XiangShan: An Open-Source Project for High-Performance RISC-V Processors Meeting Industrial-Grade Standards

Kaifan Wang[1,2], Jian Chen[3], Yinan Xu[1,2], Zihao Yu[1], Zifei Zhang[1,2], Guokai Chen[1,2], Xuan Hu[1,2], Linjuan Zhang[1,2], Xi Chen[1,2], Wei He[3], Dan Tang[1,3], Ninghui Sun[1,2] and Yungang Bao[1,2]

[1] *Institute of Computing Technology, Chinese Academy of Sciences*

[2] *University of Chinese Academy of Sciences*

[3] *Beijing Institute of Open-Source Chip*

August 2024

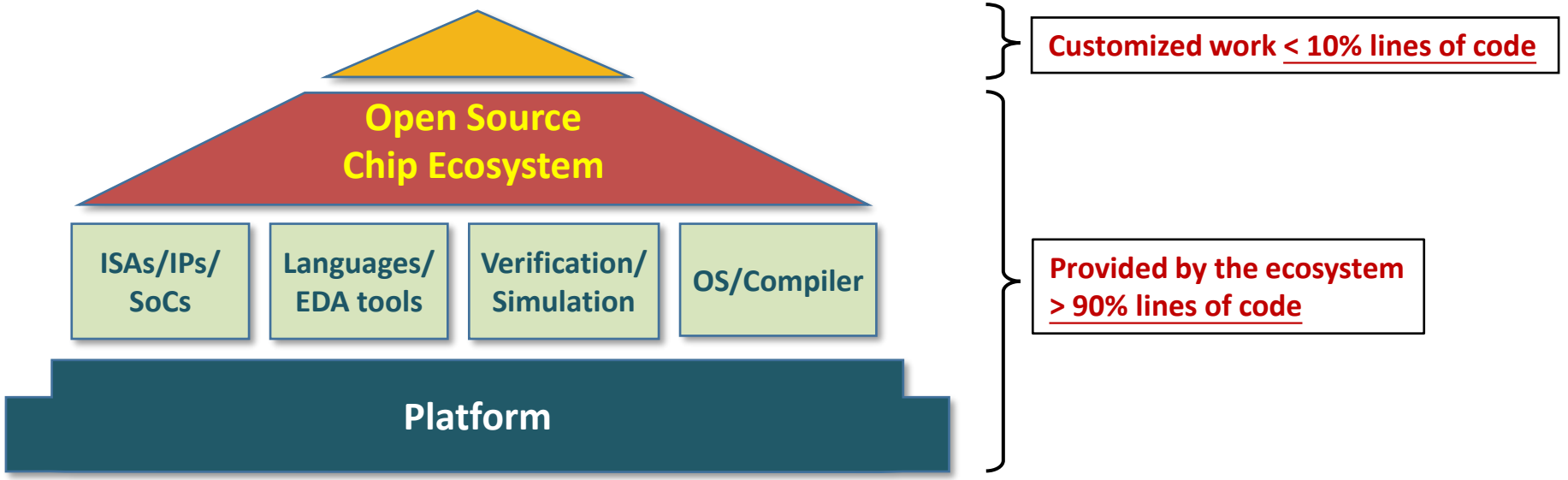
HOT CHIPS

Outline

- Overview
- Microarchitecture design
- Agile development platform
- Applications in industry & academia
- Summary

Open-Source Chip Ecosystem (OSCE)

- **Empowered by RISC-V**
- **Lower the barrier to chip development**: reduce time-to-market and costs (IPs, EDA tools, engineers, etc.)
- **Three steps of OSCE:** Open ISA → Open Design → Open Tools



XiangShan: Open-Source High Performance Processors

- **Highest performing open-source processor series by far**
  - Open RTL design with comprehensive documentation
  - Open development tools and platform
- **Address two major challenges in OSCE**

XIANGSHAN

Hosted on GitHub

> 4.5K stars

> 630 forks

High Performance

- Expanding computing demand
- Lack of high performance open source processors due to design and verification complexity

High Customizability

- Industry demand customization and rapid dev iteration
- Enables domain specific solutions with myriad diverse requirements

➢ **"The Linux of processor"**

Two-tier CPU Core Roadmap

Kunminghu Architecture

- Designed for high performance
- Targeting server/data-center segment

- RVA-23 profile
- Leading RISC-V feature sets (H/V ext.)

Versus ARM Neoverse N2

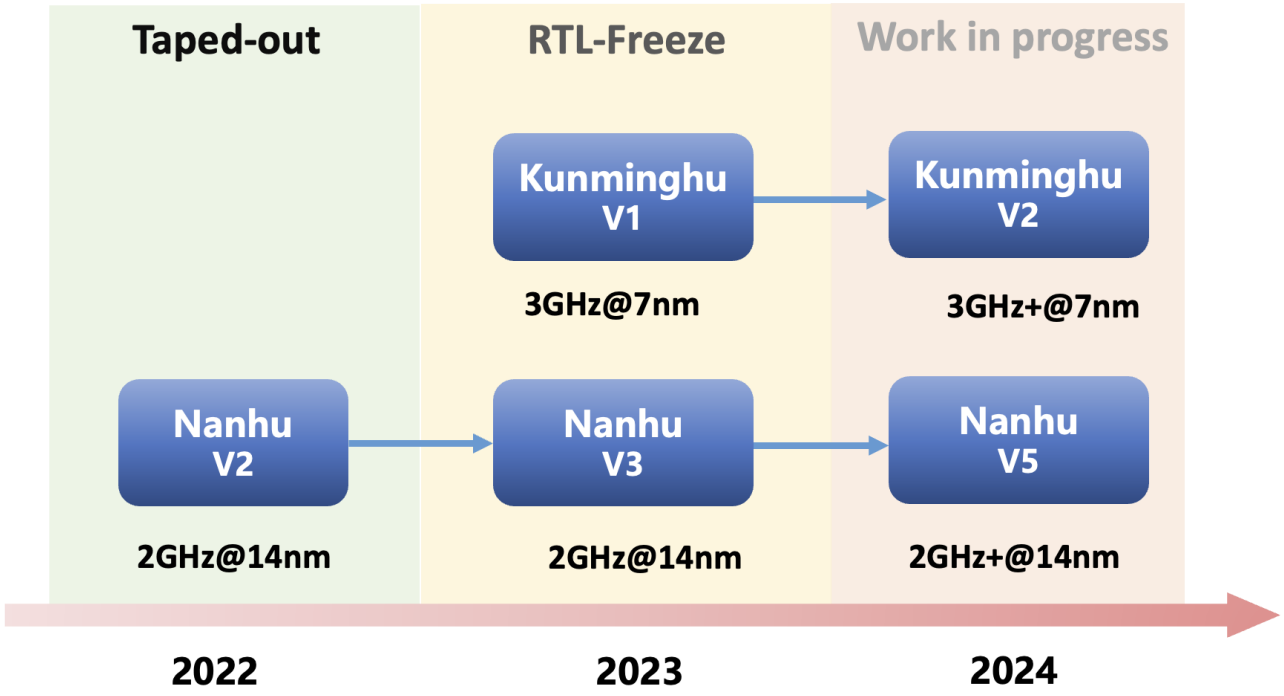
Nanhu Architecture

- Designed for power/area efficiency
- Targeting industry-control segment

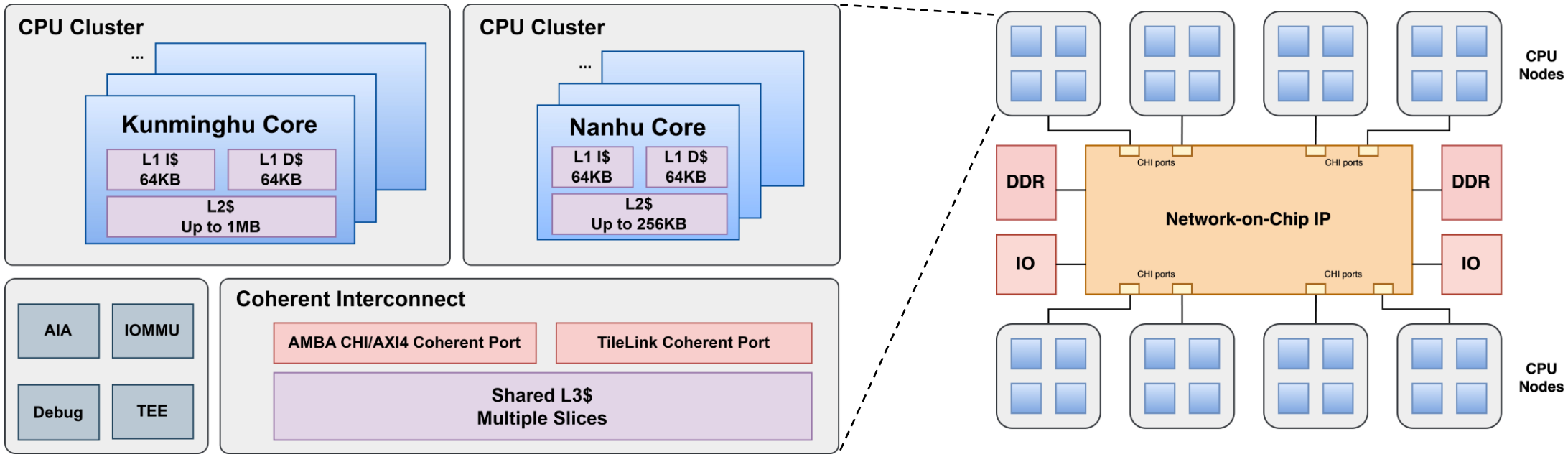
- RVA-20 profile
- Taped out and tested

Versus ARM Cortex A76

- **Industrial-grade µarch design and development workflow**
- **Highly configurable with agile development methodology**
  - Written in Chisel, delivered in both Chisel and Verilog
  - Leverage Chisel's high configurability and efficiency



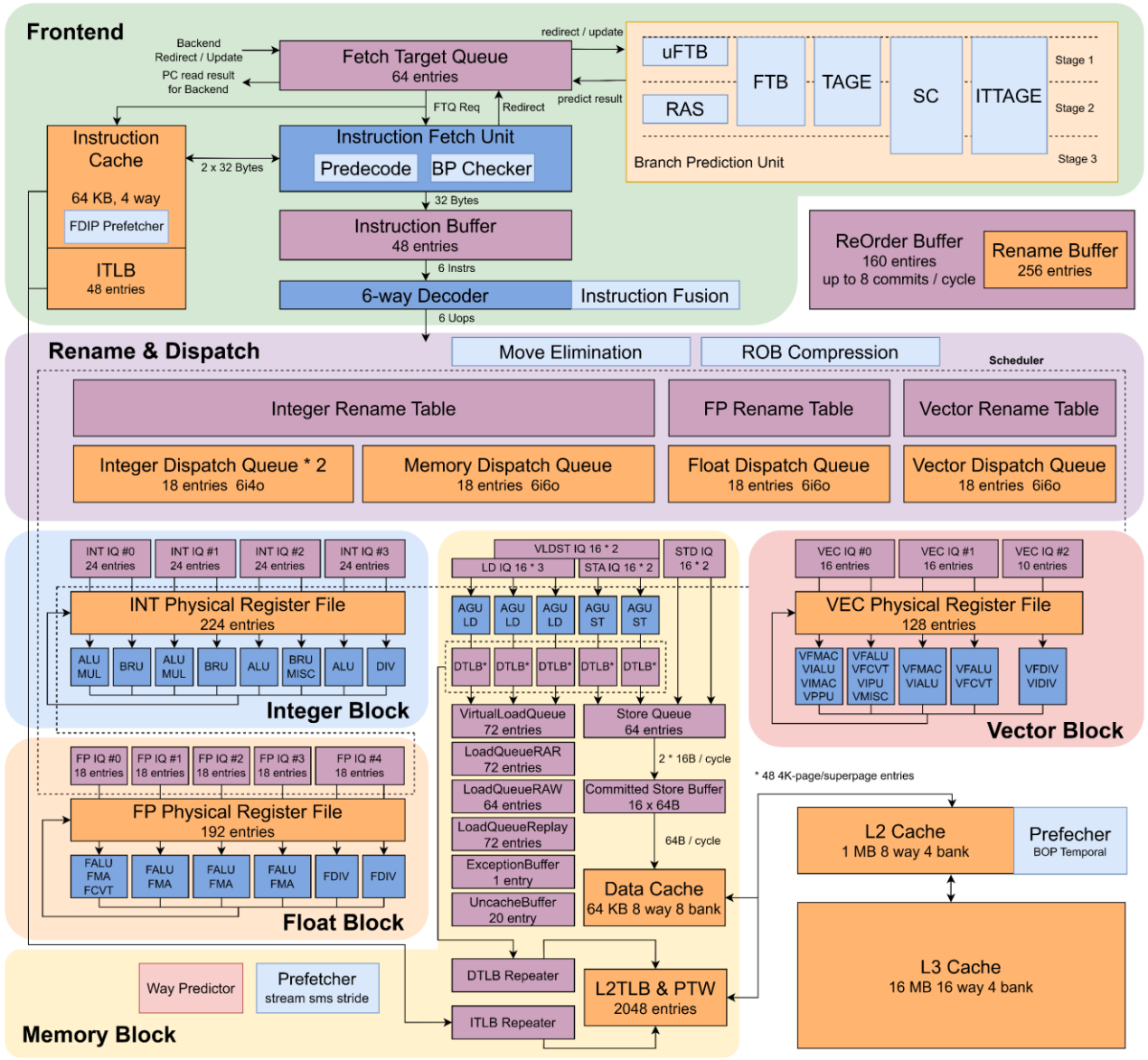
Configurable and Scalable SoC Solutions



Support

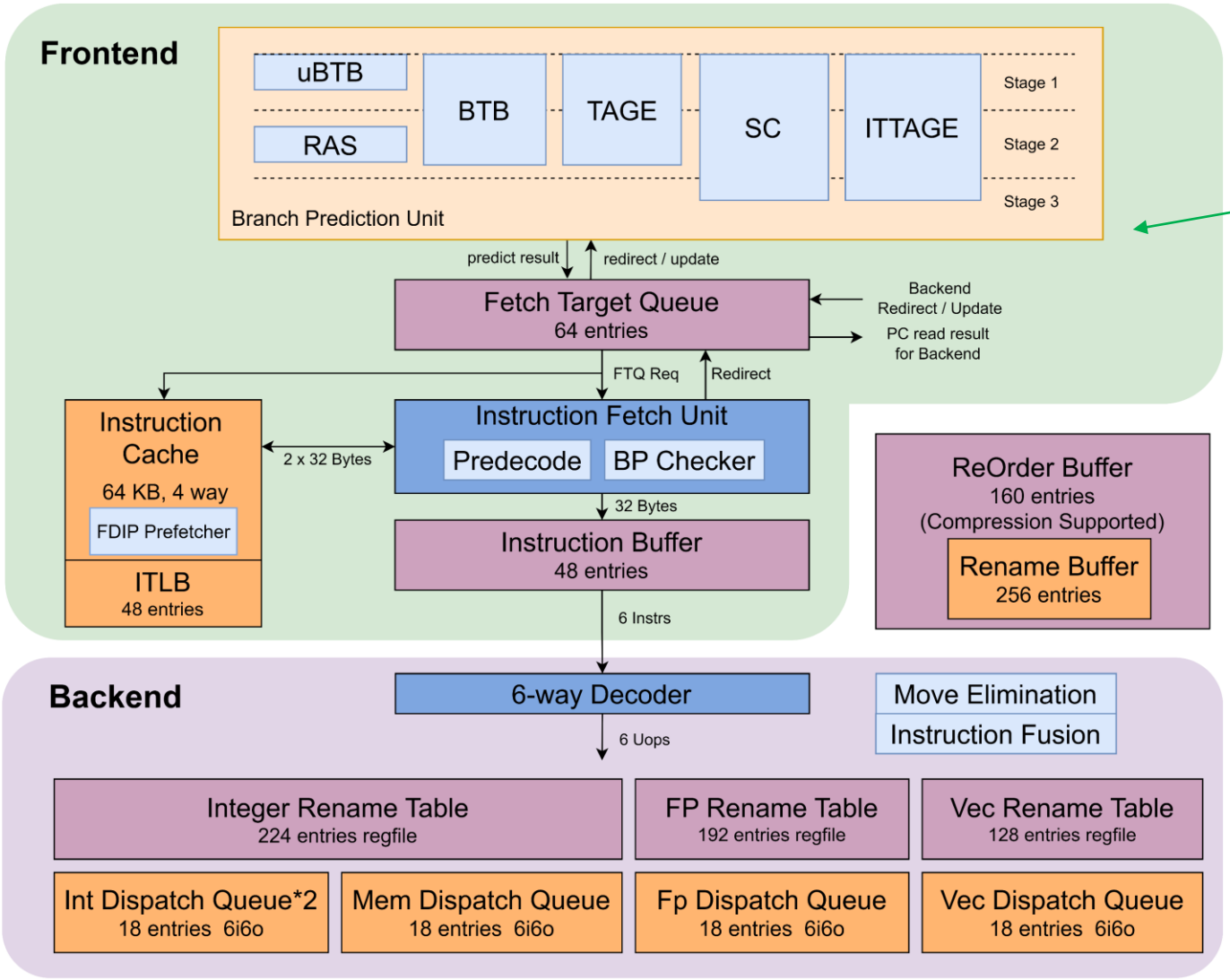
- **RISC-V Advanced Interrupt Architecture**
  - MSI handling & interrupt virtualization
- **RISC-V IOMMU Architecture**
- **RISC-V Debug/Trace Architecture**
- **TEE optimized specifically for RISC-V**
- **Optional cluster-level shared L3 cache**
  - Directory based, Inclusive/non-inclusive
- **CHI/AXI4/TileLink protocol**
- **3rd party NoC ready for current generation**
  - NoC for XiangShan work-in-progress

Kunminghu µArch Overview



- **Decoupled frontend**
  - Reduce fetch bubbles
  - Instruction prefetching friendly
- **Aggressive outstanding instruction window**
  - Large ROB/LQ/SQ
- **Low latency & high BW $ access**
  - Closely-coupled load/store pipe & L1/L2$
  - Highly-banked design
  - Multi-level composite prefetching
- **ISA support**
  - Vector/Hypervisor extension
  - RVWMO (RISC-V Weak Memory Ordering)

Kunminghu µArch Overview

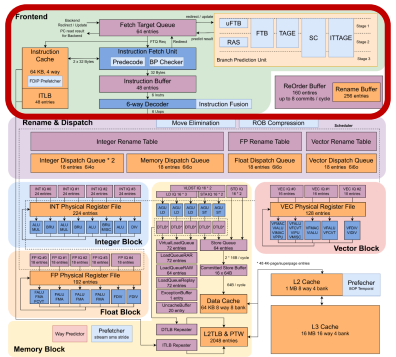


Multi-level branch predictors

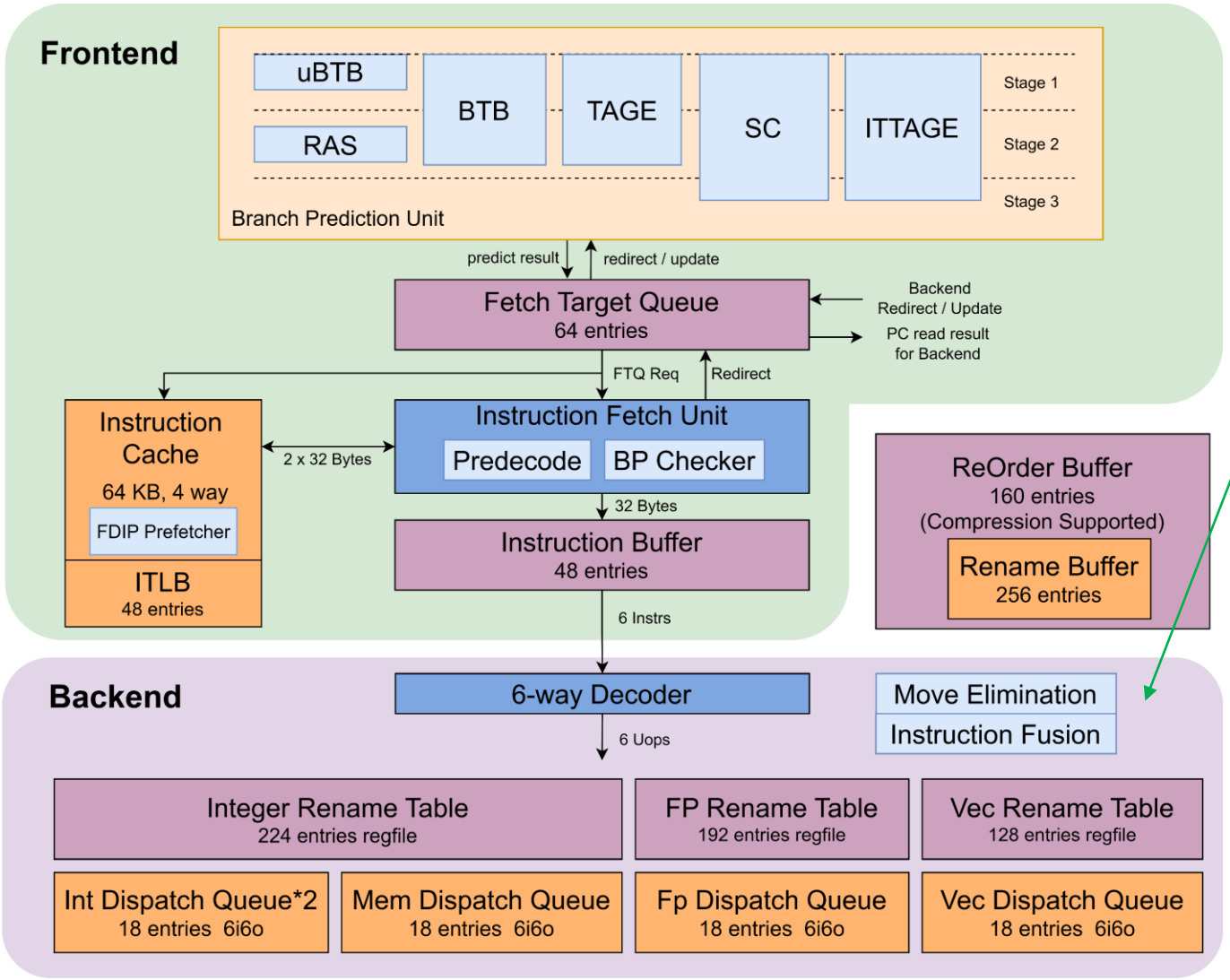
- **256**-entry µBTB
- **2K**-entry BTB, optional L2 BTB
- **16K** TAGE-SC direction predictor
- **2K** ITTAGE indirection predictor
- **48**-entry return address stack

Instruction cache and TLB

- **64KB 4-way** ICache
- **48**-entry ITLB
- Fetch-directed instruction prefetcher



Kunminghu µArch Overview



6-wide decode/rename/dispatch

Register rename

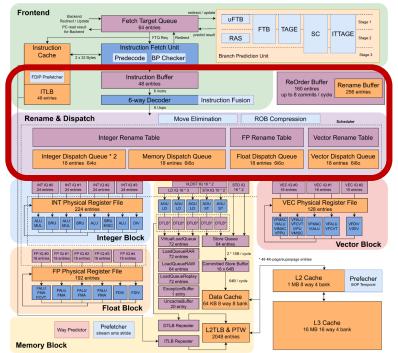
- **224**-entry Int regfile
- **192**-entry FP regfile
- **128**-entry Vec regfile
- Move elimination
- Instruction fusion

160 entry ROB

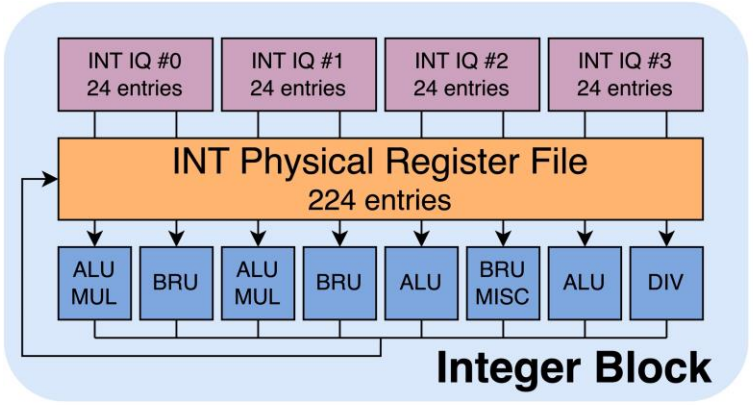
- Support compression (up to **6**-µop/entry)
- **8**-entry retire/cycle
- Recovery via checkpoint + rollback

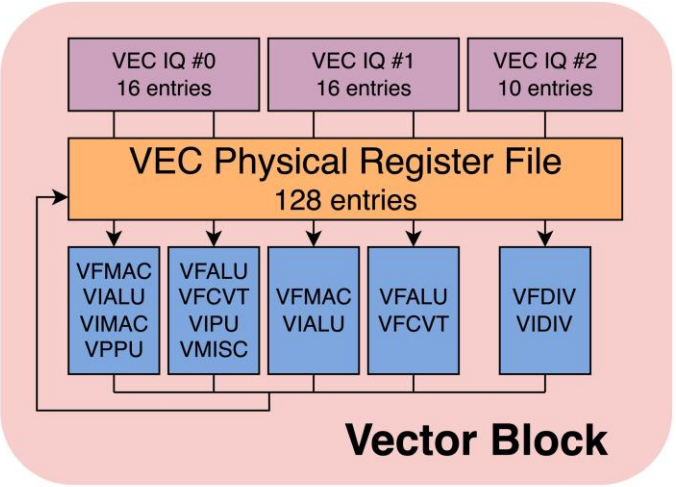
256 entry Rename Buffer

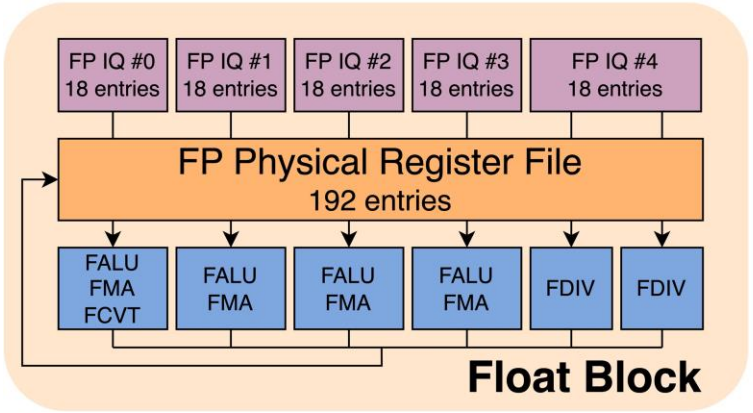
- Bridge the gap between commit & rename table update



Kunminghu µArch Overview







Integer block

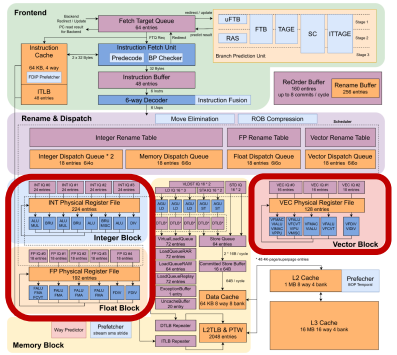
- **4** ALU (2 with 3-stage multiplier)
- **3** BRU for branch resolution
- **1** SRT radix-16 divider

Float-point block

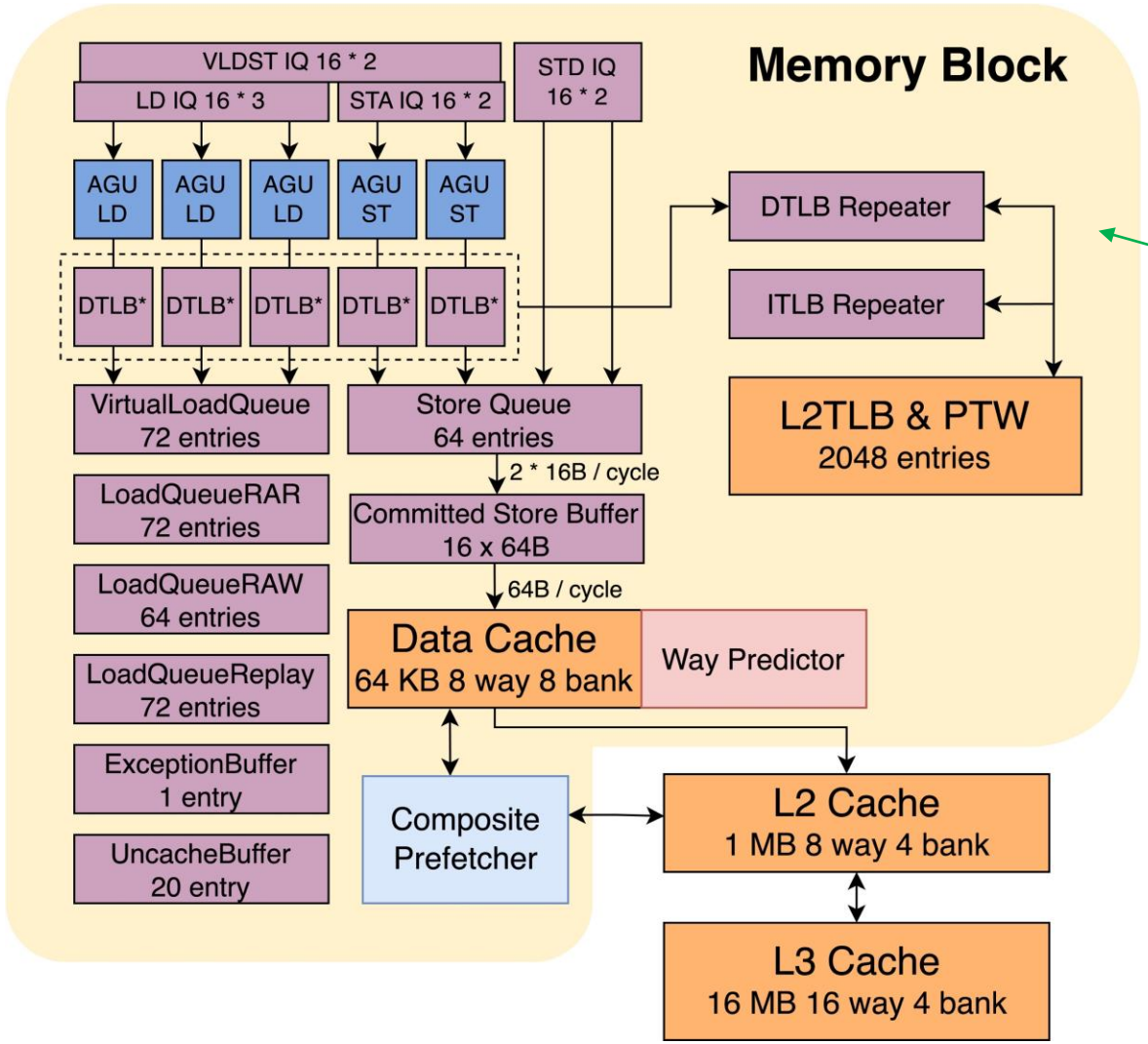
- **4** FPU + **2** FP Div

Vector block

- **4** VPU + **1** Vec Div
- **V1.0** SPEC (VLEN = **128**)



Kunminghu µArch Overview



Load/Store pipeline & queue

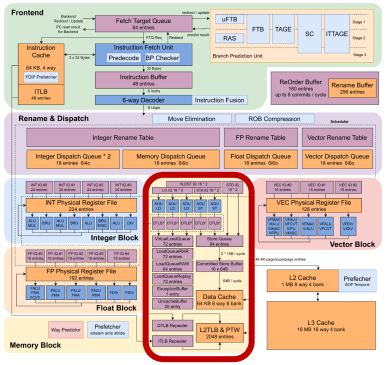
- **3** load pipes, **2** store pipes
- **72** inflight load, **64** inflight store

MMU

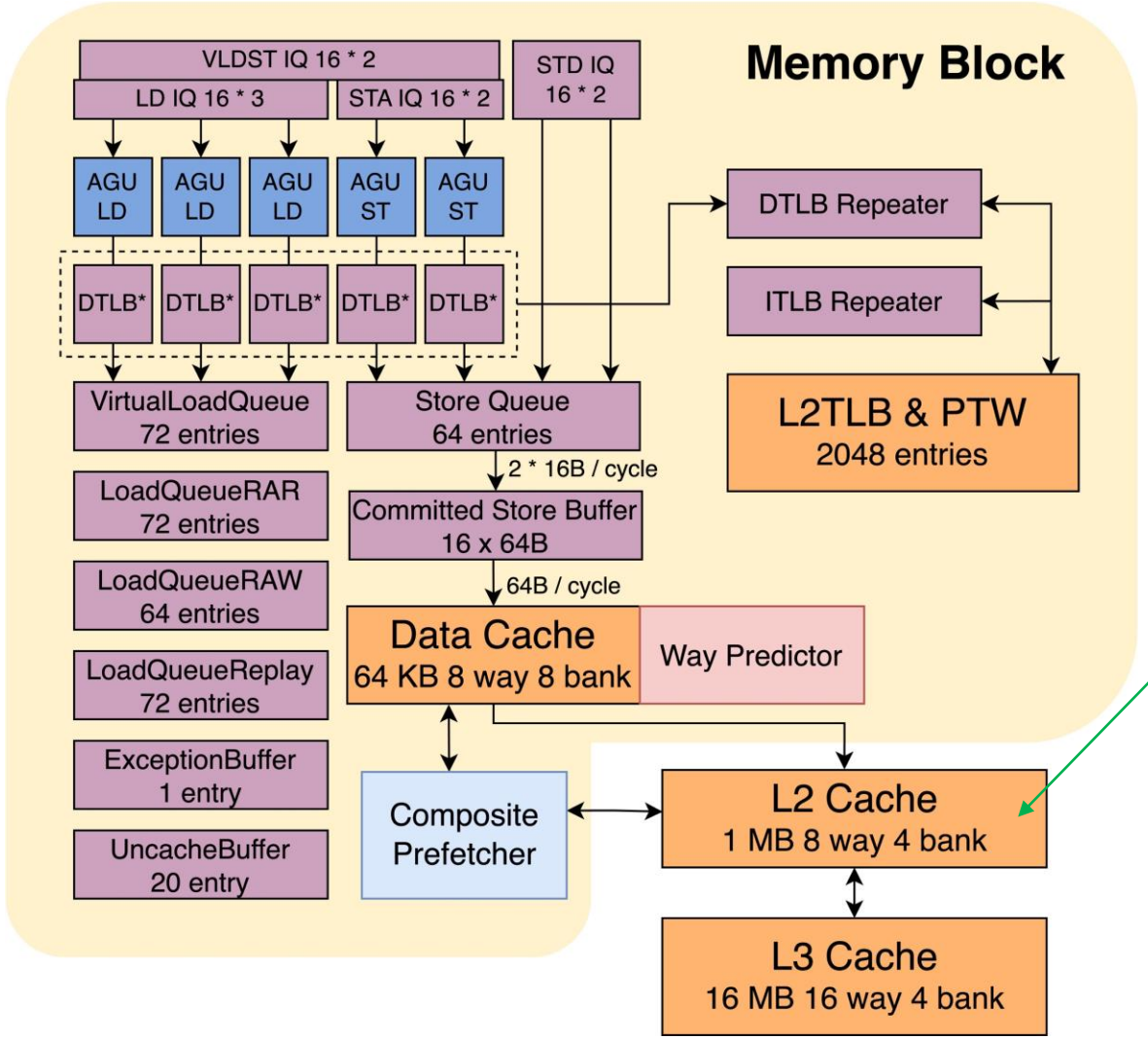
- **48b** virtual/**52b** physical addressing
- **48**-entry L1 DTLB, **2K**-entry L2 TLB
- Up to **7** outstanding page table walks
- Nested walk for virtualization

Data cache

- **64KB 8-way** VIPT (HW anti-aliasing)
- Way predictor for power efficiency
- Composite prefetcher
  - Stream & Stride prefetching
  - SMS & BOP prefetching
  - Temporal prefetching



Kunminghu µArch Overview

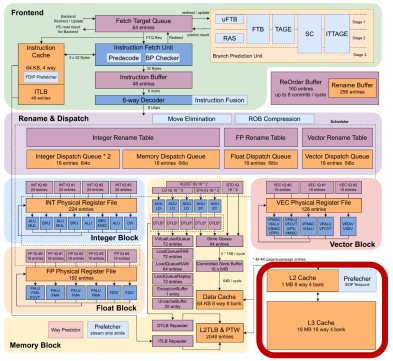


Private L2 cache

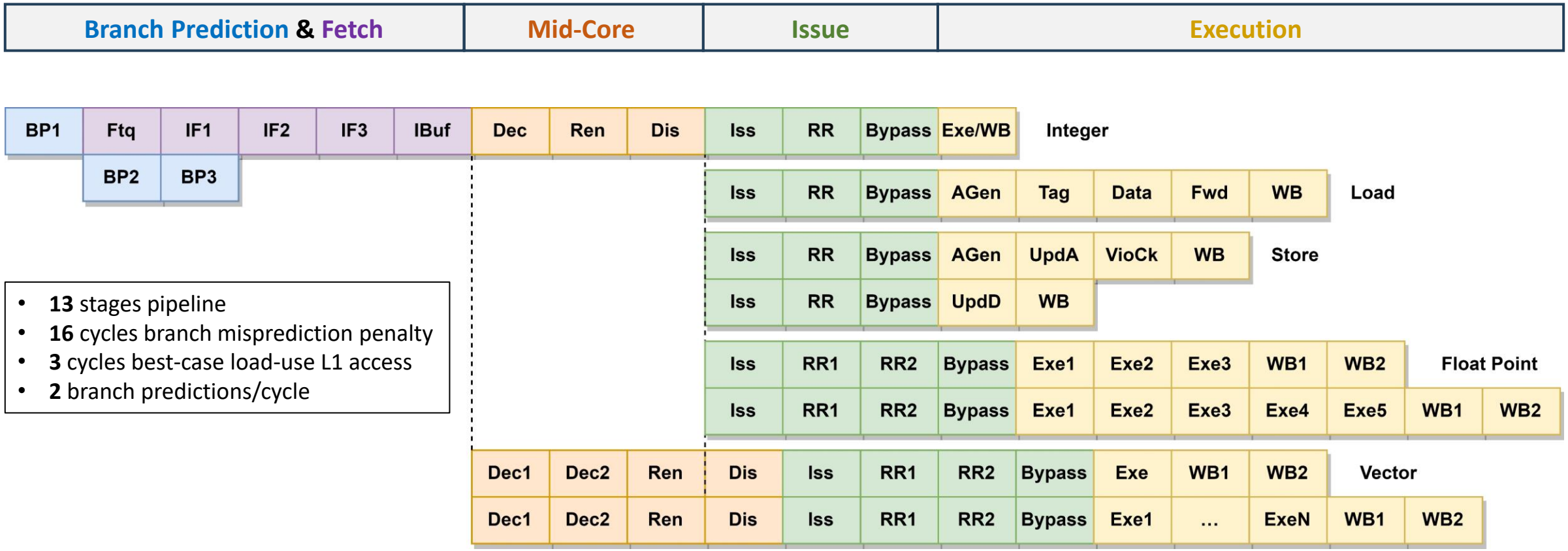
- **8-way**, up to **1MB** per core
- Inclusive to D$, non-inclusive to I$
- **64+** OTs (configurable)
- **10-cycle** load-use latency
- PLRU/RRIP replacement

Shared L3 cache

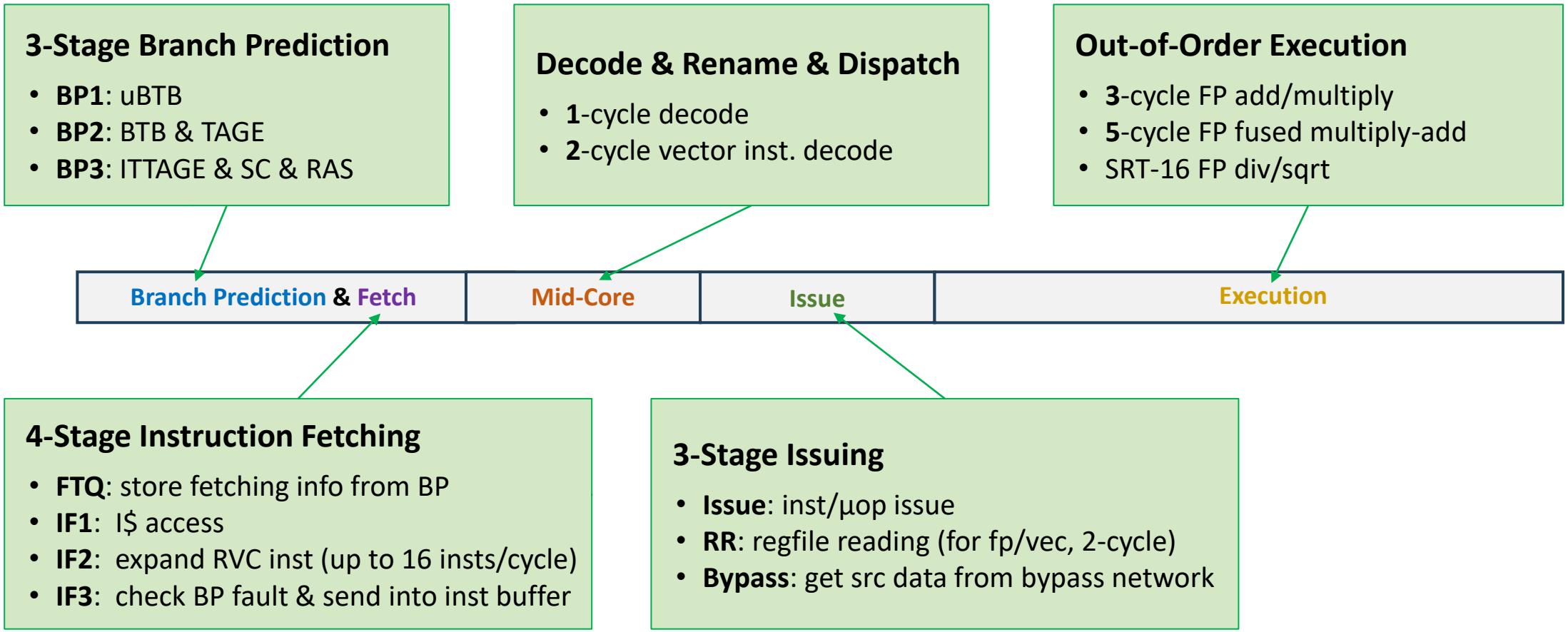
- **16-way**, up to **16MB**
- Inclusive/Non-inclusive to L2$
- **150+** OTs (configurable)
- **~30-cycle** load-use latency
- PLRU/RRIP replacement



Kunminghu Pipeline Diagram



Kunminghu Pipeline Highlights



Comparison between Kunminghu & Nanhu

- Nanhu architecture basically similar, tailored to strike a balance in PPA
- Most features are comparable to Neoverse N2 & Cortex A76, respectively

Feature	Kunminghu	Neoverse N2	Nanhu	Cortex A76
Pipeline depth	13	10	13	13
Rename width	6	5	4	4
Rename checkpoint	Y	Y	N	N
ROB size	160 (x6)	160+	192	128
ALUs	4	4	4	3
L1 instruction cache	64KB	64KB	64KB	64KB
L1 data cache	64KB	64KB	64KB	64KB
L2 cache	1024KB	512/1024KB	256KB	256/512KB
L3 cache	Up to 16MB	4MB per slice	Up to 4MB	Up to 4MB
NoC support	Y	Y	N	N
L2 outstanding txns	64	64	32	46
ITLB	48	48	32	48
DTLB	48	44	128 direct mapped	48
L2 TLB	2048	1280	2048	1280
Vector	Y	Y	Y	Y
Virtualization	Y	Y	N	Y
ECC support	Y	Y	Y	Y
PMA/PMP support	Y	Y	Y	Y
Debug support	Y	Y	Y	Y
External interface	AXI4/TL/CHI	AXI4/CHI	AXI4/TL	AXI4/CHI

Competitive

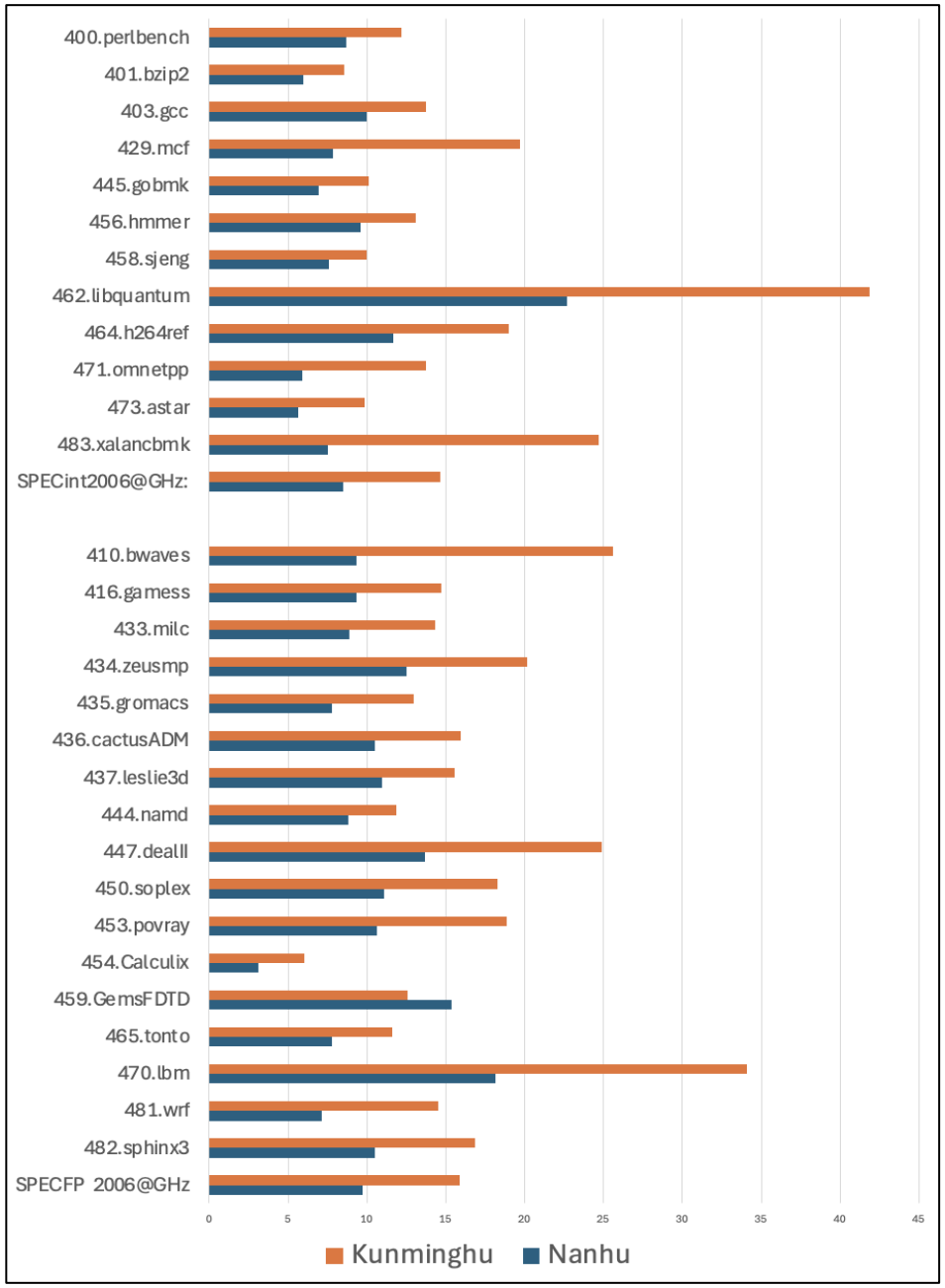
Further optimization needed

Performance Evaluation

- **Methodology: RTL simulation using checkpoints selected via Simpoint**
  - Compiler: GCC 12 -O3, RV64GCB, jemalloc
  - Cache: 64KB I$/D$ + 256KB L2$ + 4MB L3$ (Nanhu)
    64KB I$/D$ + 1MB L2$ + 16MB L3$ (Kunminghu)
  - Memory modeled by DRAMsim3 DDR4@3200MHz
    - 70ns latency
    - Dual channel, 2x64
- **Evaluation results**

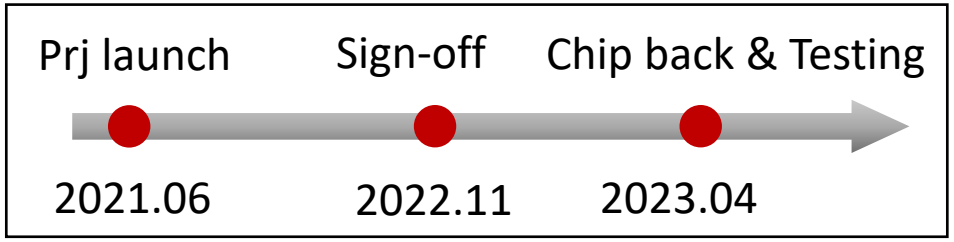
Base score	SPECint 2006	SPECfp 2006
Nanhu@2GHz	16.94	19.42
Kunminghu@3GHz	44.00 → 49.96*	47.63

* With compiler optimizations

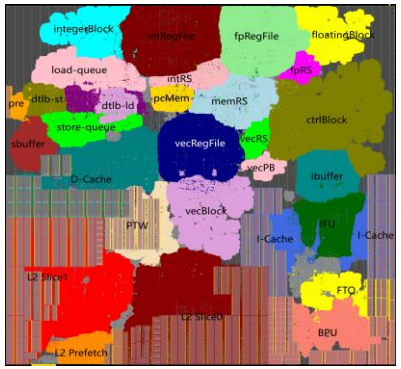


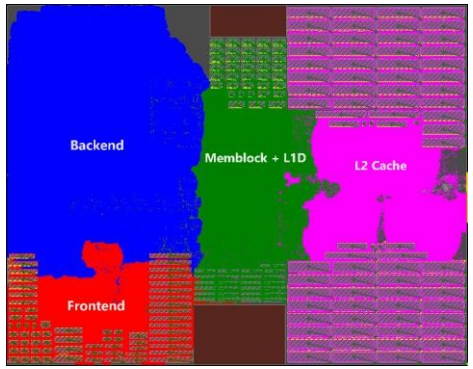
Tape-out Status

- **Nanhu V2 has been taped out**

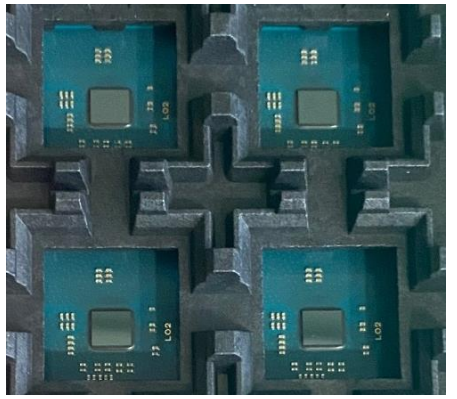


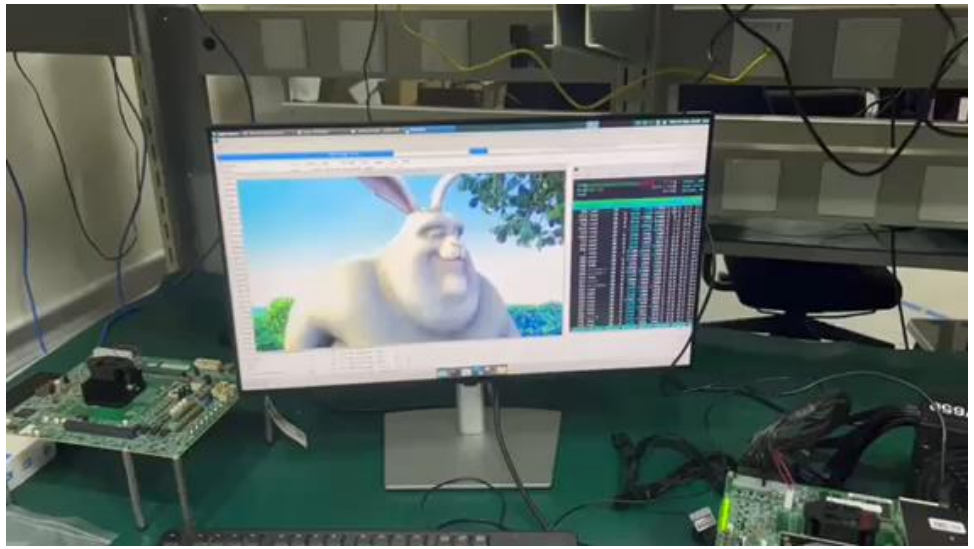
- **Nanhu V2 chip evaluation**
  - 2.5GHz, SPEC CPU 2006 ~10/GHz
- **Ready to tape-out next**
  - Nanhu V3 & Kunminghu V1





Floorplan of Nanhu V3 & Kunminghu V1 (single core)

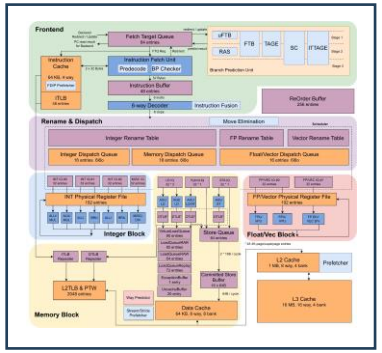


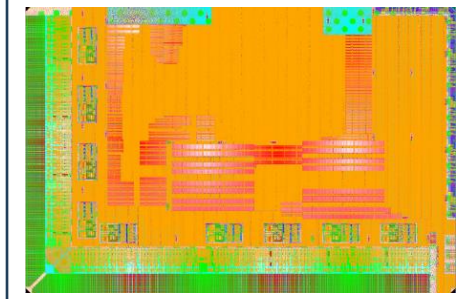


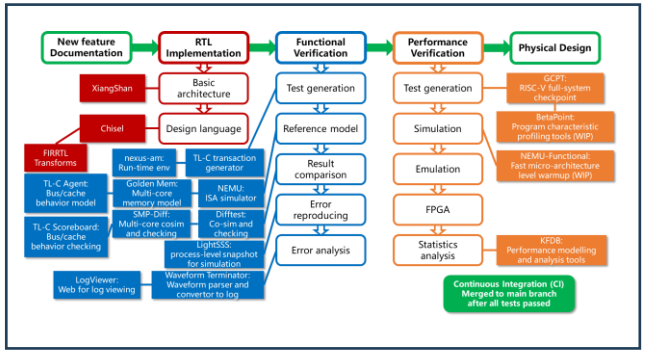
Nanhu V2 Test Chip Demo

Beyond Two Architectures

- **XiangShan = Baseline µArch + Chip Generator + Dev Infrastructure**



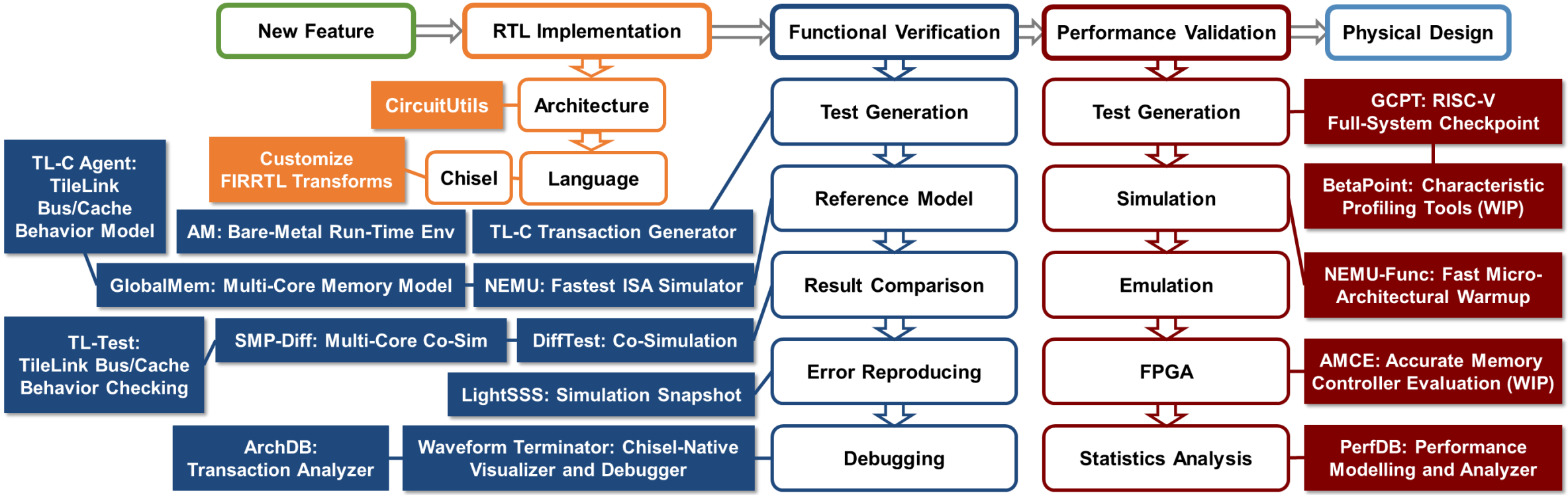




- **Provide a full-stack processor development platform**
  - Infrastructure is a key deliverable of the XiangShan project
  - Open-sourced to empower customization capabilities
- **Support**
  - Rapid design variants — powered by **agile development language**
  - Rapid feature implementation — powered by **agile development toolchain**

Minjie Agile Development Toolchain

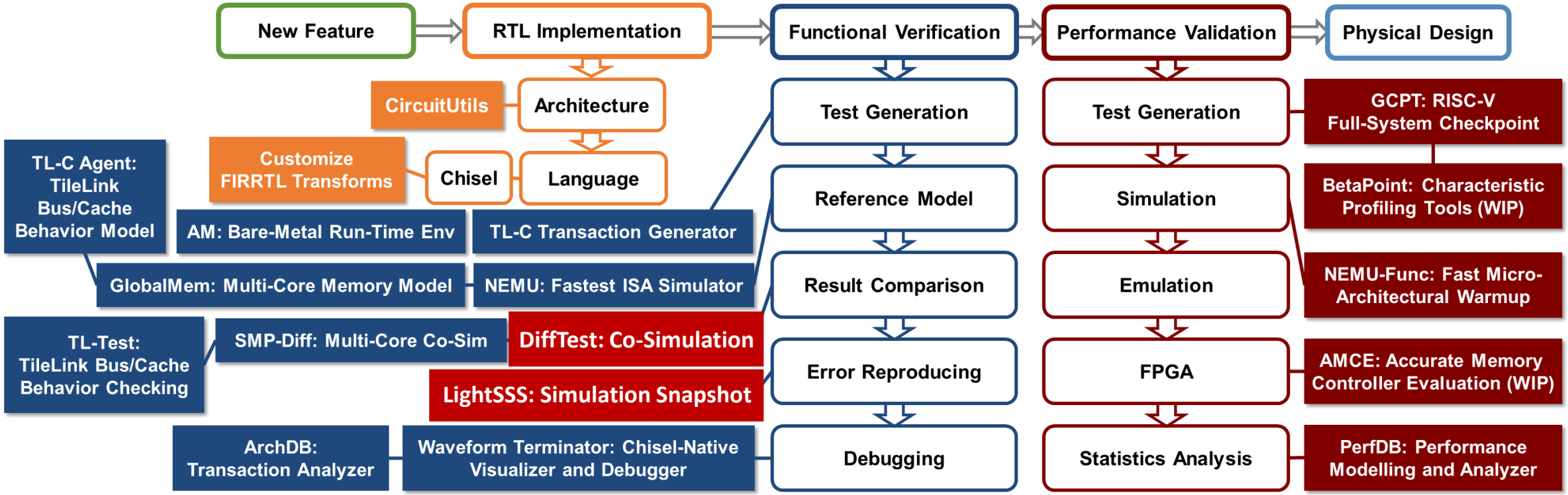
- Cover multiple phases of microarchitecture design and verification
- Tackle complex designs with emphasis on simulation-based verification process



➢ **Only three months** from conception to Debian OS boot in simulation

Minjie Agile Development Toolchain

- Cover multiple phases of microarchitecture design and verification
- Tackle complex designs with emphasis on simulation-based verification process

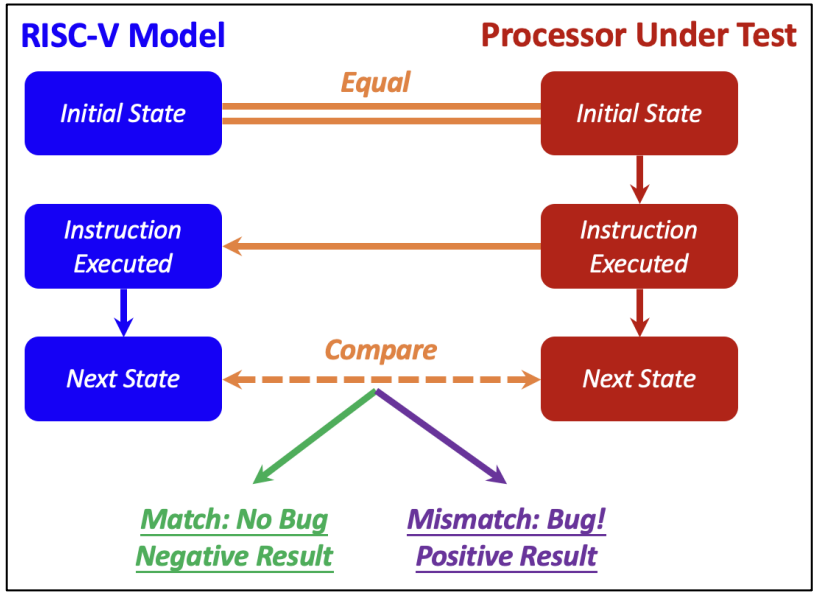


➢ **Only three months** from conception to Debian OS boot in simulation

Minjie Highlights — Difftest

- **Goal:** Isolate RTL functional errors in a timely manner
- **Idea:** Co-simulate RTL design against ISA reference and compare the results
  - **Challenge:** Non-deterministic ISA Spec vs. Deterministic RTL simulation

➢ **Difftest:** Use rules to identify and eliminate non-determinism at runtime



Difftest Co-simulation Workflow

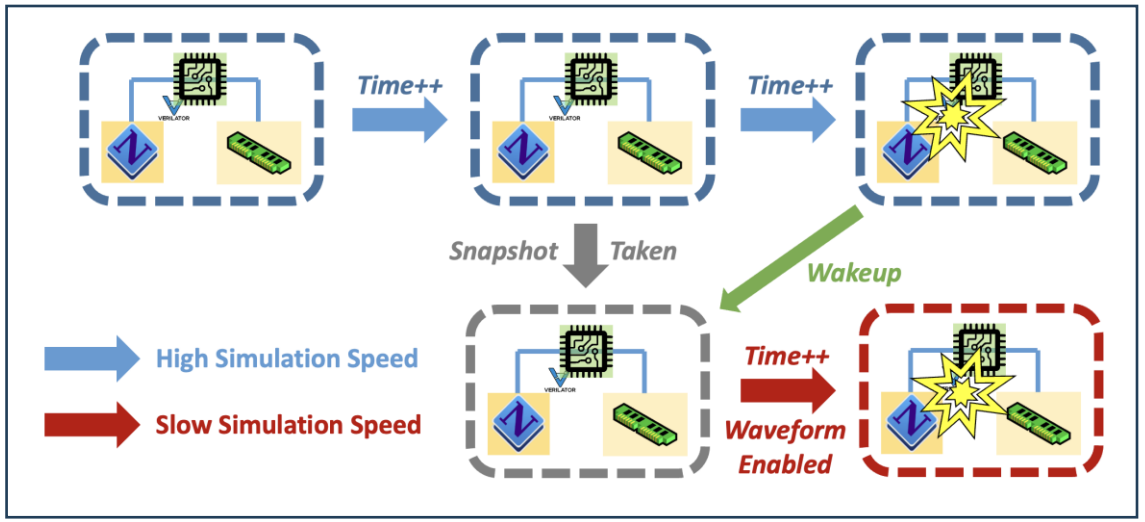
Categories	Sub-Categories	Examples
Static	Impl. Dependent Registers	CSR, MMIO Devices
Dynamic	Impl. Dependent Registers	Timer, Counters
	Asynchronous Events	External/Timer Interrupts
	Speculative Execution	Inst./Load/Store Page Fault
	Memory Model	Memory Accesses in Multi-core
	Hardware Timing	LR/SC, Instruction Fusion, Caches

Behavioral Non-Determinism Identification

Minjie Highlights — LightSSS

- **Goal:** reproduces debug info without significantly slowing down the simulation
- **Idea:** take periodic snapshots to allow replay in debug-mode in case of errors
  - **Challenge:** high time/storage overhead of snapshots

➢ **LightSSS:** use *fork* system call to take snapshots of the simulation process



LightSSS Workflow

Snapshot Interval	Overhead for Coremark	Overhead for Linux Kernel
Every 1min	< 0.1%	< 0.1%
Every 30s	< 0.1%	< 0.1%
Every 10s	< 0.1%	< 0.1%
Every 1s	0.3%	1.8%

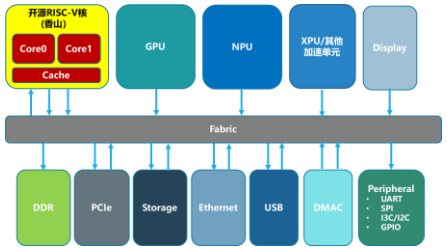
Minor performance overhead

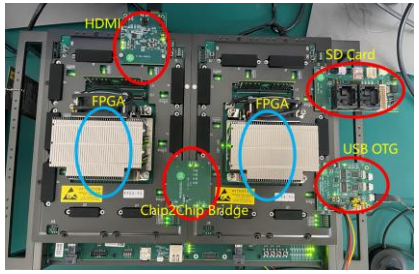
Deep Cooperation with Top Companies & Academia

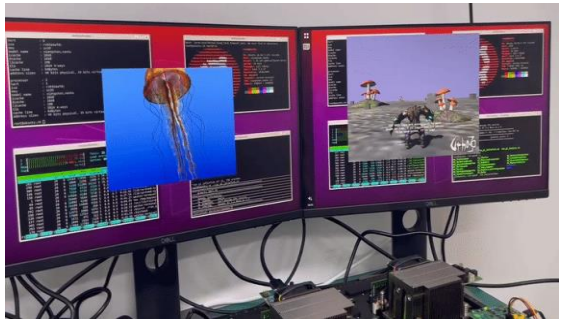
Extensive corporate partnerships

- **7nm Test SoC (Taped-out)**
- **5nm AI Acceleration Chip (Delivered)**
- 28nm Heterogeneous SoC
- 12nm AI Video Monitor Chip
- 7nm/12nm Server CPU
- 7nm DPU Controller

Prototype evaluation for startups





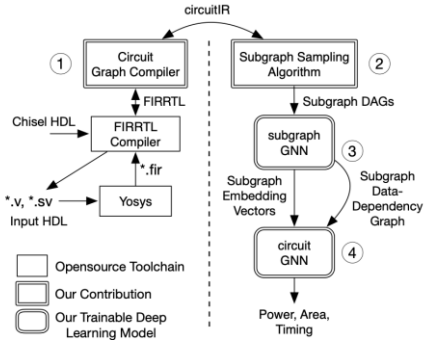


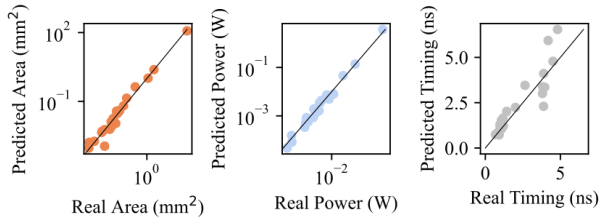
Source: Xinchen Technology

Effective Platform for Academic Research

midgard EPFL

Imprecise Store Exceptions, ISCA'23 (EPFL)





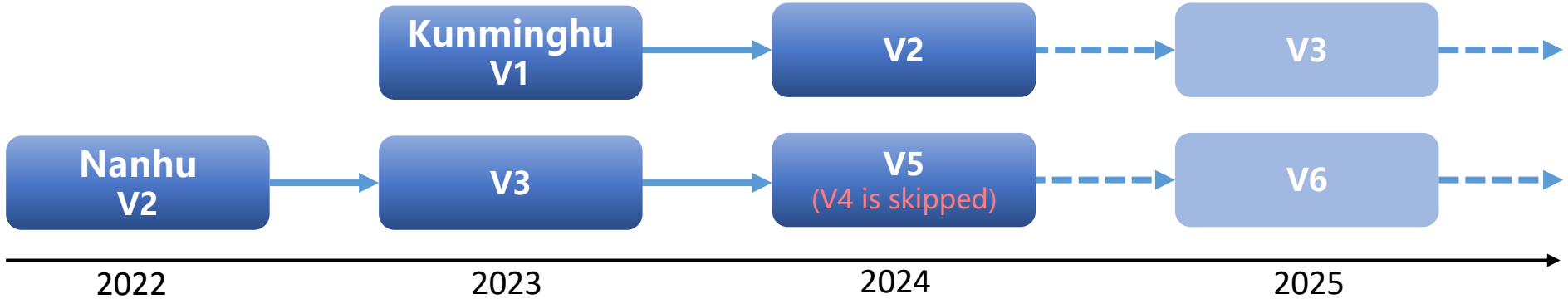
SNS v2, MICRO'23 (Duke University)

Summary & Future Plans

XiangShan Project

- **Embraces innovative agile development workflow**
- **Fills the gap in open-source high-performance processors**
- **Addresses the needs from both industry and academia**

- **Microarchitecture:** Parallel execution by dual development teams
  - Yearly test chip tape-out on cadence for each architecture



- **Development tools:** Further refinement for performance optimization

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

北京开源芯片研究院
BEIJING INSTITUTE OF OPEN SOURCE CHIP

中国科学院大学
University of Chinese Academy of Sciences

Thank you!

XIANGSHAN

Scan to follow us on GitHub

HOT CHIPS