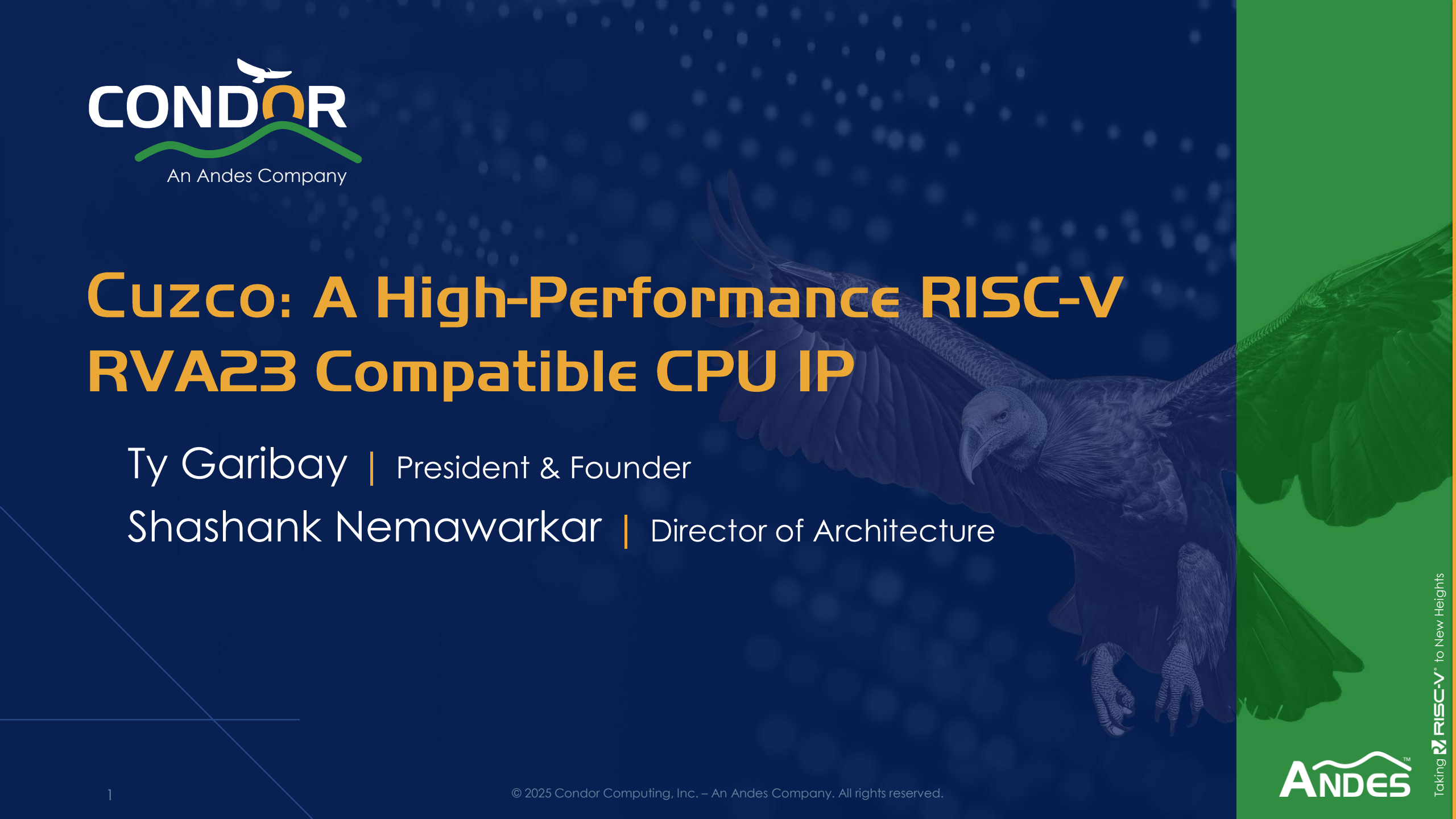
CONDOR

An Andes Company

Cuzco: A High-Performance RISC-V RVA23 Compatible CPU IP

Ty Garibay | President & Founder

Shashank Nemawarkar | Director of Architecture



Taking RISC-V to New Heights

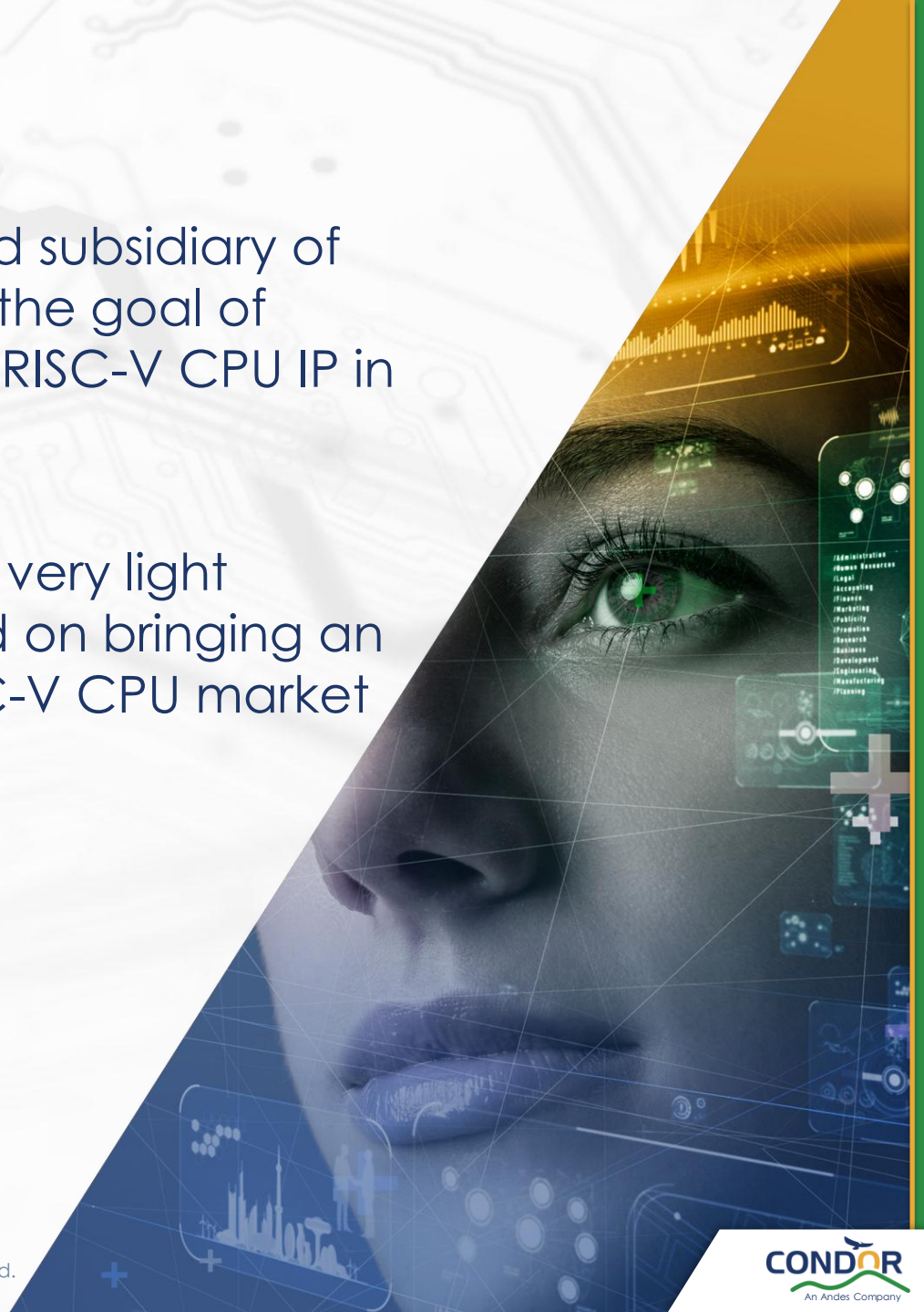
ANDES

Intro to Condor Computing

- Condor Computing, a wholly owned US-based subsidiary of Andes Technology, was founded in 2023 with the goal of creating the highest performance, licensable RISC-V CPU IP in the industry

- We are a tight knit team (~50 engineers), with very light management and overhead, entirely focused on bringing an innovative new micro-architecture to the RISC-V CPU market

- We intend to demonstrate that RISC-V can be competitive in any high-performance computing application, from datacenters to handsets to automotive

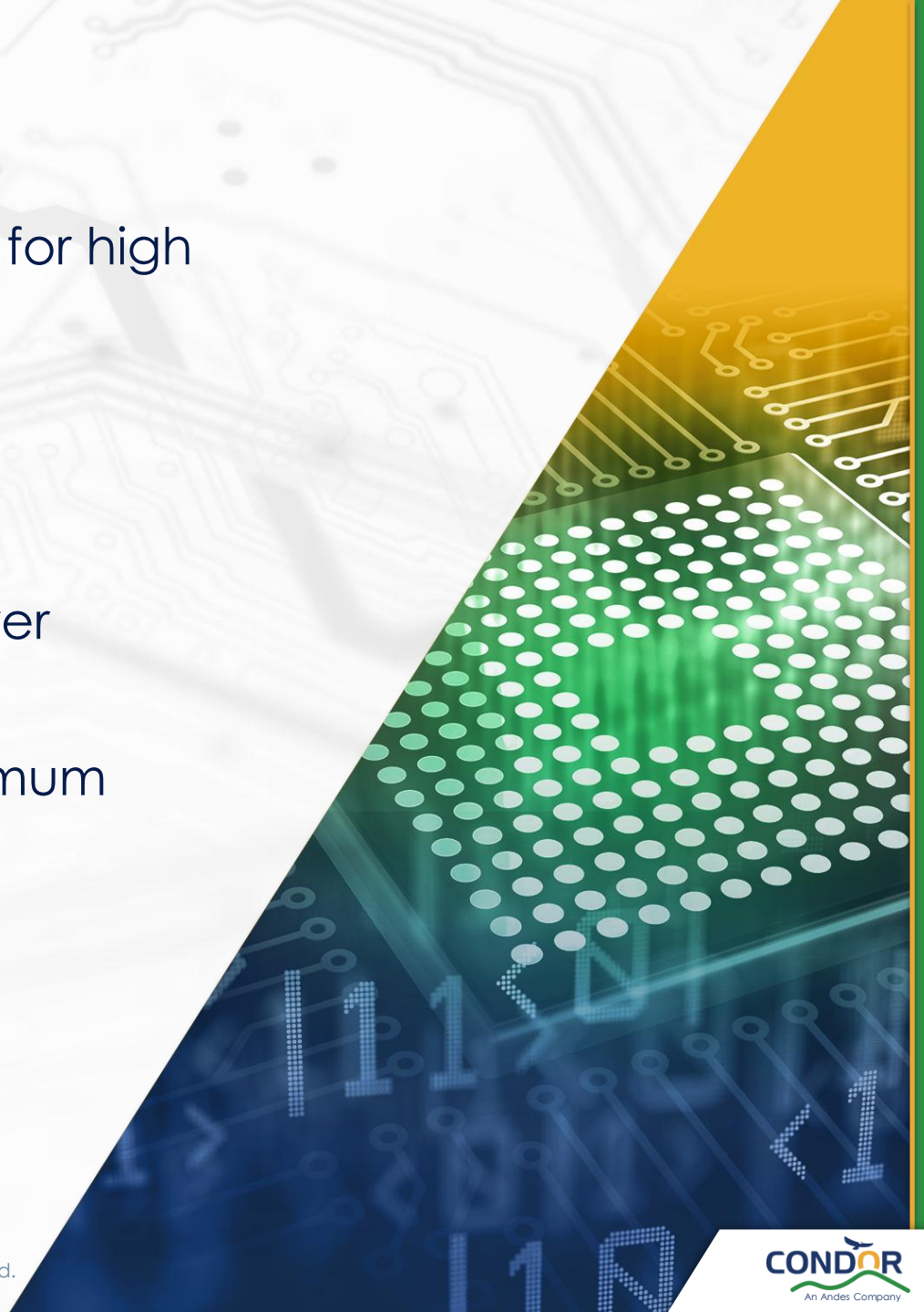


Taking RISC-V® to New Heights

CONDOR An Andes Company

Cuzco Processor Summary

- Innovative, time-based O-O-O core designed for high performance application processors
- Support for up to 8 cores with private L2$s in a coherent cluster with a shared L3$
- Much better performance than other high performance licensable CPUs with similar power consumption
- Latest RISC-V profile support (RVA23) for maximum software compatibility
- Full support for ISA customization

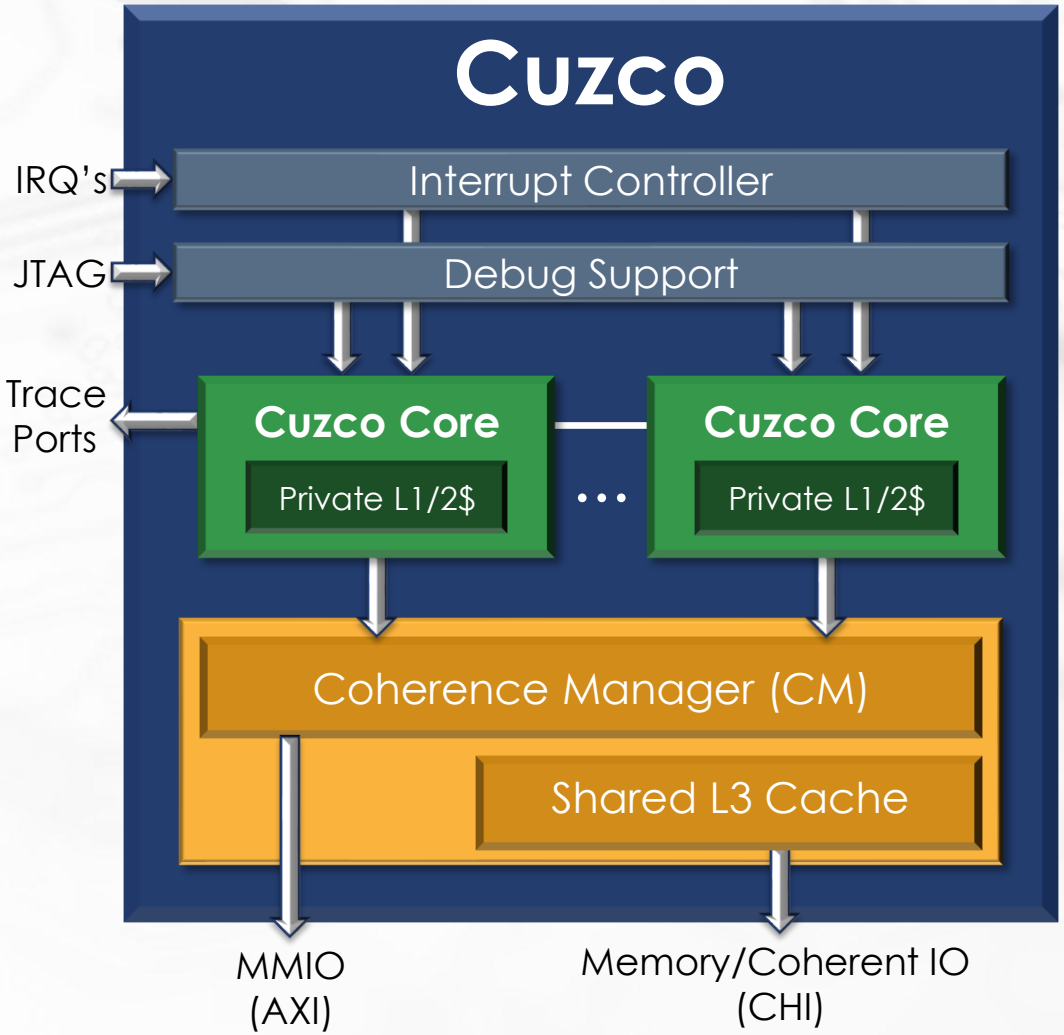


Taking RISC-V to New Heights

CONDOR An Andes Company

Cuzco Feature Overview

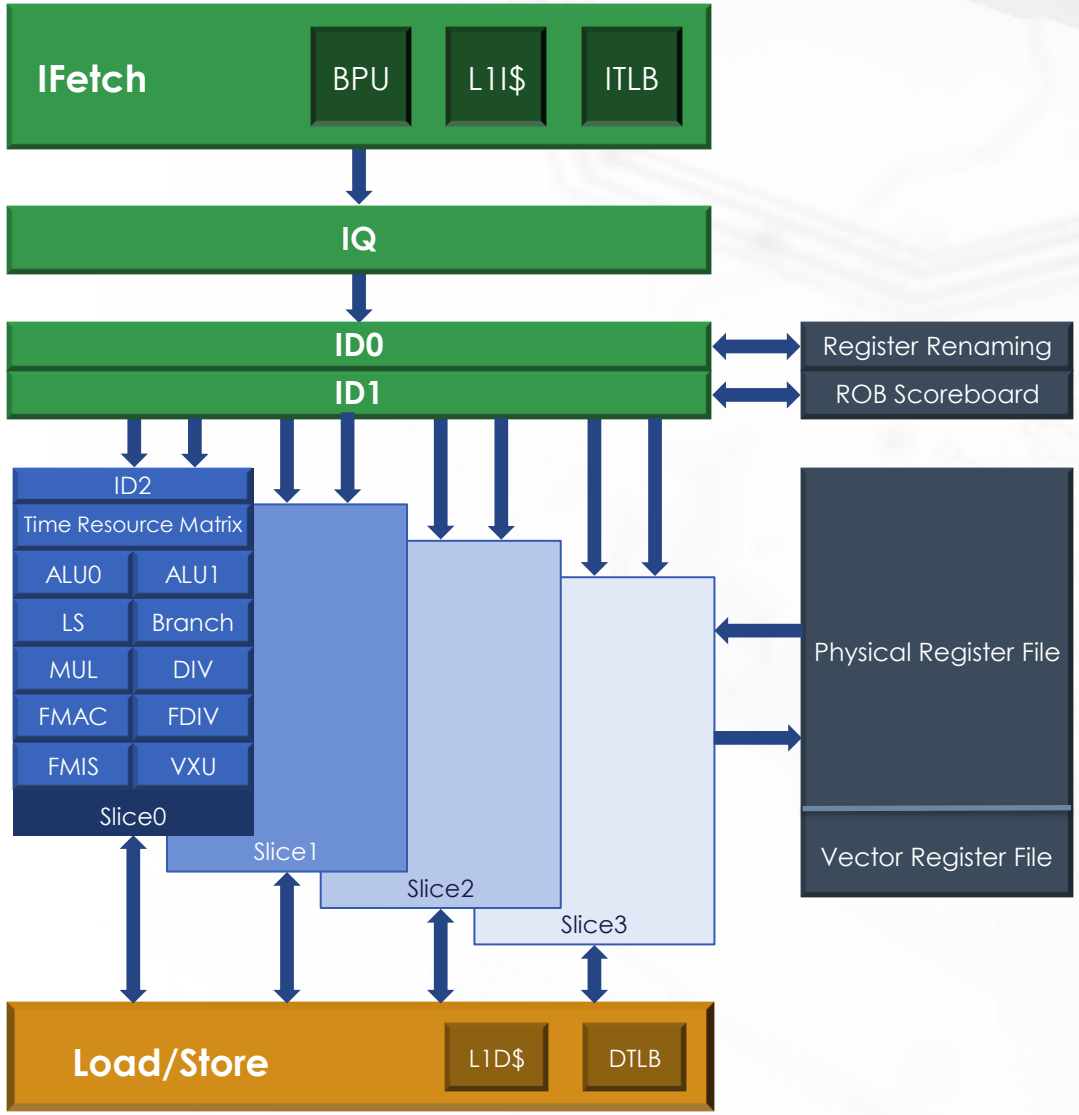
- 64-bit, RV64GCBKV* + CMO
- RVA23 profile compliant, with Hypervisor
- Innovative time-based microarchitecture
- 12-Stage Pipeline
- 8-Wide Frontend Decode
- 256-Entry Reorder Buffer (ROB)
- 8 Execution Pipelines
- RISC-V Vector 1.0 + Vector Crypto, 512b VLEN
- 2-Level Branch Target Buffer (BTB)
- TAGE-SC-L & Tournament Branch Predictor
- 1K/2K/4K 4-Way L2 TLBs
- 64 KB, 8-Way Private I/D Caches
- Up to 8 MB, Private L2$
- 8-Core Multiprocessor w/ Shared L3$ (up to 256MB)
- 256/512-bit CHI and 64b/512b MMIO Buses



Cuzco: CPU Core Features

- **RISC-V RV64GCBKV* and CMO**
  - FP: Half(Zfh), single(F) and double(D) precision
  - B: new instructions for bit manipulation
  - K: crypto functions such as SHA and AES
  - V & Zvk: Vector and vector crypto
  - CMO: cache management instructions
  - AndeStar 5.0
- **Innovative, time-based instruction issue microarchitecture**
- **12-stage, out-of-order execution pipeline**
- **256-entry reorder buffer (ROB)**
- **Vector: VLEN 256/512b**
  - 4 Exec Units
  - Merged Scalar-Vector pipelines
- **8 instruction decode/execute pipes**
  - 2 instructions per slice
  - 8 ALUs
  - 4 Branch units
  - 4 Load/Store units
  - 12 FPU units: 4x FMAC, FMIS, FDIV
- **Dynamic branch prediction**
  - TAGE-SC-L (TAgged GEometric history with Loop)
  - Tournament predictor
  - Indirect predictor
  - 2-level Branch Target Buffer (BTB), 8K entries
  - Global History Table (GHT), 16K entries
  - Return Address Stack (RAS), 32 entries
  - Up to 64 predicted branches outstanding
  - 10 cycle branch mispredict penalty

Cuzco CPU Core Block Diagram



Taking RISC-V to New Heights

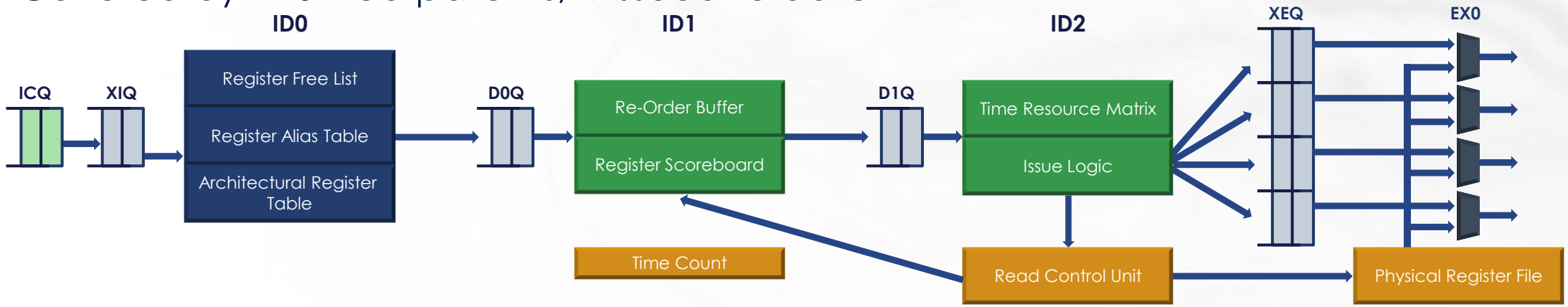
CONDOR An Andes Company

Cuzco Core Pipeline

IF0	IF1	IF2	IF3	ID0	ID1	ID2	EX0	EX1/DC1	DC2	DC3	DC4
Calculate next address & access iTLB	Access IC tag array, IC(hit/miss) BTB, and GHT, TAGE-SC-L	Access IC data array access, BTB hit/miss and taken/non-taken	Write IC cache line to ICQ & bypass/ read N instructions to XIQ	Read N instrs from XIQ, 1st instr decode, & access RFL & RAT	Access RSB to calculate execution times, 2x (read_time and read_time+ 1) for each instruction	Access TRM to issue instruction, secondary instruction decode, write issued instructions to XEQ	Read RF/fwd data, check RSB, send instr from XEQ to functional unit	Execute instrs, write result to PRF. AGU and access dTLB	Access DC tag array (DC hit/miss)	Access DC data array	Align and send load data to write back to PRF

Cuzco Time-Based Microarchitecture

- 1st CPU designed with hardware compilation for optimal instruction sequencing:
  - Register Scoreboard: record write time of an instruction to a register which becomes the read time of dependent instruction
  - Time Resource Matrix (TRM): busy indicators for read & write buses, and other resources
  - Issues instructions with precise predicted future times for execution
- Similar to a performance model with a perfect view of all past instructions to plan ahead for execution of instructions
- Targeting reduced gate count and power vs typical O-O-O machine mid-cores
- Covered by >10 filed patents, 4 issued to date



Time Resource Matrix (TRM)

Time	Scheduled		Read ports				Write ports		Br	ALU		LSU
83	O	P	A	A	J							
84	Q	R	L				J		A	J		
85	S	T	N				M		M			L
86	U	V	H	I	T		N			N		
87	W	X	K	K			H	I		H	I	T
88	Y		B	B			K			K		
89	a	b	C	C	D	D	B			B		
90	c	d	E	E	F	F	D		C	D		
91	e	f	G	G	P		F		E	F		
92	g	h	U						G			P
93	i	j	O	O	V	c						U
94	k	l	X				O	c		O	c	V

- Instructions are efficiently scheduled in the TRM, after all required inputs operands are predicted to become available by Scoreboard, taking advantage of all available resources
- In this example snippet from Dhrystone, instruction O is scheduled in cycle 83, issued in cycle 93, and executed in cycle 94

Why Time-Based Microarchitecture for Cuzco?

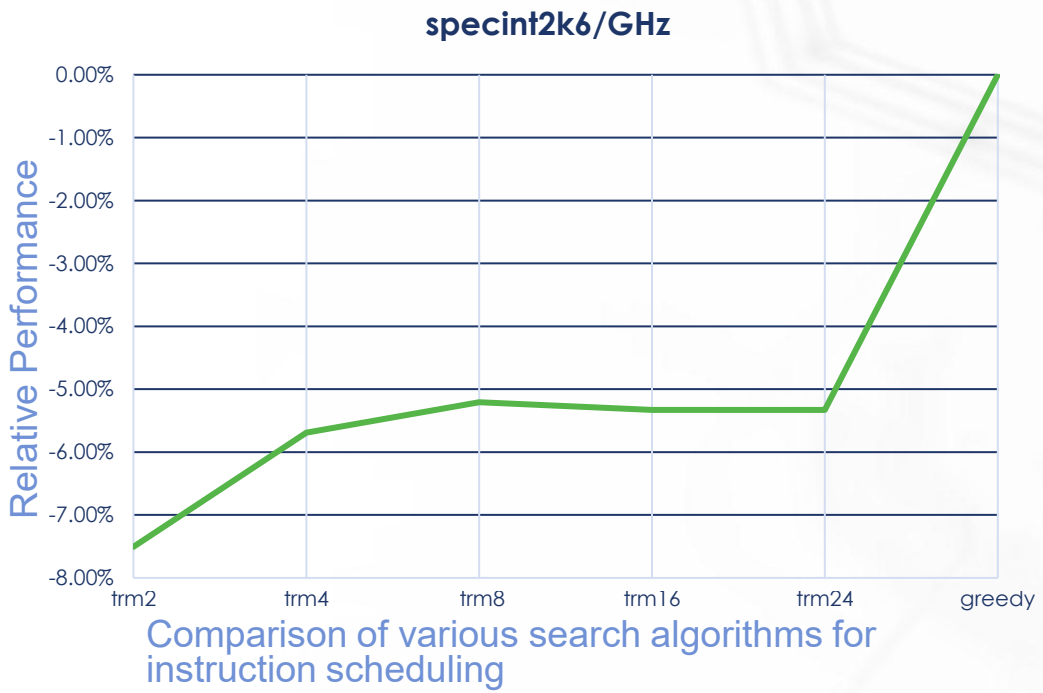
- Well-researched in compilers and deterministic compute
  - Certainty of scheduling reduces logic complexity for wide machines
  - Removes complex runtime per-cycle scheduling reducing dynamic power
- Q: How to handle dynamic behavior of general-purpose compute?
  - What are the different dynamic program execution behaviors encountered?
  - How frequent are they?
  - How to address them at runtime?

Cuzco performance model and design provides a direction to this microarchitecture approach

Detailed Performance Modeling

- Starting point: Open-source Sparta and Olympia
  - RISC-V SIG Perf Modeling
- Active Condor contributions:
  - Dromajo and now Spike-based trace and simpoint (for up-to-date R-V Spec)
  - FSL: Fusion Spec Language
  - Guidance for Olympia development including supervising interns
- Extended internally for Cuzco modeling
  - Typical design intricacies and abstractions
  - Scheduling: Best case and realistic flavors of TRM

Pros of Time-Based Microarchitecture



Comparison of various search algorithms for instruction scheduling

- Build execution schedule with known/projected dependencies and latencies:
  - Activate operands, resources, and execution units only at scheduled times
  - No search or prioritized selection at reservation stations
  - Reduced scheduler complexity (logic and area)
  - Dynamic power reduction vs conventional scheduler
- Two instructions scheduled/slice/cycle, scalable with slices
  - TRM2: Identify functional unit slots within 2 cycles of operand availability
  - TRM8: Within the next 8 cycles

Dealing With Dynamic Behaviors

- TRM assumes optimal latencies and dependences, and adds dynamic runtime updates
  - Core: Variable exec latencies, misprediction initiated flushes
  - Memory: Load data L1 cache and TLB misses, bank conflicts
    - Dynamic memory response: L2/L3/Mem latency prediction
    - Load/Store RAW hazard prediction

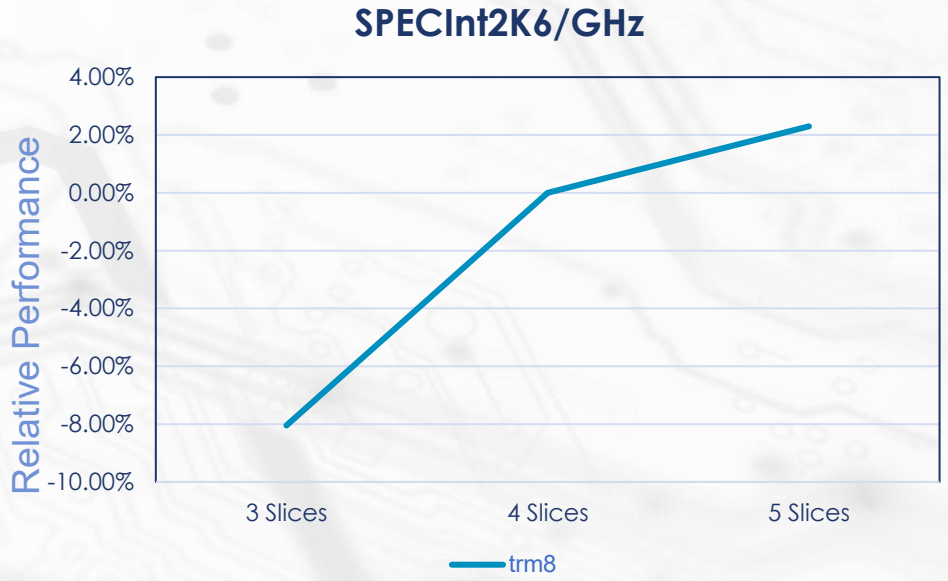
Metric/Config	Default	Mem Dep Predictor	Reg based Predictor	No RAW hazards
Specint2k6/GHz (%diff)	100%	-1.0%	+1.4%	+4.2%
RAW Flush PKI	1.2	0.97	0.68	0

- Rescheduling of dependency chains
  - Instructions (L/S) will replay until successful completion (dynamic exec power)
  - Reschedule (ALU) can result in wasted issue slots (scheduling power)

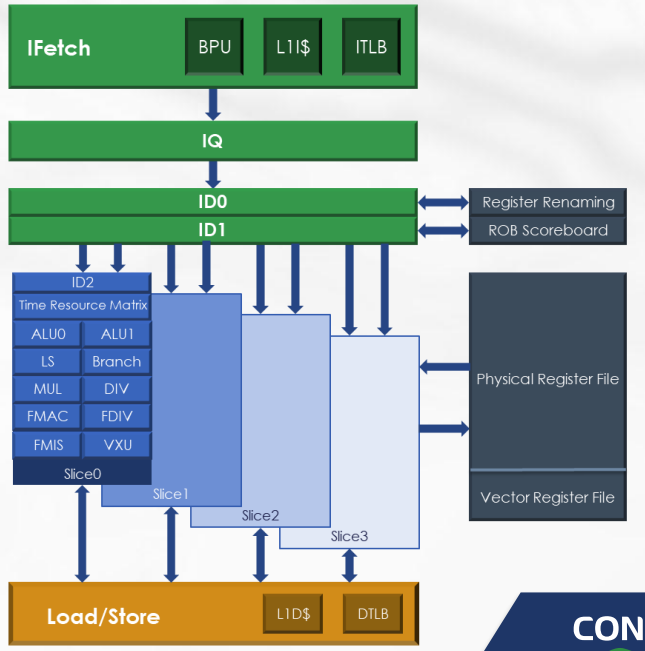
Config	trm2	trm4	trm8	greedy
Replays in PKI	68.12	68.75	70.07	99.74
Replays (%diff)	-31.7%	-31.0%	-29.4%	0%

Slice-based Microarchitecture

- Baseline: 4 slices, 2 instruction pipelines/slice
- Uniformly scalable design IP
  - Each slice implements a fully compatible RISC-V CPU
  - Each slice adds symmetric set of resources to the machine
- Static PPA control through IP configuration
- Dynamic power vs. throughput control with slice-based enables and scheduling



Relative performance of different sized configs



Cuzco Memory System Microarchitecture

Private L1 Caches

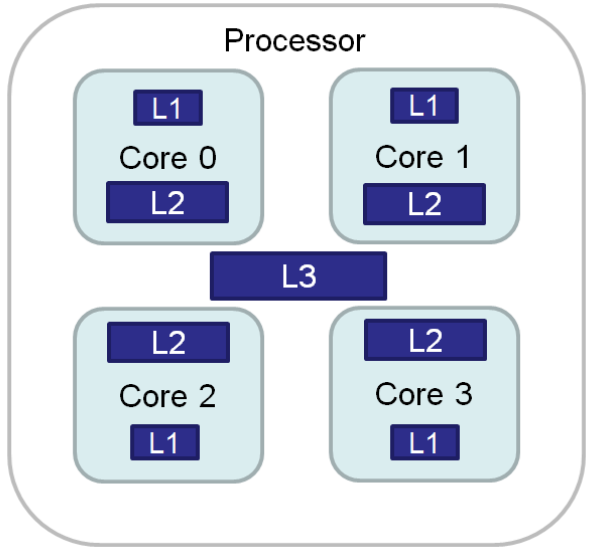
- 64KB I$/D$, 64B line size, 8-way, pLRU
- PIPT (Physical Index and Physical Tag)
- 64-byte cache line size
- I$/D$ prefetch and D$ writearound
- SECDED ECC error protection
- Up to 64 pending miss requests
- 4 cycle load->use penalty

Private L2 Cache

- Up to 8MB, 64B line size, up to 16-way, pseudo-random replacement
- I/D prefetch
  - Preset and configurable prefetch policies
- Configurable multi-cycle SRAM accesses
- SECDED ECC error protection
- +14 cycle delay on L2 hit

Privilege and Memory Management

- Machine (M), hypervisor (H), supervisor (S) and user (U) privilege modes
- Memory management unit (MMU)
  - Bare, Sv39, Sv48, and Sv57 VA translations
  - Svnapot, Svpbmt, Svinval VM extensions
  - L1 I/D TLBs: 64-entry, fully associative
  - L2 TLB: up to 4K-entry, 4-way
  - PMP and ePMP support with 16 PMP entries
- 16 PMA regions



Cuzco: Cluster Microarchitecture

Shared L3 Cache

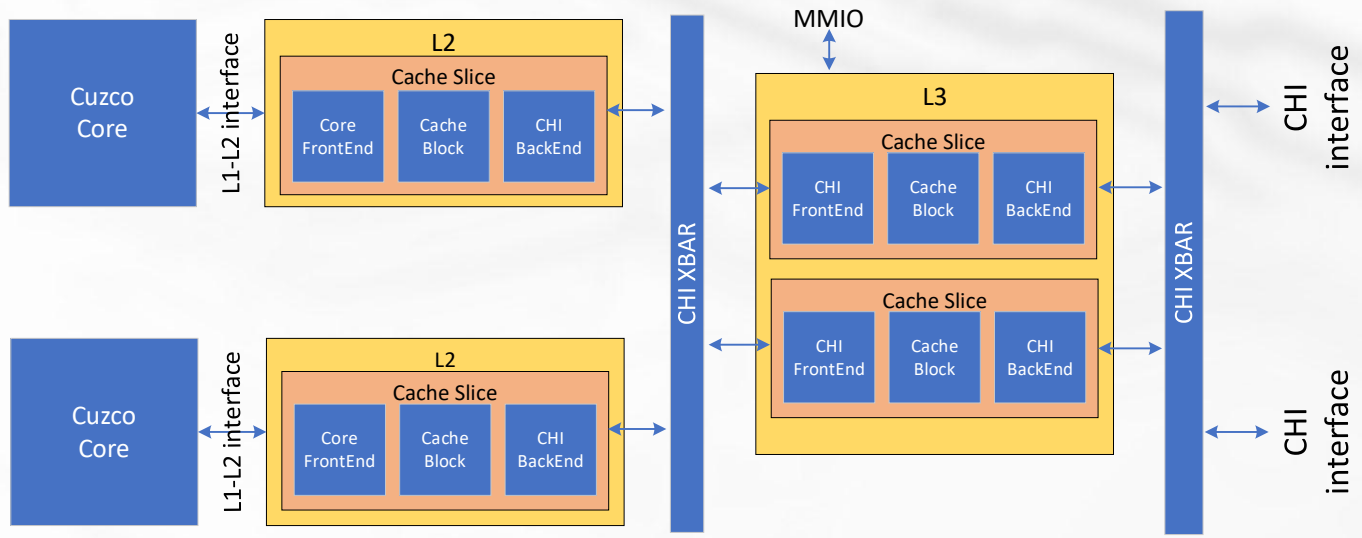
- Up to 256MB, 64B line size, up to 16-way, pseudo-random replacement
- I & D prefetch, configurable policies
- Max outstanding reads/writes (cacheable and uncacheable): 32-128
- Max 64 outstanding snoop transactions
- Configurable multi-cycle SRAM accesses
- SECDED ECC error protection

Cluster with Multicore Cache Coherency

- Up to 8 cores+L2$s in a cluster
- Coherence Manager and L3$

Bus Interfaces

- 512-bit main memory CHI bus interface
- 256/512-bit memory mapped I/O (MMIO) interface
- Core+L2 vs. external-bus clock
- Asynchronous, and Synchronous N:1 clock ratios



Cuzco: Software

- Industry standard RISC-V ISA
  - RV64GCBKV* + CMO
  - RVA23 profile
  - Diverse and growing software eco-system
- Complete and stable RISC-V tool chain, provided by an industry leader, Andes Technology, with years of proven delivery
  - Compiler based on GCC & LLVM
  - Eclipse-based IDE
  - FPGA development boards
  - RTOSes:
    - Open-source: FreeRTOS, Zephyr
    - Commercial: ThreadX, µC/OS-II
  - Linux SMP kernel and platform drivers
- Support for adding custom instructions

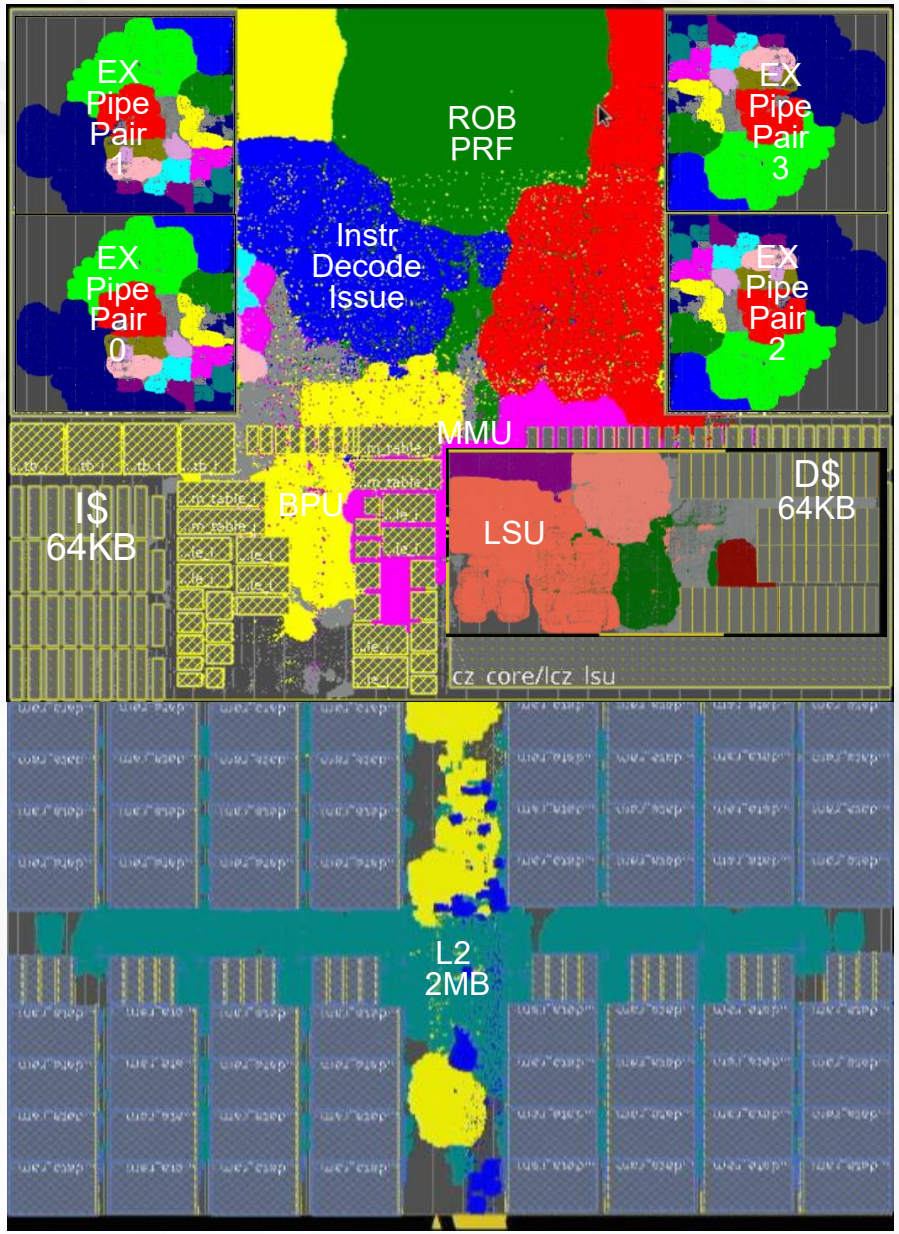
Comparison of Andes O-O-O Processors

CPU Cores	Andes AX65	Cuzco
Pipeline stages	13	12
Issue width	4	8
ROB	128	256
Branch prediction	TAGE-L	TAGE-SC-L-IT
Int ALU & FPU Load/Store Unit	4 & 2 2 L/S	8 & 4 4 L/S
L1 Caches	I$: 64KB D$: 64KB	I$: 64KB D$: 64KB
L2 Cache	8MB Shared	Up to 8MB Private
L3 Cache (Shared)	None	Up to 256MB
DMIPS/MHz	4.82	8.5
SPECint2k6/GHz	8.78*	>17.5*

**with 8MB L2 and no vector ISA usage*

Cuzco Implementation

- CPU configuration
  - 8 execution pipelines
  - 64KB I/D caches
  - 1.3mm x 1.0mm in N5
  - Target 2GHz/SS, 2.5GHz/TT
  - 3.5M stdcells
  - 7M gate equivalents
- L2 configuration
  - Private L2
  - 2MB example
  - 1.3mm x 0.8mm
  - 1.5M stdcells for 2MB config
  - 4.5M gate equivalents



Cuzco: Benefits Overview

- **Cost reduction:** Better performance / $ of die size
- **Lower power:** Better performance / µW of power
- **Fully MP-ready:** up to 8 HPC CPU cores per cluster
- **Industry standard RISC-V ISA Compliant**
  - RV64GCBKV* + CMO
  - RVA23 profile
  - Extensible
- **Industry standard toolchain and development environment**

CONDOR

An Andes Company

Thank you!

www.condorcomputing.com

Taking RISC-V to New Heights

ANDES