UNCLASSIFIED

DECENTRALIZING THE PROCESSING UNIT:

DISTRIBUTING COMPUTE THROUGHOUT THE SYSTEM

David Donofrio, Chief Hardware Architect

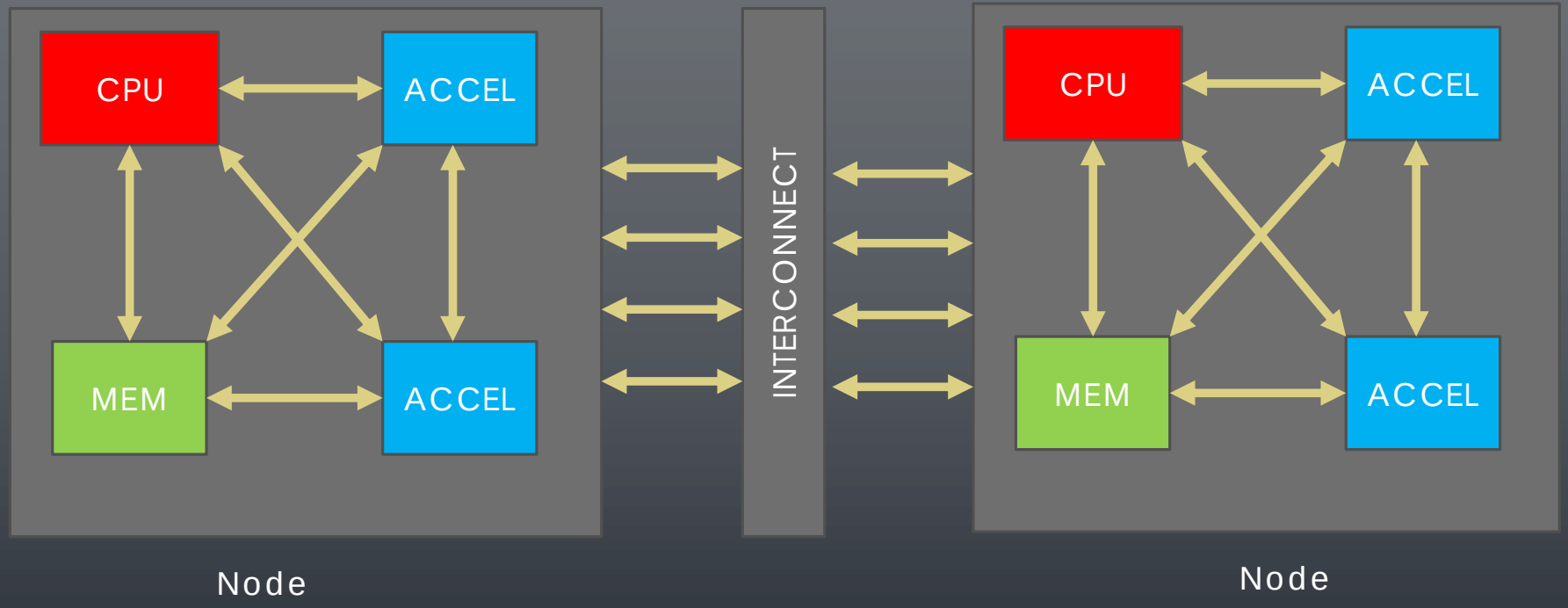
John D. Leidel, Chief Scientist

{ddonofrio,jleidel}@tactcomplabs.com

2020/28/10

UNCLASSIFIED

SYSTEMS TODAY



SPECIALIZATION AS THE PATH TO PERFORMANCE

- Process Scaling Tapering off
- Specialized architectures are everywhere
  - GPUs
  - AI Accelerators
  - Re-configurable
  - Matrix accelerators
- Increasing heterogeneity in systems
  - Heterogenous in both types of computation as well as where computing is performed
  - As computing becomes more decentralized we will need a greater variety of accelerators
  - This drives a need for frameworks that allow rapid design space exploration

A BIT OF HISTORY

- LBNL & TEXAS TECH WERE FUNDED TO DEVELOP A SPECIAL-PURPOSE, RISC-V BASED SOC
  - FPGA-BASED
  - HMC MEMORIES, CUSTOM ISA EXTENSIONS, SCRATCHPADS
- PHD DISSERTATION
  - GOBLINCORE-64: RISC-V BASED SOC FOR DATA INTENSIVE COMPUTING
  - ISA EXTENSIONS, PIPELINE CHANGES. SMT
  - NO CACHES, MEMORY COALESCING

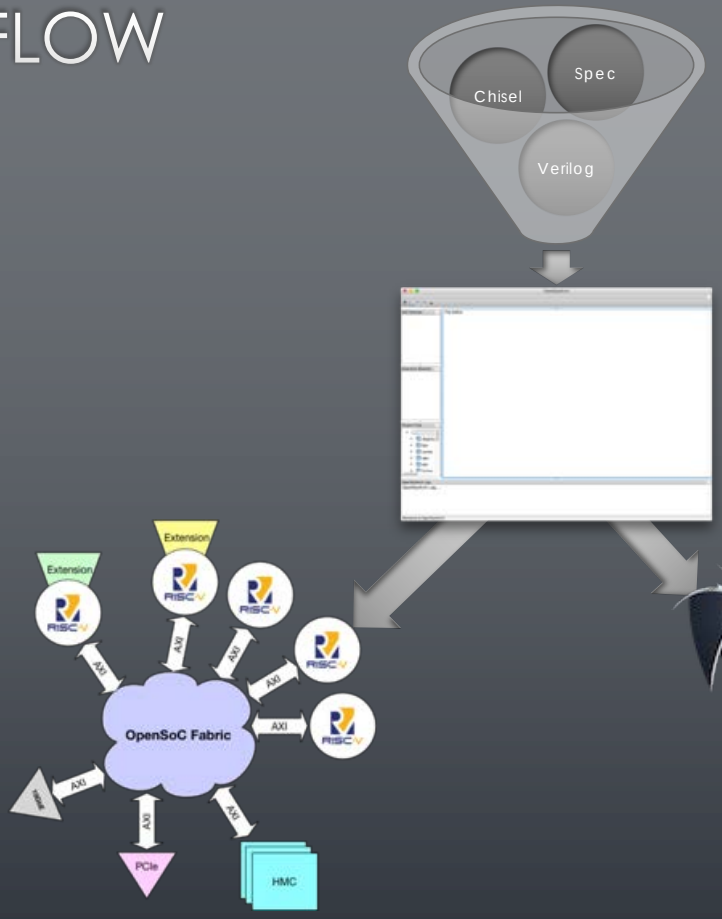
SYSTEM ARCHITECT GEN1

- THIS "RINSE AND REPEAT" DESIGN FLOW CAN BE AUTOMATED!
- SYSTEM ARCHITECT GEN1 WAS BORN AS A TOOLCHAIN FOR DEVELOPING RISC-V EXTENSIONS
- UTILIZED COMPILER-TECHNIQUES TO HANDLE DEPENDENCE ANALYSIS OF DESIGN PARAMETERS TO RISC-V INTERNALS

- **INPUT:** RISC-V CORE & DESIGN PARAMETERS
- **OUTPUT:** CHISEL RTL FOR ENTIRE SOC & LLVM COMPILER

SYSARCH TOOL CHAIN & FLOW

- OpenSoC Tool Chain & Flow consists of several integral moving parts:
  - OpenSoC System Architect Frontend
  - CoreGen IR & Backend
  - OpenSoC Fabric NoC Interconnect
  - RISC-V Standard Cores (in Chisel)
  - LLVM Compiler Infrastructure
- Users input design specs and extensions via frontend
- CoreGen ensures design correctness and continuity
  - Also generates backend Chisel and LLVM implementation
- OpenSoC Fabric "glues" all intermediate modules together with scalable NoC

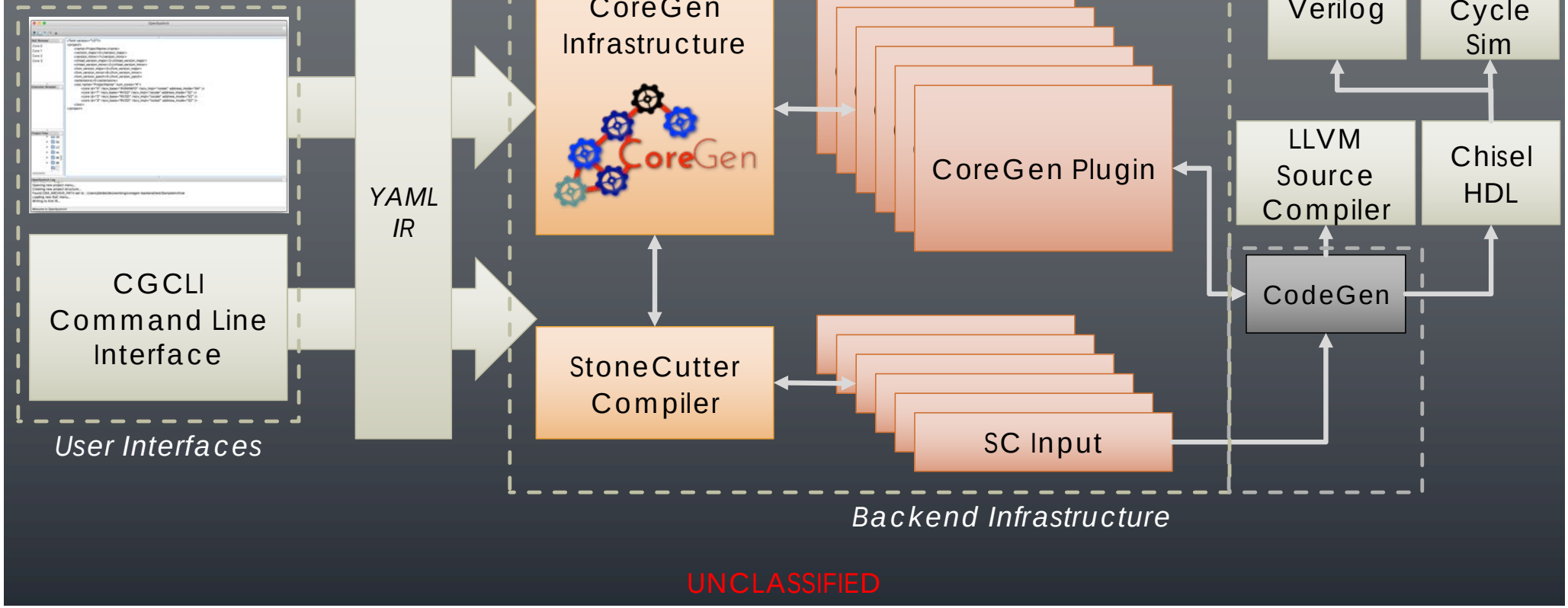


SYSTEM ARCHITECT GEN2

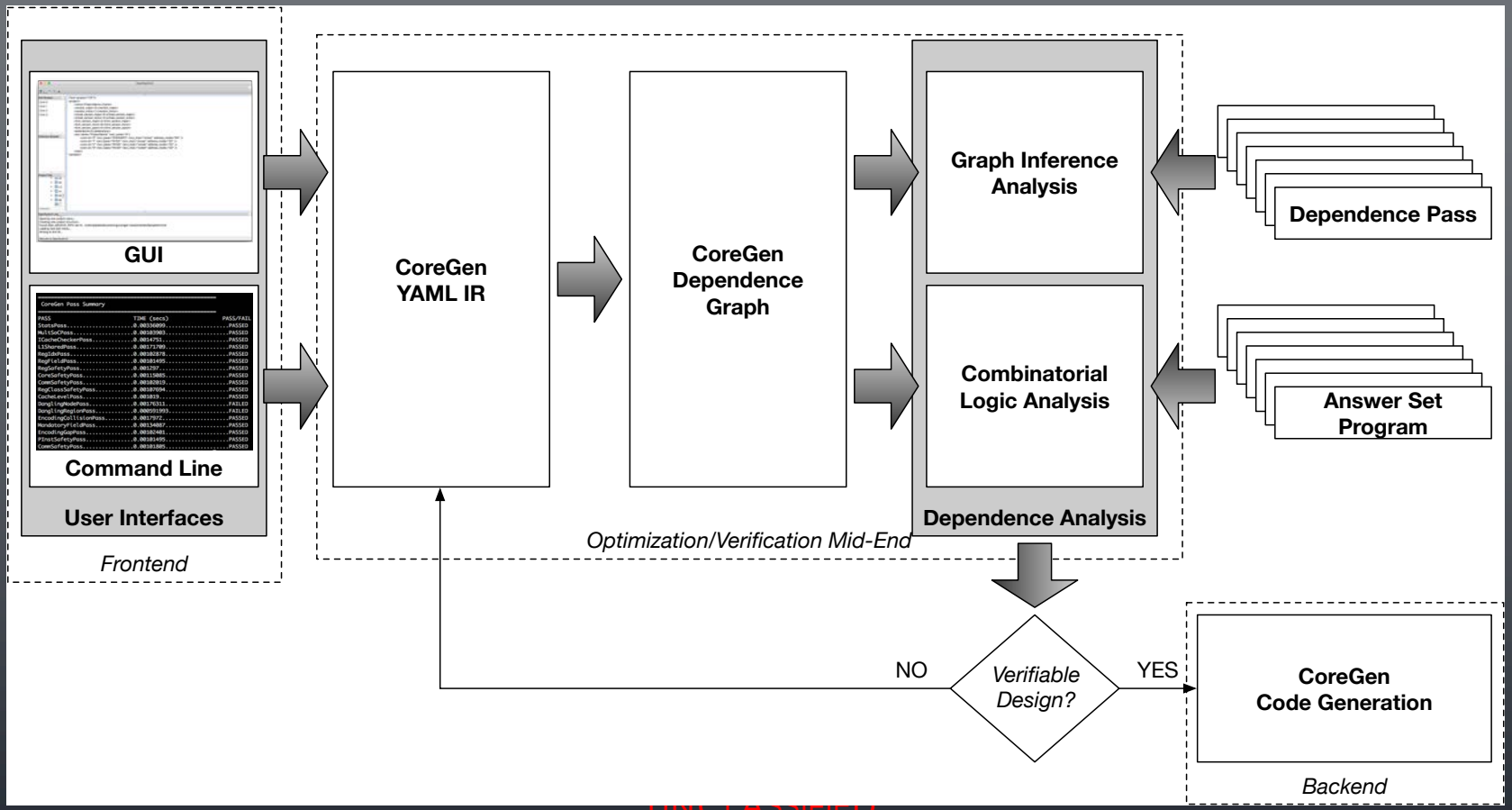
- Several early users posed an interesting question:
- ***What about developing arbitrary ISA's?***
- **Gen2 is a complete re-write of the infrastructure to support arbitrary ISA development**
  - New intermediate representation (formatted in YAML)
  - New dependence analysis engine (CoreGen)
  - High Level instruction definition language (StoneCutter)
  - Support for Plugins

Maintain verification performance similar to a modern optimizing compiler!

GEN2 HIGH LEVEL DESIGN

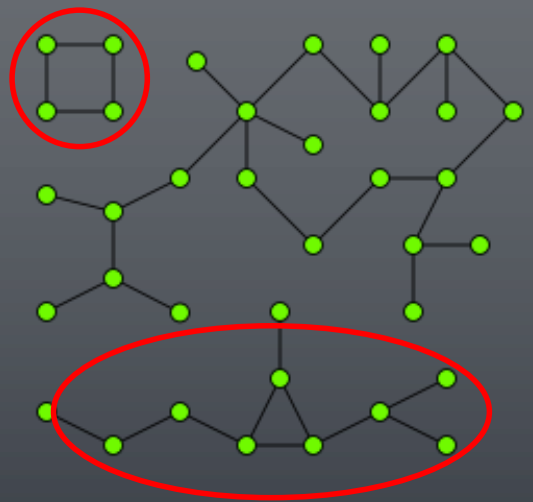


COREGEN VERIFICATION INFRASTRUCTURE



SYSTEM ARCHITECT GEN2: RAPID DESIGN GENERATION AND ANALYSIS

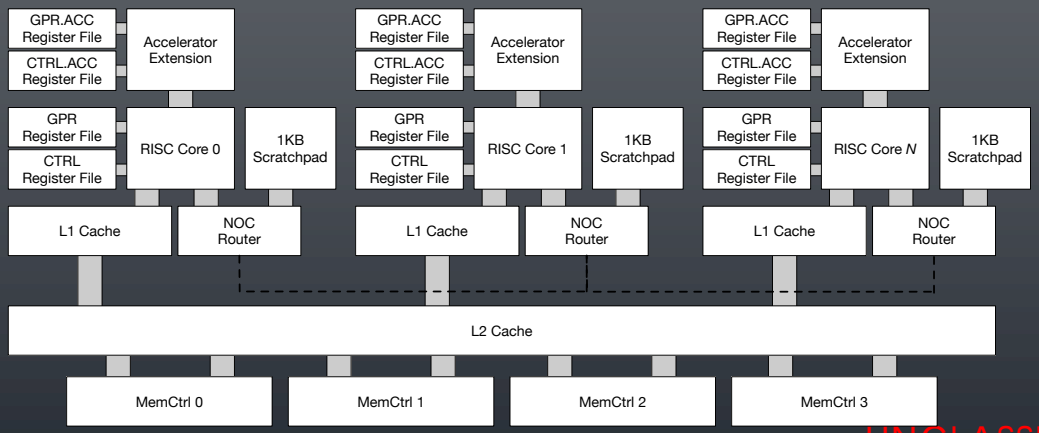
- Graph-Based dependence Analysis for design verification
- Hardware modules are defined as nodes in the DAG inside CoreGen
  - Dependence graph processed in multiple stages to expose connectivity, communication links, instruction and register encodings
- Multiple passes performed on the DAG for design verification
  - Elimination of dangling nodes / regions
    - Early detection to avoid issues with later synthesis of the design
  - ISA Encoding Collision Check
    - Early detection with decode logic and assembly/disassembly
  - Combinatorial logic analysis
  - User-defined passes coming soon

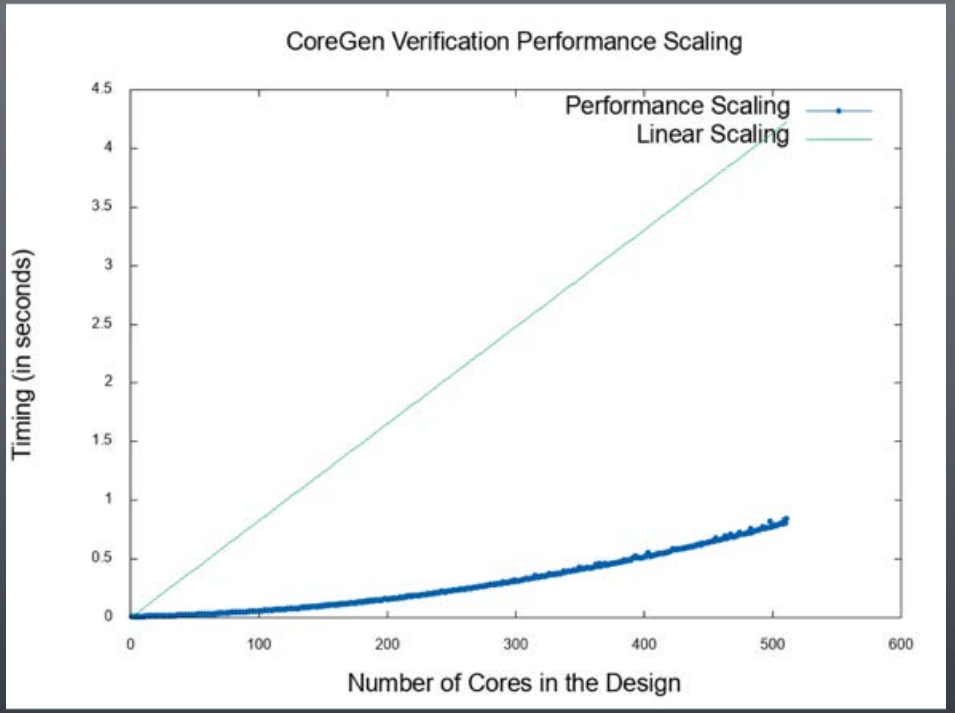


https://en.wikipedia.org/wiki/Component_(graph_theory)

SAMPLE PERFORMANCE

- HETEROGENEOUS, TILED DESIGN WITH MULTI-LEVEL CACHES AND NOC
  - MULTIPLE ISA'S WITH RISC CORES AND ATTACHED ACCELERATORS
- DESIGN SCALED FROM 1 TO 511 TILES
- **FULL VERIFICATION RUN OF 511 CORES < 1 SEC**





UNCLASSIFIED

PROGRAMMABLE ACCELERATOR NETWORKS: AMORTIZING HPC INTERCONNECTS

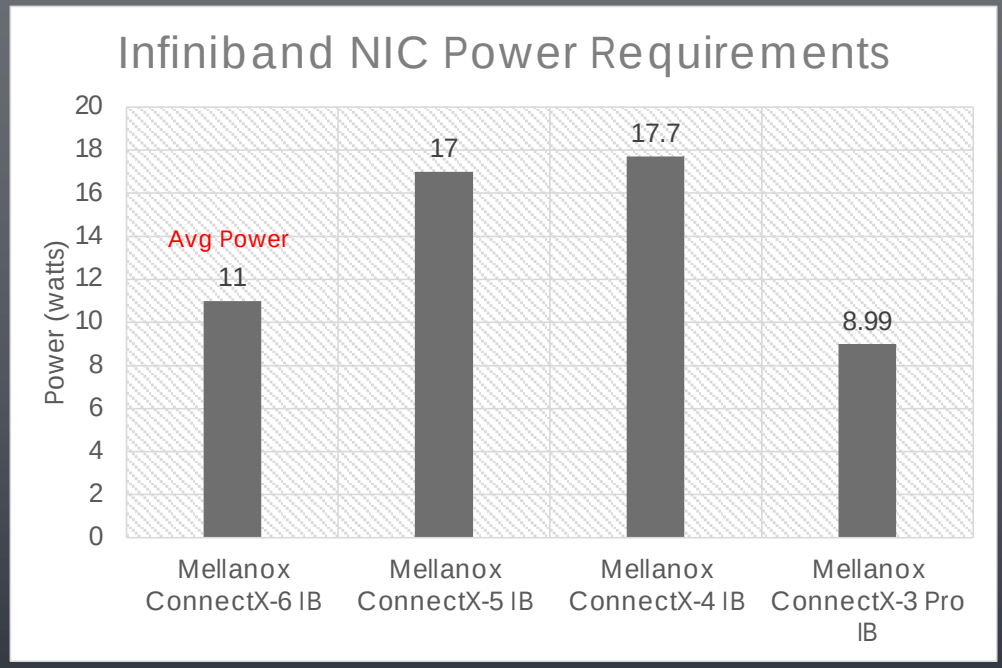
UNCLASSIFIED

PAN: PROGRAMMABLE ACCELERATOR NETWORK

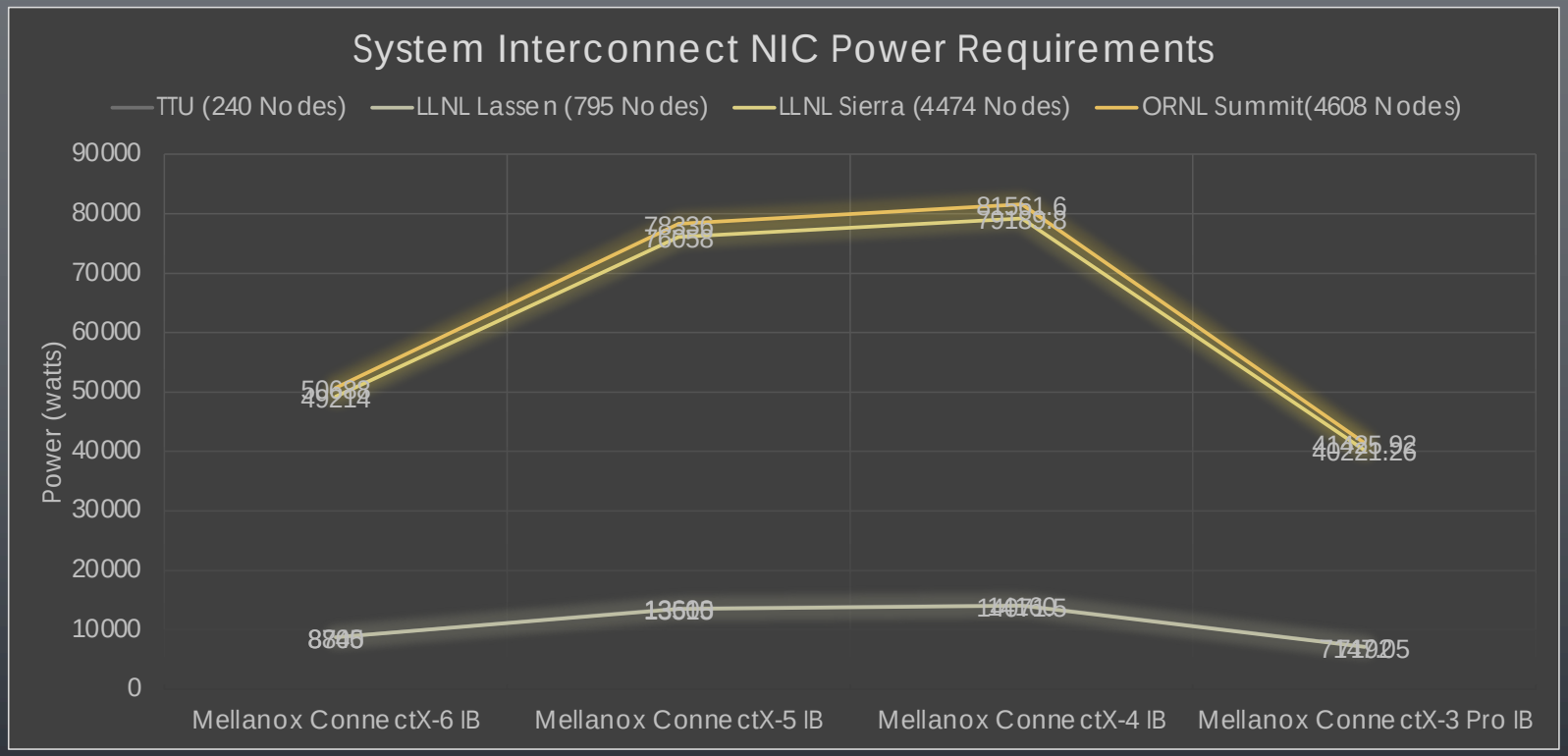
- Current HPC interconnects move significant data – and consume significant power
  - Opportunity to amortize the power and latency incurred in traversing the network by compute embedded in the interconnect
- PAN combines SDN concepts with HPC interconnects embedded with in-situ compute accelerators
- Goal is to support both traditional network operations as well as compute-in-network
- Accelerate applications with low compute efficiency and small messages by combining these operations into a single solution
  - Large scale sparse solvers and graph algorithms.

INFINIBAND POWER REQUIREMENTS

- MELLANOX INFINIBAND NIC POWER REQUIREMENTS
- DATA WAS OBTAINED FROM MELLANOX DOCUMENTATION
- ** CONNECTX-6 IS AVERAGE POWER
- ** CONNECTX-{3-5} IS PEAK POWER

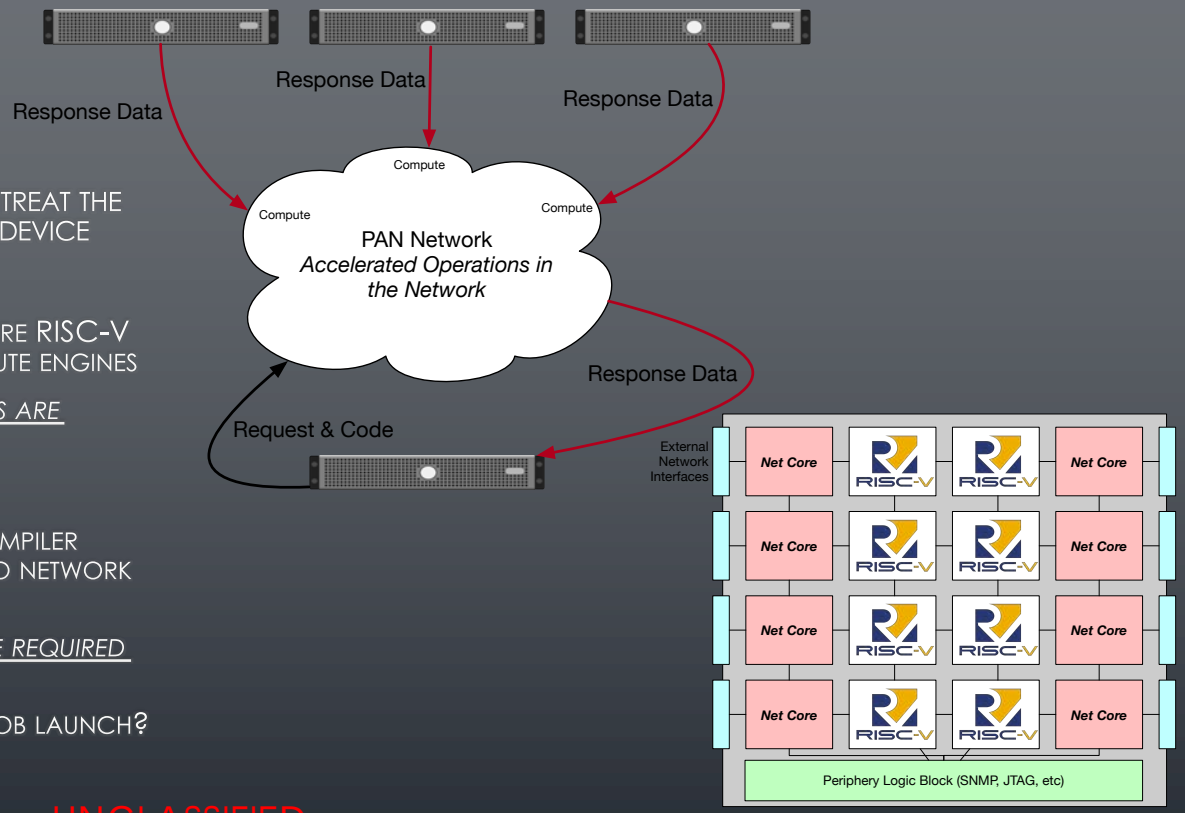


SCALABLE SYSTEM INTERCONNECT POWER



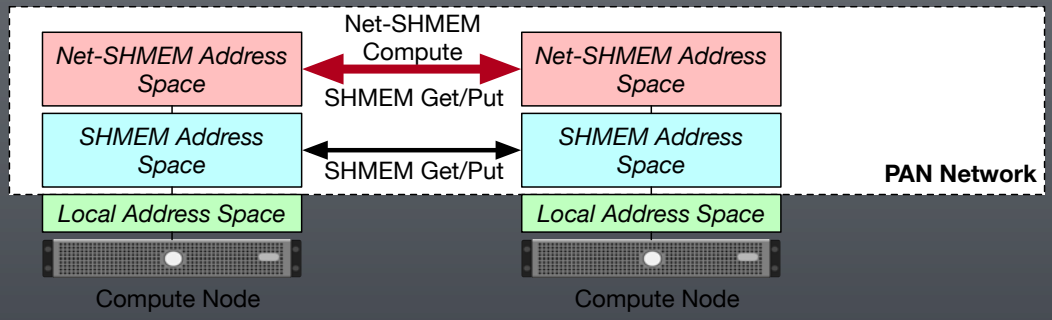
PROGRAMMABLE ACCELERATOR NETWORKS

- DEVELOP SIMULATION AND DEMONSTRATION HARDWARE/SOFTWARE IP TO PERMIT USERS TO TREAT THE INTERCONNECT AS A **COMPUTE** ACCELERATOR DEVICE
- HARDWARE IP:
  - COUPLE NETWORK BLOCKS TO ONE OR MORE RISC-V DEVICES TO SERVE AS THE NETWORK COMPUTE ENGINES
  - *WHAT ADDITIONAL ISA/UARCH EXTENSIONS ARE REQUIRED?*
- SOFTWARE IP:
  - DEVELOP PROGRAMMING MODEL AND COMPILER EXTENSIONS TO PERMIT SEAMLESS ACCESS TO NETWORK COMPUTE CORES
  - *WHAT ADDITIONAL UARCH EXTENSIONS ARE REQUIRED FOR SEAMLESS FUNCTIONALITY?*
  - DEBUGGING? CONTEXT SAVE/RESTORE? JOB LAUNCH?



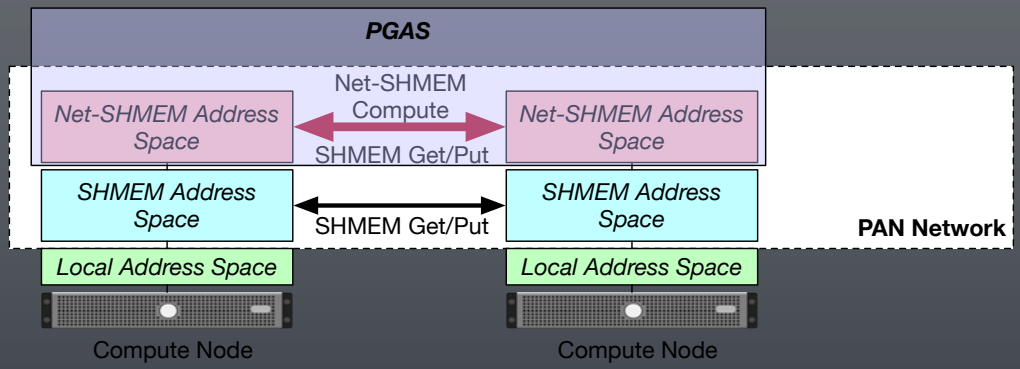
EXAMPLE PROGRAMMING MODEL: NET-SHMEM

- EXTENDS OPENSHMEM PROGRAMMING MODEL TO EXPOSE NETWORK MEMORY/COMPUTE SPACE
- THE NIC/SWITCH MEMORY IS EXPOSED VIA A SEPARATE NAMED ADDRESS SPACE
- UTILIZES RUDIMENTARY SHMEM OPERATIONS FOR DATA MOTION BETWEEN HOST/NETWORK SPACE AND BETWEEN NETWORK/NETWORK SPACE



EXAMPLE PROGRAMMING MODEL: PGAS

- EXTENDS OPENSHMEM, CHAPEL, UPC/UPC++ TO PROMOTE NETWORK MEMORY SPACE AS DIRECTLY ACCESSIBLE PGAS MEMORY
- DMA OPERATIONS FOR DATA MOTION BETWEEN HOST/NETWORK MEMORY SPACE
- EXTENDED LAUNCH MECHANISMS FOR DIRECT ACCESS TO NETWORK COMPUTE FACILITIES



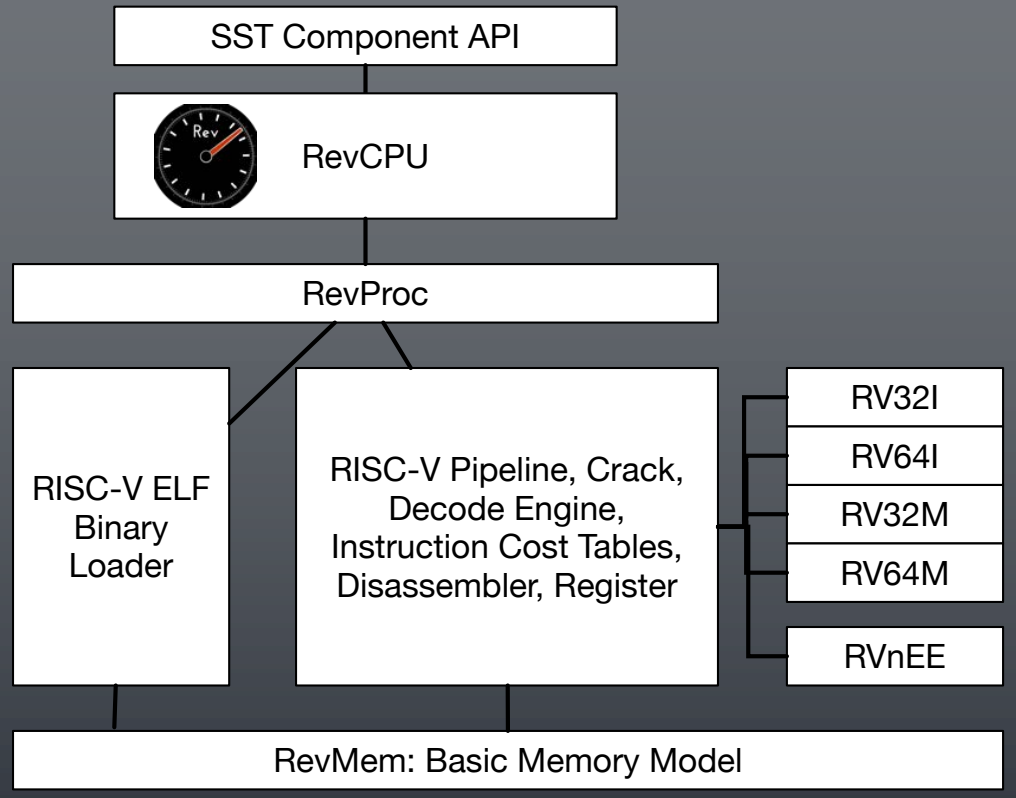
PAN SIMULATION INFRASTRUCTURE

REV SIMULATION INFRASTRUCTURE

- NATIVE SST COMPONENT THAT EXISTS OUTSIDE THE NORMAL SST TREE
  - DOES NOT REQUIRE ANY MODIFICATIONS TO SST
- TOP-LEVEL COMPONENT IS MAPPED TO A CPU DEVICE
- THE CPU (OR CPUS) IMPLEMENT SINGLE OR MULTI-CORE RISC-V DEVICES
- EACH "SOCKET" CAN CONTAIN HETEROGENEOUS CORES
- UNLIKE OTHER CPU MODELS, REV CONTAINS A **REAL** BINARY LOADER
  - REV RUNS APPLICATIONS! NOT JUST SIMPLE TESTS
  - IN-ORDER, CYCLE-BASED MODEL WITH CONFIGURABLE MEMORY LATENCY AND INSTRUCTION LATENCY
- CONTAINS A SIMPLE MERLIN NIC (REVNIC)

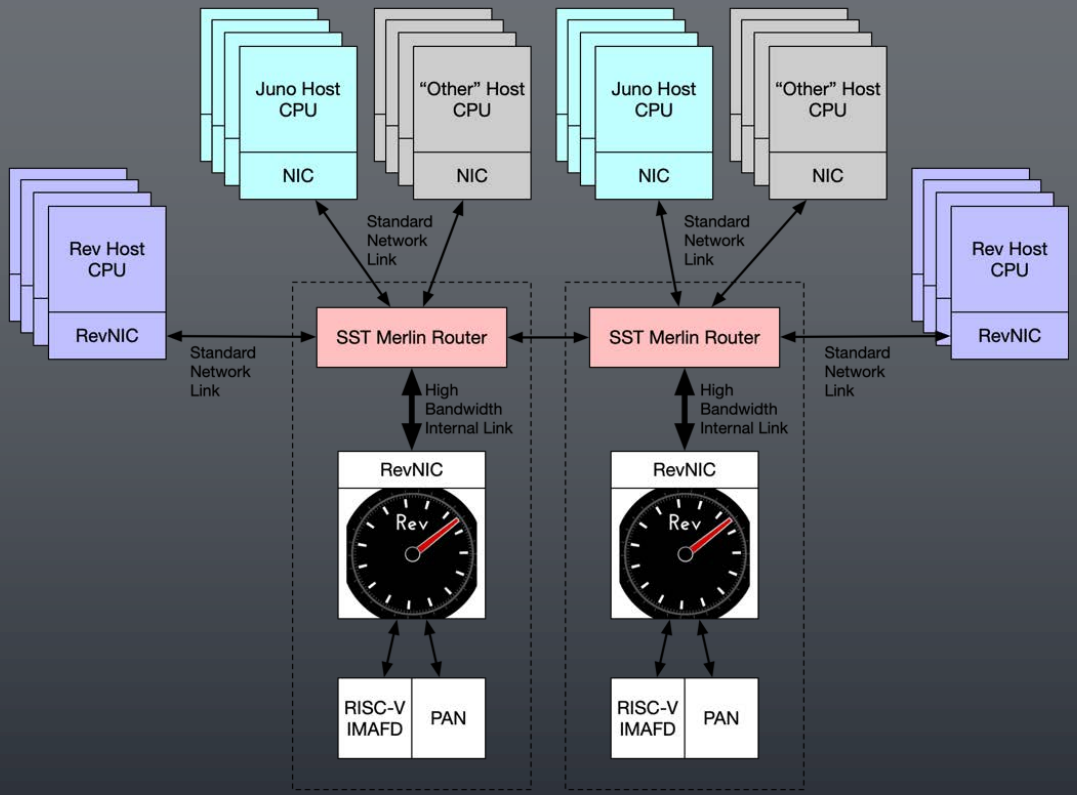
REV ARCHITECTURE

- CYCLE BASED CPU MODEL
  - CURRENTLY IN-ORDER ONLY
  - INSTRUCTION COST TABLES ARE CONFIGURABLE AT RUNTIME (TEXT FILE)
- EXECUTES NATIVE ELF BINARIES
  - NO CUSTOM COMPILER REQUIRED!
- SUPPORT FOR NEW INSTRUCTION EXTENSIONS WITH MINIMAL EFFORT
- CURRENTLY ALL INTEGER INSTRUCTIONS ARE COMPLETE
  - WORKING ON FLOATING POINT
- RELEASED UNDER APACHE 2.0 LICENSE
- HTTPS://GITHUB.COM/TACTCOMPLABS/REV



Tactical Computing Labs

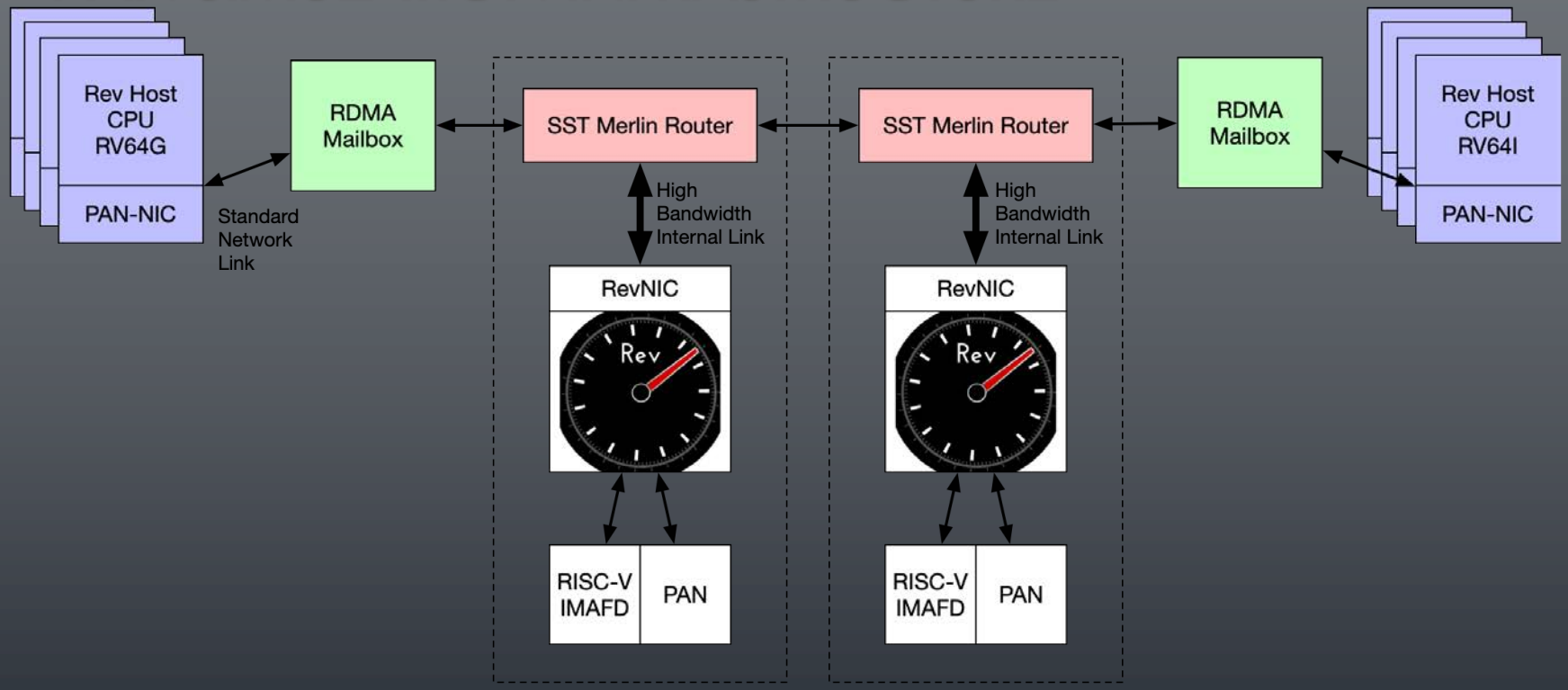
REV SIMULATION INFRASTRUCTURE



PAN SIMULATION INFRASTRUCTURE

- PAN EXTENDS THE BASE REV CPU COMPONENT WITH ADDITIONAL SUB-COMPONENTS
- PAN-NET (PAN-NIC) IMPLEMENTS THE PAN COMMAND PACKET INFRASTRUCTURE
  - THIS INCLUDES AN RDMA MAILBOX COMPONENT IMPLEMENTED AT THE CPU LEVEL
  - THE PAN-NIC DOES NOT REQUIRE THE PAN RISC-V ISA EXTENSIONS (VANILLA HOST, ACCELERATED NETWORK)
- PAN RISC-V EXTENSION: IMPLEMENTS "FUTURE" OPERATIONS
  - THE REV INFRASTRUCTURE STILL SUPPORTS HETEROGENEOUS OPERATIONS

PAN SIMULATION INFRASTRUCTURE



PAN COMMAND SET

- Simulator supports all the packet command types
- Synchronous/Asynchronous Gets/Puts
- Job Management
- Debugging

Packet	Packet Type	Opc Encoding
Synchronous Get	*Base*	0b00000000
Synchronous Put	*Base*	0b00000100
Asynchronous Get	*Base*	0b00001000
Asynchronous Put	*Base*	0b00001100
Synchronous Streaming Get	*Streaming*	0b0001
Synchronous Streaming Put	*Streaming*	0b0101
Asynchronous Streaming Get	*Streaming*	0b1001
Asynchronous Streaming Put	*Streaming*	0b1101
Execute	*Base*	0b00010000
Status	*Base*	0b00100000
Cancel	*Base*	0b00110000
Reserve	*Base*	0b01000000
Revoke	*Base*	0b01010000
Halt	*Base*	0b01100000
Resume	*Base*	0b01110000
Read Register	*Base*	0b10000000
Write Register	*Base*	0b10010000
Single Step	*Base*	0b10100000
Set Future	*Base*	0b10110000
Revoke Future	*Base*	0b11000000
Status Future	*Base*	0b11010000
Command Success	*Base*	0b11100000
Command Failure	*Base*	0b11110000

UNCLASSIFIED

DAVID DONOFRIO, CHIEF HARDWARE ARCHITECT

JOHN D. LEIDEL, CHIEF SCIENTIST

{DDONOFRIO,JLEIDEL}@TACTCOMPLABS.COM

UNCLASSIFIED