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
香山开源处理器社区 XiangShan Open-Source Processor Community

At the 58th IEEE/ACM International Symposium on Microarchitecture (MICRO 2025)

DiffTest-H: Toward Semantic-Aware Communication in Hardware-Accelerated Processor Verification

Kunlin You[1,2], Yinan Xu[1], Kehan Feng[3], Luoshan Cai[1,2], Yaoyang Zhou[3] and Yungang Bao[1,2]

[1] *State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences*

[2] *University of Chinese Academy of Sciences*

[3] *Beijing Institute of Open Source Chip*

October 21, 2025

XiangShan: Open-source High Performance Processors

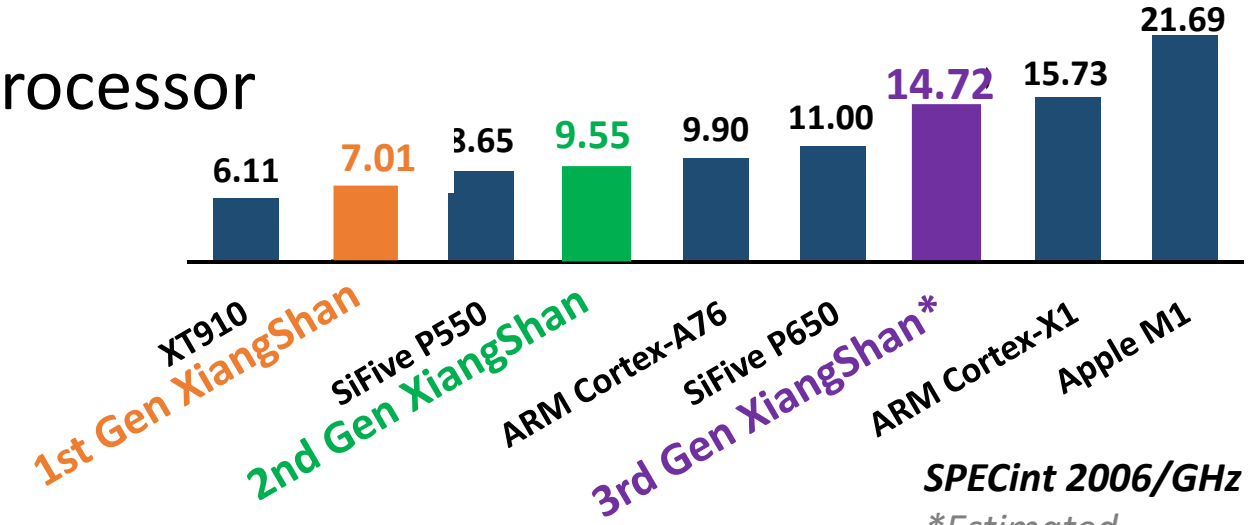
- **Top-Performing** Open-Source Processor

XIANGSHAN

Code on GitHub

> 6.7K stars

> 820 forks



- Vision: "**Linux of Processors**" for Industry & Academia

Industrial Partners

ESWIN 奕斯伟计算

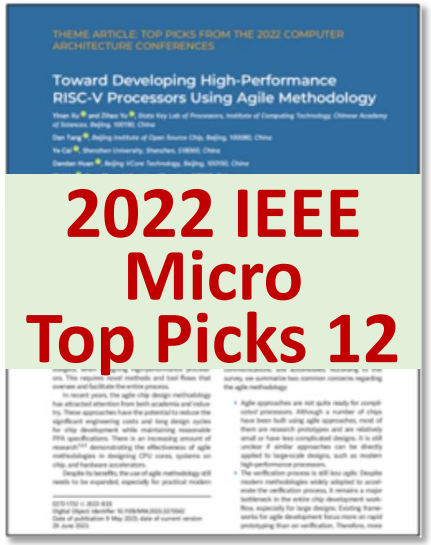
Tencent 腾讯

SANECHIPS 中兴微电子

蓝芯算力 LANXIN COMPUTING

知合计算 ZHIHE COMPUTING

SOPHGO 算能



2022 IEEE Micro Top Picks 12

Research Platform

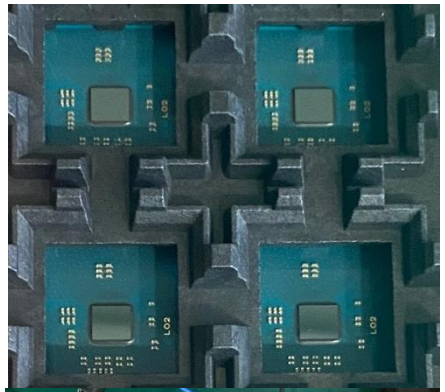
Research on XiangShan

XiangShan has been used by researchers as the underlying platform for their evaluations. We appreciate their contributions to enhancing XiangShan and strengthening the community.

- Imprecise Store Exceptions, EPFL, ISCA'23
- TEESec: Pre-Silicon Vulnerability Discovery for Trusted Execution Environments, OSU & SUSTech, ISCA'23
- of CPU, ICT-CAS, ICCAD'23
- A Distributed ATPG System Combining Test Compaction Based on Pure MaxSAT, ICT-CAS, ATS'23
- REMU: Enabling Cost-Effective Checkpointing and Deterministic Replay in FPGA-based Emulation, ICT-CAS, ICCD'23
- Asynchronous Memory Access Unit: Exploiting Massive Parallelism for Far Memory Access, ICT-CAS, TACO
- Single-Address-Space FaaS with Jord, EPFL, ISCA'25

XiangShan: Powering Desktop, GPU SoC, Server …

1st Gen XiangShan Chip

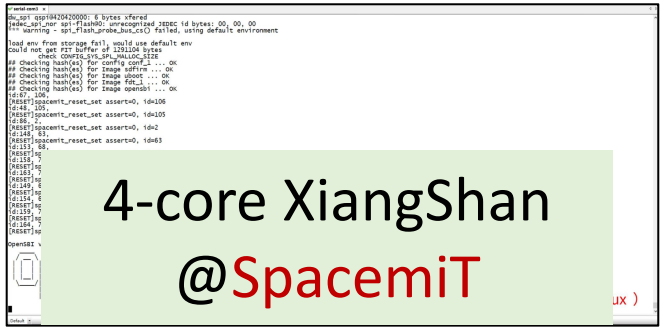


XiangShan Laptop @ISCAS

2nd Gen XiangShan Chip

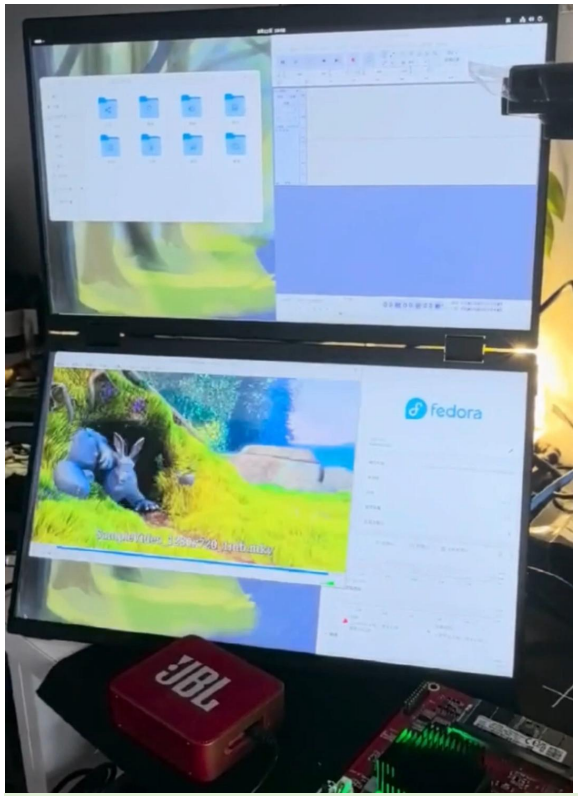
Gaming on 2nd Gen XiangShan Chip

XiangShan on GPU @InnoSilicon



4-core XiangShan @SpacemiT

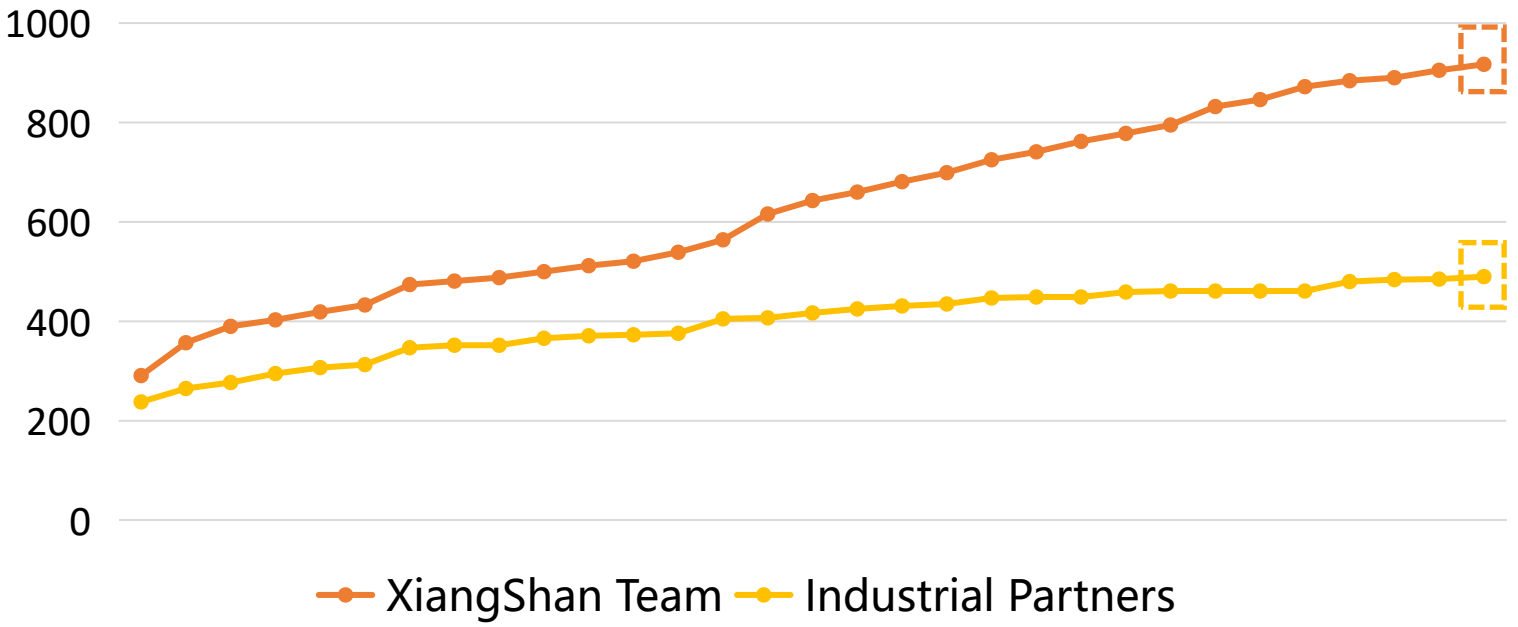
8-core XiangShan @LanXin Computing

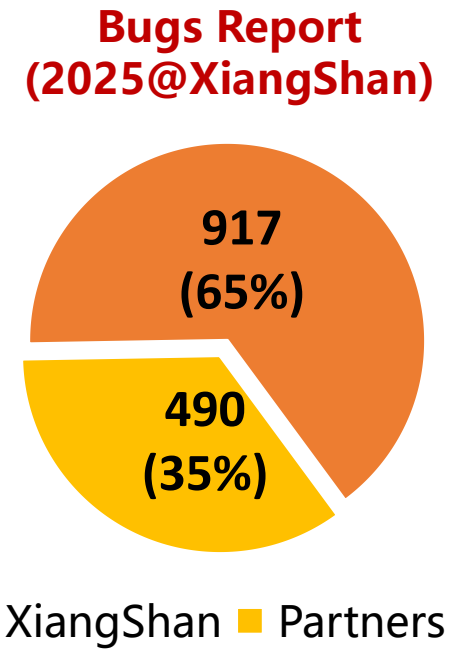


Fedora on XiangShan @Fedora-V Force

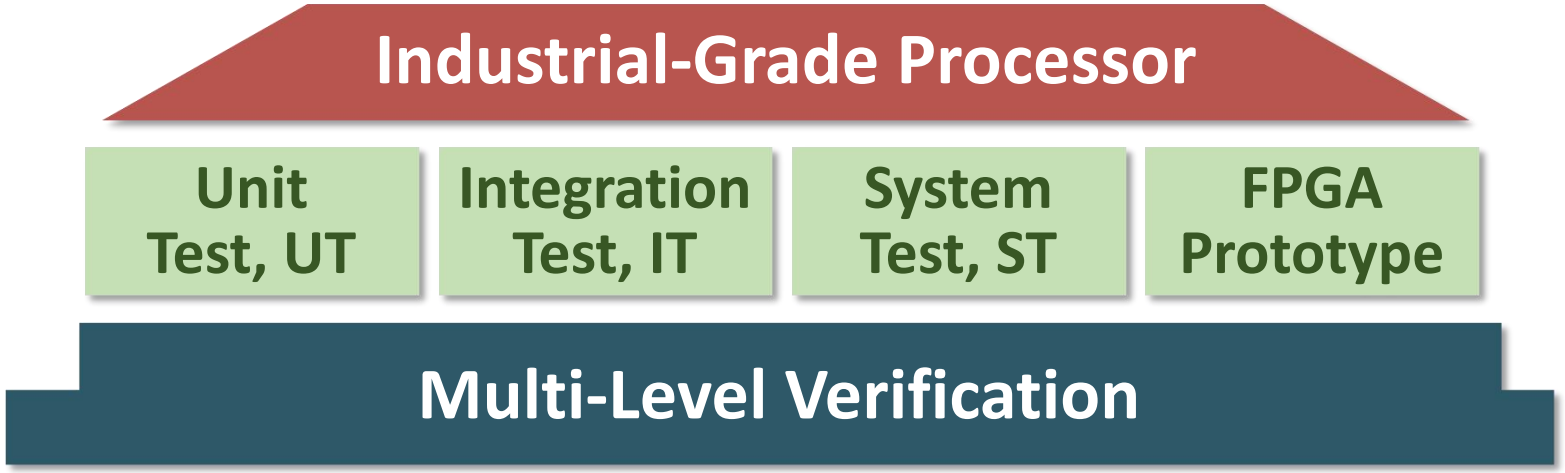
Open-source Demands Agile Verification

- Highly configurable ISA
- Rapid code iterations
- **1,400+** bugs uncovered in XiangShan (2025)

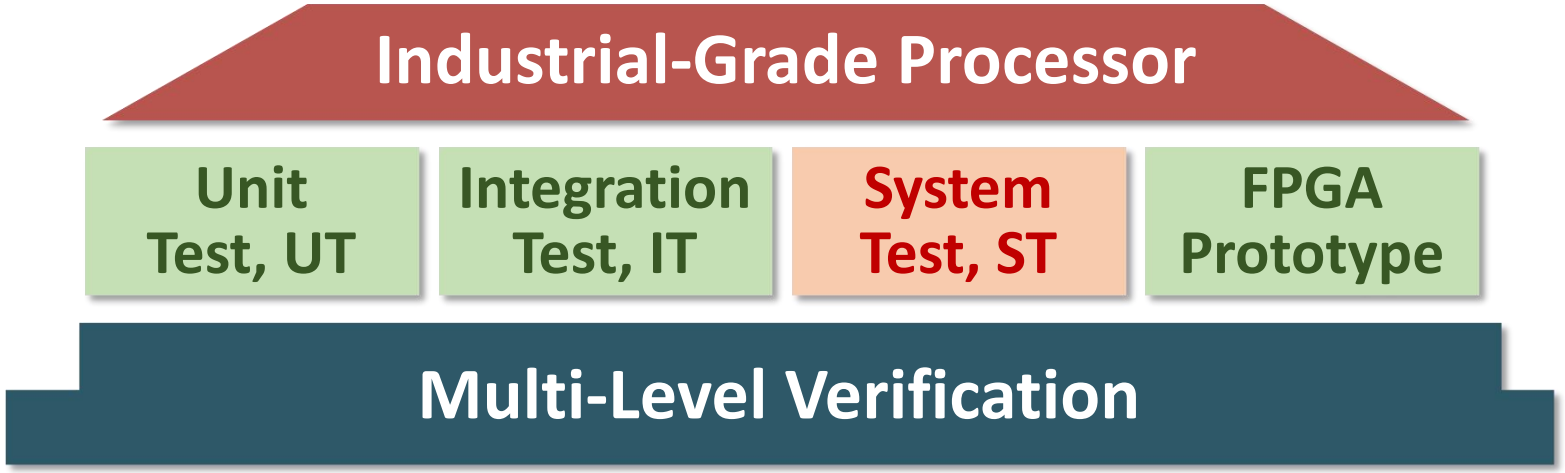




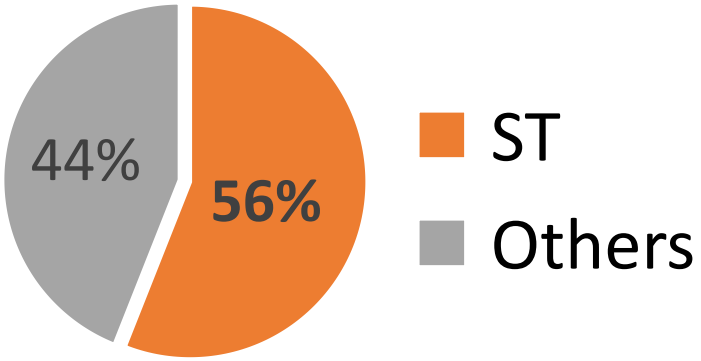
The Era of Verification



The Era of Verification



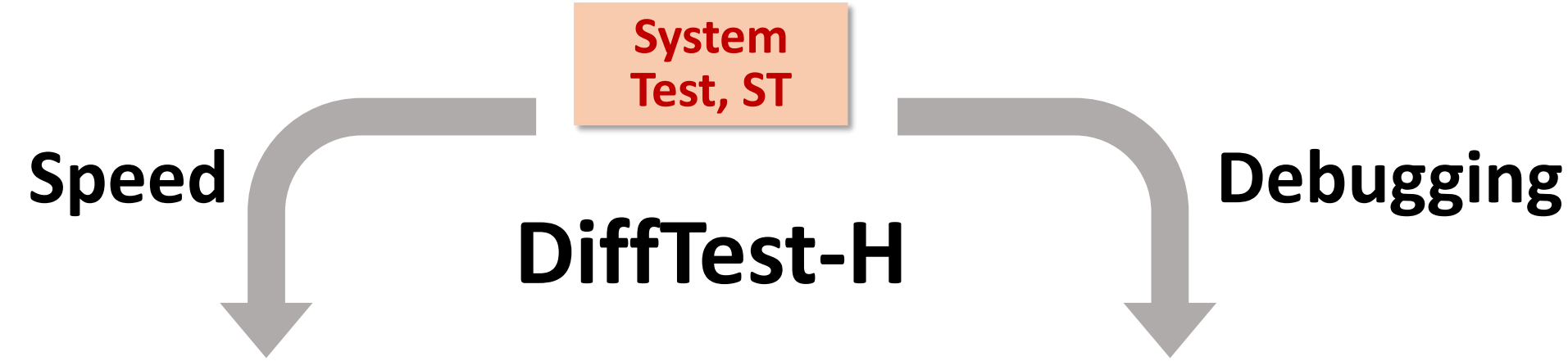
Covering 56% bugs of XiangShan



Checking under real-world cases

Linux
NGINX 20 Years
PARSEC ['pär-''sek] A unit of measure
RT-Thread
CoreMark®
MySQL
KVM
litmus FROM VALIDITY
SPEC CPU®

This Work: Fast & Debug-friendly System Test



13.8 MHz@FPGA

Hardware-Accelerated RTL Simulation

Instruction-level Checking

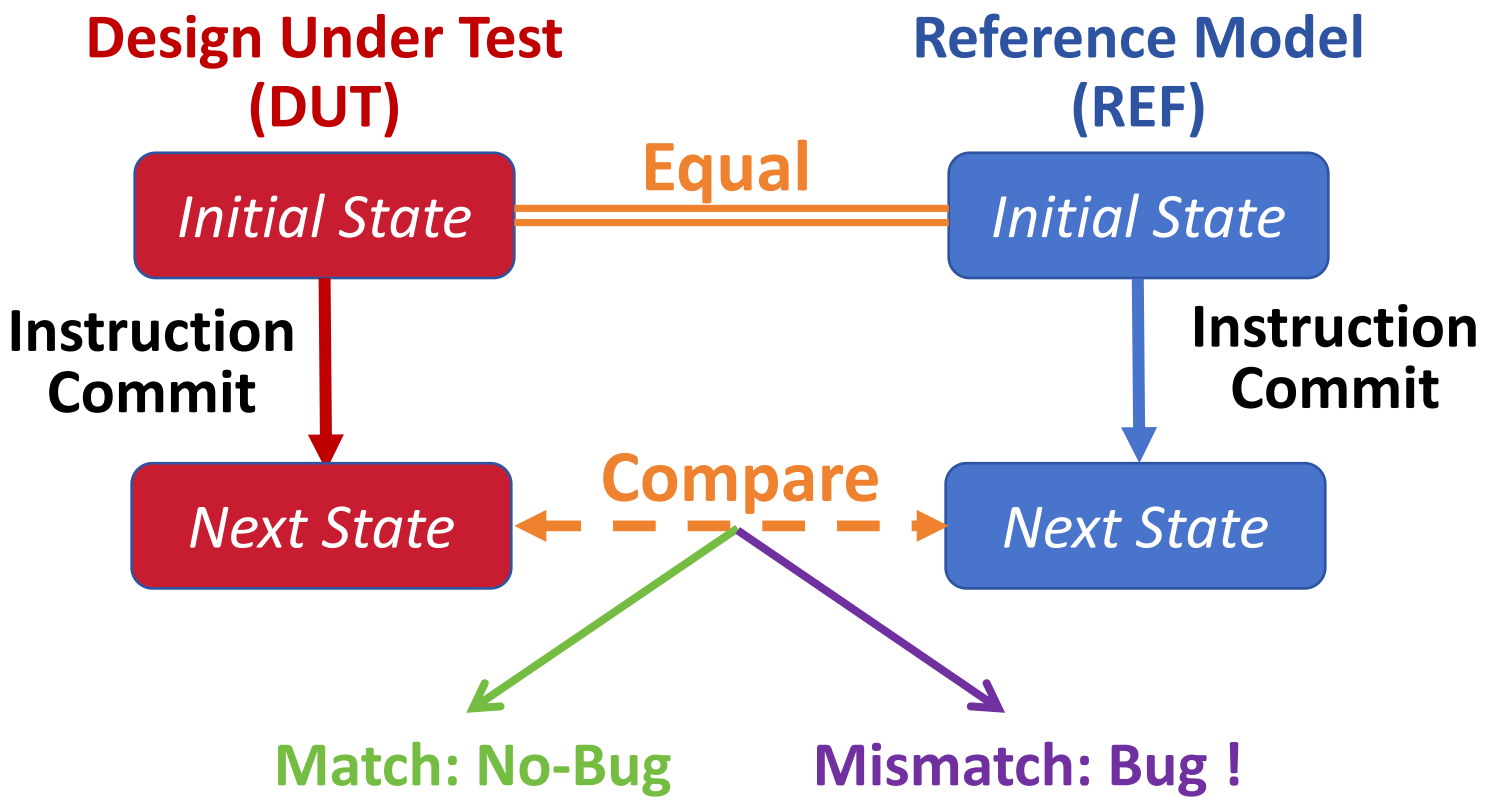
Per-Instruction Comparison with Reference Model

Practical & Effective

151 bugs uncovered in XiangShan

Co-simulation: Instruction-level Debugging

- **DiffTest[1]: Instr-by-Instr Comparison** of **Architectural State** (DUT vs. REF)

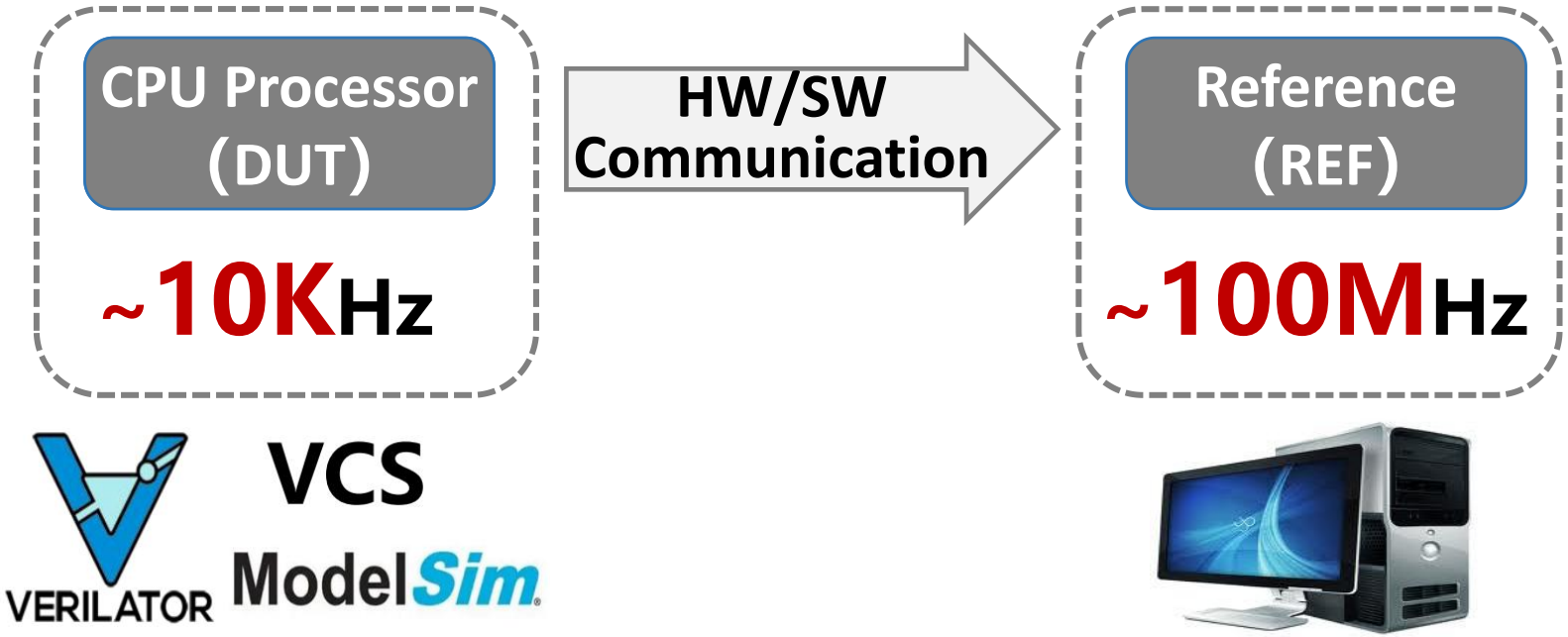


Well debugging, but what about speed?

[1] Xu et al. Towards Developing High Performance RISC-V Processors Using Agile Methodology. (MICRO 2022)

Software-based Co-simulation

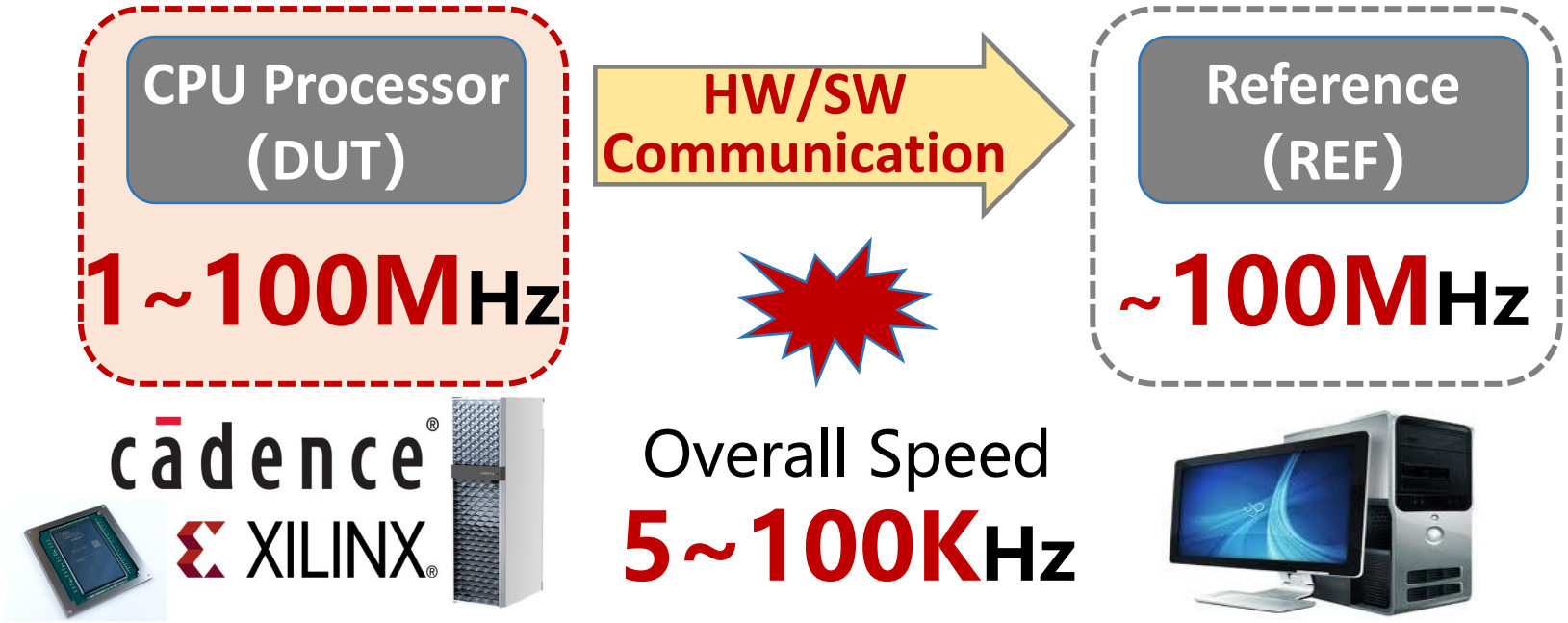
- Based on **RTL Simulator:**
  - Verilator, VCS: **only KHz-scale speed** (limited by RTL simulation)
  - Even with optimizations[1,2,3], still limited at **KHz-scale**



[1] Chen et al. GSIM: Accelerating RTL Simulation for Large-Scale Designs. (DAC 2025)
[2] Wang et al. Repcut: Superlinear parallel rtl simulation with replication-aided partitioning. (ASPLOS 2023)
[3] Zhou et al. Khronos: Fusing memory access for improved hardware RTL simulation. (MICRO 2023)

Hardware-accelerated Co-simulation

- Based on **Emulator/FPGA:**
  - Emulator/FPGA: DUT at 1~100 MHz, **up to 10,000× speedup**
  - Reference Model (software): ~100 MHz

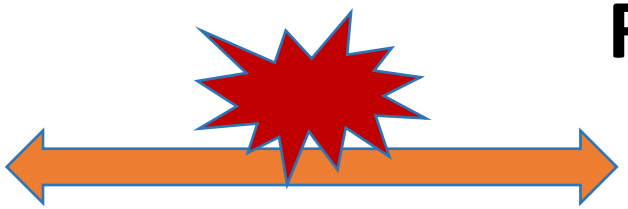


Speed GAP: HW/SW Communication

Ideal: DUT Speed

Up to 10,000× Speedup

Emulator: 1 MHz
FPGA: 100 MHz



GAP: HW/SW Communication

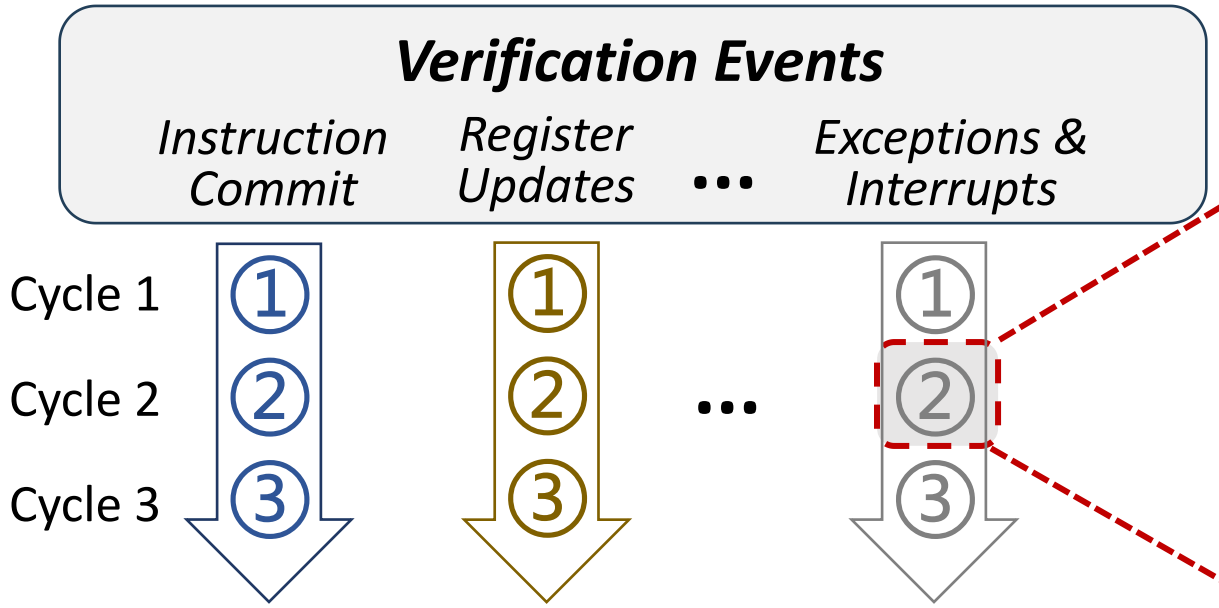
Reality: Co-sim Speed

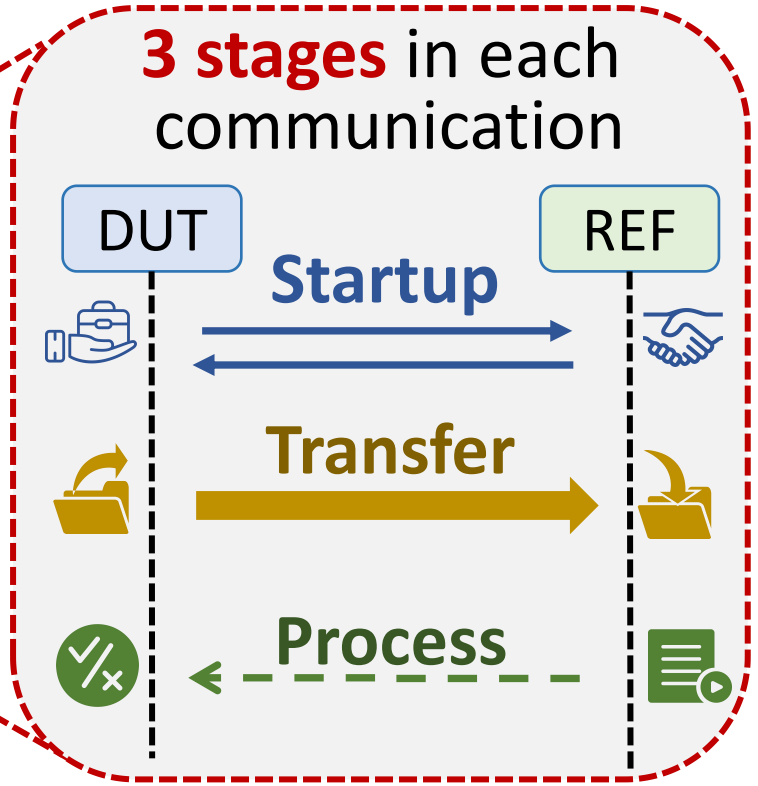
Less than 20× Speedup

Emulator: < 10 KHz
FPGA: < 100 KHz

98%~99.8% of co-sim time

What is HW/SW Communication?

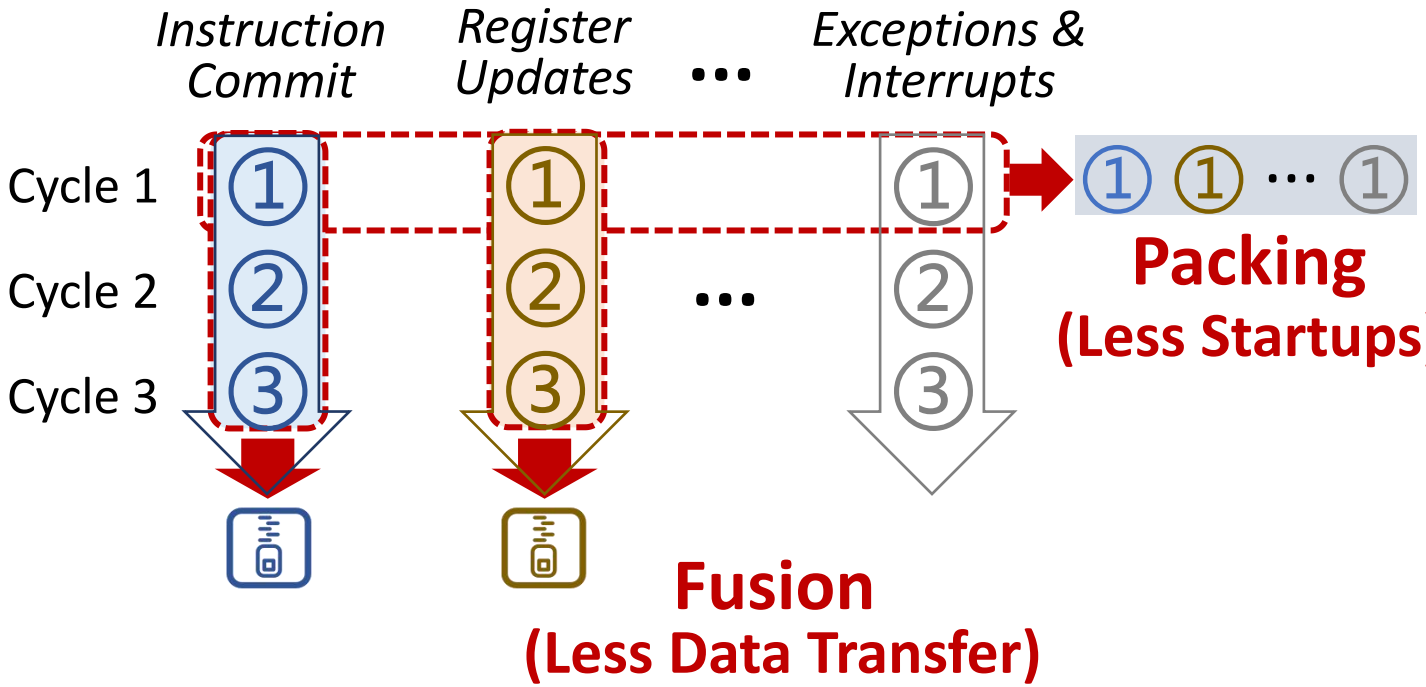


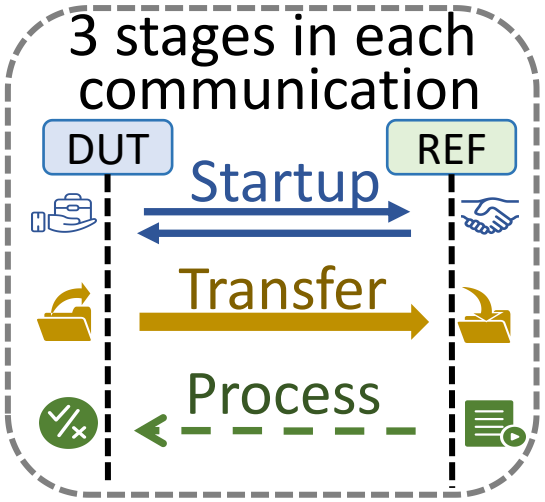


- Example: DiffTest[1] (XiangShan@MICRO 2022)
  - **32 types** of architectural events
  - 1 event trigger 1 communication
  - **~15 communications, ~1.2 KB data** per cycle

[1] Xu et al. Towards Developing High Performance RISC-V Processors Using Agile Methodology. (MICRO 2022)

How to optimize 3-stage communication?

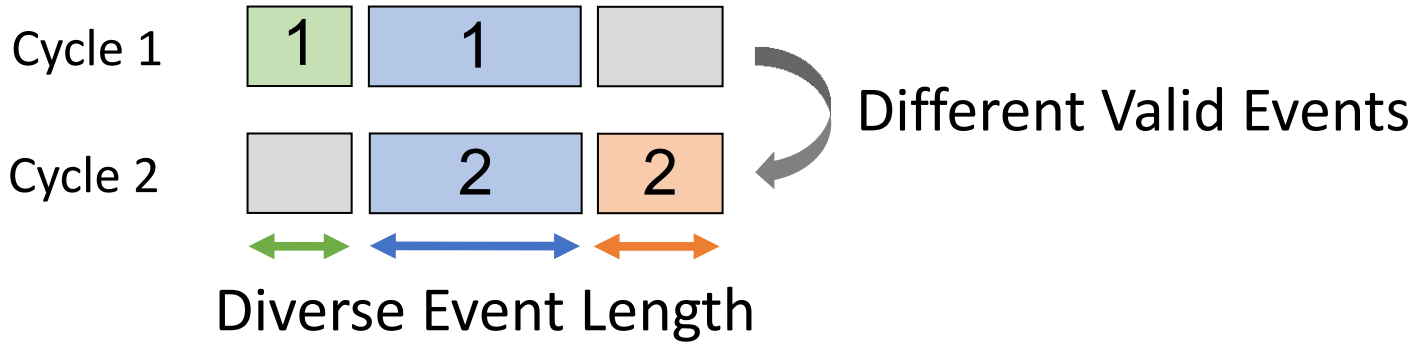




Focus on **Startup & Transfer**

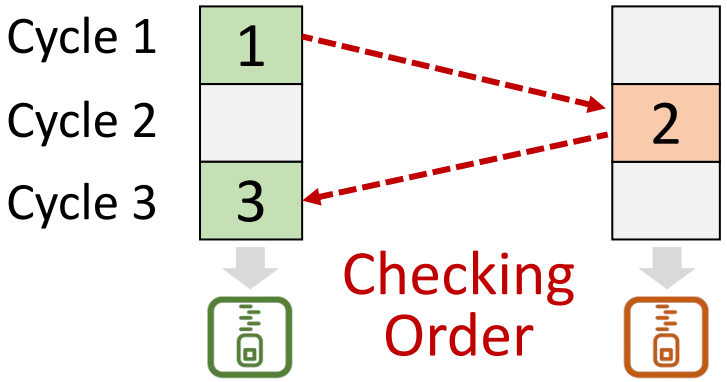
Process latency is hidden by Async Transmission

Toward Fast & Debug-friendly Communication



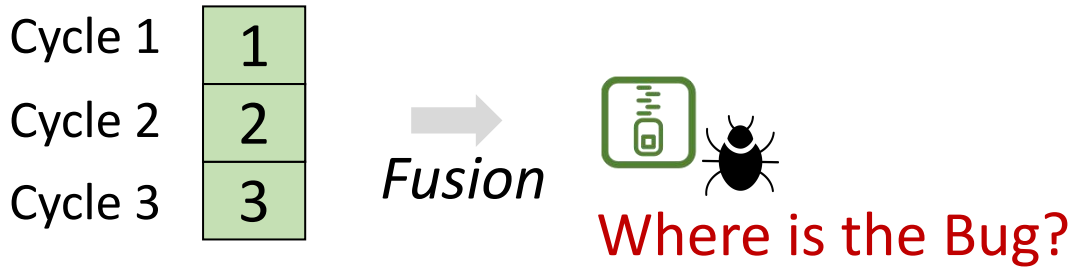
- ***Challenge 1:*** Packing Under Structural Diversity
  — 170× Length Diversity + Per-cycle Event Variation

Toward Fast & Debug-friendly Communication



- ***Challenge 1:*** Packing Under Structural Diversity
  — 170× Length Diversity + Per-cycle Event Variation
- ***Challenge 2:*** Fusion Under Order Constraints
  — Same-type Fusion vs. Cross-type Ordering?

Toward Fast & Debug-friendly Communication



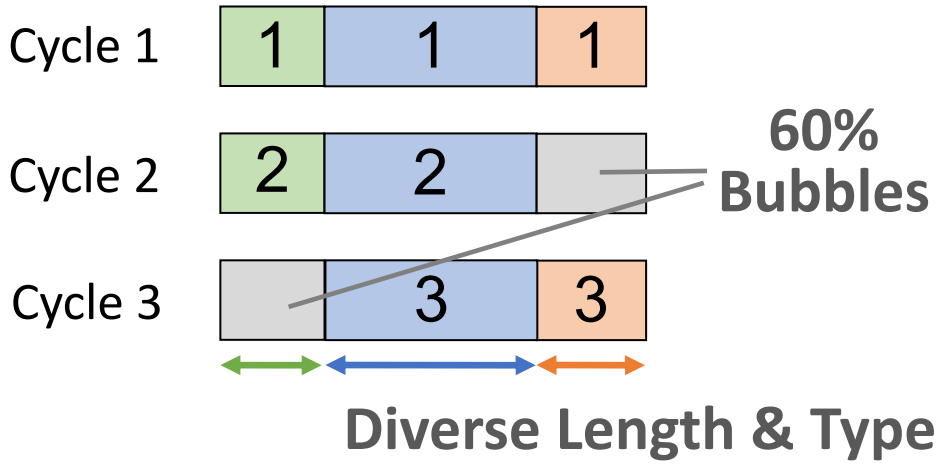
- ***Challenge 1:*** Packing Under Structural Diversity
  — 170× Length Diversity + Per-cycle Event Variation
- ***Challenge 2****:* Fusion Under Order Constraints
  — Same-type Fusion vs. Cross-type Ordering?
- ***Challenge 3****:* Debugging after Fusion
  — Instr-level Debugging vs. Fusion Detail Loss?

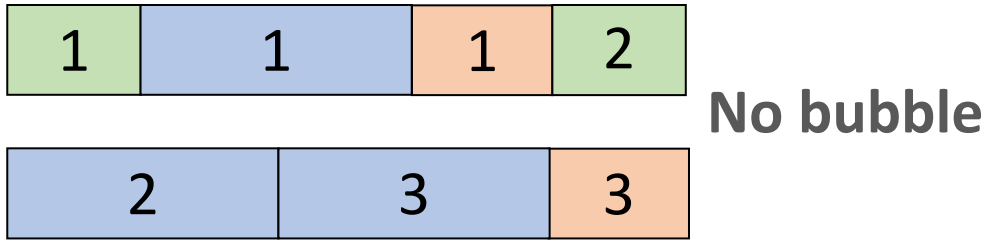
DiffTest-H Overview

A Semantic-aware, Hardware-accelerated Framework
Toward Fast & Debug-friendly Communication

Challenge 1 (Frequency) Packing Under Structural Diversity	***Batch: Structure-wise Packing***
Challenge 2 (Data Volume) Fusion Under Order Constraints	***Squash: Order-decoupling Fusion***
Challenge 3 (Debuggability) Debugging after Fusion	***Replay: Instruction-level Debugging***

Batch: Packing under Structural Diversity





HW Pack: Offset from length
SW Unpack: Extract by type

Prior: Fixed Space

Padding invalid space with bubbles

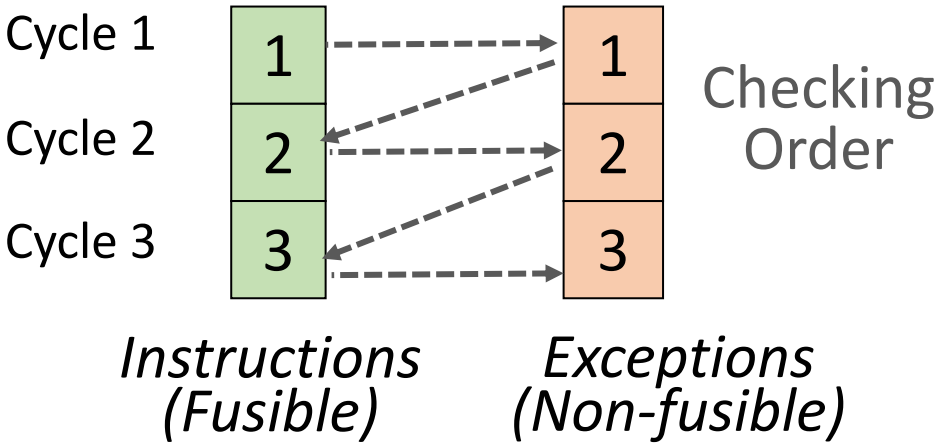
60% Bubbles, **1.67×** Frequency

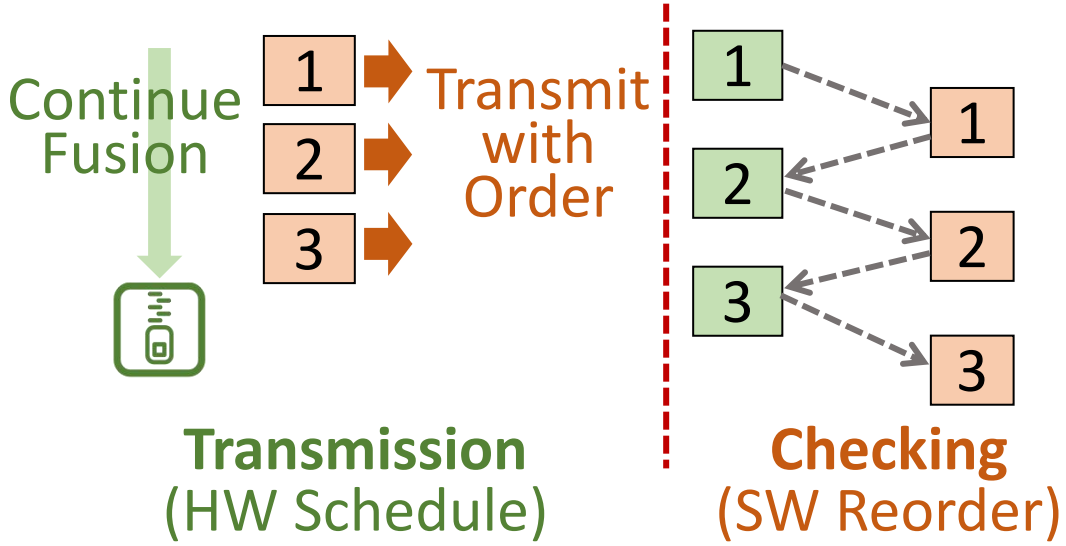
Now: Structure-wise Packing

Packing with length & type

No bubble, Less frequency

Squash: Order-decoupling Fusion





Prior: Order Constraints

e.g. Exception after specific instrs

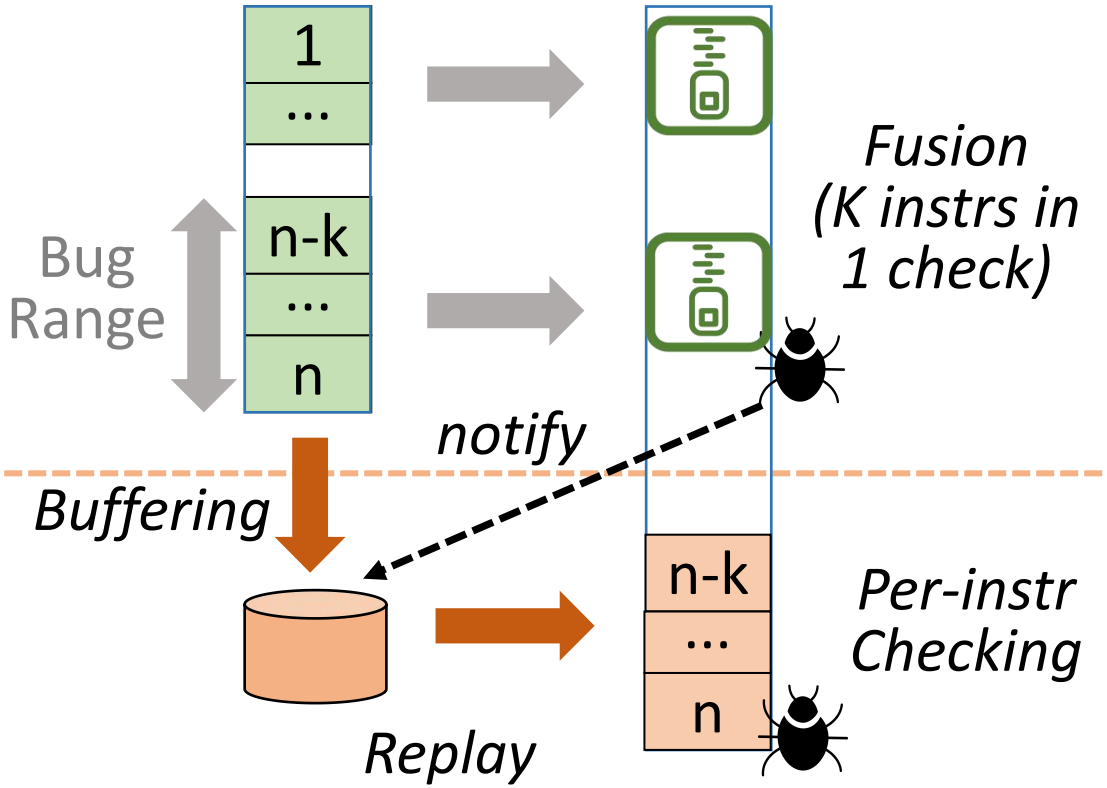
Frequent Fusion Breaks

Now: Order-decoupling Fusion

Decouple fusion & checking order

Better fusion ratio, Less Data

Replay: Instruction-level Debugging



Prior: Where is the Bug?

K instrs in 1 fused check

Hard for bug localization

Now: Instr-level Debugging

Per-instruction replay

Localize bug to single instr

DiffTest-H Workflow

Accelerated Co-simulation:

①**Monitor**: Capture DUT events

③**Squash**: Fusion for less data

④**Batch**: Packing for less frequency

⑤**Nonblock**: Overlap SW latency

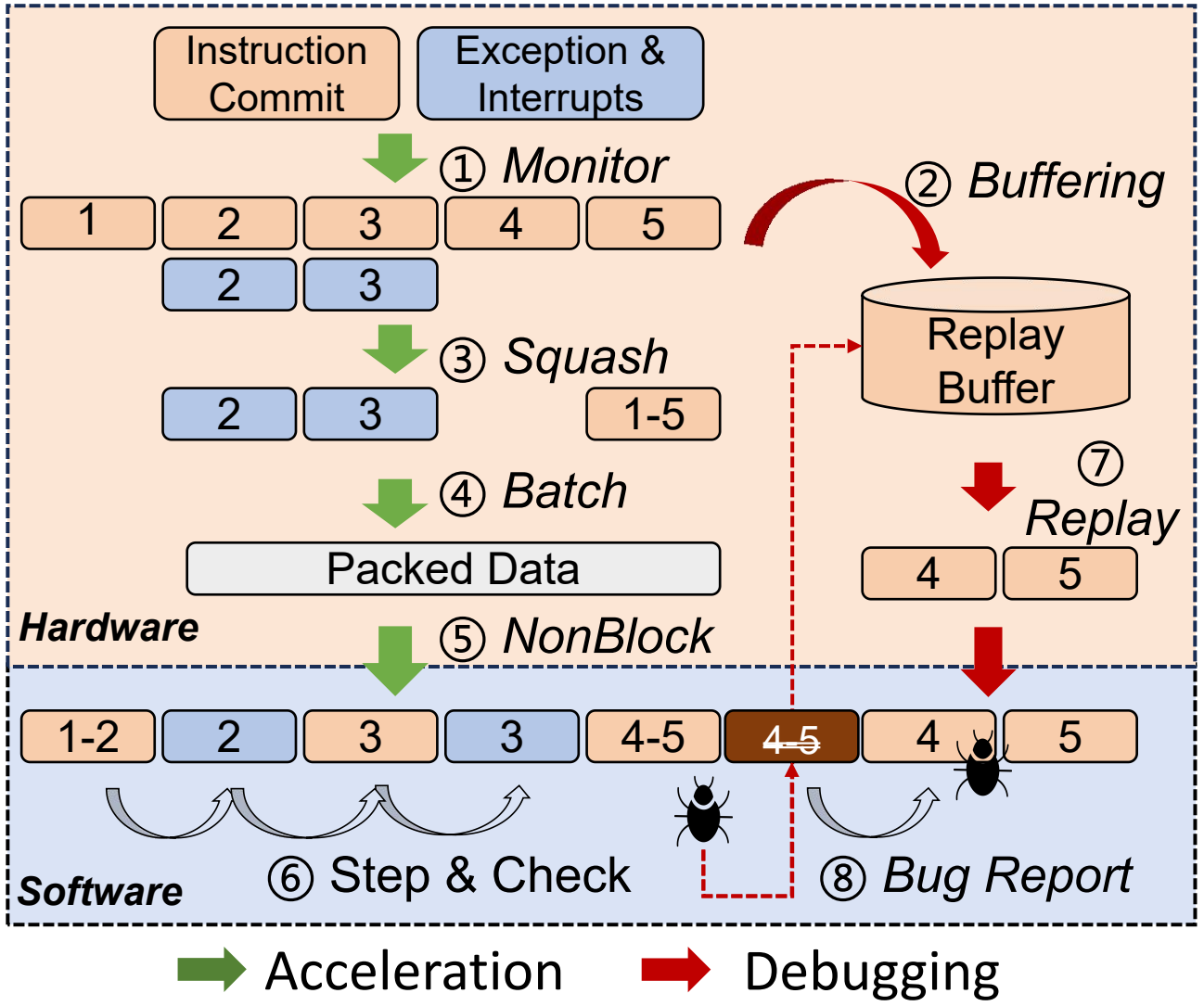
⑥**Check**: Compare DUT vs. REF

Debugging on Error:

②**Buffering**: Backup unfused events

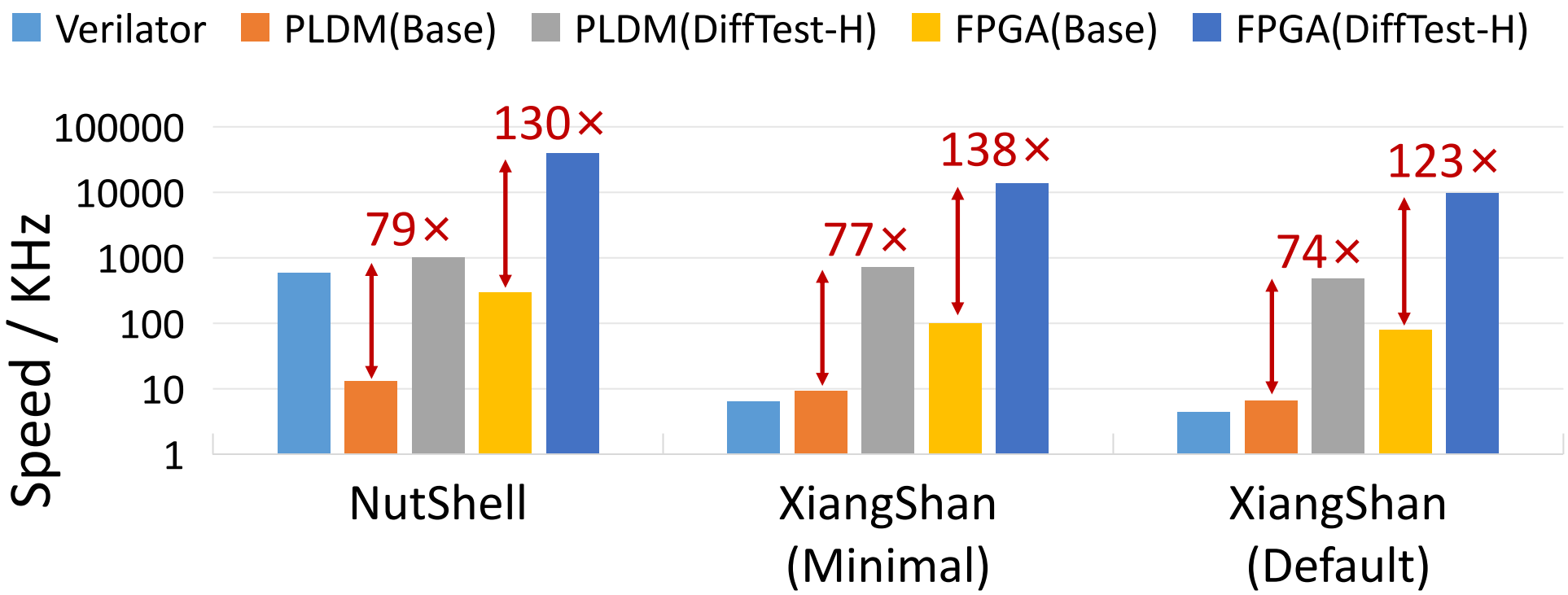
⑦**Replay**: Recheck unfused events

⑧**Report**: Bug localization

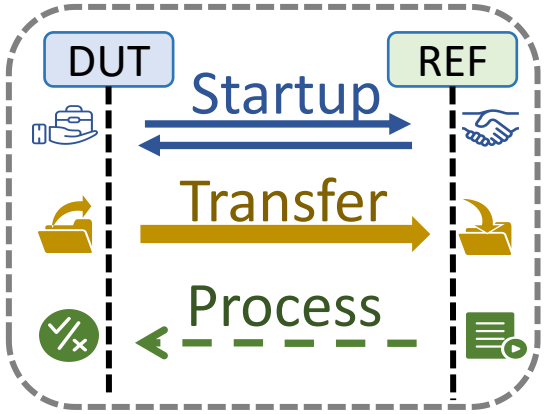


Performance Evaluation

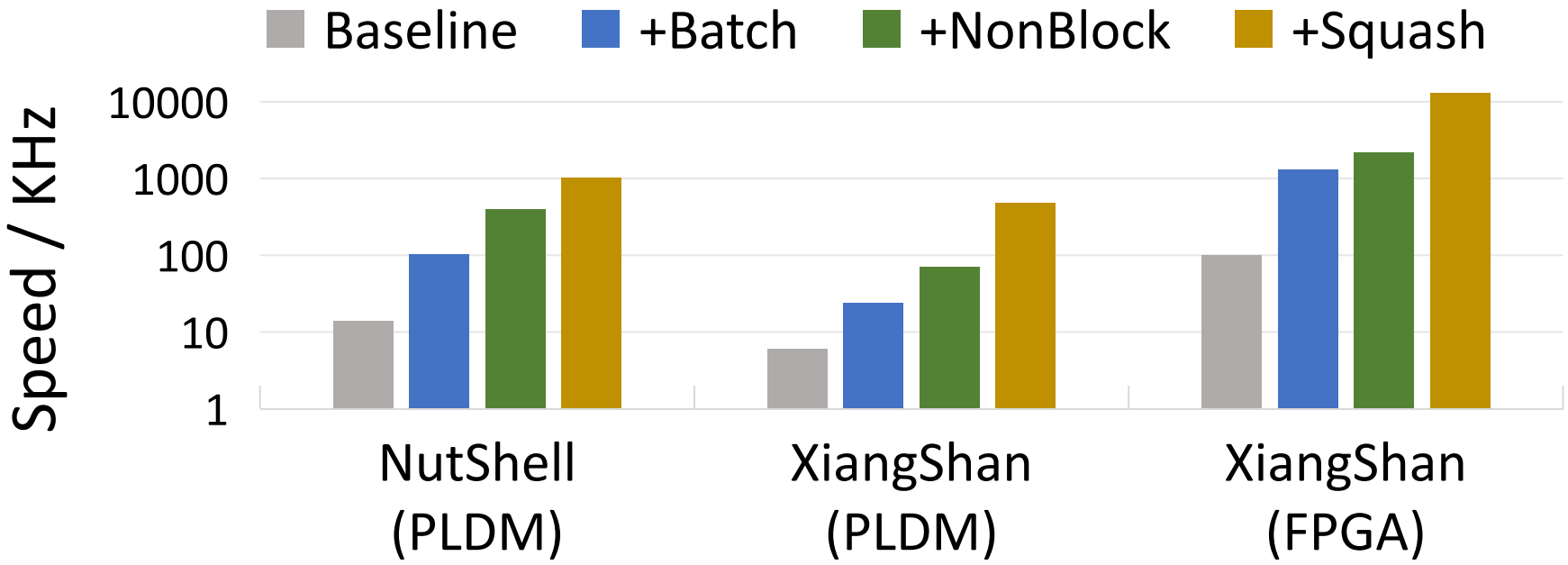
XiangShan: **13.8 MHz**@FPGA, **2,300× faster** than Verilator



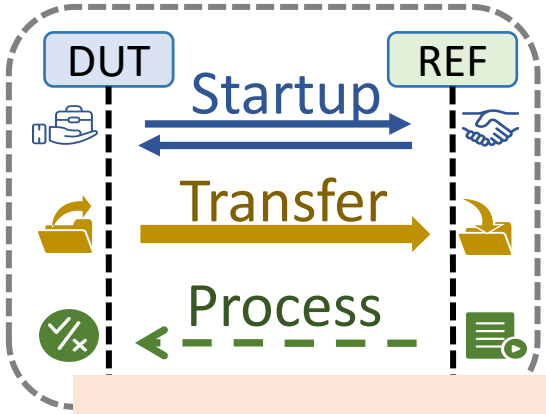
Optimization Breakdown



- **Frequency**: **Batch** reduces by **43x**
- **Process Latency**: **NonBlock** overlaps with async
- **Data Volume**: **Squash** reduces by **47x**

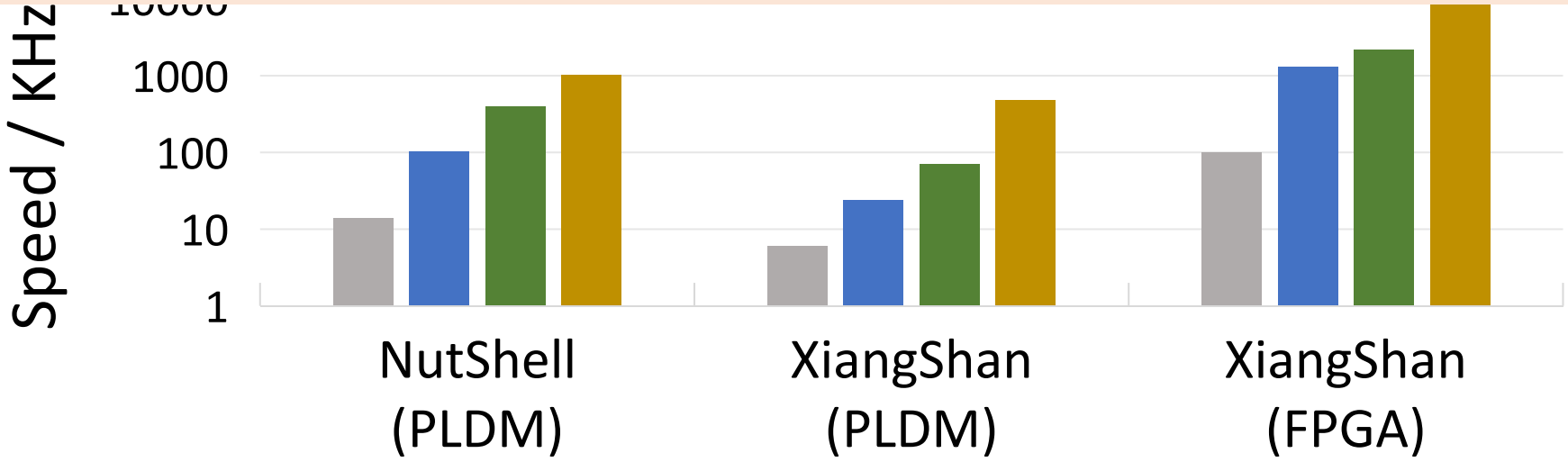


Optimization Breakdown



- **Frequency**: **Batch** reduces by **43x**
- **Process Latency**: **NonBlock** overlaps with async
- **Data Volume**: **Squash** reduces by **47x**

74 ×-138 × speedup, 98.0%-99.8% overhead reduction



Comparison: Fast & Debug-friendly

- **13.8× faster** than FPGA SOTA
- **32** verification event with **1.2KB** size, **Instruction-level** Debugging

Platform	Work	Checking Types	Checking Size B/cycle	Co-sim Speed
Emulator	IBI-Check[1]	2	7	80 KHz
	This work	32	1200	**478 KHz**
FPGA	Fromajo[2]	7	24	1 MHz
	This work	32	1200	**13.8 MHz**

[1] Chatterjee et al. Checking architectural outputs instruction-by-instruction on acceleration platforms. (DAC'12)
[2] Zhang et al. Integrating a High-Performance Instruction Set Simulator with FireSim to Co-simulate Operating System Boots. (ASPLOS '23 Workshops)

Bug Discovery: 151 Bugs Uncovered in XiangShan

- In the past 6 months, uncover **151 Bugs** in XiangShan
- Over **780+ lines** change with 19 Pull Request

Exception & Interrupts

fix(LoadUnit): fix misalign load wrong wakeup ✓
type: bug/confirmed
#4263 by cz4e was merged on Feb 16

fix(LSU): fix misalign store exception logic ✓
#4239 by Anzooooo was merged on Jan 26

fix(CSR): fix
#4157 by sinceforYy was merged on Jan 10

fix(interrupt): `vset` should not respond to interrupts ✓
#3991 by Anzooooo was merged on Dec 7, 2024

fix(VecExcp): isEnqExcp should be set 0 when writeback has older exception ✓
#3778 by huxuan0307 was merged on Oct 24, 2024

fix(exception): fix exception vaddr generate logic ✓
#3639 by good-circle was merged on Sep 27, 2024

Memory hierarchy & coherence

fix(LoadUnit): uncache should not be generated when page fault ✓
#4442 by Anzooooo was merged on Mar 20

fix(StoreUnit): no uncache store misalign of mmio ✓
#4441 by Anzooooo was merged on Mar 20

fix(StoreQueue): fix the `vecExceptionFlag` setting

fix(TLB): avoid

fix(ICache): block waylookup if there is a pending gpf ✓
#3719 by ngc7331 was merged on Oct 12, 2024

fix(TLB, RVH): delete the s1tagfix which maybe cause the tag check to fail ✓
#3685 by pxk27 was merged on Sep 30, 2024

submodule(CoupledL2): bump CoupledL2 ✓
#3621 by linjuanZ was merged on Sep 21, 2024

Vector Control Logic

fix(vector): do not set vs.dirty for some type of vecInsts ✓
#3965 by Tang-Haojin was merged on Dec 1, 2024

fix(vstart): fix vstart wrong update when other instruction handling interrupt ✓
#3887 by Tang-Haojin was merged on Nov 18, 2024

feat(rv64v): fix

feat(rv64v): Added support for vector's vstart, first-only-fault, trigger ✓

feat(Difftest): add multi-core vector load check ✓
#4361 by Anzooooo was merged on Mar 7

fix(vtypegen): fix initial condition after receive redirect ✓
#3664 by Ziyue-Zhang was merged on Sep 27, 2024

fix(VLSU): fix bug in flush of pipeline connect & skid buffer ✓
#3646 by weidingliu was merged on Sep 26, 2024

DiffTest-H Summary

- **DiffTest-H:** Semantic-Aware, Hardware-Accelerated Framework
  - **13.8 MHz**@FPGA, **Instruction-level** Debugging
  - Deployed in **XiangShan**, with **151 bugs uncovered**
  - **Open-Source** *@github/OpenXiangShan/difftest*

GitHub Repo
DiffTest-H

Batch	***Squash***	***Replay***
Structure-wise Packing	Order-decoupling Fusion	Instruction-level Debugging

XIANGSHAN

Thanks for Your Attention