Extended-Base Global Address Space for Scalable and Secure Data Center & HPC Systems

Asst. Prof. Michel A. Kinsy

Adaptive and Secure Computing Systems (ASCS) Laboratory

TEXAS TECH UNIVERSITY

Massachusetts Institute of Technology

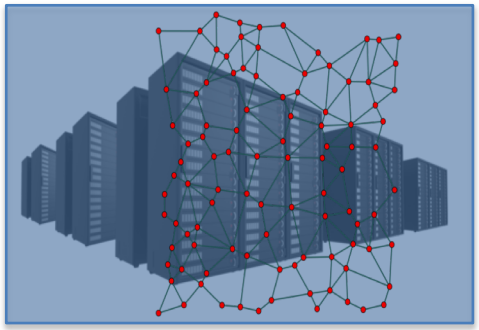
BOSTON UNIVERSITY

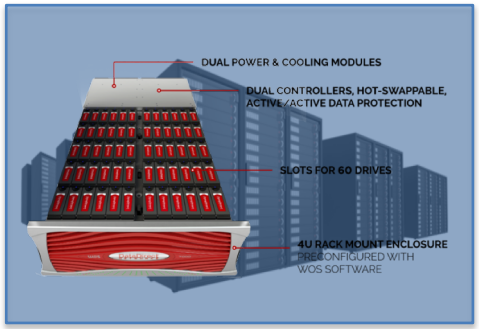
HPC & Data Center Scale Addressing

- Extended Base Global Address Space (xBGAS)
- Goals:
  - Provide extended addressing capabilities without ruining the base ABI
    - EG, RV64 apps will still execute without an issue
  - Extended addressing must be flexible enough to support multiple target application spaces/system architectures
    - Traditional data centers, clouds, HPC, etc..
  - Extended addressing must not specifically rely upon any one virtual memory mechanism
    - EG, provide for object-based memory resolution
- What is xBGAS NOT?
  - …a direct replacement for RV128

Application Domains

- HPA-FLAT
  - High performance analytics flat addressing
  - For extremely large datasets that are too difficult/time consuming to shard
- MMAP-IO
  - Map storage tiers into address space
  - Potential for object-based addressing
  - See DDN WOS
- Cloud-BSP
  - Potential for global object visibility for in-memory cloud infrastructures (Spark)
  - Reduce the time/cost to port Java to a full 128-bit addressing model
- Security
  - Fine grained, tagged security extensions to base addressing model
  - Tags are stored/maintained as ACL's for secure memory regions
- HPC-PGAS
  - High Performance Computing: Partitioned Global Address Space





xBGAS Motivation

- Shared Memory vs. Message Passing Models
  - Message passing is generally viewed as having more efficient and scalable communication infrastructure with low possibility of false sharing and race conditions
    - But it is complex to program
    - Data partitioning is a real challenge
    - Coordination of communication (send/receive pairs) needs to be explicit
    - Data replication leads to more memory requirement
  - On the other hand, with the shared memory model,
    - Programming is substantially easier
    - Shared data can be directly accessed
    - Implicit communication, direct reads, writes

xBGAS Motivation

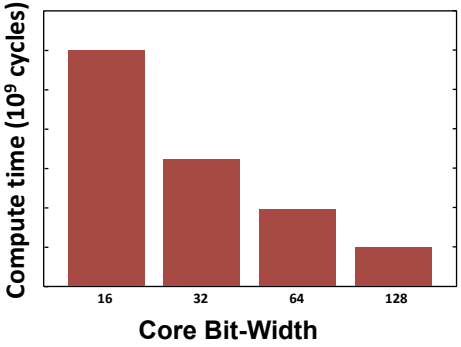
- How do get the best of both world?
  - Many previous research and design efforts have aimed at this
    - E.g., Efforts around Partitioned Global Address Space (PGAS)
      - Unified Parallel C, Chapel, X10, UPC++, Coarray C++, DASH and SHMEM
    - Asynchronous partitioned global address space (APGAS) with both local and remote task creation
      - Chapel and X10
- The key differences in xBGAS
  - Open-source RISC ISA based
  - Security-aware communication and data sharing

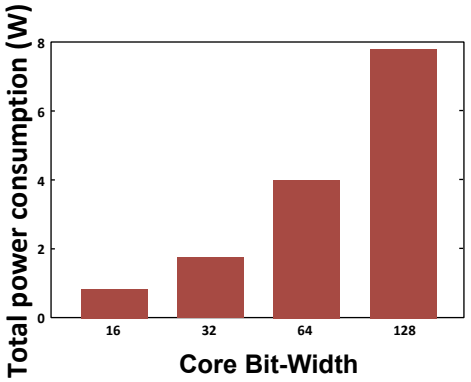
xBGAS Motivation

- The key differences in xBGAS
  - RISC-V ISA
    - A new, open, free ISA from Berkeley
    - Several variants
      - RV32, RV64, RV128 – Different data widths
      - 'I' – Base Integer instructions
      - 'M' – Multiply and Divide
      - 'A' – Atomic memory instructions
      - 'F' and 'D' – Single and Double precision floating point
      - 'V' – Vector extension
      - And many other modular extensions
    - Design principles
      - Simplicity favors regularity
      - Total number of instructions in the instruction set
      - Smaller is faster
      - Make the common case fast

xBGAS Motivation

- The key differences in xBGAS
  - RISC-V ISA
    - Why RISC?
      - You never really wanted a CISC (e.g., 128-bit) Architecture!
    - Key insights
      - Jaguar – 2005/2009 Oak Ridge
        - 224,256 CPU cores - 1.75 petaflops





You do not need complex or wide instructions, but you do need long memory addressing mode to reach all these cores for a shared memory configuration!

xBGAS Motivation

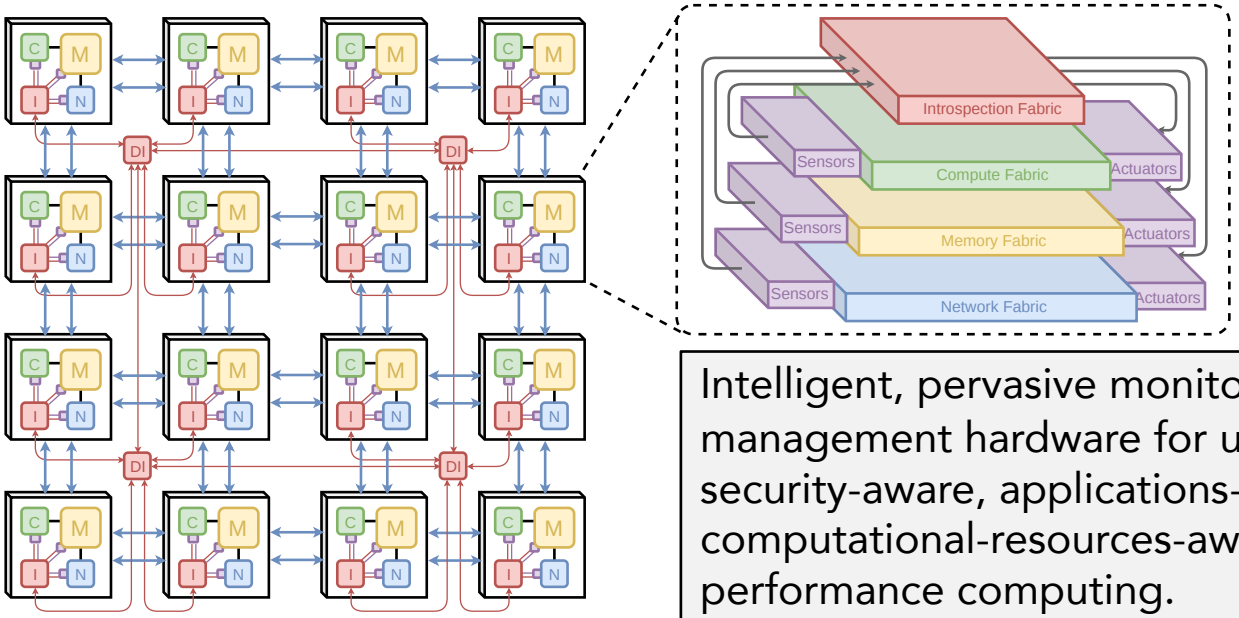
- The key differences in xBGAS
  - Security-aware communication and data sharing
    - Traditionally a small number of systems is commissioned using
      - Proprietary intellectual property (IP) blocks
      - Obscure or highly under-specified system architecture
    - The broader HPC community is compelled to narrow their algorithm optimization and software development efforts to these architectures
  - One can argue that the lack of open-architectures has severely hindered innovation and progress
    - Security-related research efforts are generally abandoned due to the lack of visibility or access to micro-architecture details to test, analyze or validate vulnerabilities, threats or risks

xBGAS Motivation

- The key differences in xBGAS
  - Security-aware communication and data sharing
  - One can argue that the lack of open-architectures has severely hindered innovation and progress
    - Security-related research efforts are generally abandoned due to the lack of visibility or access to micro-architecture details to test, analyze or validate vulnerabilities, threats or risks
    - Almost complete lack of hardware-level support for process monitoring and behavior analytics in HPC and large-scale computing systems
    - The architectural security framework in HPC and large-scale computing systems is still fairly obscure

xBGAS Motivation

- The key differences in xBGAS
  - Security-aware communication and data sharing
    - One such approach is Helios Architecture effort in our laboratory – an introspective HPC system



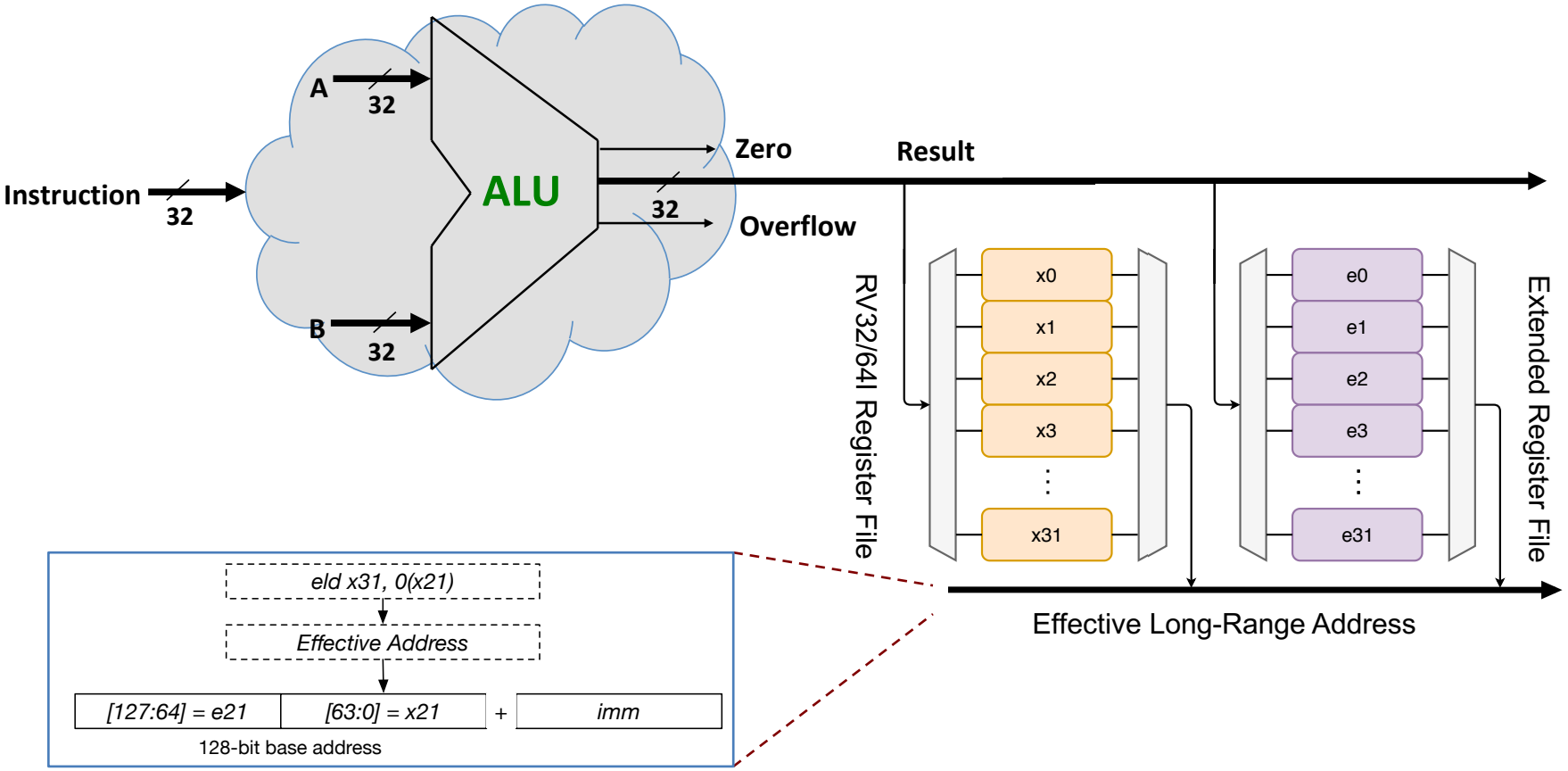
Intelligent, pervasive monitoring and management hardware for user-specified security-aware, applications-aware, and computational-resources-aware high-performance computing.

xBGAS Motivation

- The key differences in xBGAs
  - Flat address space for very large data
    - No need for multiple address spaces
  - Flat address space for lots of IO/Storage
    - Map all SSDs in one range
  - Global object visibility with programmable access security policy for in-memory data
    - No software translation
    - Thousands of cycles saved per operation
    - Provides an extra 64 to 96 bits of address space
  - This is an open and open-source project
    - RTL Cores, processors, nodes, and the whole software stack

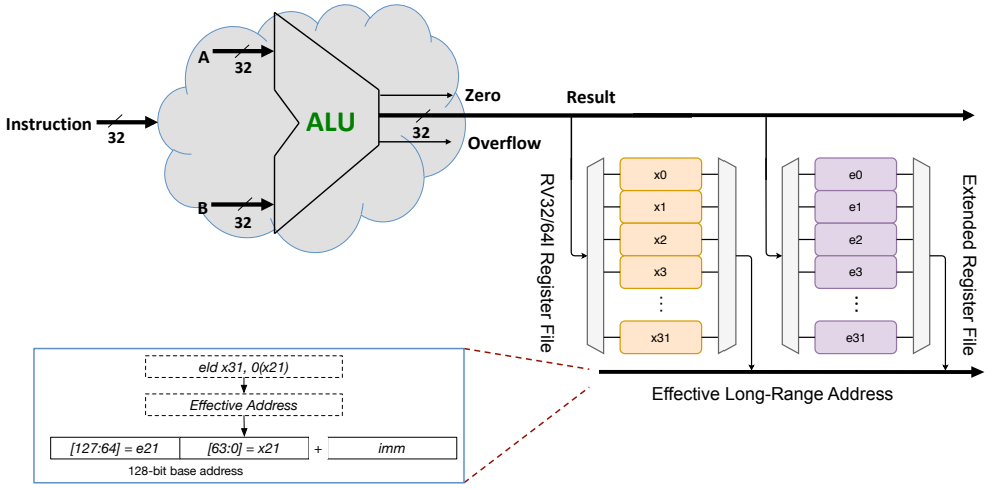
Improve the secure posture of HPC and data center systems through open and formally verified secure cores, nodes, and even systems.

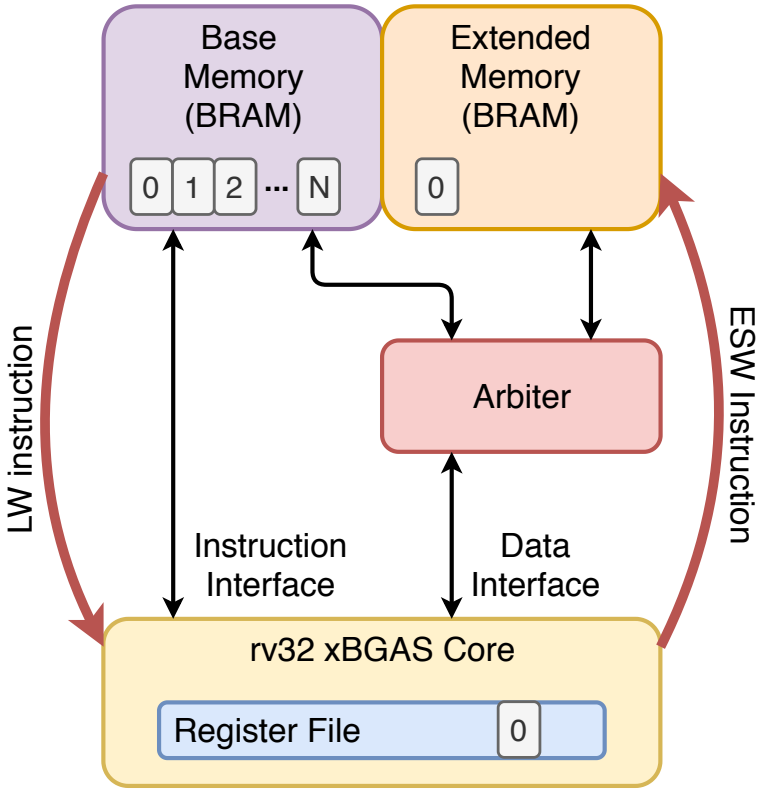
xBGAS Essence



xBGAS Essence

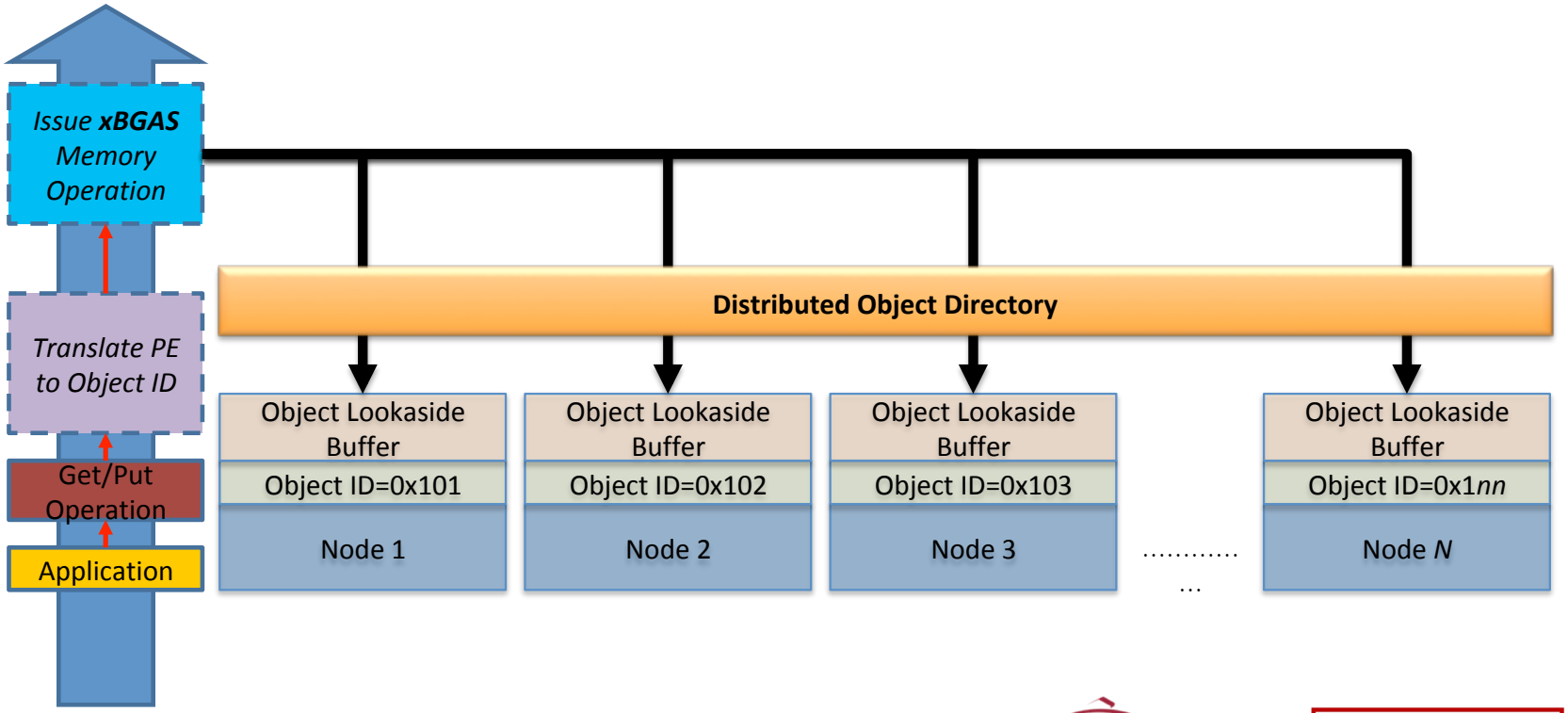
- The customizable instruction capabilities of the RISC-V ISA allows for this extension seamlessly





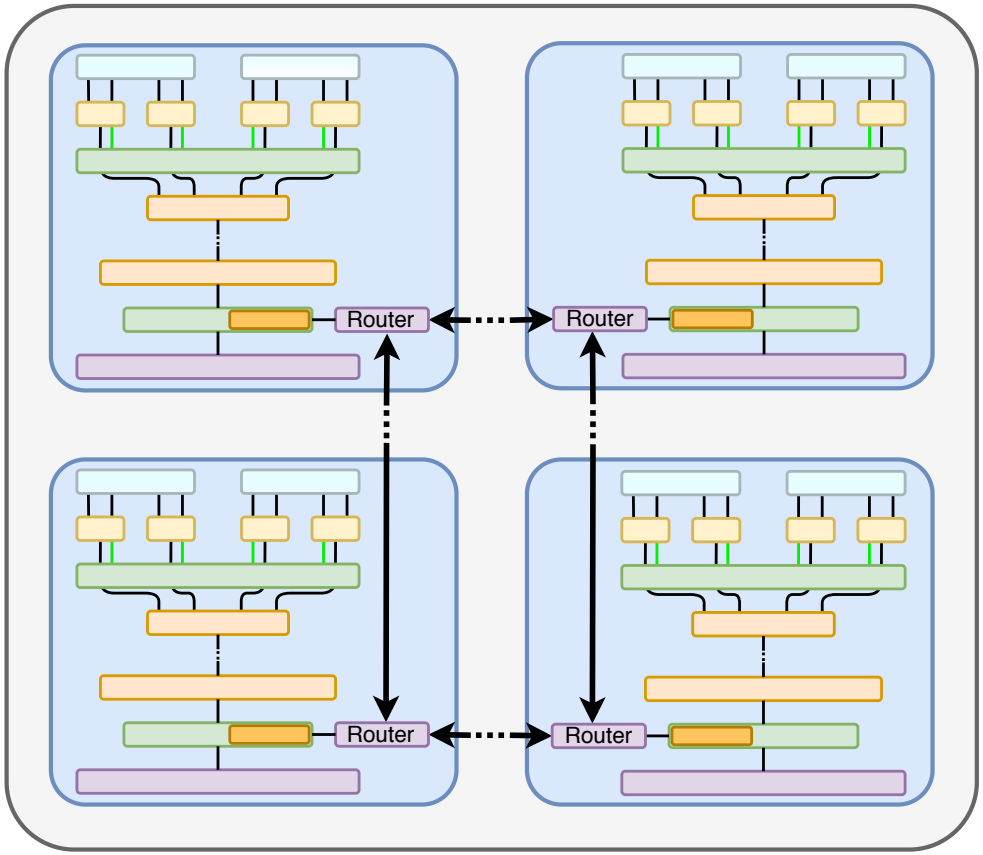
xBGAS Essence

- The Object-Based Global Memory Space (OBGMS) paradigm for in-memory data allows to formally verify the security properties and access policies of the data objects



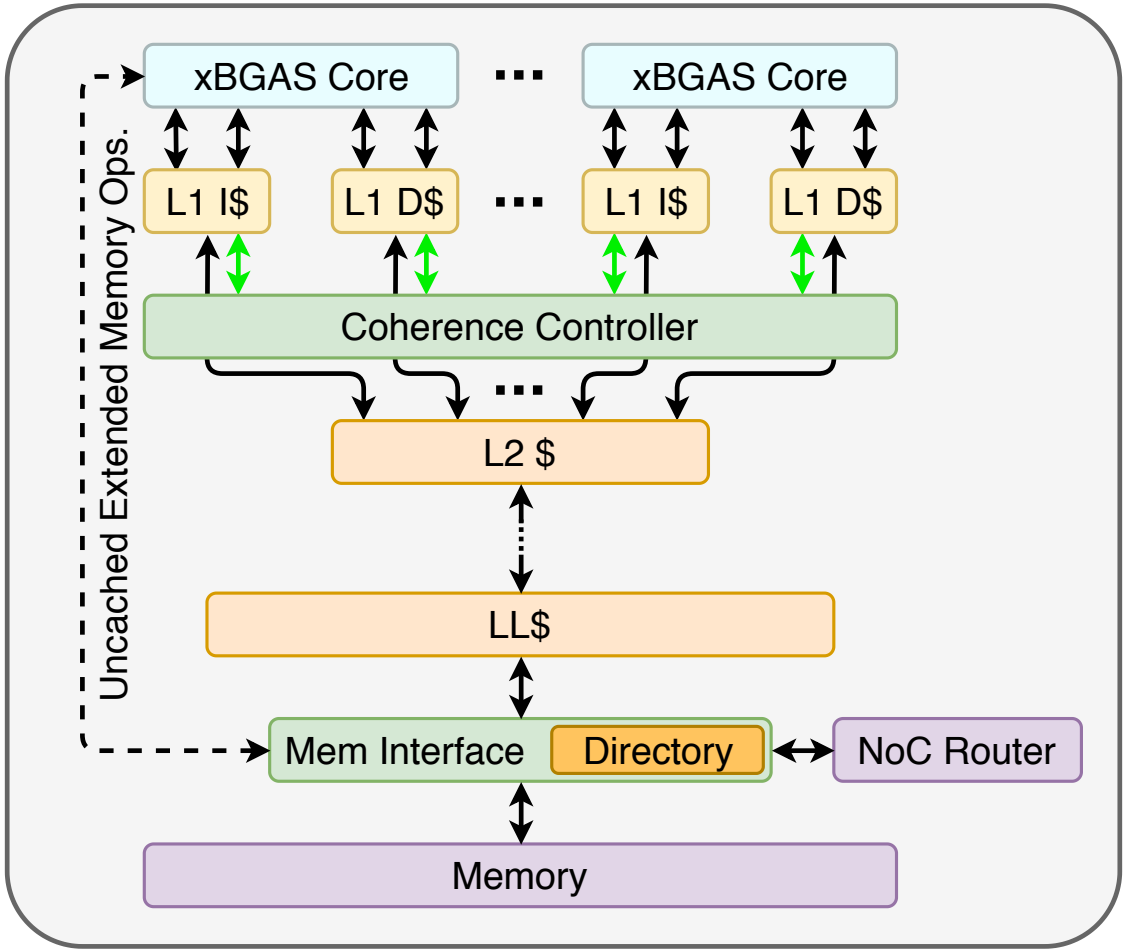
Full xBGAS Multi-Node Architecture

- Features
  - System granularities
    - Hardware thread
    - Core
    - Processor
    - Node
  - Processors
    - Single-, 5-, 7- Cycle, Out-of-Order, hardware-Threaded RISC-V Cores
  - Memory subsystem
    - A multi-level caching system
    - Both snoopy and directory-based coherence
  - Network-on-Chip
    - Wormwhole router
    - Table- and logic-based routing

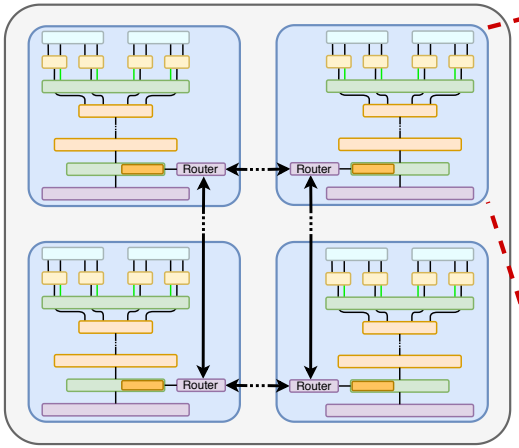


xBGAS Multi-Core Processor Node

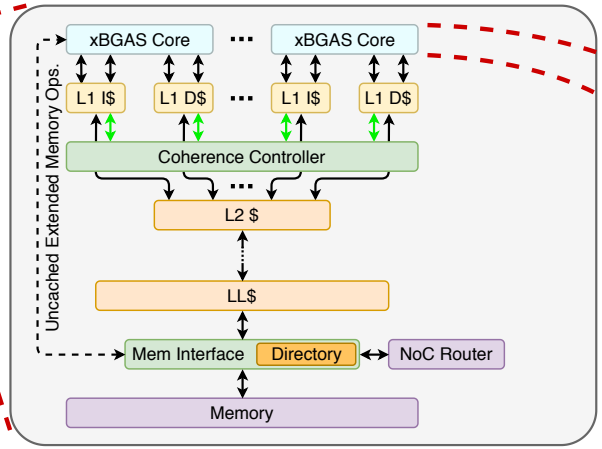
- Extended Memory ops un-cached
- Multi-port memory interface supports NoC and ext-mem connection



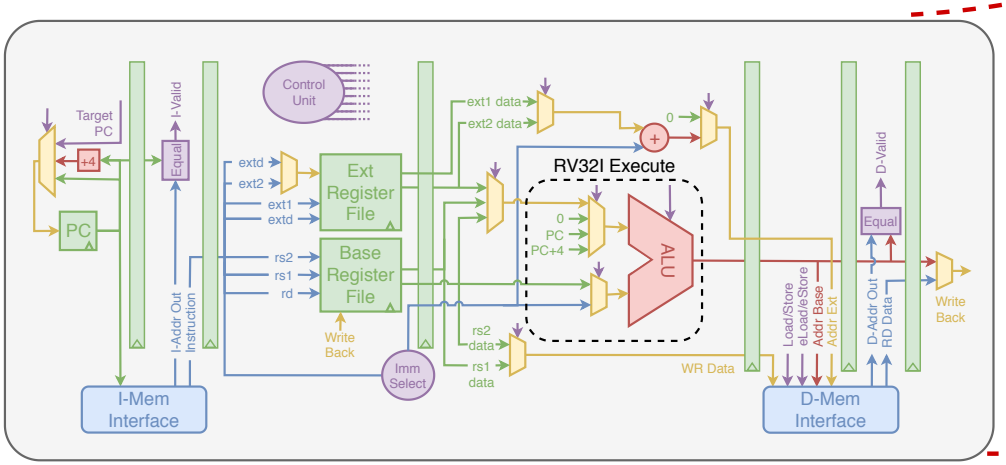
xBGAS Core Architecture



xBGAS Multi-Processor Node



xBGAS Multi-Core Processor



xBGAS Core

TEXAS TECH UNIVERSITY

Tactical Computing Labs

Massachusetts Institute of Technology

BERKELEY LAB

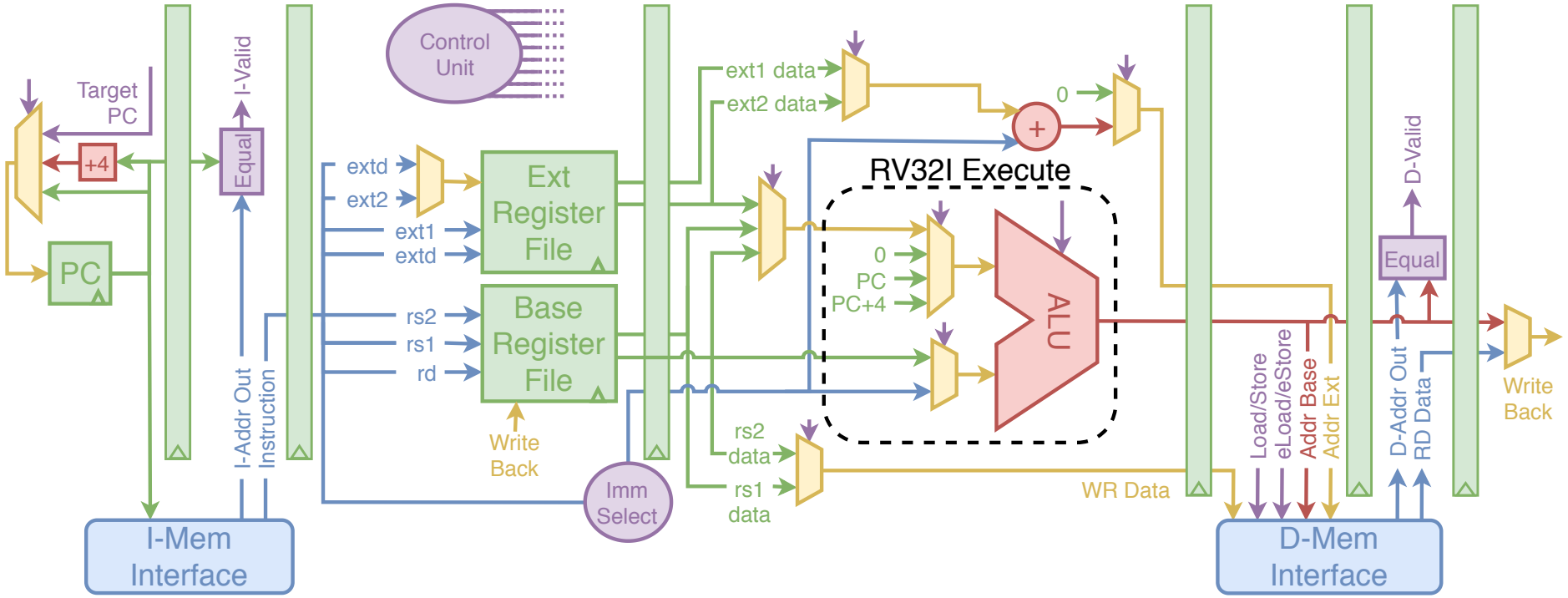
BOSTON UNIVERSITY

xBGAS Core

- ISA: RV32I
  - User level ISA 2.2
- Decode Stage
  - Extended register file
  - Additional control logic
- Execute Stage
  - New extended address adder
  - Several new multiplexers
- Memory Stage
  - New extended address port

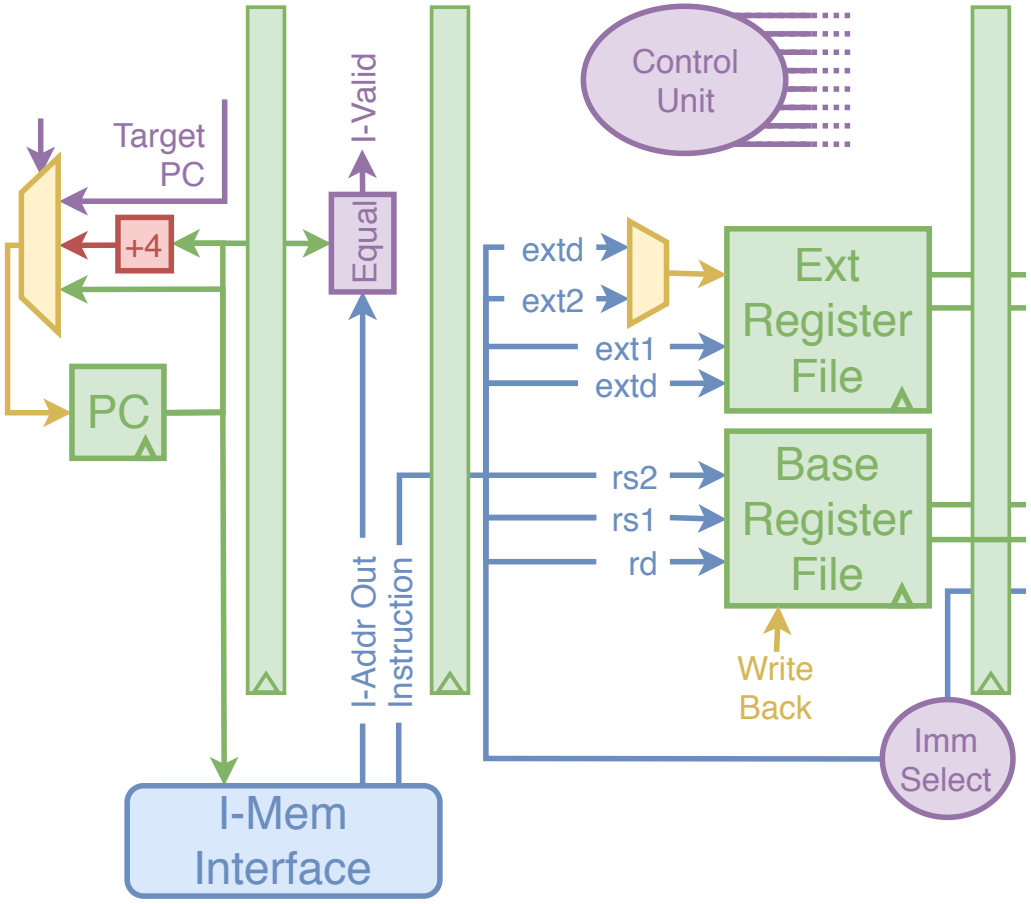
xBGAS Core

- Five new multiplexers w/ control logic
  - Mostly in execute stage



xBGAS Core – Fetch Stage

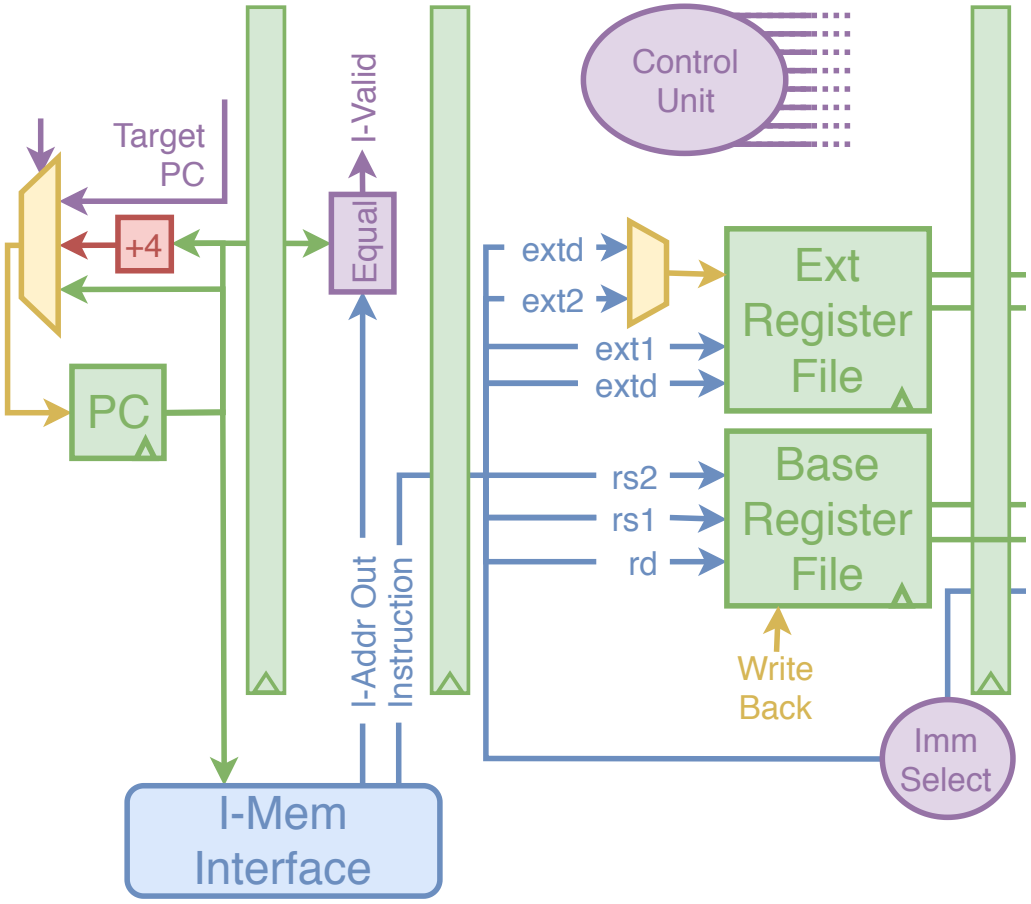
- Fetch Stage is unchanged
- PC limited to XLEN bits
  - 32 or 64 bits
- Cannot execute in extended address space



Fetch Issue, Fetch Receive, Decode Stages

xBGAS Core – Decode Stage

- Adds extended register file
- Mux before ext2 port
  - For three operand stores
  - esr[d,w,h,b,e]



Fetch Issue, Fetch Receive, Decode Stages

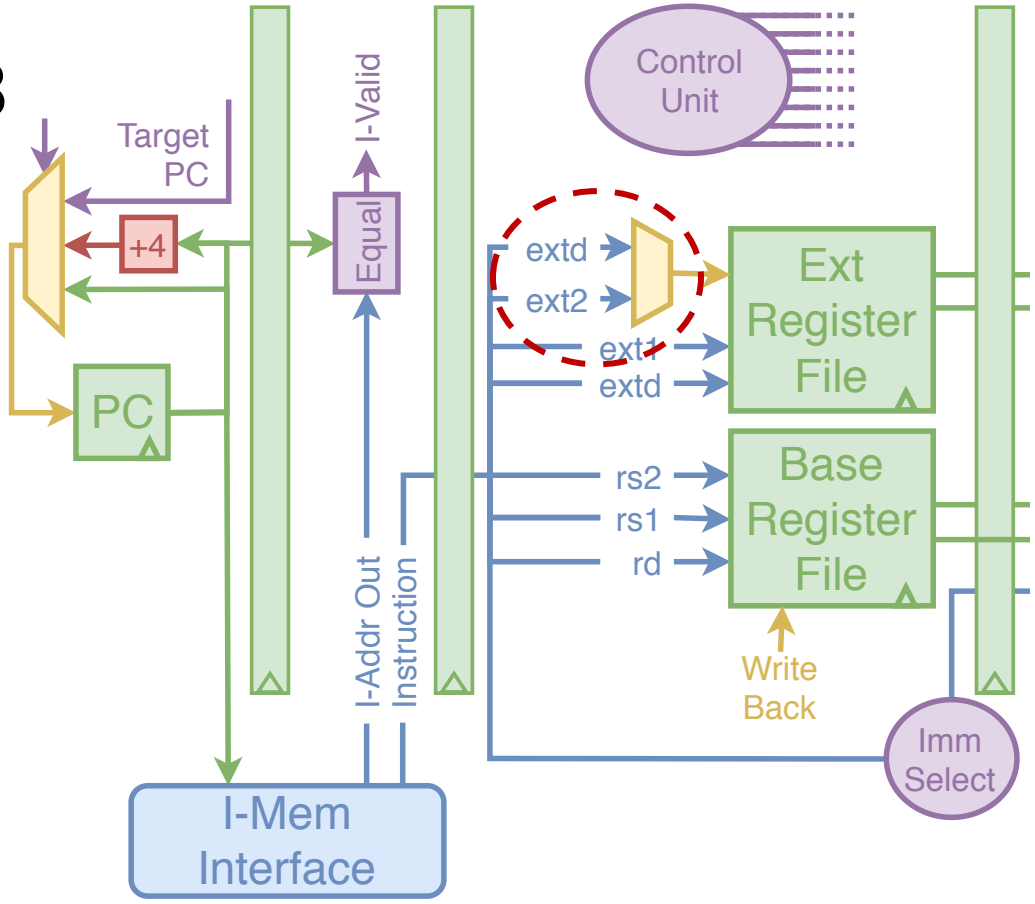
xBGAS Core – Decode Stage

esre ext1, rs2, ext3

- Store ext1 at {ext3, rs2}
- ext3/rd field to ext2 port

esw rs1, imm(rs2)

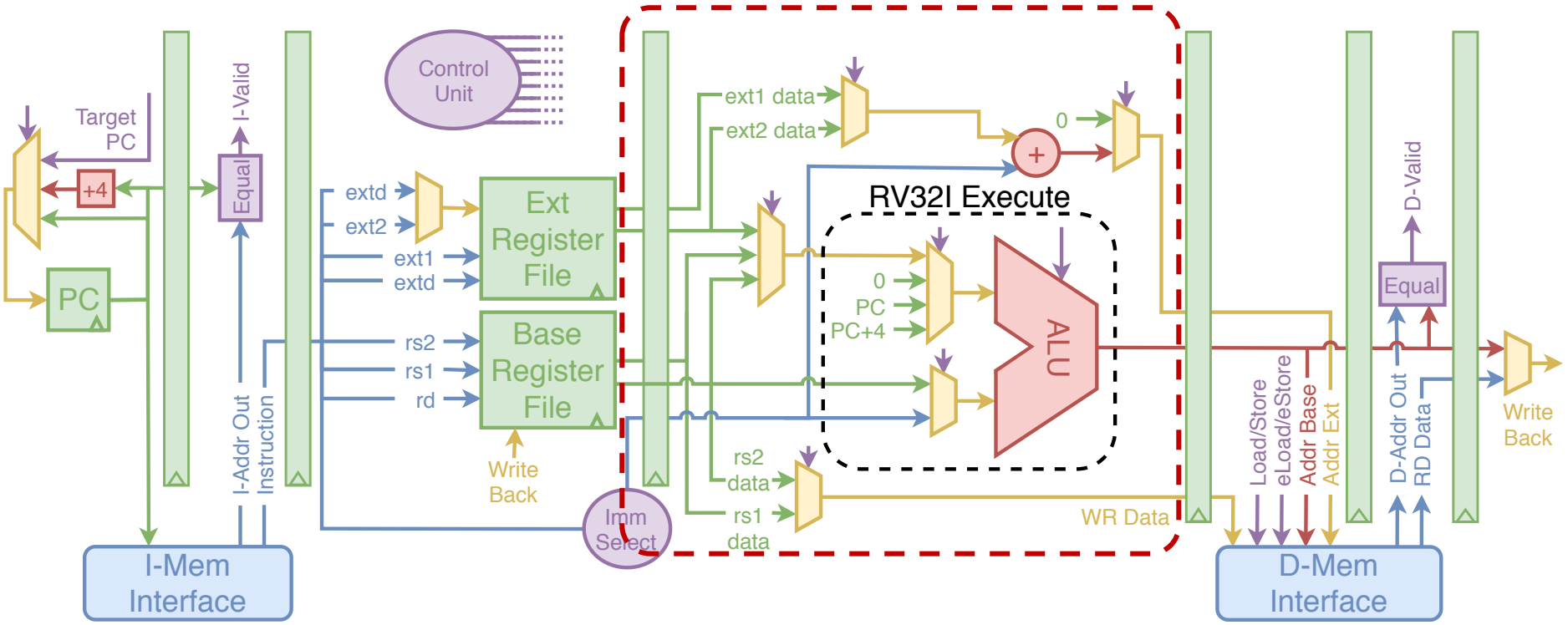
- Store rs1 at {ext2, rs2}
- rs2 field to ext2 port



Fetch Issue, Fetch Receive, Decode Stages

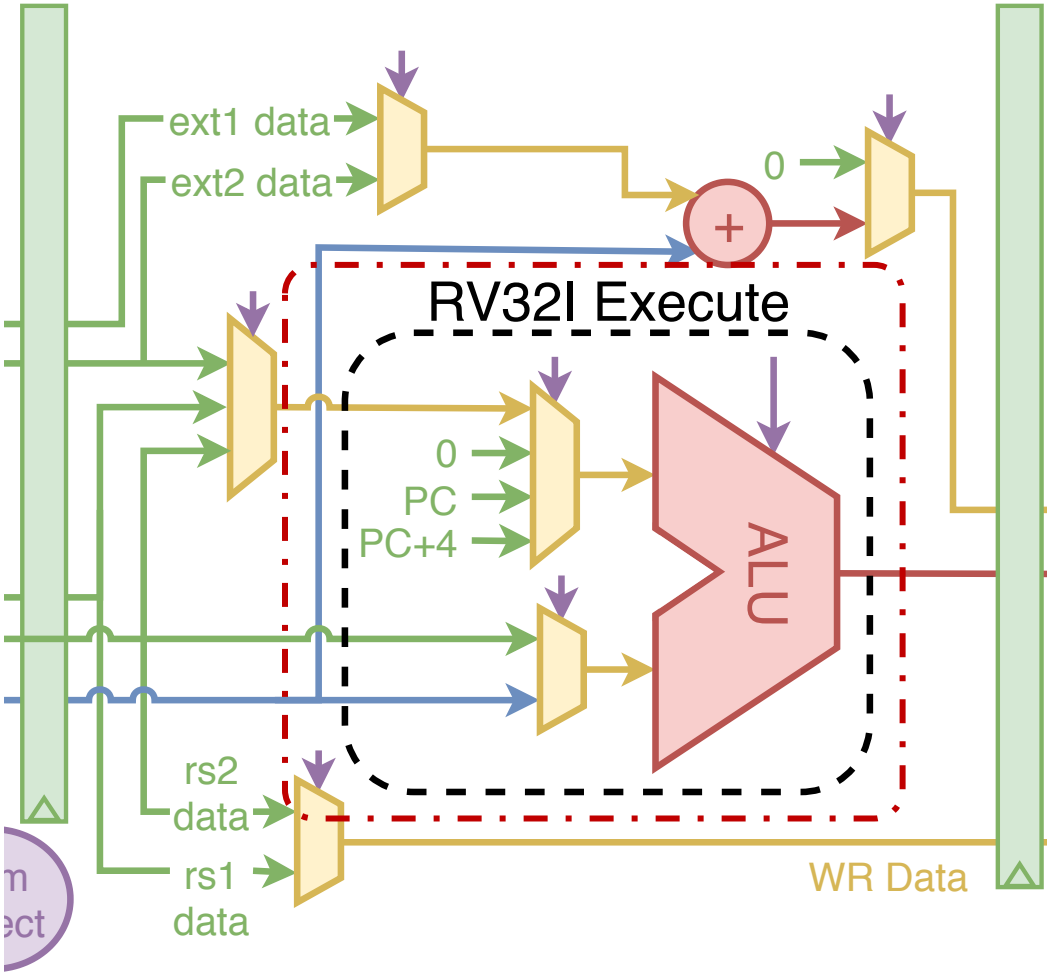
xBGAS Core – Execute Stage

- Receives all four read ports
  - rs1, rs2, ext1, ext2



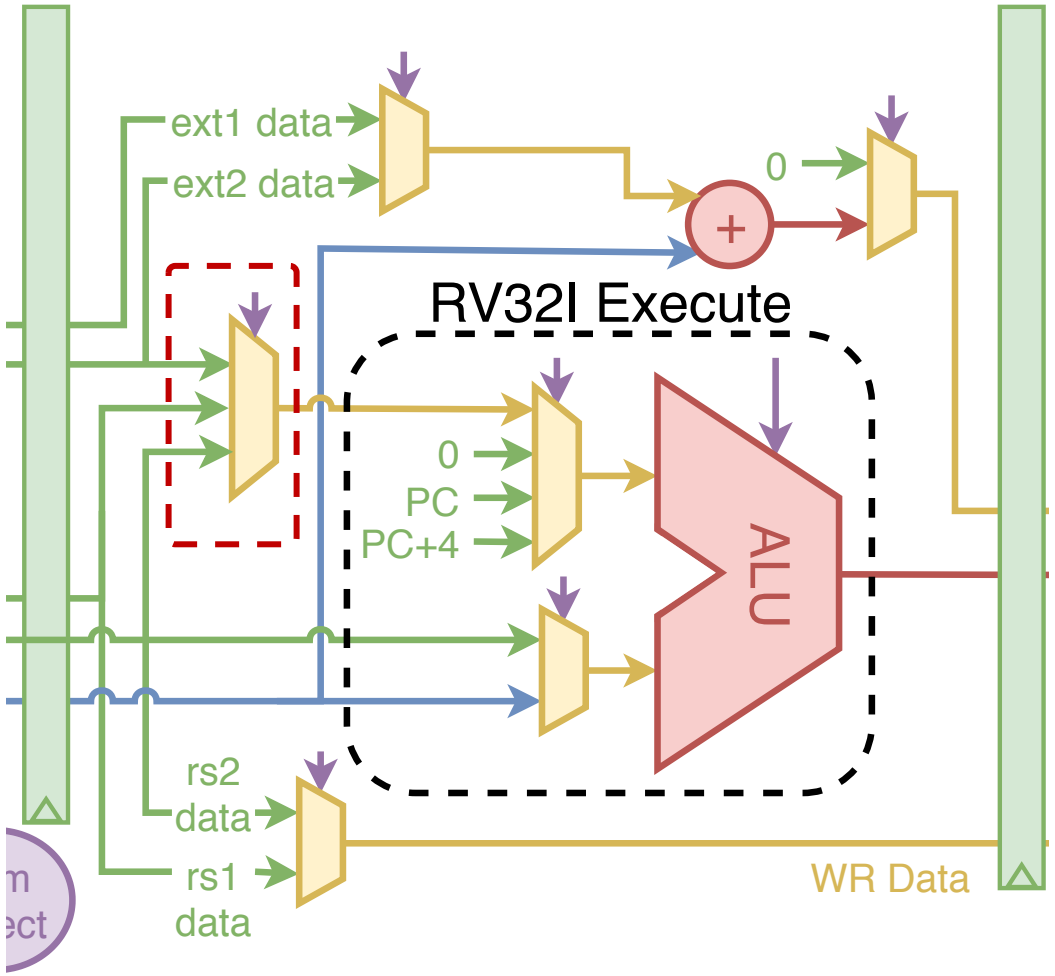
xBGAS Core – Execute Stage

- Wrapped around RV32I execute module



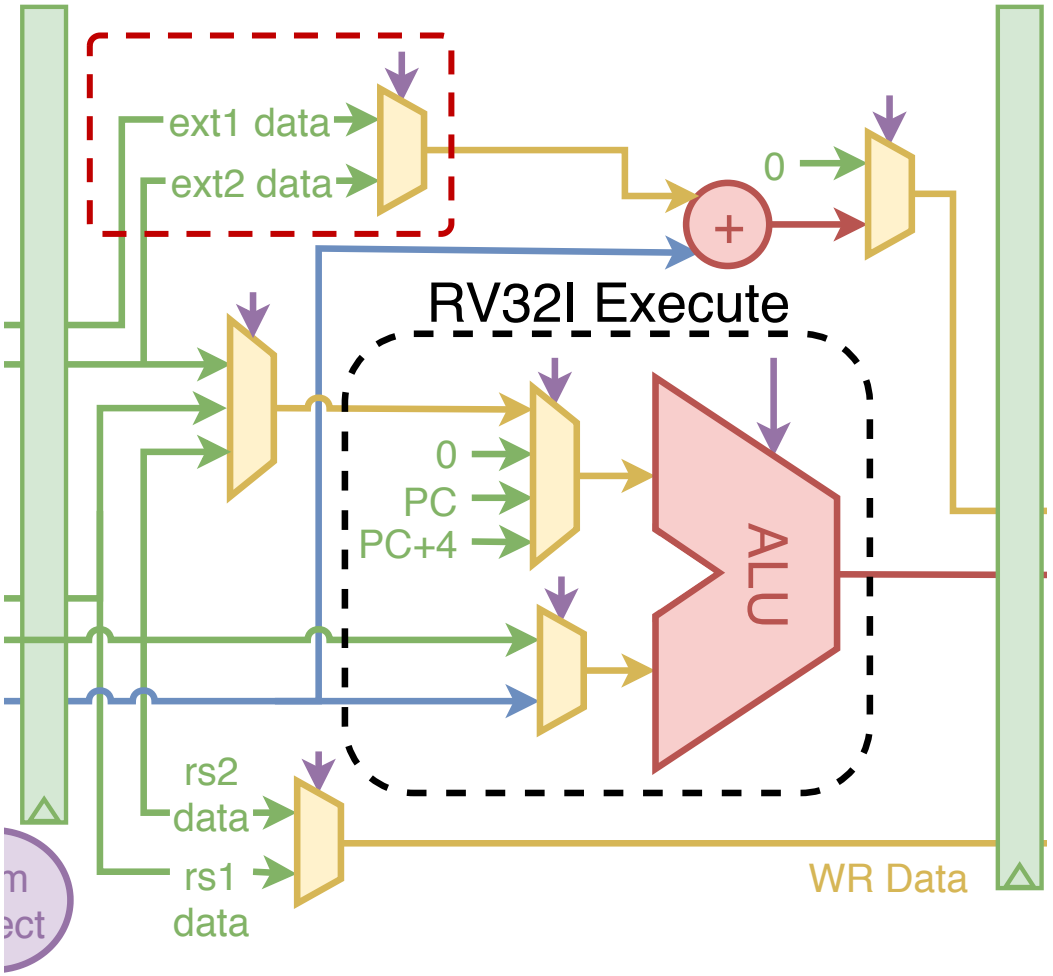
xBGAS Core – Execute Stage

- Wrapped around RV32I execute module
- Mux rs1, rs2, ext1 into ALU



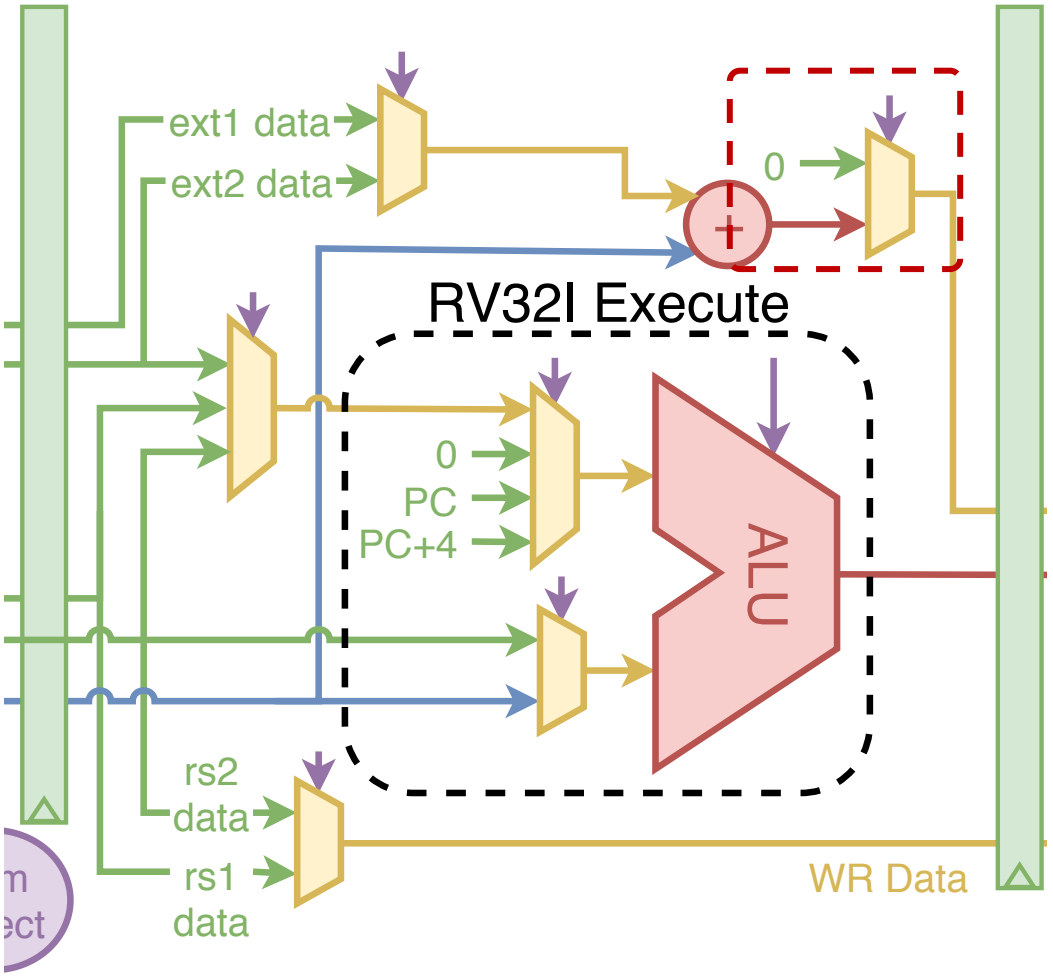
xBGAS Core – Execute Stage

- Wrapped around RV32I execute module
- Mux rs1, rs2, ext1 into ALU
- Mux ext1, ext2 to high address bits



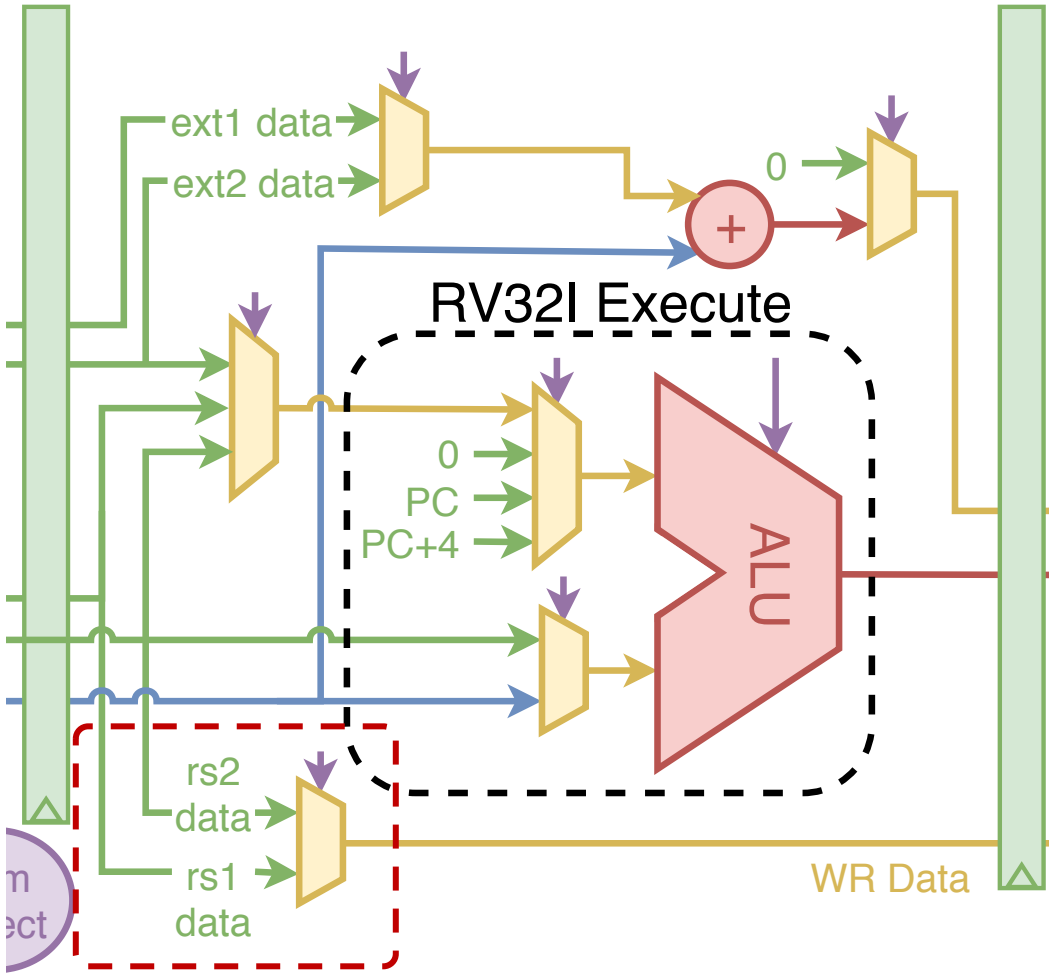
xBGAS Core – Execute Stage

- High address bits to 0 when not used
- Could use local extended address instead
  - Make every OP an extended memory OP



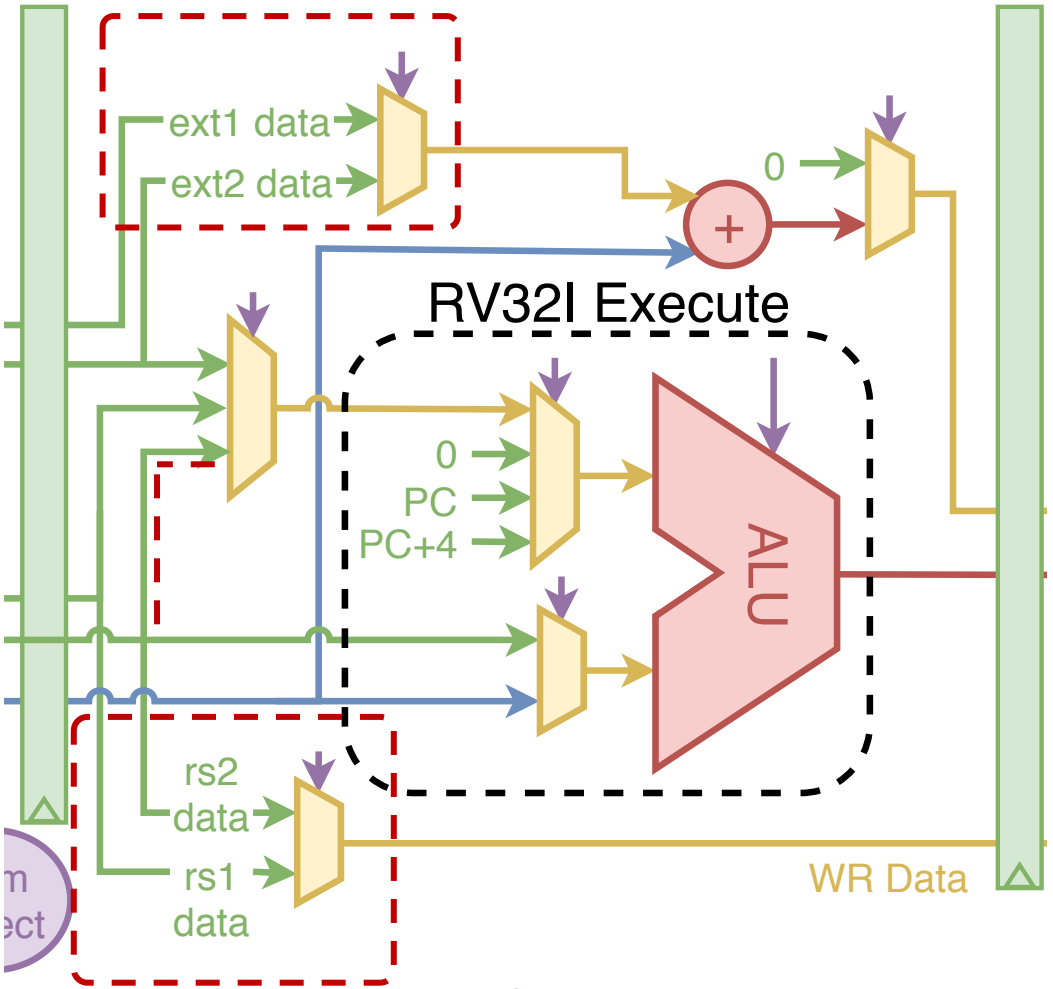
XBGAS Core – Execute Stage

- Mus rs1, rs2 to mem write port
  - esw rs1, imm(rs2)
  - sw rs2, imm(rs1)
- RV32I uses rs1 as address
- XBGAS uses RS2 as address



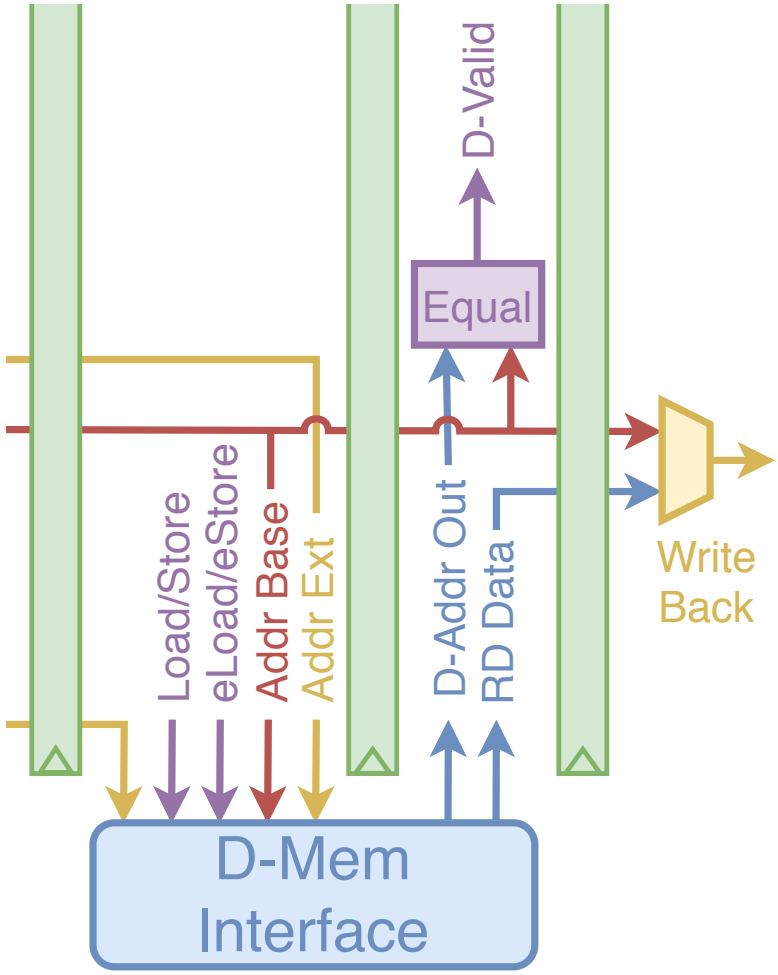
RS2 Address Overhead

- Two of Five new multiplexers
- Third input on another mux
  - 1-bit->2-bit control signal



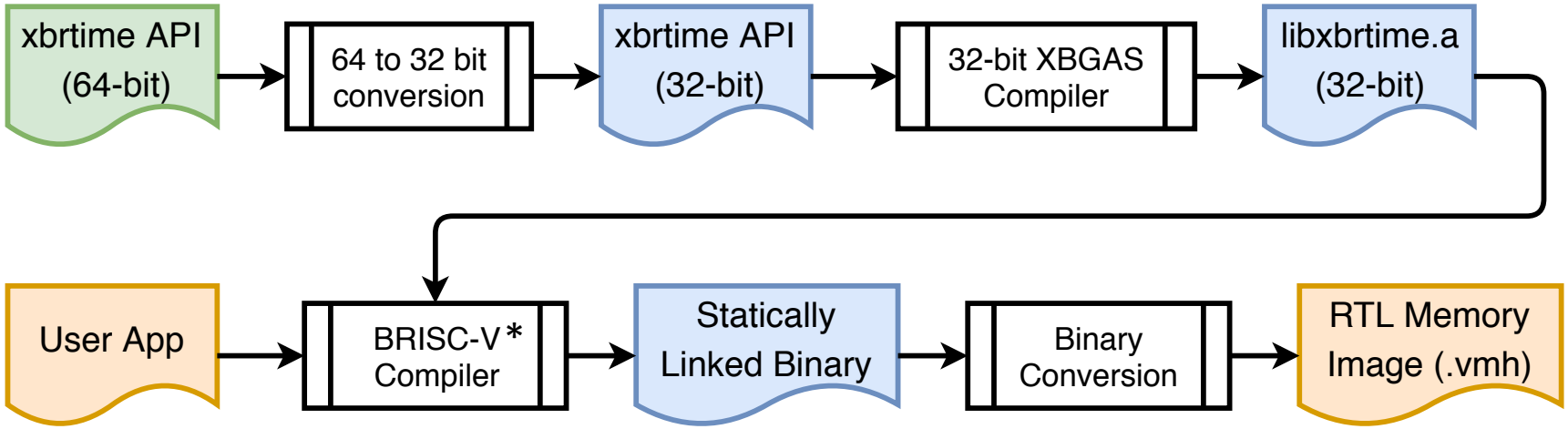
xBGAS Core – Memory Stage

- Separate R/W signals for base and extended Ops
- Extended address port
- Interface separates core and interconnect
  - Can easily update extended address space interconnect
- No changes to writeback stage



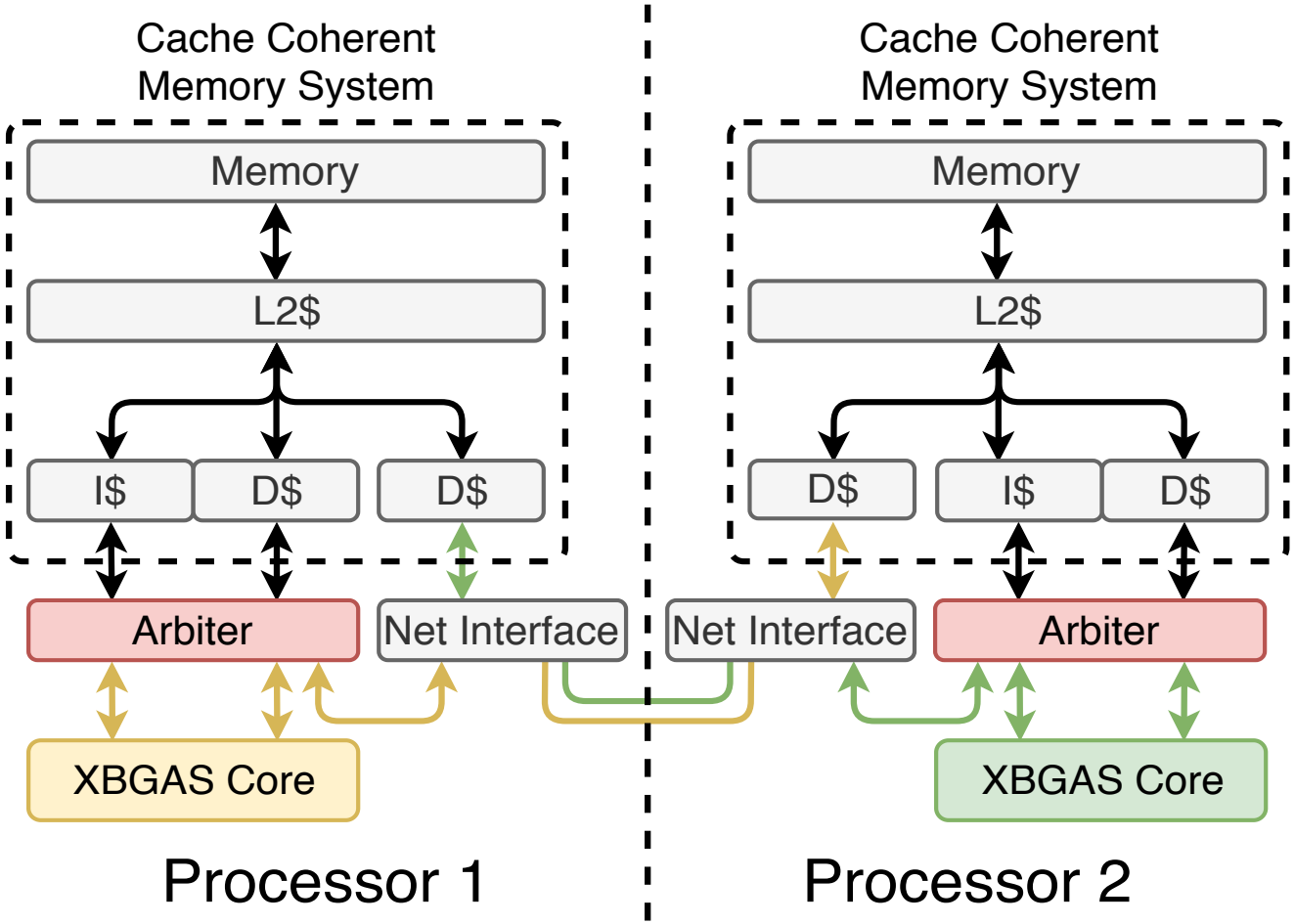
xBGAS Compiler Flow

- Compile runtime with modified xBGAS tools
- Use cross-compiler scripts
  - Link to xBGAS runtime API

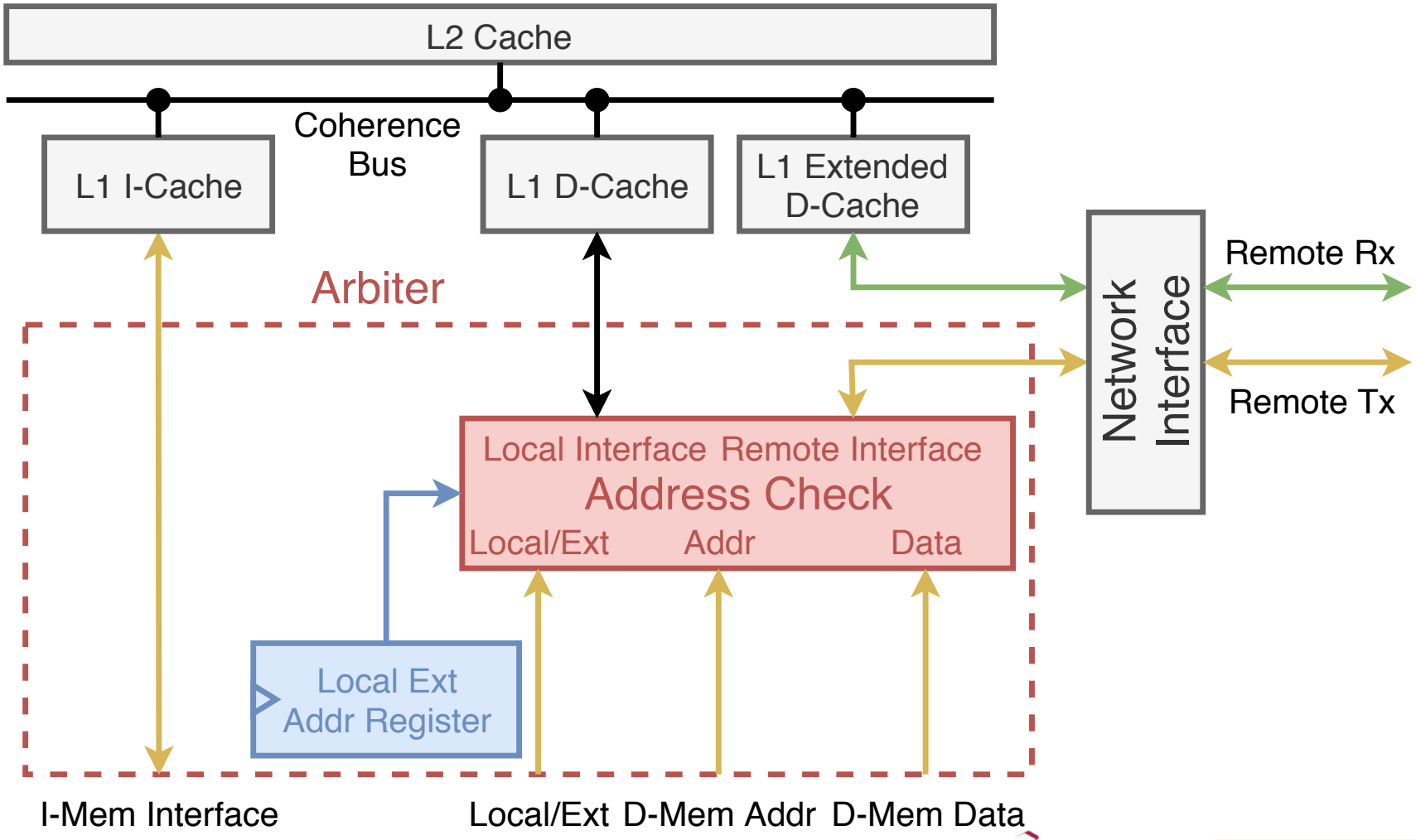


* Boston RISC-V Development Platform - https://ascslab.org/research/briscv/index.html

Extended Address Interconnect

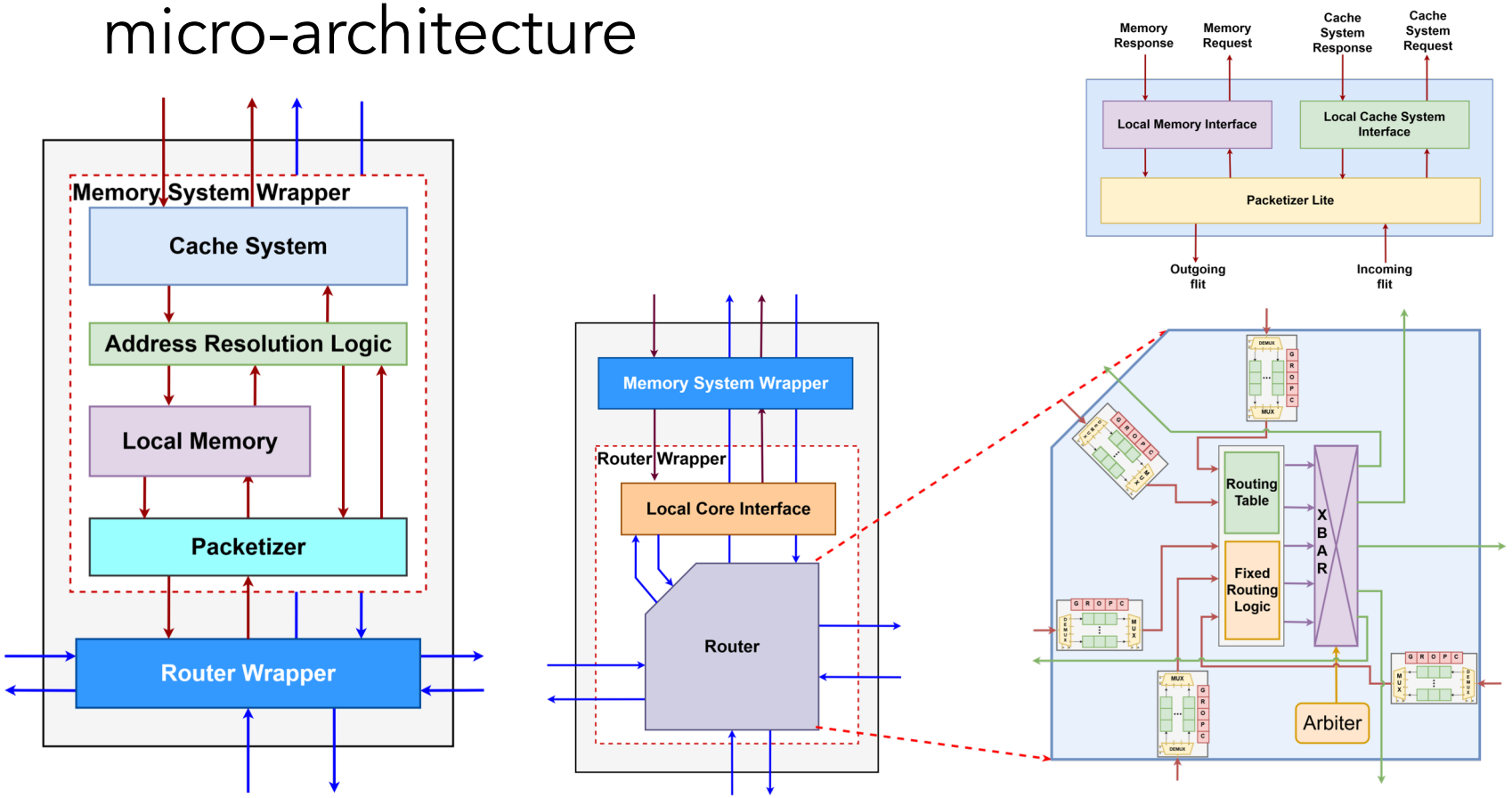


Address Arbiter



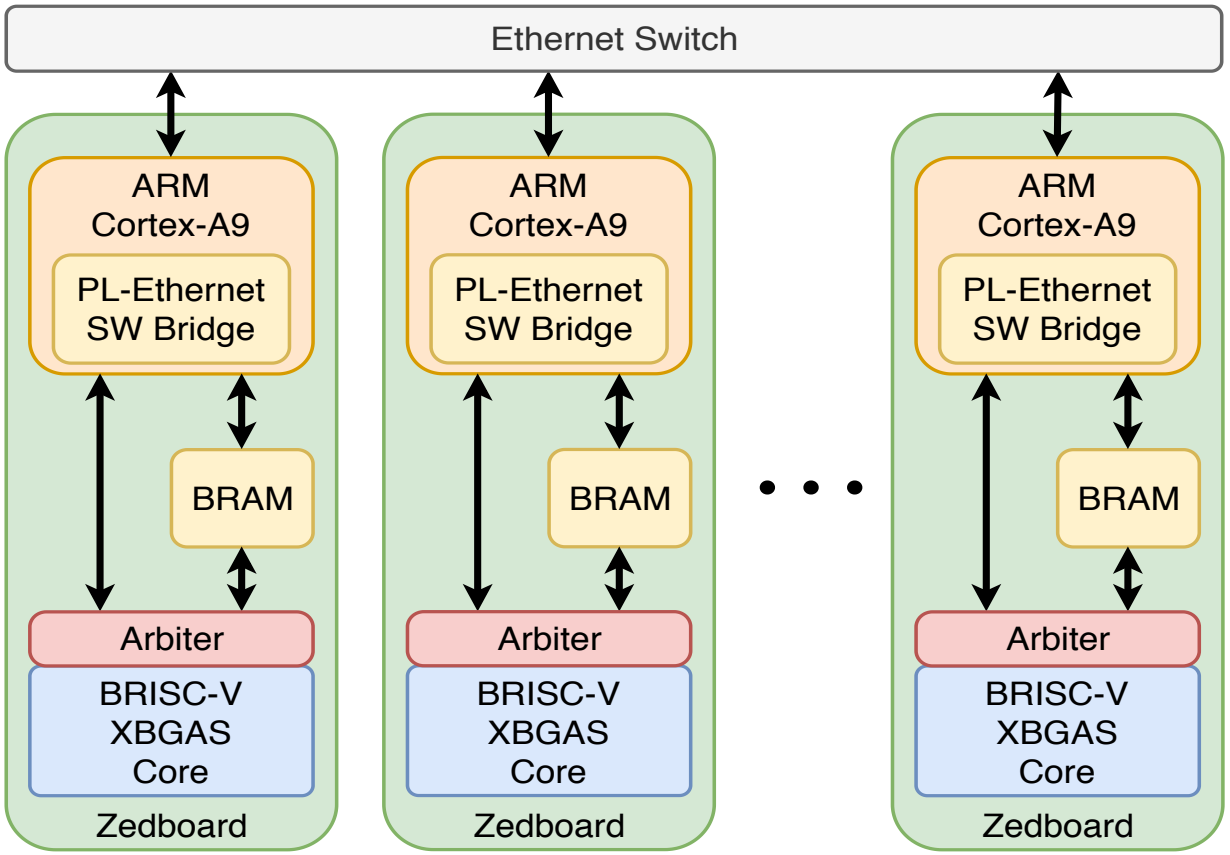
Network-on-Chip

- For brevity, we will not go into the network micro-architecture



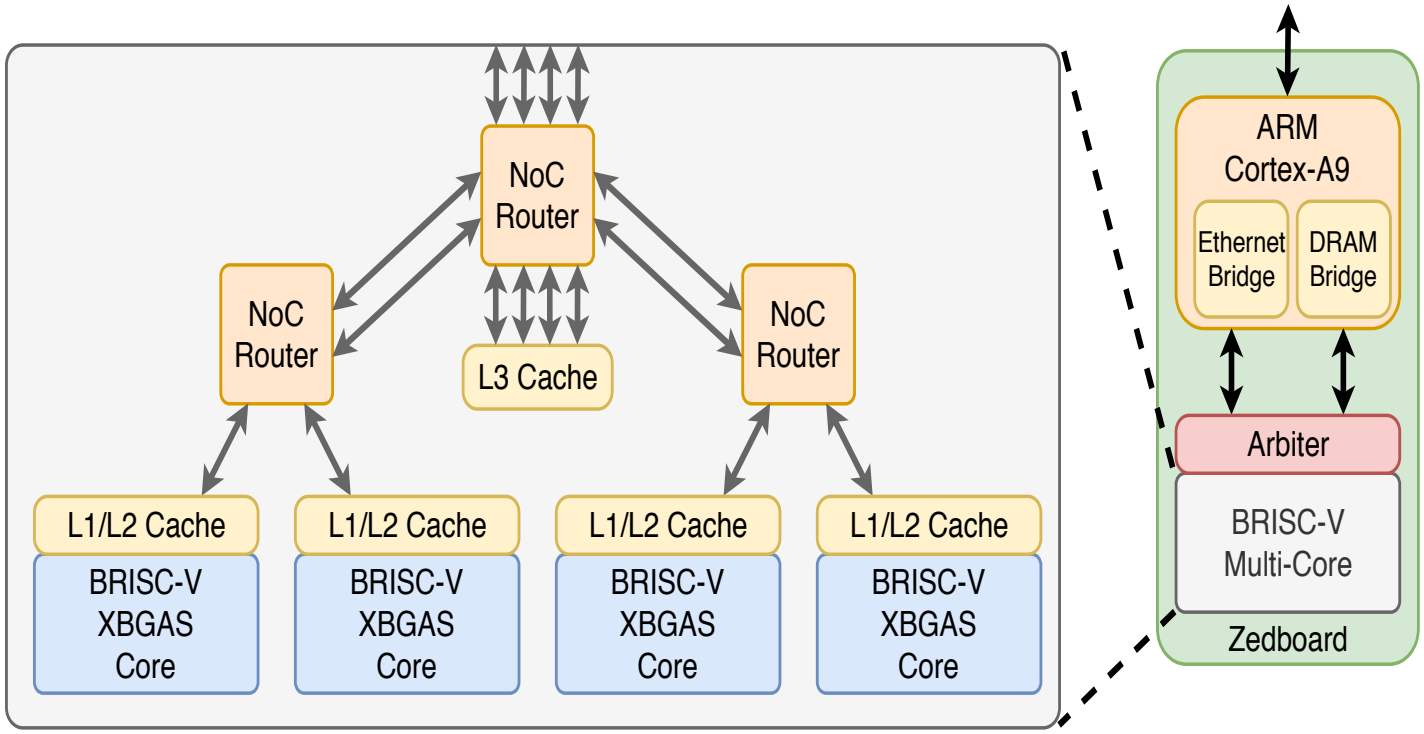
xBGAS Physical Implementation

- Partitioned address space
- Run SW from xBGAS runtime



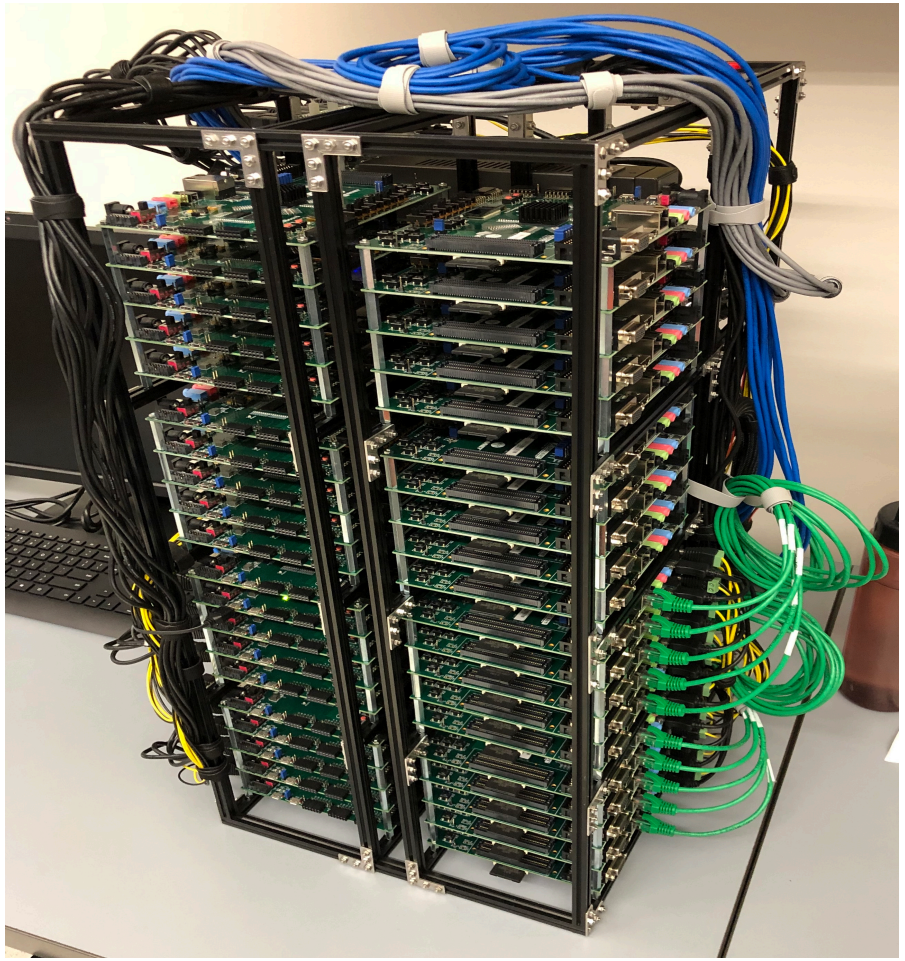
xBGAS Physical Implementation

- Directory based cache
- Fat tree NoC (simplifies directory based cache)



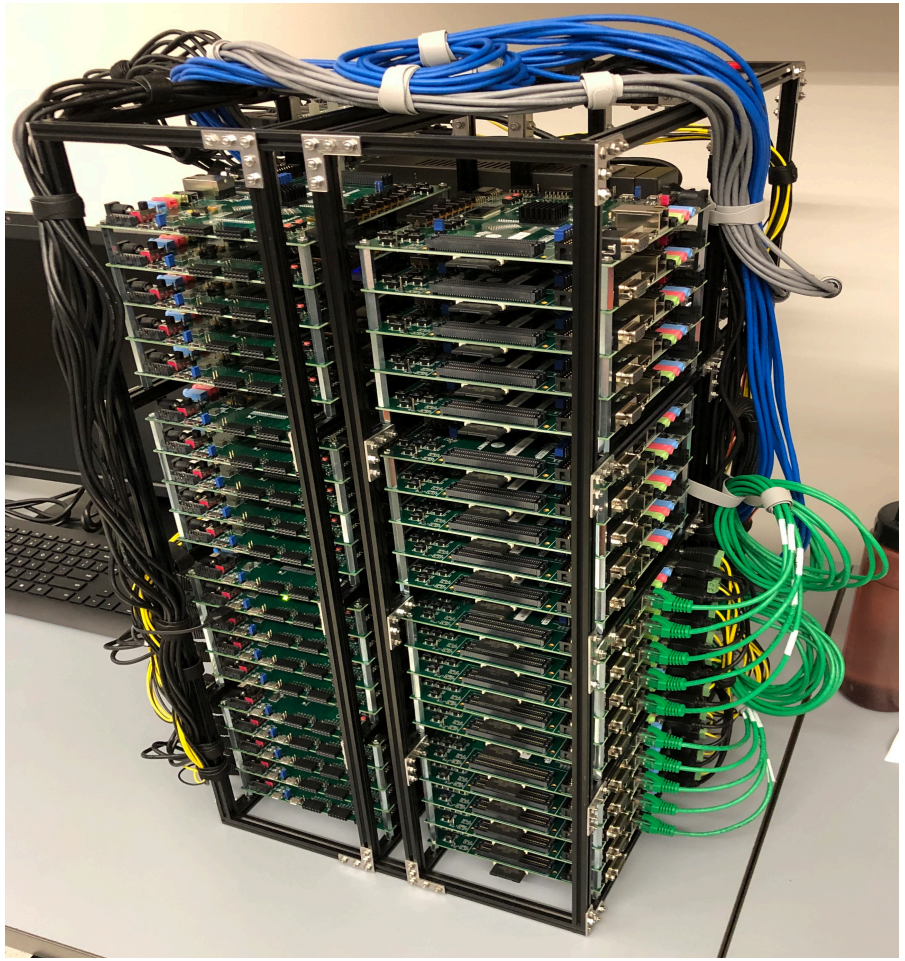
First FPGA Cluster Prototype

- 40 Zedboards
  - Zynq-7000 SoC
  - 85k Logic cells
  - Dual core ARM Cortex-A9 (666MHz)
- 3.4M Total logic cells
- 50 port gigabit ethernet switch

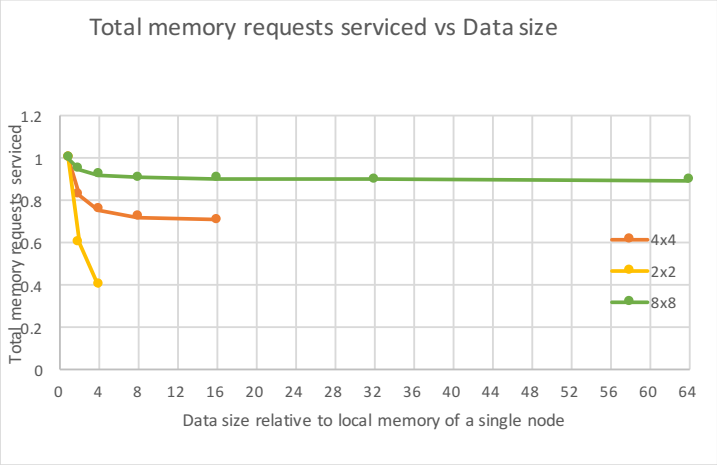


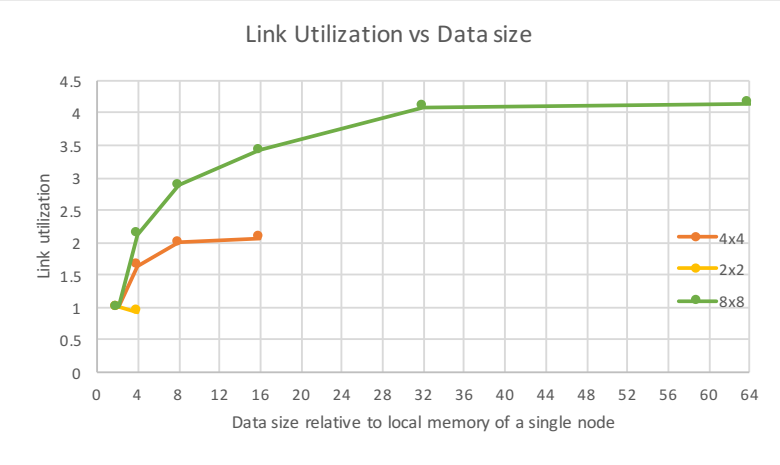
First FPGA Cluster Prototype

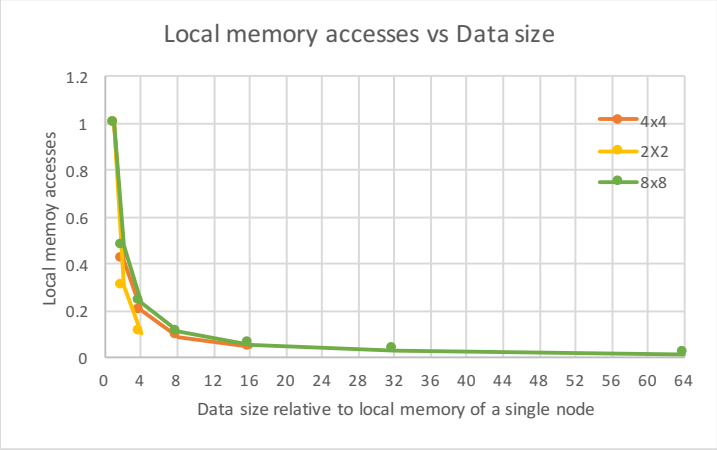
- Head server
  - Ubuntu 18.04
  - 9 Gbit network interfaces (8 to cluster, 1 to internet)
- 1500-2000 Watts
  - Will probably need bigger power supply
  - Might need some fans

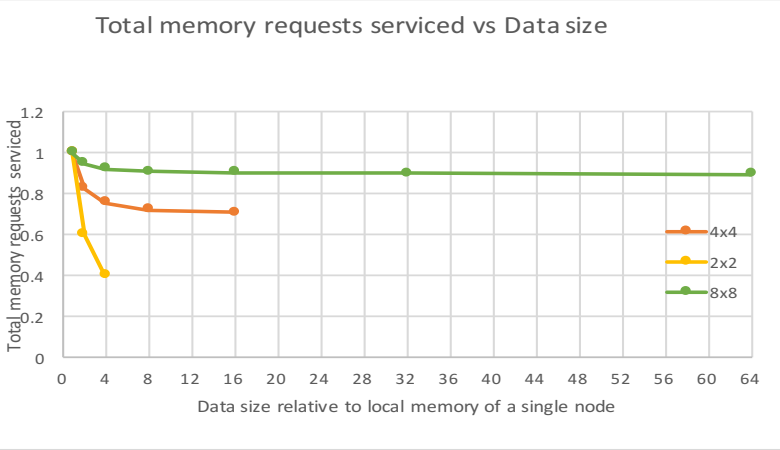


Scalability Study



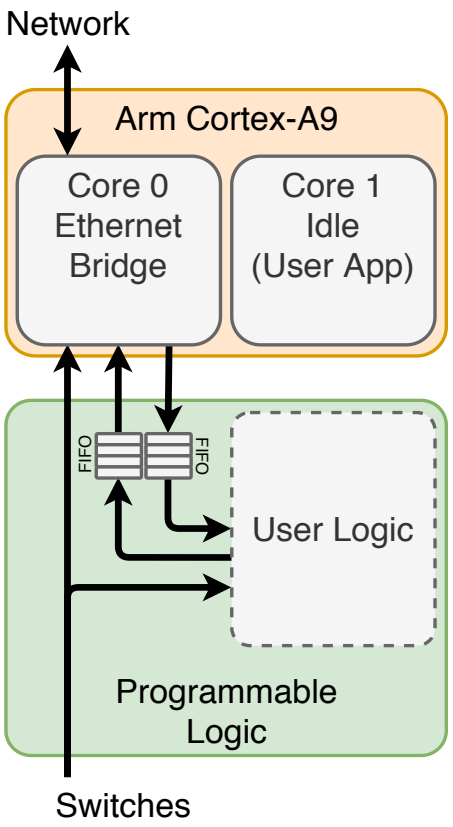






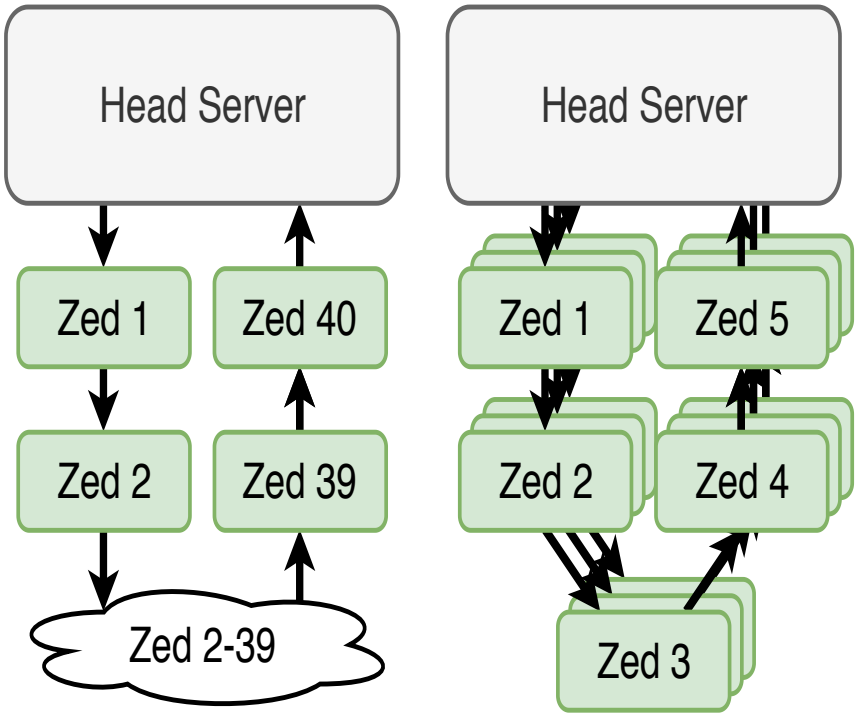
Ethernet Bridge

- Light Weight IP (LWIP) Library
  - Open source TCP/IP stack
  - Single thread
  - Bare metal
- Programmable Logic Interface
  - Streaming or Memory mapped
    - Choice will depend on application
  - Memory mapped good for writing directly to BRAM
  - Otherwise streaming will probably be necessary



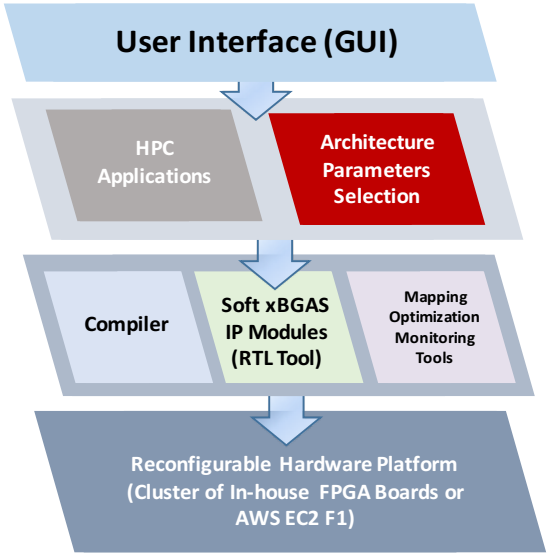
Benchmarking Ethernet Switch

- Ensure switch can sustain 1 Gbit/port
- Test NIC Bond on head server
- Two tests:
  - Loop of 40 zedboards
  - Eight loops of 5 zedboards
    - Stress NIC Bond



Next Phase of xBGAS

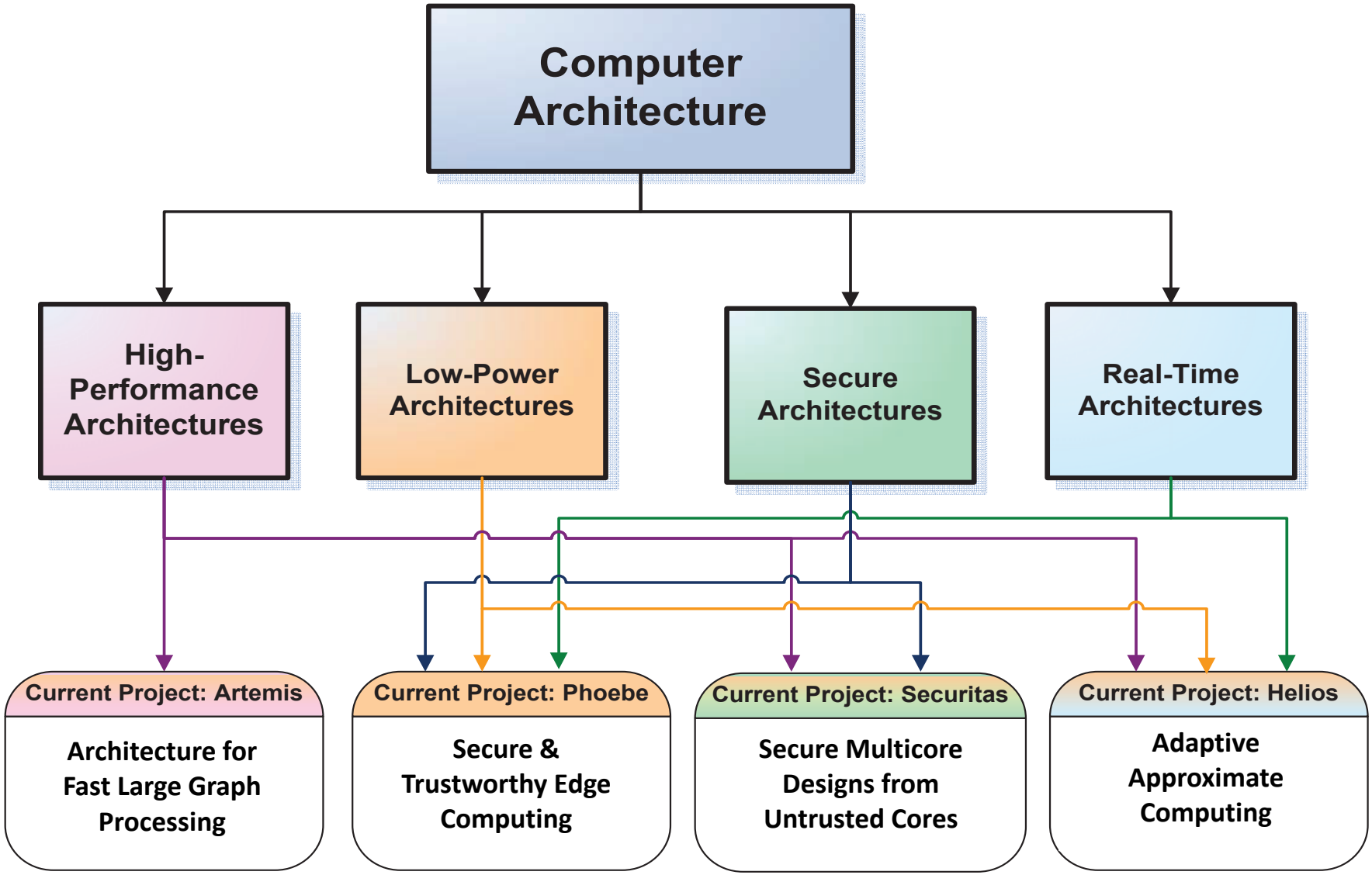
- Multi-FPGA scale prototyping
- Refinement of the software stack
- Operating system development work
- Security feature exploration and implementation
- Public Release
- Get application domain users to experiment with the multi-node version of the xBGAS architecture through an FPGA cluster or through the cloud
  - E.g., AWS Cloud-based FPGA compute environment



Acknowledgements

- Team Members
  - ASCS Lab Members
    - Boston University
  - John D. Leidel
    - Tactical Computing Laboratories, Muenster, Texas
  - Frank Conlon
    - Tactical Computing Laboratories, Muenster, Texas
  - Xi Wang
    - Texas Tech University, Lubbock, Texas
  - Yong Chen
    - Texas Tech University, Lubbock, Texas
  - David Donofrio
    - Lawrence Berkeley National Laboratory, Berkeley, California
  - Farzad Fatollahi-Fard
    - Lawrence Berkeley National Laboratory, Berkeley, California
  - Kurt Keville
    - MIT Lincoln Laboratory, Cambridge, Massachusetts

More Information at http://ascslab.org/



TEXAS TECH UNIVERSITY

TCL Tactical Computing Labs

Massachusetts Institute of Technology

BERKELEY LAB

BOSTON UNIVERSITY