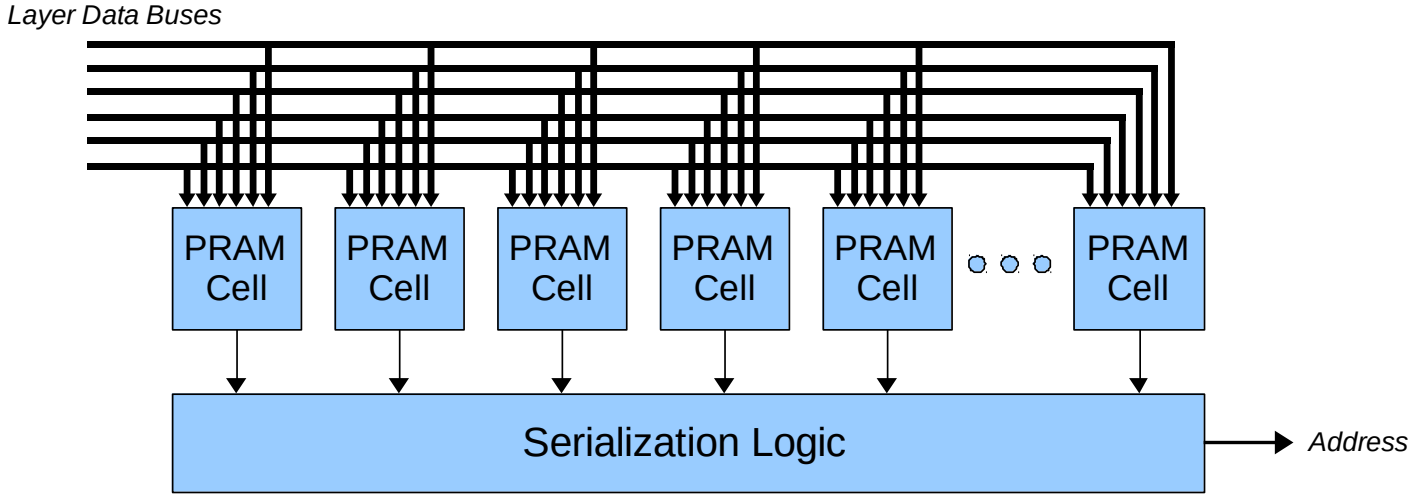
L1 Silicon Track Trigger



PRAM Implementations

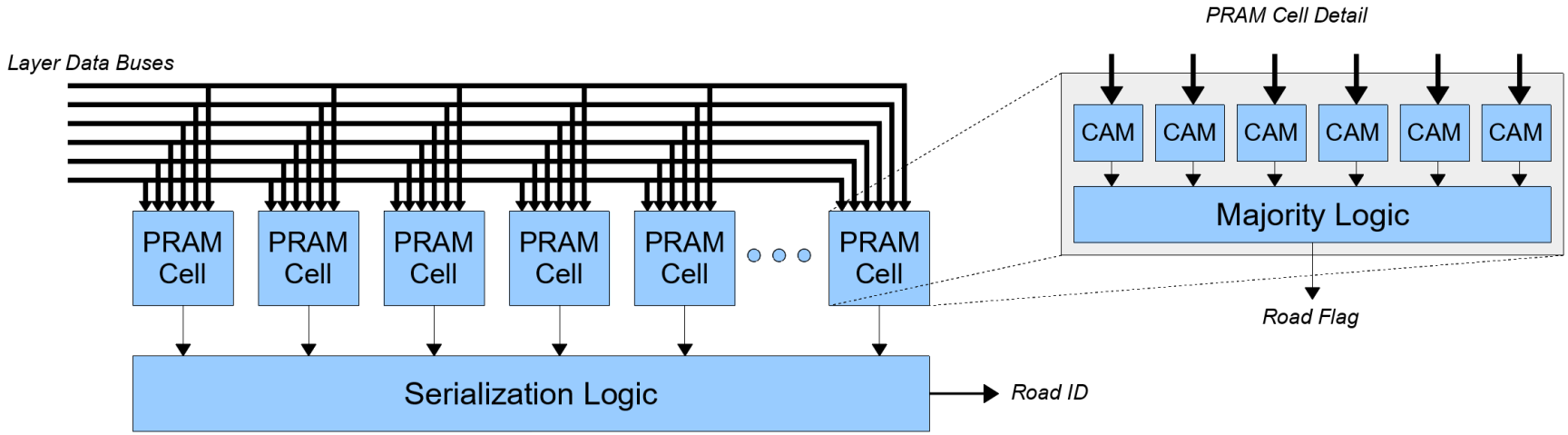
- ASIC
  - High pattern density, high performance
  - Predictable timing allows for some clever asynchronous logic structures
  - Expensive and time consuming development cycles
  - Difficult to change the system interface
    - High speed serial transceivers, etc.
- FPGA
  - Pattern density and performance challenges
  - Super flexible; quick development cycles
  - Design, test, evaluate new PRAM-FPGA interfaces, priority encoder logic, sorting, etc.

FPGA PRAM Architecture



- Functionally matches our ASIC implementation
- Fully pipelined operation
  - PRAM cells process the current event while previous event is read out
- Front End: system interface + array of PRAM cells
- Back End: capture road flags and serialize for readout

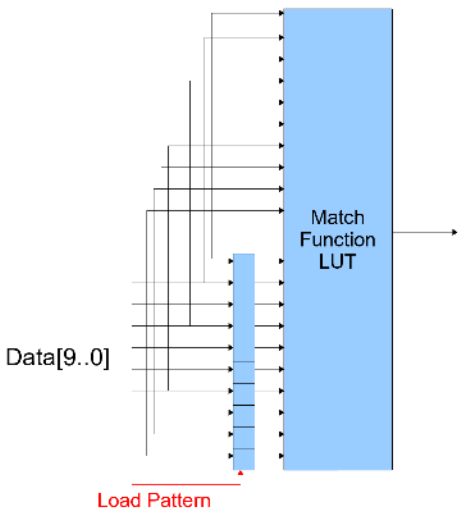
PRAM Cell

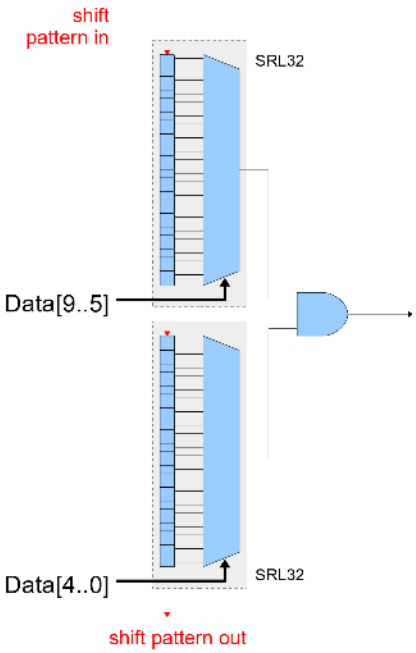


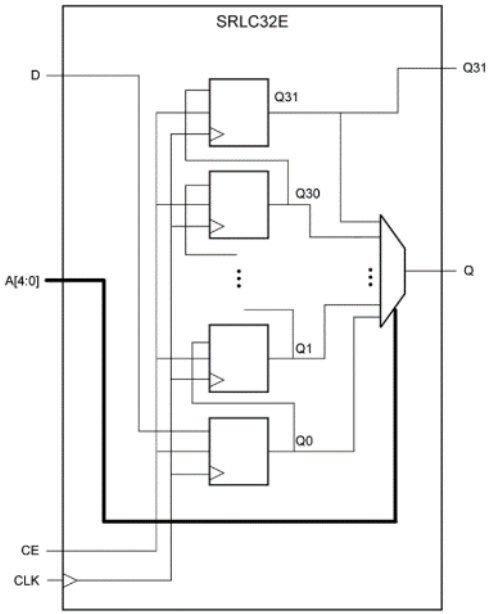
- PRAM cell = CAMs + majority logic
- CAM is 8~15 bits wide, 1 word deep
- CAM match output latches until reset by next event
- May support ternary logic (‘X’) in certain bit positions
- Majority logic (6/6, 5/6, 4/6) set globally

CAM Cell

- The key is to make it COMPACT
- Xilinx SRLC32E primitive in 7-series and Ultrascale architectures (SLICEM)
  - Intended for variable length shift registers
- Load pattern bits post-configuration





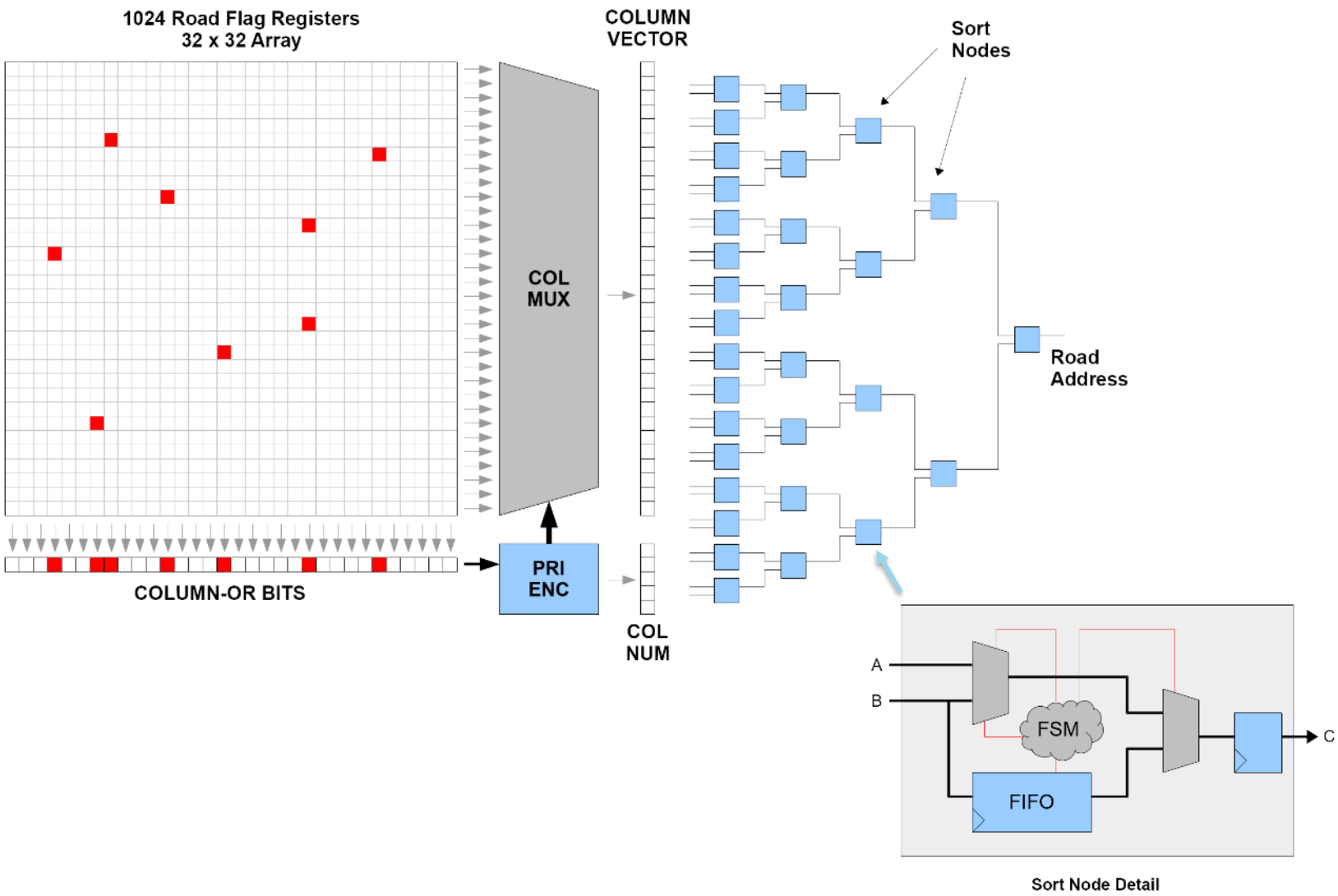


No extra logic required to support 'X' bits in pattern

Backend Serialization

- LARGE array of road flags, sparsely populated
- Challenge is to determine the address of each road in the array
  - High speed readout
  - Low latency
  - Zero suppressed
- Pipelined operation
  - Serialization logic copies road flag bits at end of event
- Multiple levels of priority encoders are typically used
- ASIC and FPGA designs are cycle accurate, but internally very different
  - ASIC binary “Fischer Tree”
  - FPGA pipelined buffered merge…

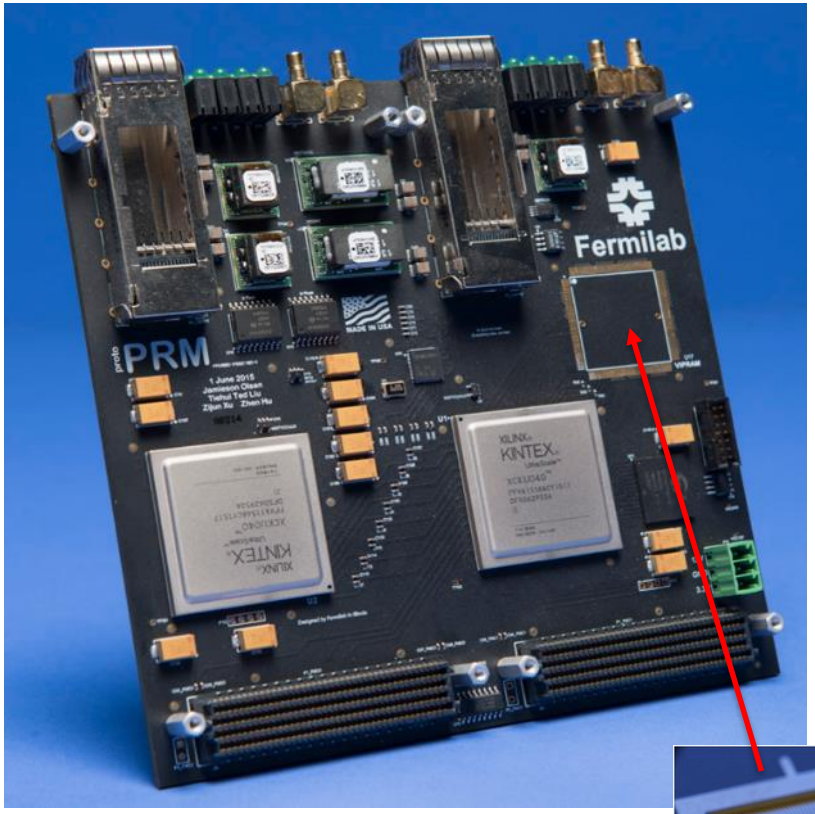
Backend Serialization

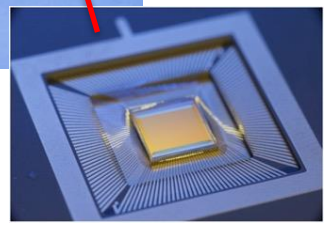


Implementation Summary

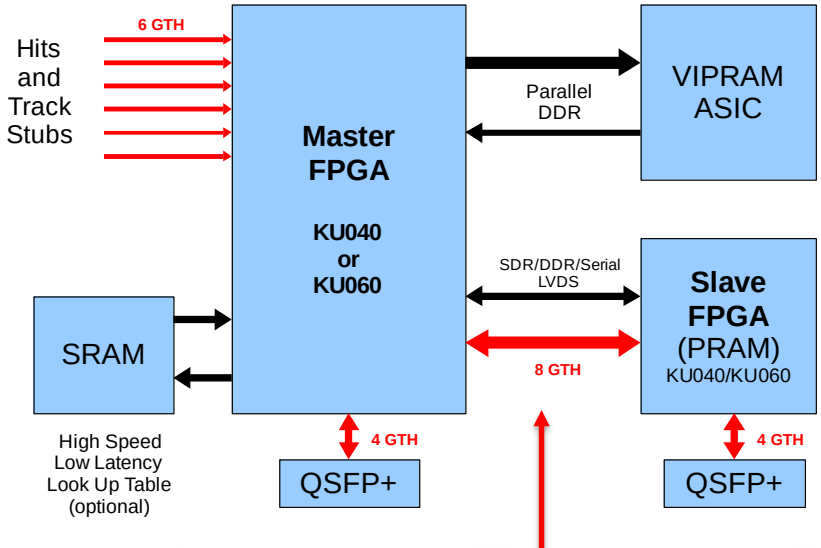
- 1k to 16k patterns tested
  - 6 layers
  - 9 or 10 bits per layer
  - Global majority logic controls (6/6, 5/6, 4/6)
- Pattern is loaded into PRAM cells serially
- Input and output buses are DDR parallel
- Meets timing with 240MHz clock
- CLB SLICEM resources limit PRAM size (KU060)
  - 1k patterns = 8%
  - 4k patterns = 33%
  - 8k patterns = 65%
- 4k~8k patterns is still useful for proof of concept demo
  - Carefully choose which patterns are loaded

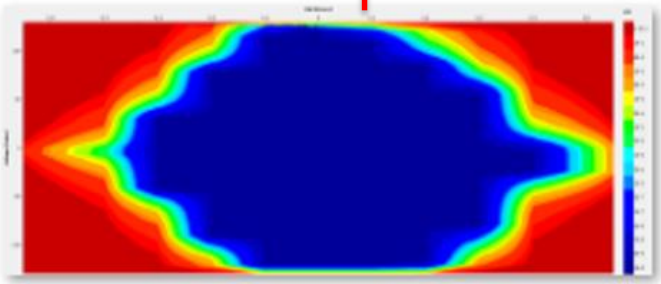
FMC Mezzanine Card





Socket for the 3D vertically integrated VI-PRAM ASIC



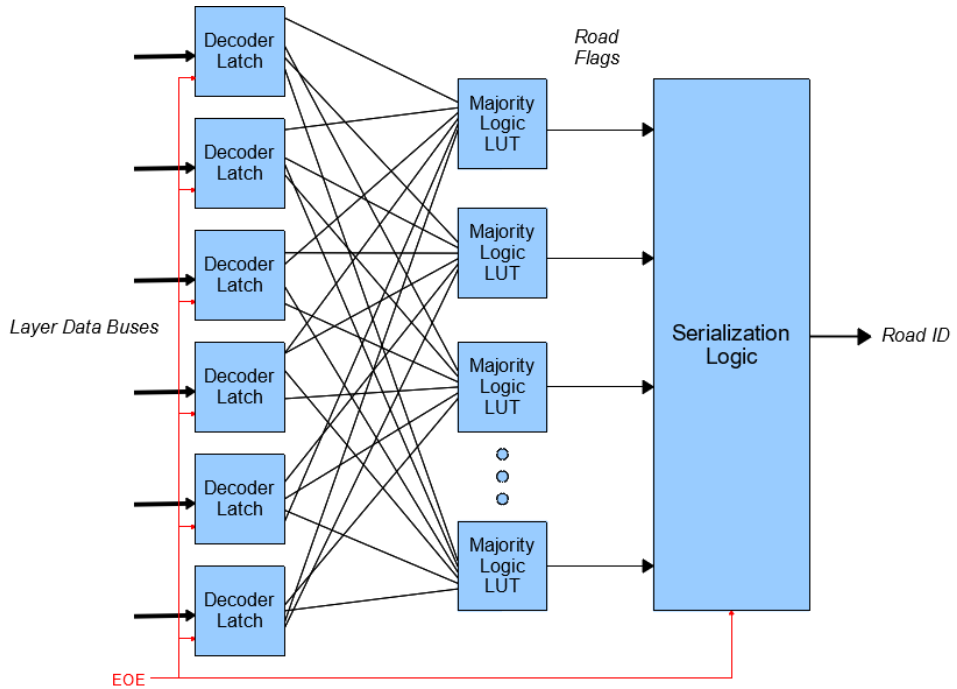


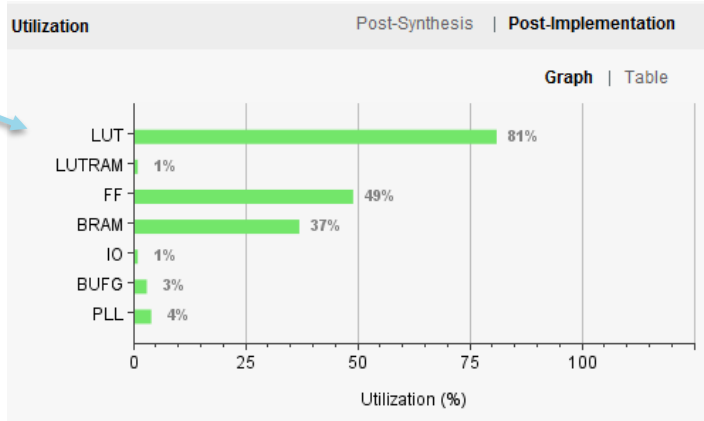
GTH Transceiver local bus @ 16Gbps

The double-wide FMC mezzanine supports both ASIC and FPGA PRAMs for direct side by side evaluation in hardware.

Increasing Pattern Density

- Hard code patterns in HDL
- Front end latching superstrip ID decoder
- Pattern stored in the internal routing
- Ternary bits are not free
- Serialization logic dominates resource usage
- Designs with 128k random patterns have been synthesized and fit into KU060 FPGA
- Serialization logic optimization may allow for up to 256k patterns in KU060 (or larger device)





Pulsar3a

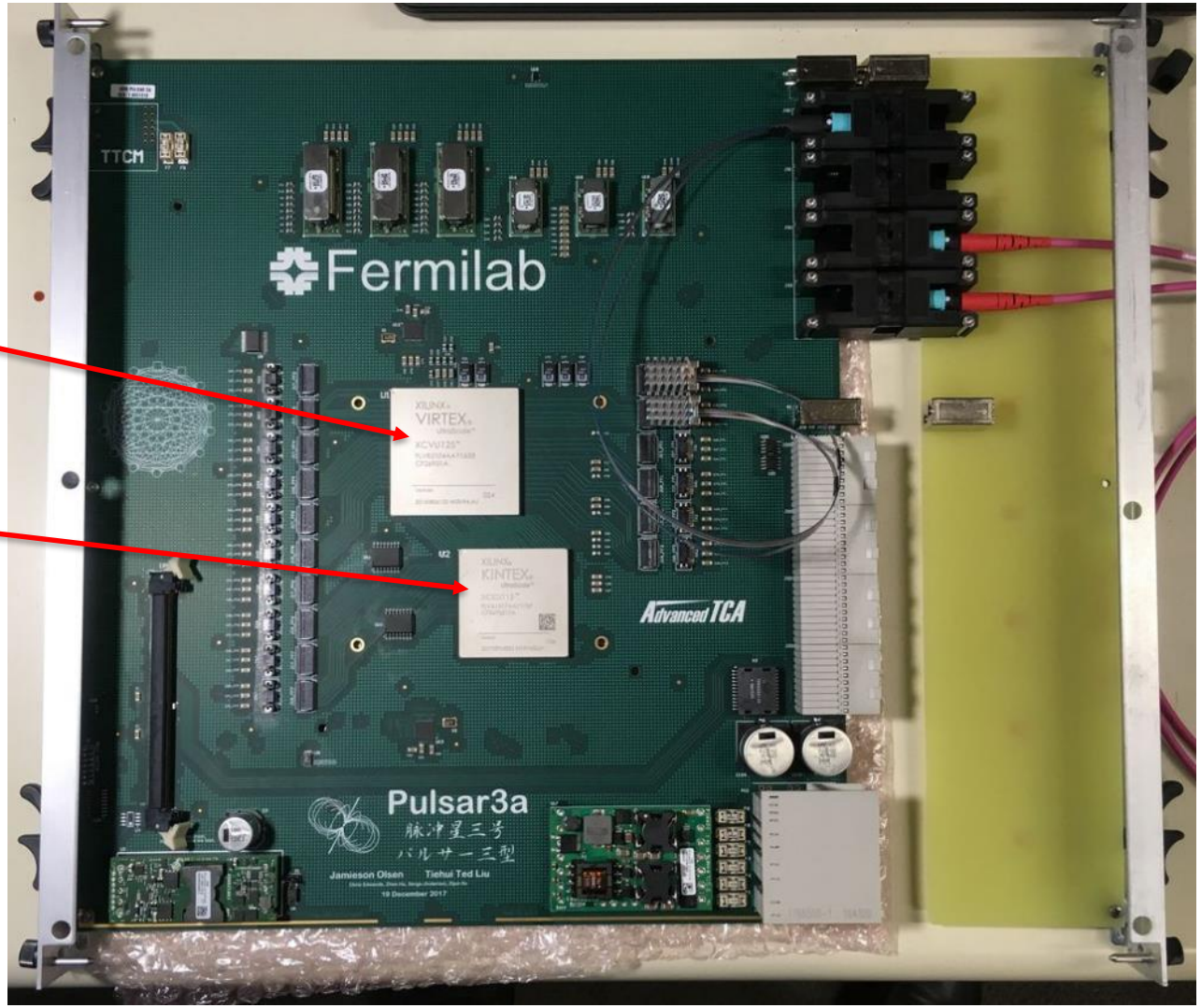
First board arrived May 2018!

Master FPGA (VU125)
Routing event data and track fitting

Slave FPGA (KU115)
PRAM!

For more information about Pulsar boards coming up!

Vertical Slice Tracking Trigger Demo System see Zhen's talk from Monday.



Conclusion

- PRAM builds on CAM and is well suited for high bandwidth, low latency pattern recognition systems
- A usable PRAM can be realized in modern FPGAs
- FPGA flexibility allows us to try out new ideas
  - System interface
  - Pattern loading / storage
  - Backend sorting logic
- We have designed and successfully implemented a few types of PRAM
  - ~10k patterns loaded at runtime
  - ~100k patterns hardcoded
- Optimization studies continuing
- New hardware for testing

Thanks!