SAND2019-1840PE

DOD/DOE Joint Hardware Exploration & Modeling Work

PRESENTED BY

David Donofrio(LBNL) and Arun Rodrigues(SNL)

Project38: A Joint DoD/DoE Architecture / Application

- Project Organization
- Technologies
- Open Proxies
- Tools

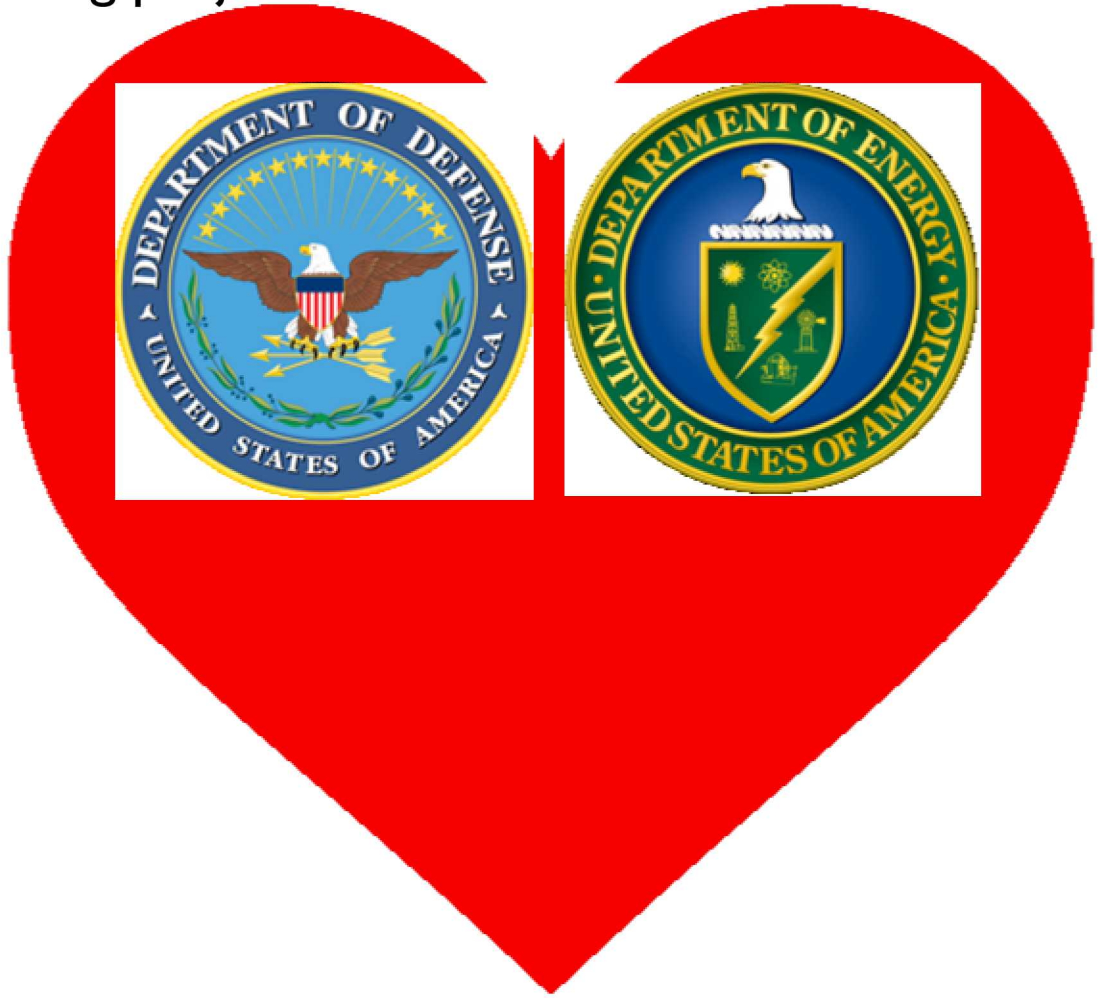
Project Organization

Project Goals

Project 38

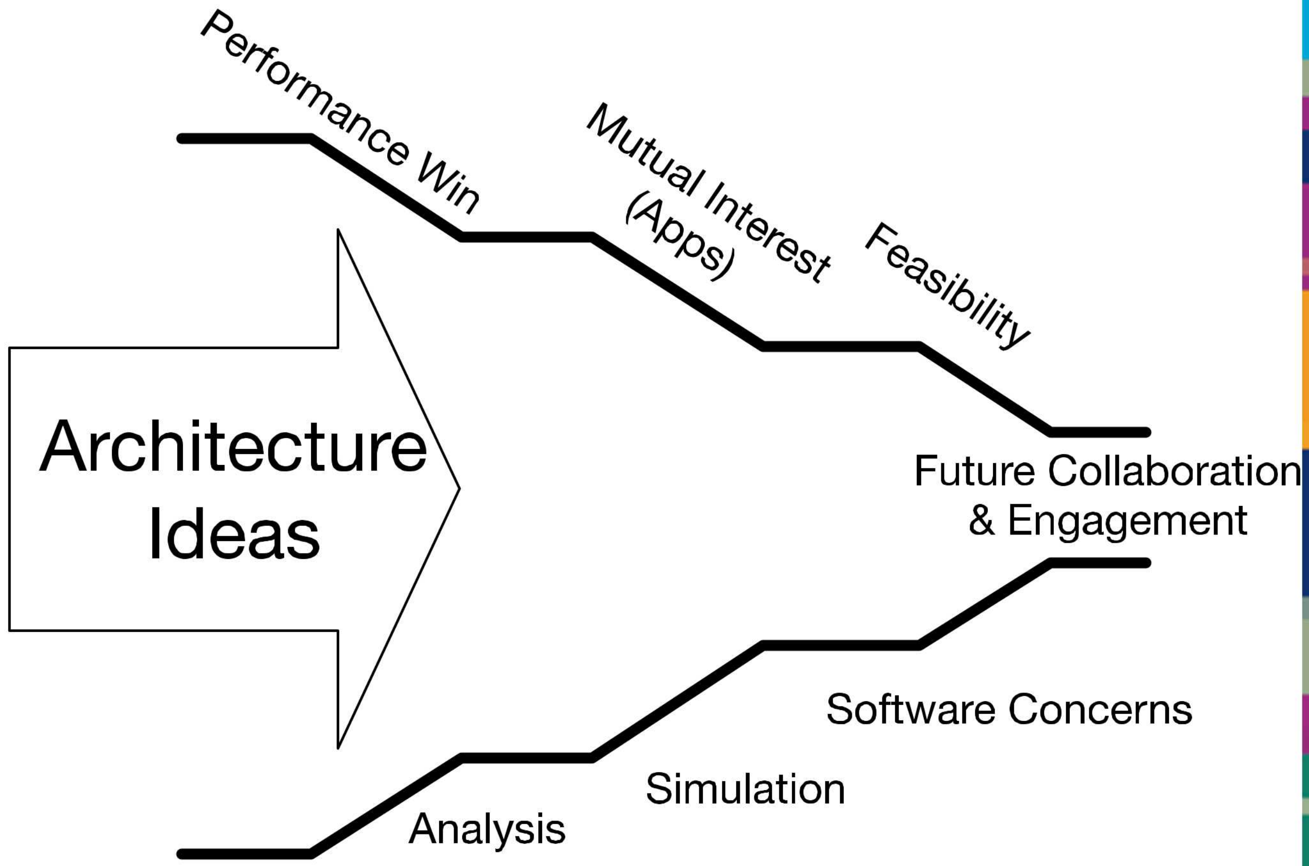
BERKELEY LAB

- Why?
  - Budgets always tight, need to share resources, understand each other's constraints
  - Discussions outside a specific procurement / existing project
  - Identify potentially useful architectures
  - Areas of interest / overlap
  - Guide future collaboration
  - Guide future interactions with vendors
- Tools focus
  - Harness different strengths & knowledge
  - Share what we know about our applications
  - Discover what we don't know



Process

- Initial Discussions (Sept 2017)
  - Brainstorm Architecture Ideas
  - Discuss points of overlap (procurement issues, need for specialization)
  - Discuss points of divergence (application longevity, software constraints)
- Summer 2018
  - Formalize 3 focus architectures
  - Define Applications scope
- Current
  - Study applications
  - Refine Architectural ideas
- Goal: Future collaboration



Key Institutions

BERKELEY LAB

Pacific Northwest NATIONAL LABORATORY

Sandia National Laboratories

Lawrence Livermore National Laboratory

BROOKHAVEN NATIONAL LABORATORY

Argonne NATIONAL LABORATORY

LPS The Laboratory for Physical Sciences

Architectural Concepts

Architectural Concepts

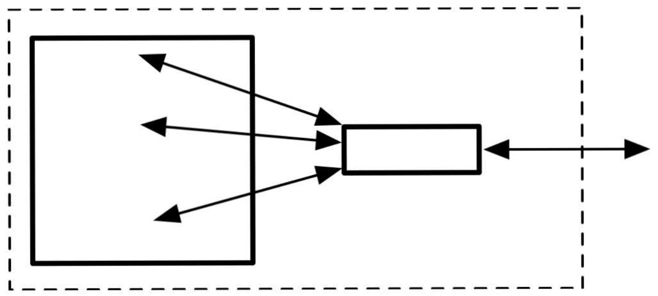
Project 38

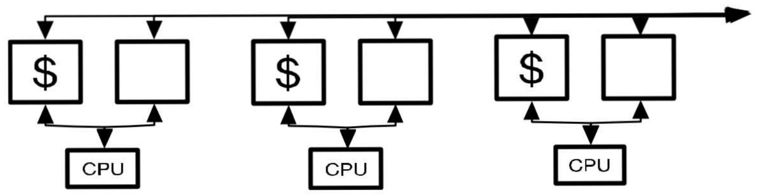
BERKELEY LAB

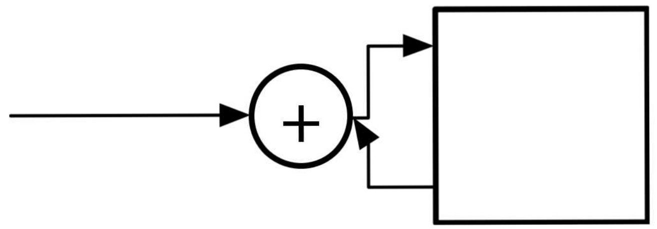
Concept	Suitable for Aggressive Vendor?	Suitable for us?
Inter-core Messaging	High (in SoC libraries)	???
Word Addressable $/Memories	Low?	
Pointer Math Unit	Medium	
Programmable Prefetch Units	Medium	
Multi-Level Memory	High	
Local Store	Medium	
Enhanced Memories	High	
Disaggregated Memory	High (on roadmap)	

Focus Architectures

- 2 Independent groups met, came up with same set of apps
- Scatter/Gather
- Word-Addressable Local Store
- Atomics



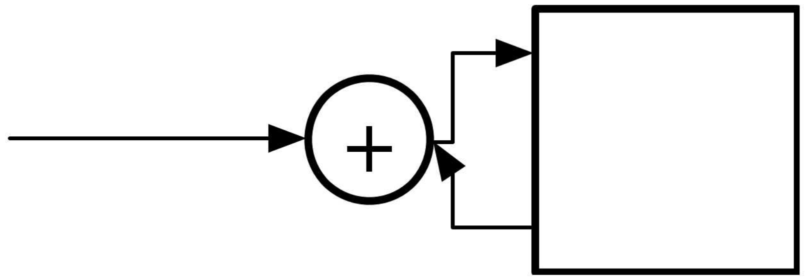




- TLB
  - not a topic for exploration, but a target for future cross-collaboration

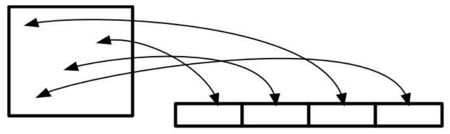
Atomics: Description

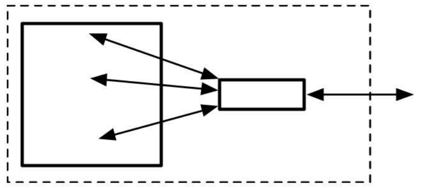
- "Better" Atomic operations, possibly occurring in the memory system or at multiple levels of cache
- Distinguishing Features
  - Fire & Forget
  - Both fetching atomics & 'one-way'
  - Floating point capable (especially atomic add)
- Existing work: IBM, Intel, EMU

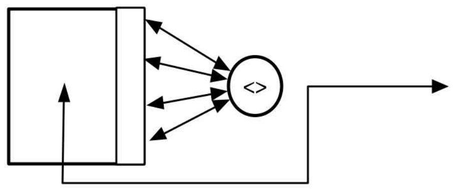


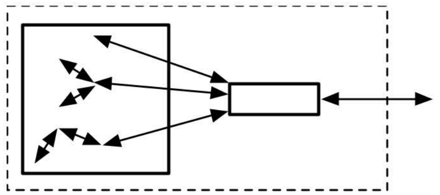
Scatter/Gather: Description

- Load multiple memory locations into a contiguous region.
- Distinguishing Features
  - Flexible
  - Pipelined
  - To a scratchpad
- Feature to Explore
  - S/g with atomic add
  - In Memory
  - More Programmable (Key/Value Lookup, pointer chasing)
  - S/g to a register (e.g.VGATHERDPD)



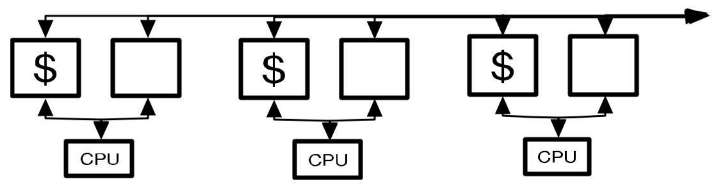


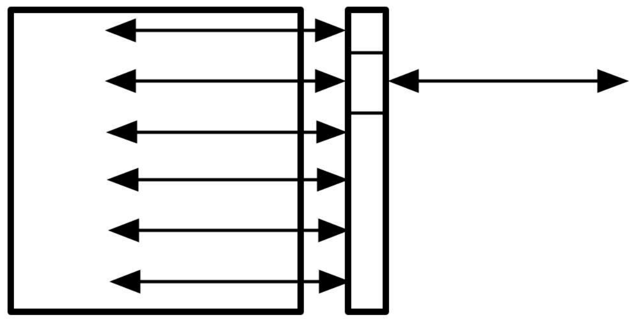




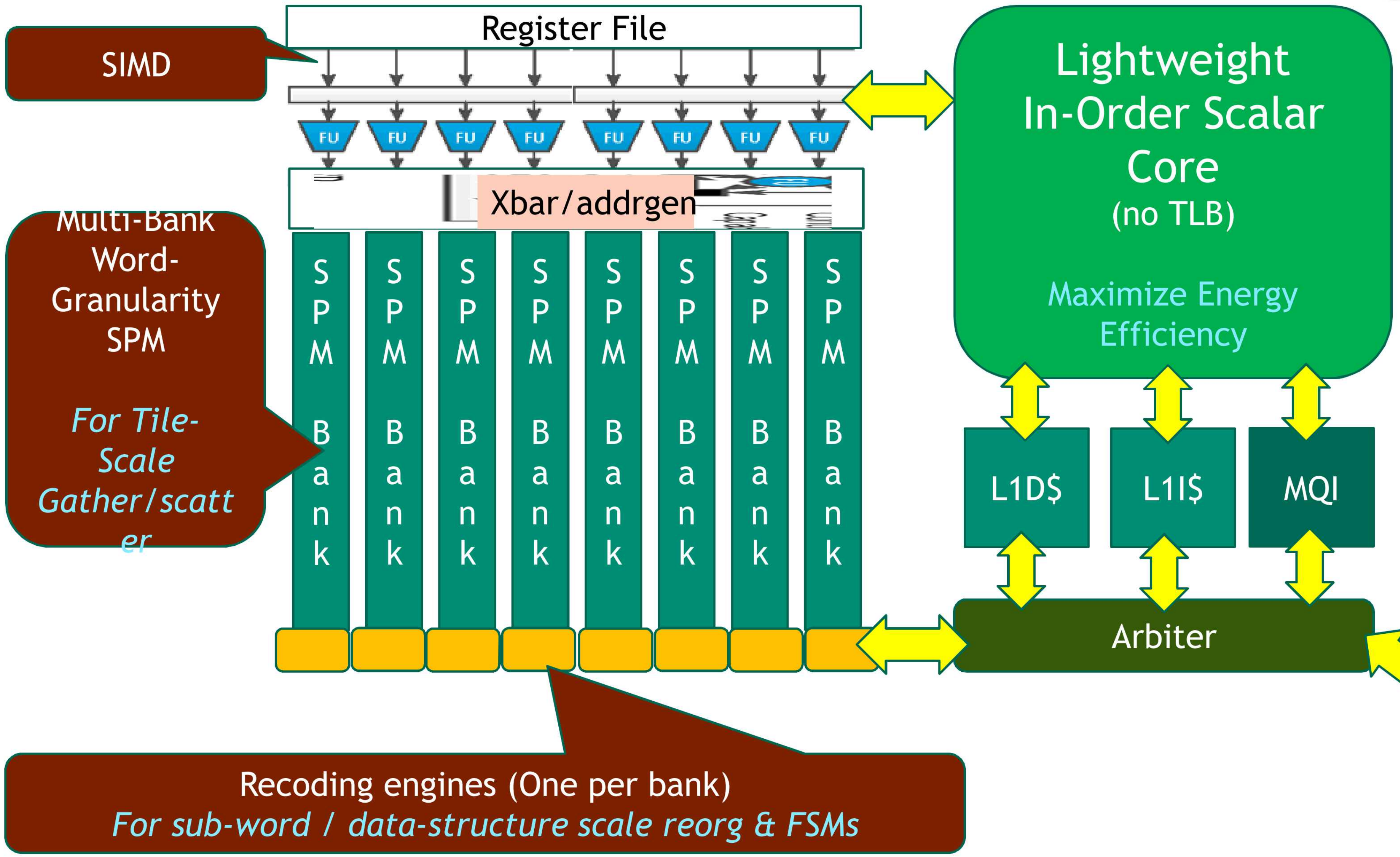
Local Store: Description

- Distinguishing Features
  - Software controlled
  - Word Addressable
  - Located at L1 or L2
  - Communication & Protection options (inclusive, exclusive, partitioned)
- Existing: CELL work, ECP, KNL





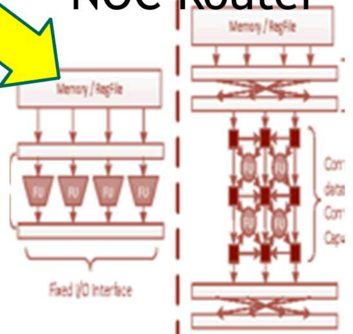
Conceptual Design (programmable accelerator tile)


Register File
SIMD
FU
Xbar/addrgen
Multi-Bank Word-Granularity SPM
For Tile-Scale Gather/scatter
SPM Bank
Lightweight In-Order Scalar Core
(no TLB)
Maximize Energy Efficiency
L1D$
L1I$
MQI
Arbiter
Recoding engines (One per bank)
For sub-word / data-structure scale reorg & FSMs

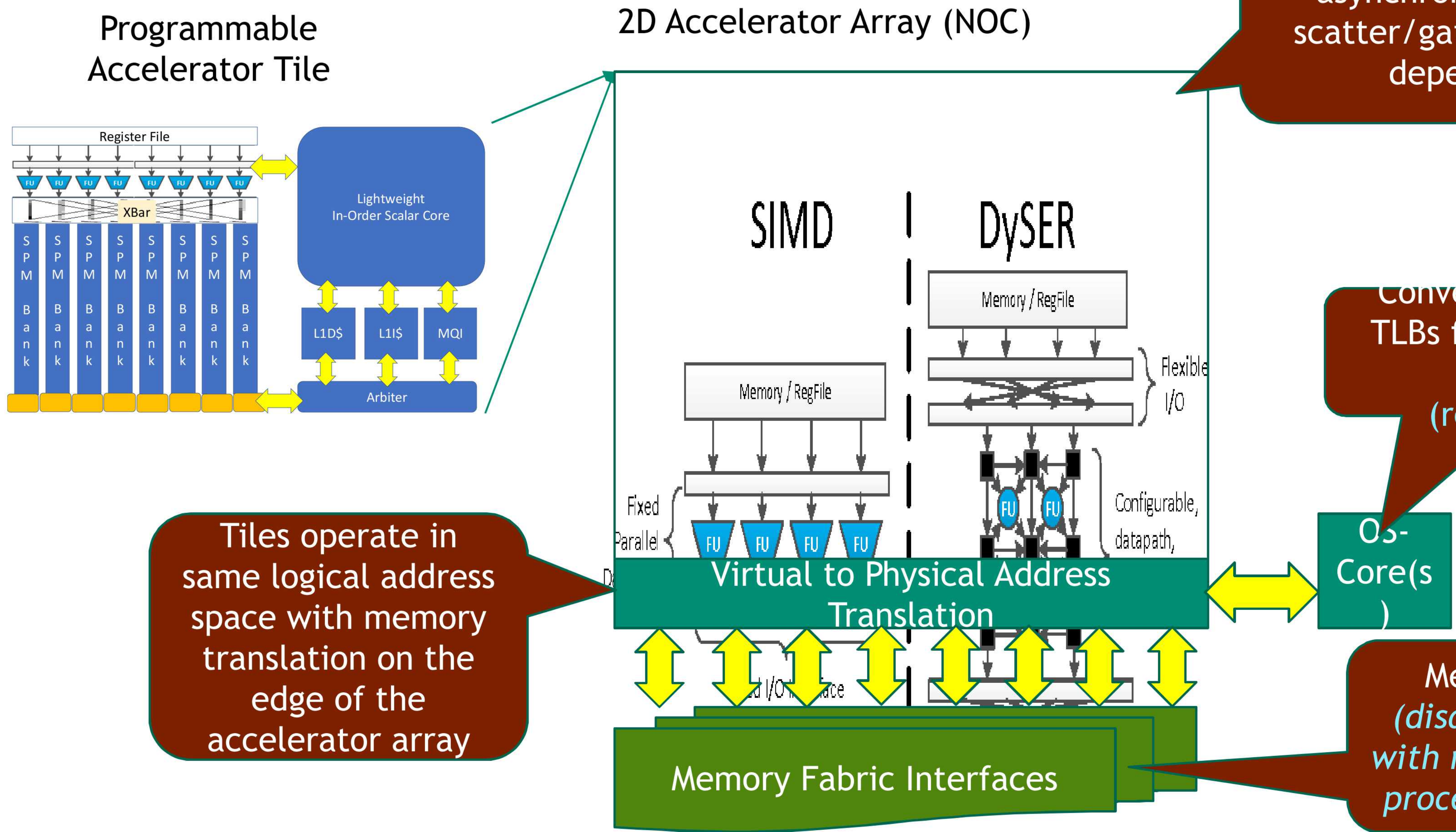
ROM Estimates
28nm@1GHz
1TF/100mm^2
70W

7nm@1GHz
17TF/100mm^2
100W

NOC Router



Conceptual Design (Chip Scale)

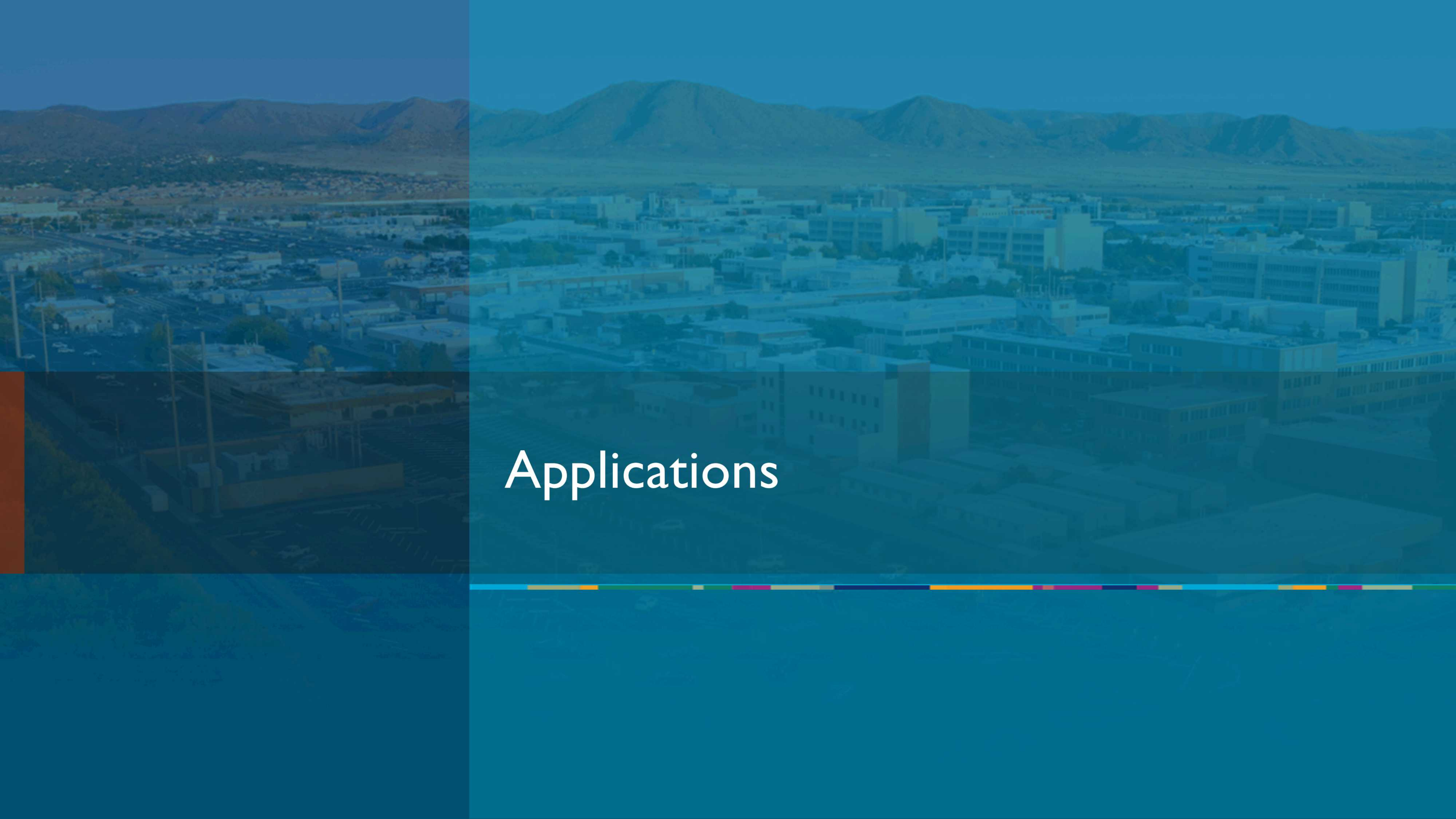


Atomic Message Queues for asynchronous chip-scale scatter/gather & enforcing dependencies.

Conventional Core with TLBs for OS, Drivers, Sys Services (reverse offload)

Tiles operate in same logical address space with memory translation on the edge of the accelerator array

Memory Fabric *(disaggregated NIC with memory & other processors as peers)*



Applications

Application Motivation

- Important applications
- Broad set
- Open
  - "digestible" - not 500,000 FORTRAN
  - Unclassified / No Export Control issues
  - Proxies for larger applications
- Common ground for conversation
  - "digestible"
  - Flexible – can be rewritten
  - Well understood

Applications Table

Proxy/Benchmark	Pattern	Hardware Feature
HPGMG (Transport_SE/HOMME microbench)	Stencil	SPM, queues
KRIPKE	Wavefront	Word Granularity SPM, queues
MerBench	Hash/Scatter	Remote atomics
Tensor?	Scatter/Gather	?
XS Bench / RS Bench	Gather/Table-lookup	Word granularity SPM + queues
Sparse Trisolve (SPMV)	Sparse Matrix	Queues + word granularity SPM, and Recoding engine (mat coding)
FFT	FFT	Word granularity SPM + queues
? Contact algorithms ?	Sort/Search	Recoding Engine
PIC	PIC	Atomics, queues

- + Graph analytics (TBD)

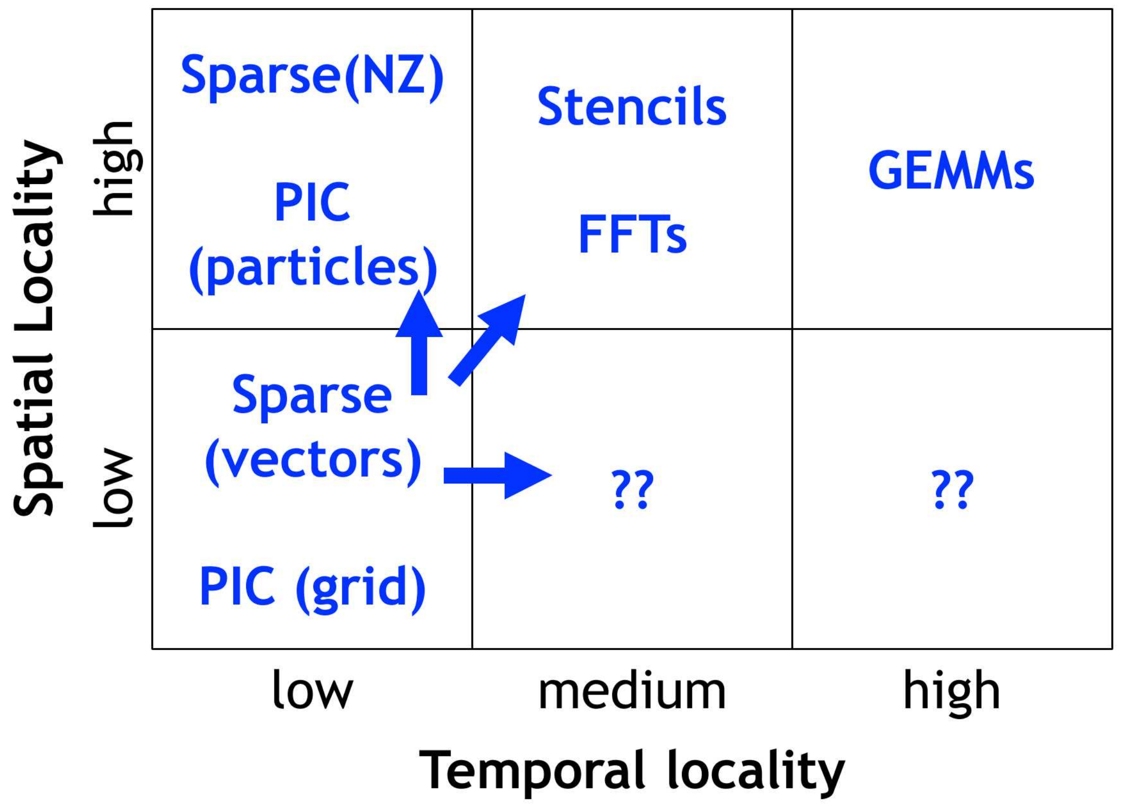
Diversity of temporal behavior

Sam Williams

Project 38

- SW Categorization: Locality manifests at different scales
  - You might not exploit locality within an L1, but could within an L3
- Orthogonal to these axes is whether it is discovered at compile time or run time.

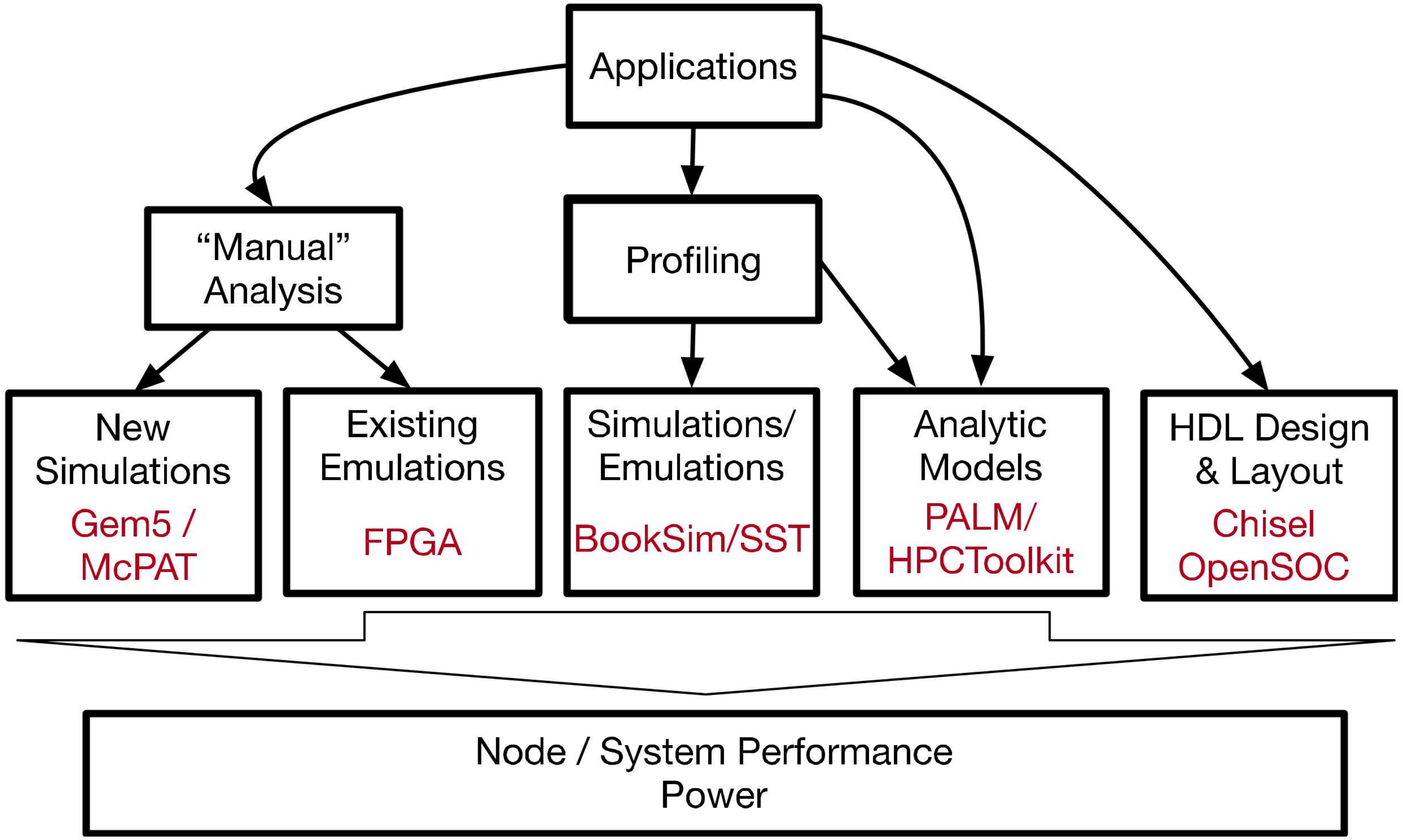
Software Characterization



Modeling & Simulation Tools

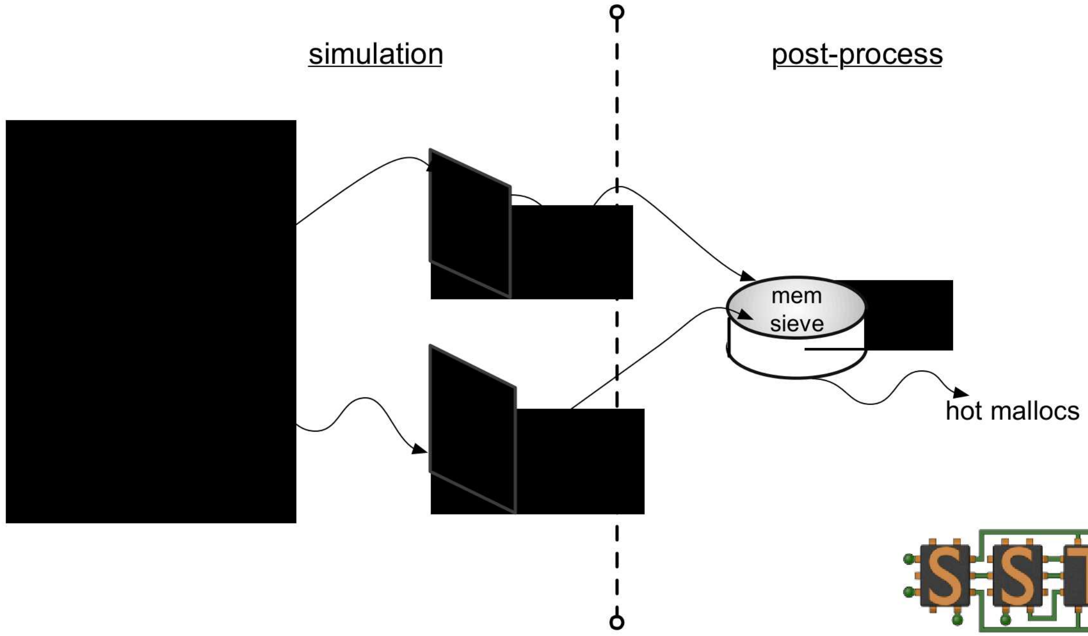
Tools

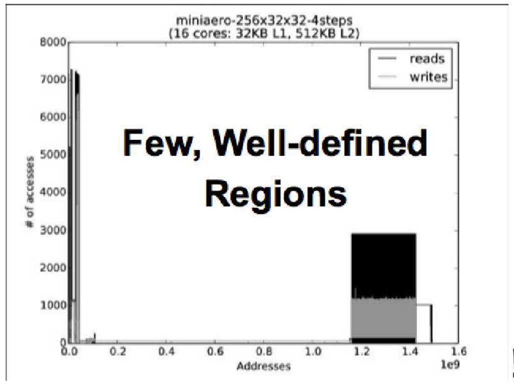
- Diverse set of tools & techniques
- "Manual" analysis & code modification
  - Leverage application knowledge
  - Explore SW impact
- Profiling
  - Understand our applications
  - What do what think we know, but really not know
- Simulation
  - Design space exploration
- HDL Design & Layout
  - Detailed feasibility studies
- Examples…



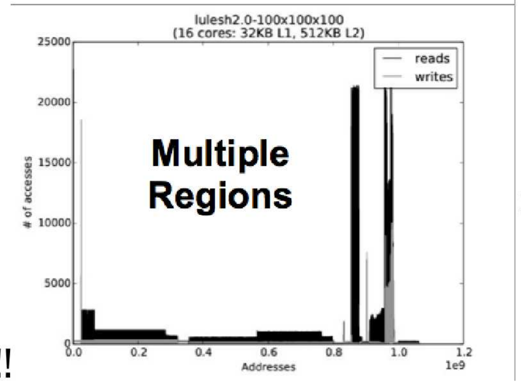
MemSieve

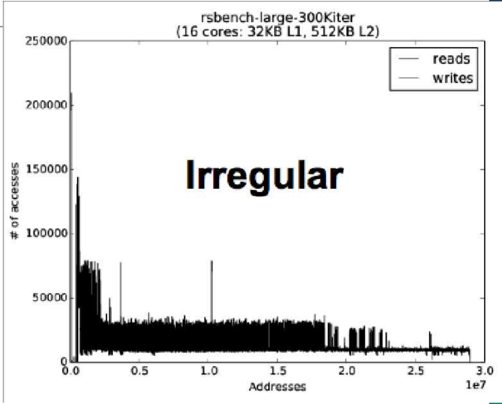
- SST-based tool
- Process:
  1. Trace application (PINTool)
  2. Detect malloc() calls
  3. Gather post-cache memory accesses
  - Associate with malloc()s
  - Build page-level histogram
- Output
  - "Hot" Malloc()s
  - Page Histograms
- Use
  - Identify regions for local store?
  - Size local store?
  - Estimate software effort?





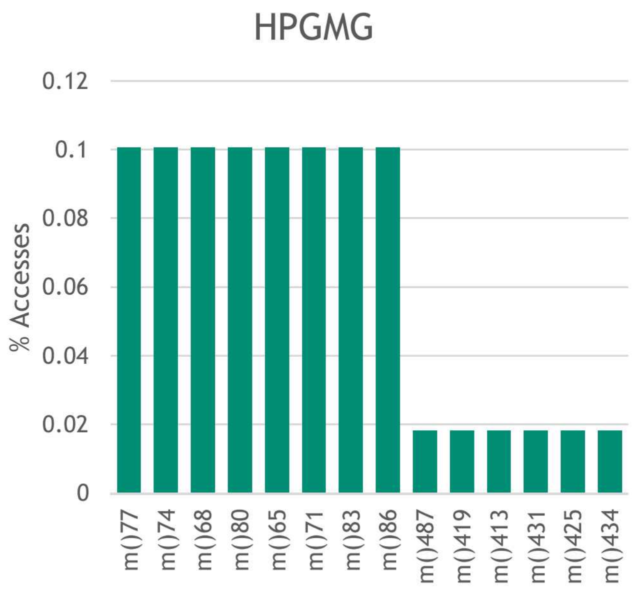
!!

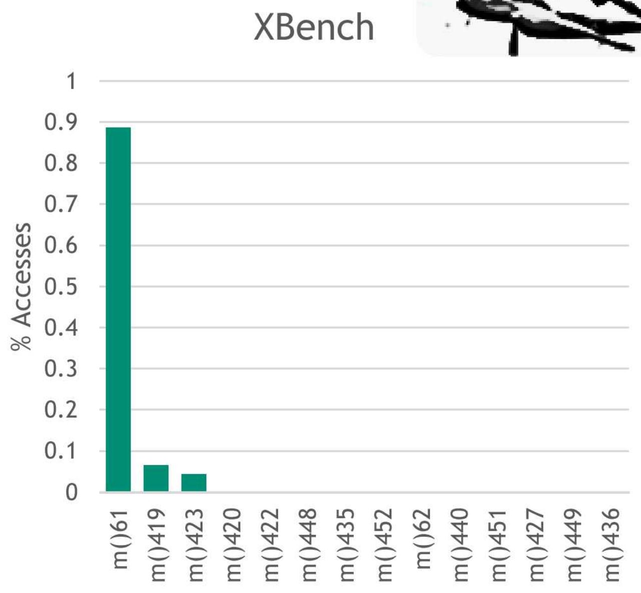


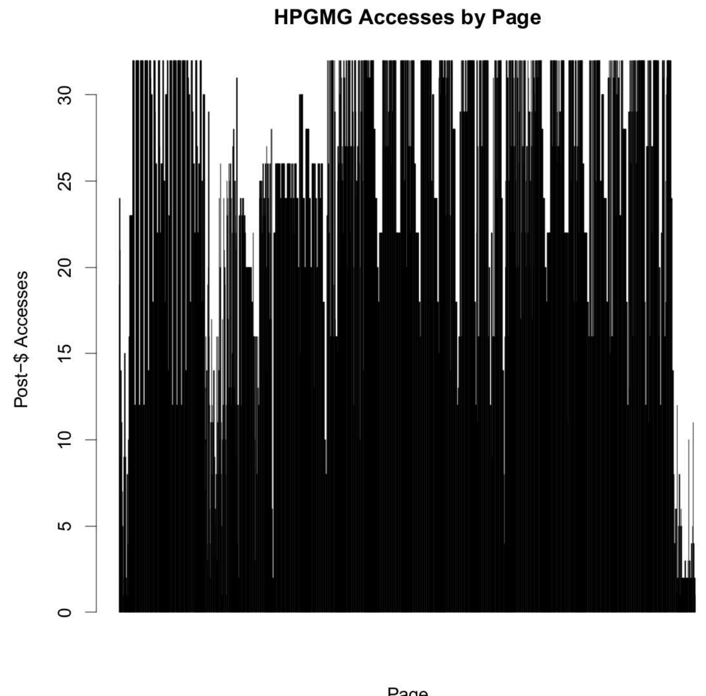


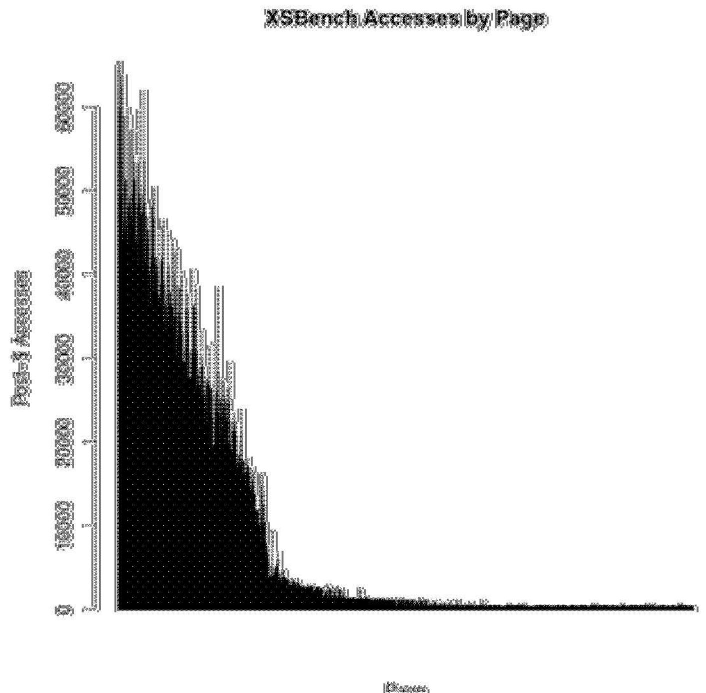
Malloc()s

- Assume 8MB Cache, 16 thr
- Mallocs
  - Captured stack traces to identify malloc()s
  - Can weight by r/w, size, accesses/size
- Histograms
  - Post-cache accesses per page
- Substantial diversity
  - HPGMG: more uniform
  - XSBench: varies considerably



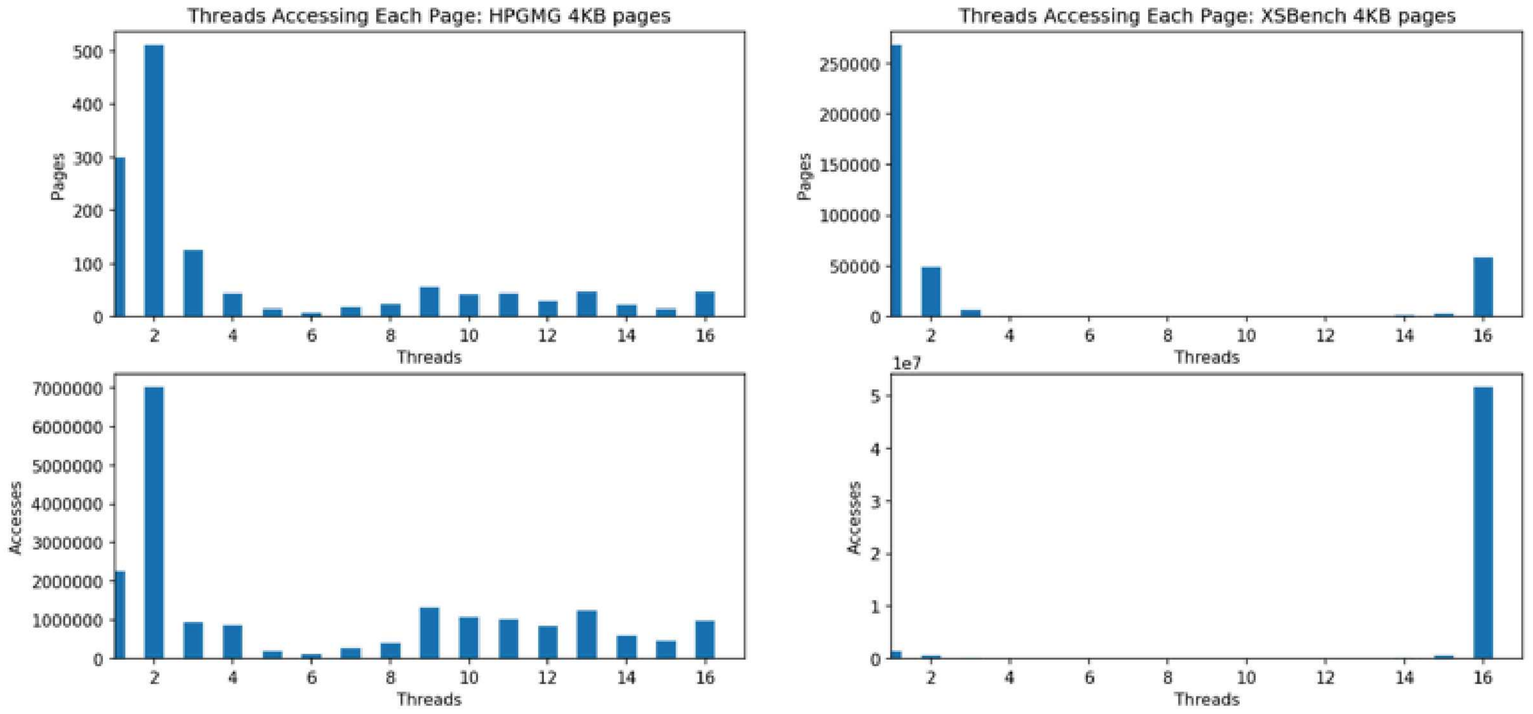






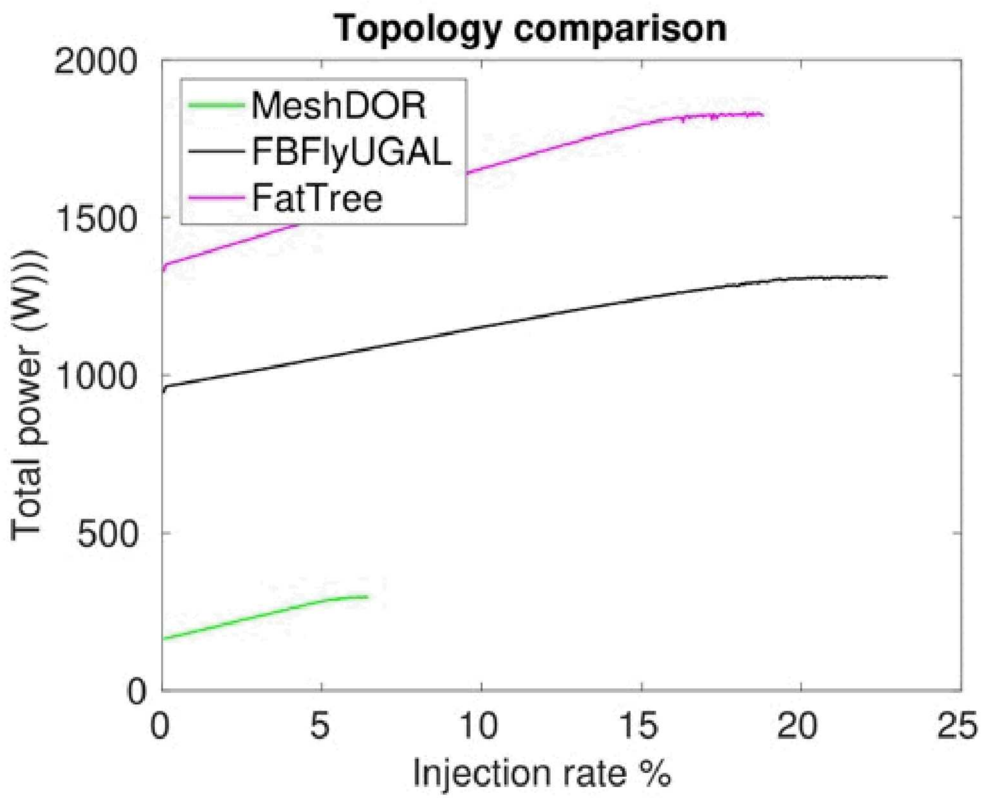
Thread Accesses

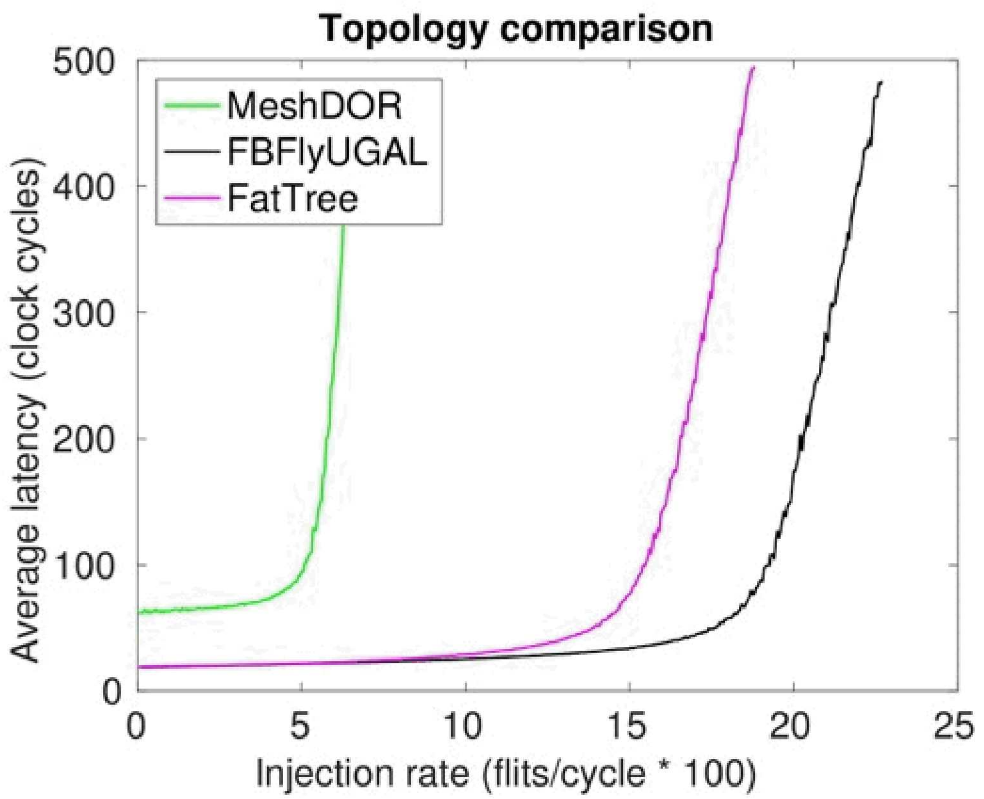
- Threads accessing each page
- HPGMG: generally few threads/page
- XSB: most pages are not accessed by many threads
- But, the pages which are accessed by lots of threads account for most accesses
- Implications for coherency?



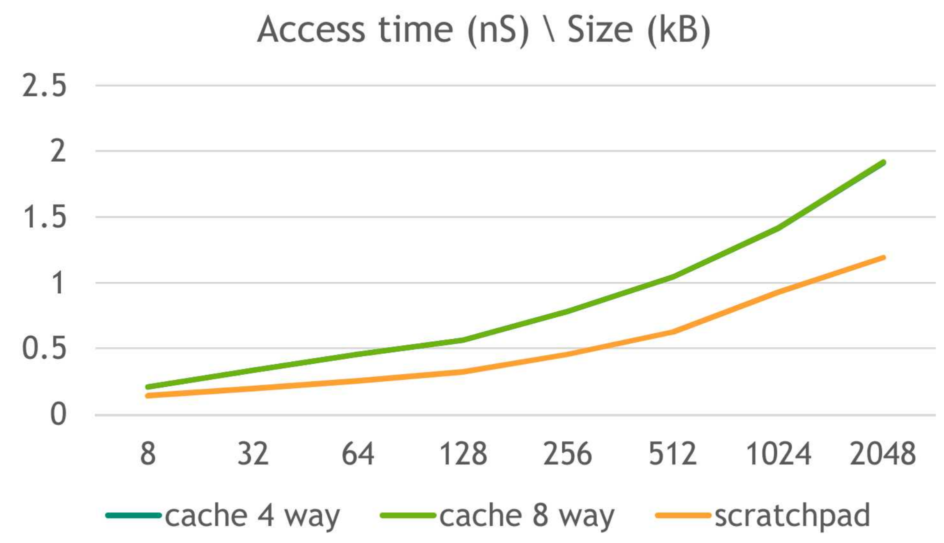
On-chip Network Topology Eval *(PIC Particle Sort Throughput)*

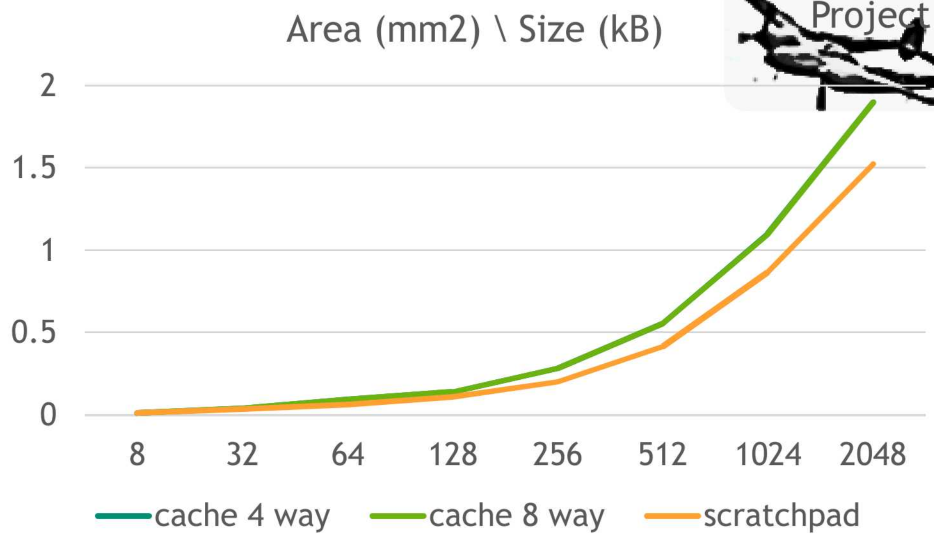
- Choosing a topology requires careful balancing of performance and power, using application-specific communication patterns
  - Lowest performing solution can be cheaper
- We can evaluate the architecture at cycle-accurate level

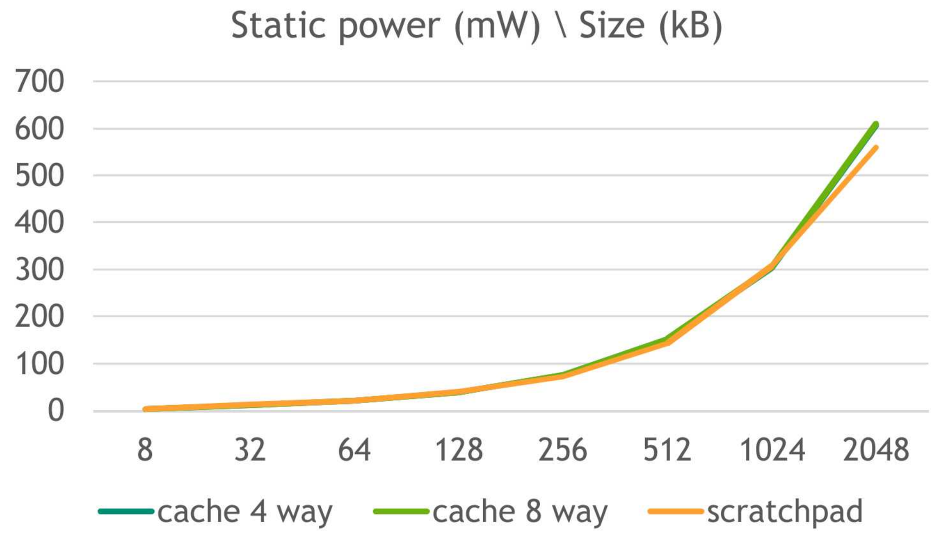


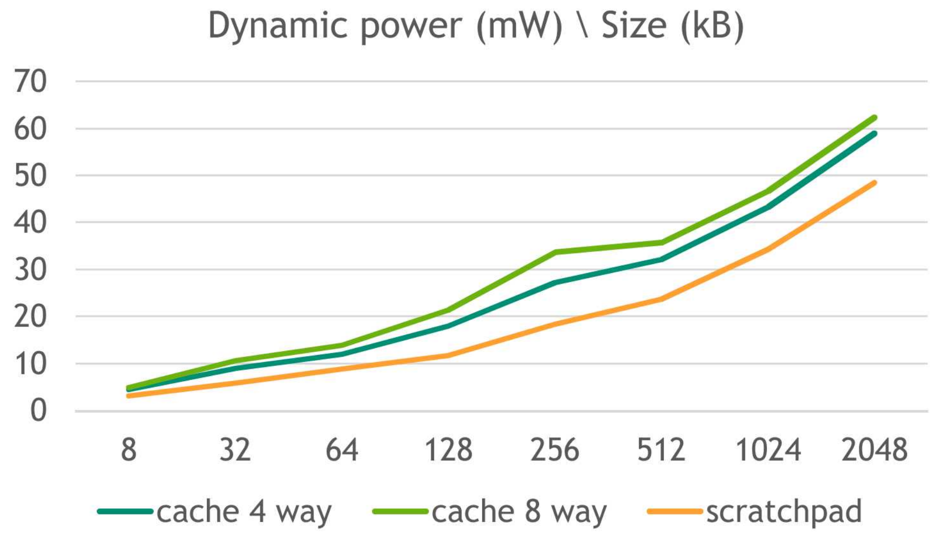


22nm


Access time (nS) \ Size (kB)
2.5
2
1.5
1
0.5
0
8
32
64
128
256
512
1024
2048
cache 4 way
cache 8 way
scratchpad


Area (mm2) \ Size (kB)
2
1.5
1
0.5
0
8
32
64
128
256
512
1024
2048
cache 4 way
cache 8 way
scratchpad


Static power (mW) \ Size (kB)
700
600
500
400
300
200
100
0
8
32
64
128
256
512
1024
2048
cache 4 way
cache 8 way
scratchpad


Dynamic power (mW) \ Size (kB)
70
60
50
40
30
20
10
0
8
32
64
128
256
512
1024
2048
cache 4 way
cache 8 way
scratchpad

Next Steps: Hardware Model Inventory

- **Update Hardware Power/Area models**
  - current models are projected from figures in literature
  - Run hardware components through Synthesis (FreePDK@45nm and 14nm)
- **TLB Design Study**
  - Quantify power/area benefit of moving TLB to memory interface
- **NOC Design Study**
  - BookSim to do performance evaluation of NOC configurations
  - OpenSOC to generate RTL for synthesis to quantify HW model