



SHAPING THE NEXT GENERATION OF ELECTRONICS

**JUNE 23-27, 2024**

MOSCONE WEST CENTER  
SAN FRANCISCO, CA, USA

# A Distributed Co-Simulation Environment and its Application in HW-FW Verification

Nitin Pundir, Pretty Mariam Jacob, Arun Joseph, Viresh Paruthi, Sandeep Korrapati, Ali Elzein, Zoltan Hidvegi, Bodo Hoppe

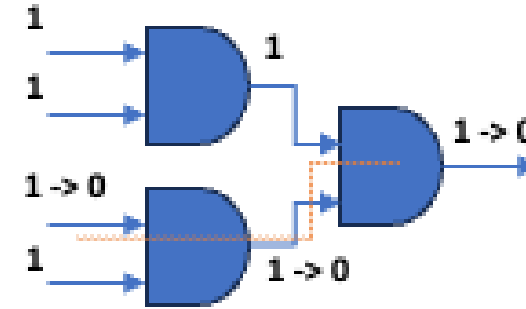
IBM Infrastructure



# Motivation

## Event Simulation

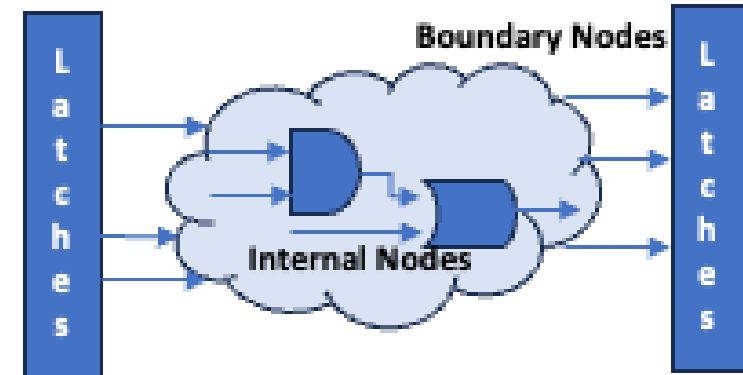
- Delay and timing information included
- Changes at time  $t$  triggers an event at  $t + \Delta t$
- Flexible and can handle asynchronous circuits
- Driven by UVM/System Verilog testbenches



*Fig. 1: Event Driven Simulation - Where changes trigger events down the line*

## Cycle Simulation

- Delay and timing information **not** included
- Assumes switching is synchronous and evaluates all gates each cycle
- Runs at constant speed regardless of model activity
- Sim is faster, scales better, and can support hardware acceleration
- Driven by performant C++ testbenches



*Fig. 2: Cycle Simulation - Boundary nodes get evaluated every cycle*

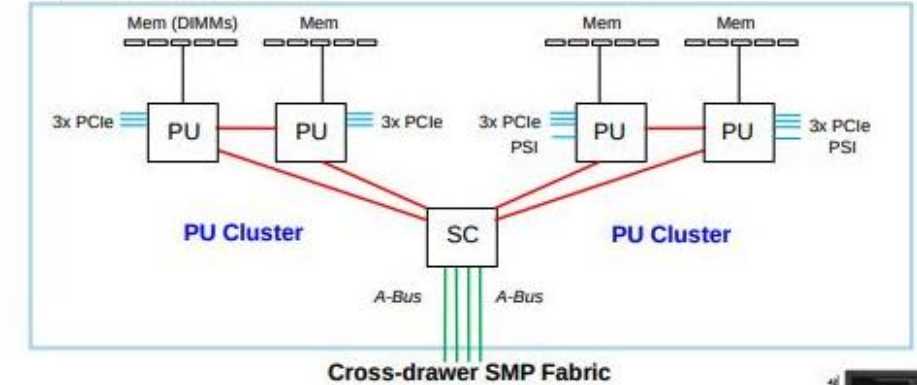
# Hybrid Solution

- Leverages advantages of both methodologies
- Event simulation at unit level to get timing accurate verification
- Cycle simulation at chip level for high speed and coverage
- XTB sim gets both by allowing timing accurate unit models to simulate with chip cycle models

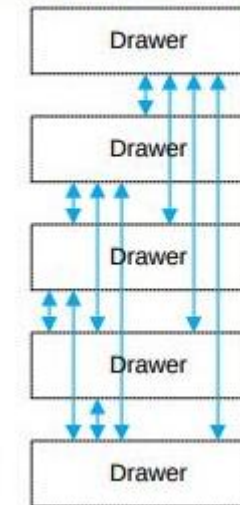
- Allows us to verify complex server chips using cycle simulation.
- While bringing in unit IPs fully verified using timing accurate event simulation
- First commercial case was IBM's POWER9 interfacing with NVIDIA's GPU over NVLink 2.0 protocol

**Reference:** Schubert, K-D., Syed Saif Abrar, Duane Averill, Ellen Bauman, Aaron C. Brown, Ron Cash, Debapriya Chatterjee et al. "Addressing verification challenges of heterogeneous systems based on IBM POWER9." *IBM Journal of Research and Development* 62, no. 4/5 (2018): 11-1.

Fully Populated Drawer



5 Drawer System Fully Interconnected



**Fig. 3:** IBM leverages cycle simulation to simulate server processors and achieve ~100% coverage

# XTB(Cross Testbench): A Distributed Co-Simulation Environment

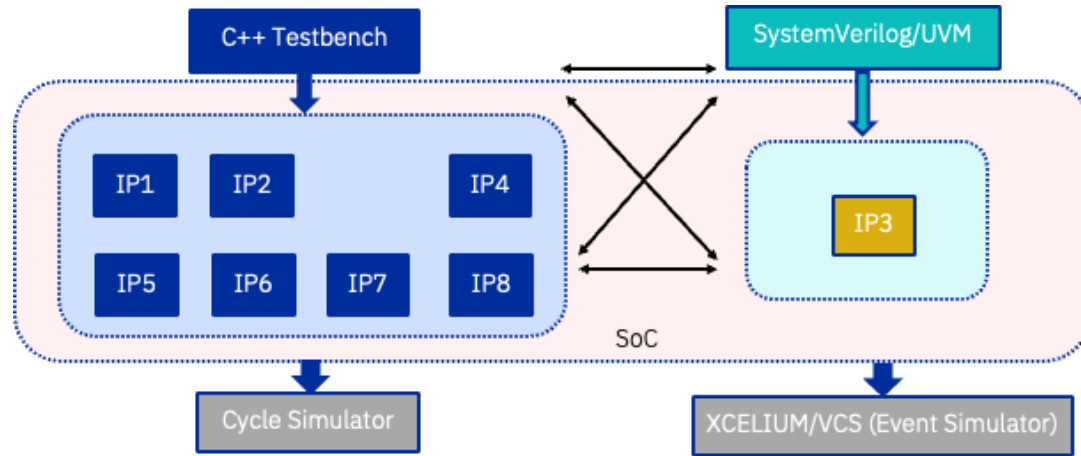


Fig. 4: XTB allows IPs to run in Individual Simulators while allowing to Exchange Values

- Supports DUT-2-DUT, TB-2-DUT, and TB-2-TB communication channels
- One can also target a hierarchical signal/wire deep within the design

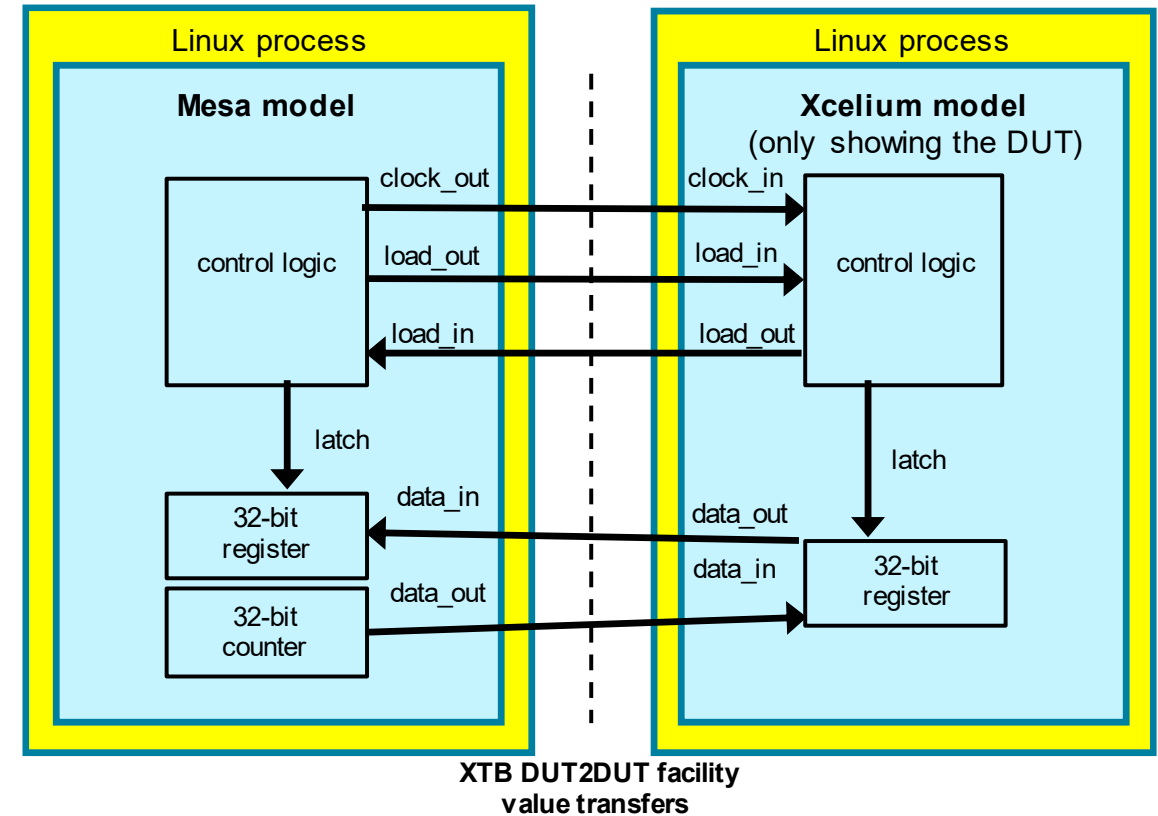


Fig. 5: Illustrative Example

# Timing Information

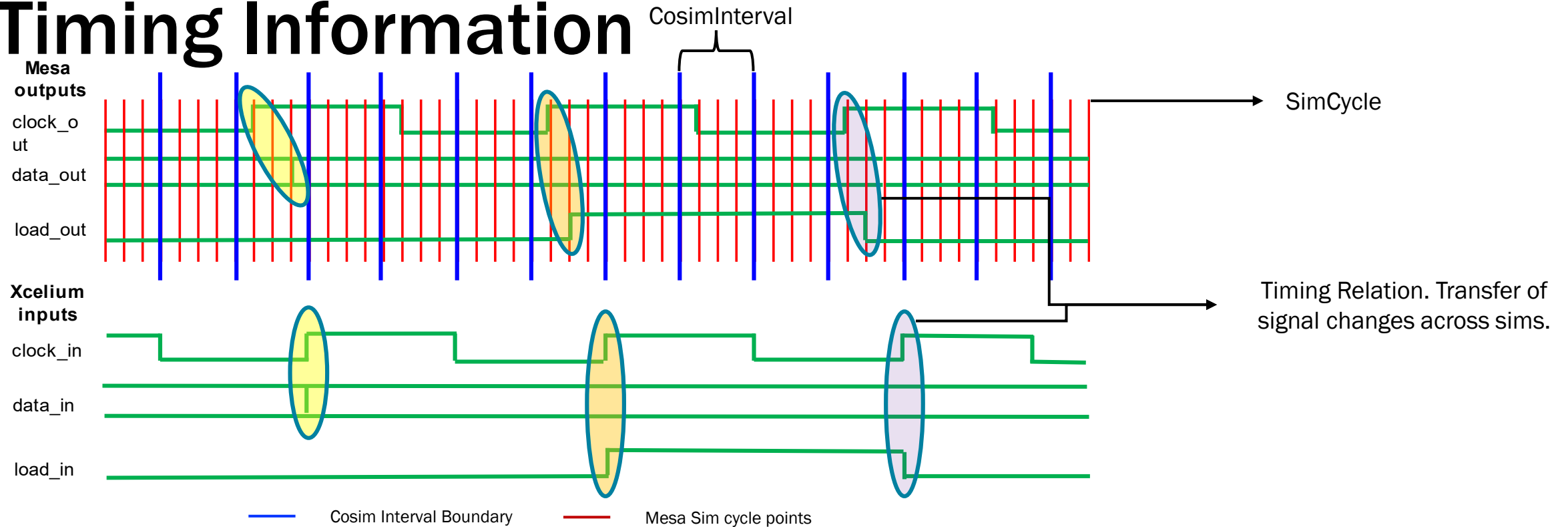


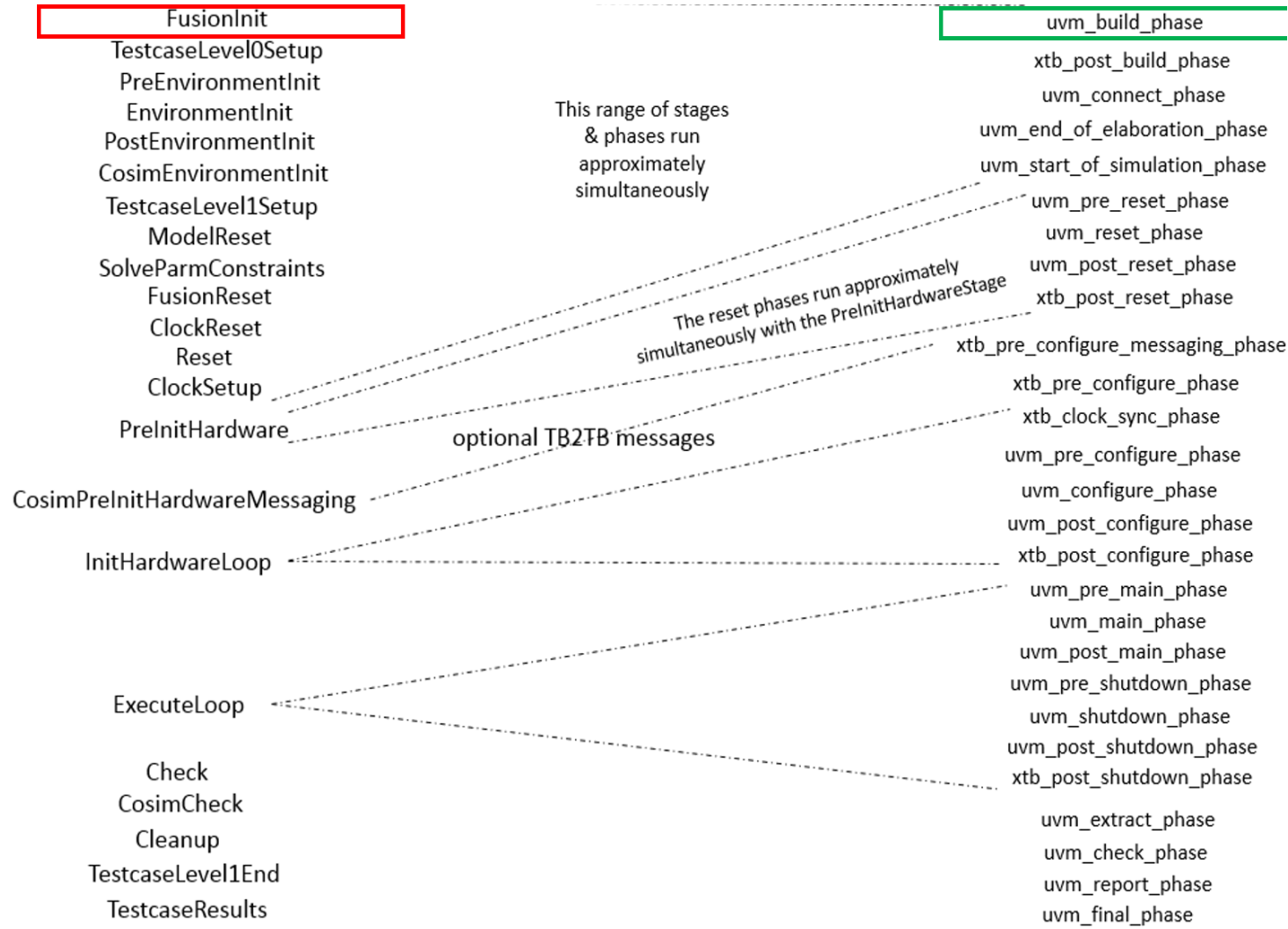
Fig. 6: Illustration of Timing Relationships across Cycle and Event Sims

- $XTB^{[5]}$  exchange facility values on the co-simulation interval boundaries
- Shorter co-sim interval improves the fidelity of the timing relationships
- Longer co-sim interval improves performance, but over time the timing gets corrupted

## How we Setup Timing Information?

1. Fix the frequency of event sim (Usually the fastest permitted speed)
2. Find the clock cycle ratio, i.e., Clock Cycle/Sim Cycles
3. Finalize the Sim Cycle to nanosecond ratio

# Synchronization



- XTB maps a correlation between C++ Testbench stages and UVM phases
- XTB introduces custom stages in between as synchronization stages/phases

Fig. 7: Correlation across UVM phases and Cross Simulation stages

# Checkpointing/Snapshot

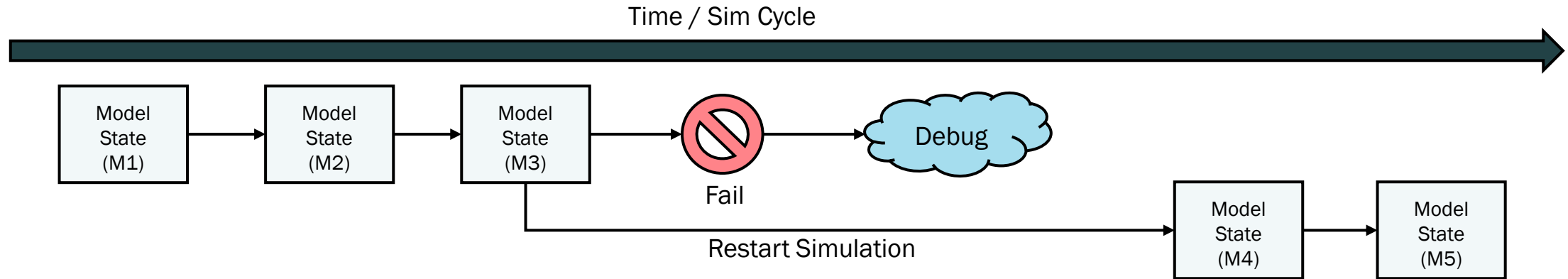


Fig. 8: Illustration of Simulation Restart in Homogenous Environment

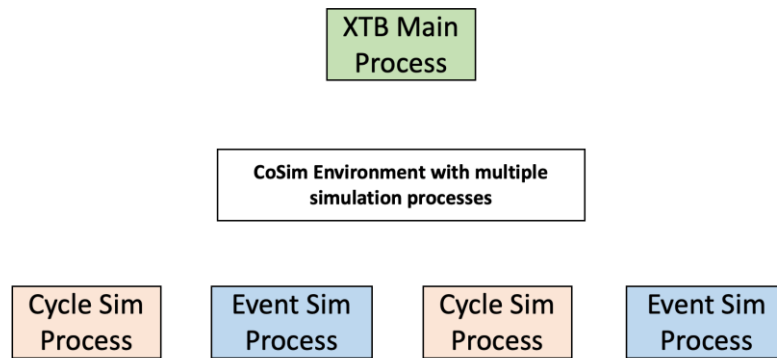


Fig. 9: Snap/Checkpoint of a Co-simulation Environment

- Simulators allows verification to restart from the last known good state of the hardware model
- XTB allows users to save the state of the entire co-simulation environment
- This allows users to restart simulation from the last known good checkpoint
- Beneficial in case of large models taking multiple days to verify

# Evidence

## Use-Case

- SoCs involves IPs from multiple vendors
- Memory buffer to connect to DIMMs needs Phy IPs
- Phy IPs are timing accurate fully verified using event sim at fastest clock
- Hardware and firmware (of both host and vendor IP – VIP ) interact for configuration and initialization of Memory
- Host/Server is Cycle sim compatible, and Vendor IP is event sim compatible
- Cross design environment - To verify the integration

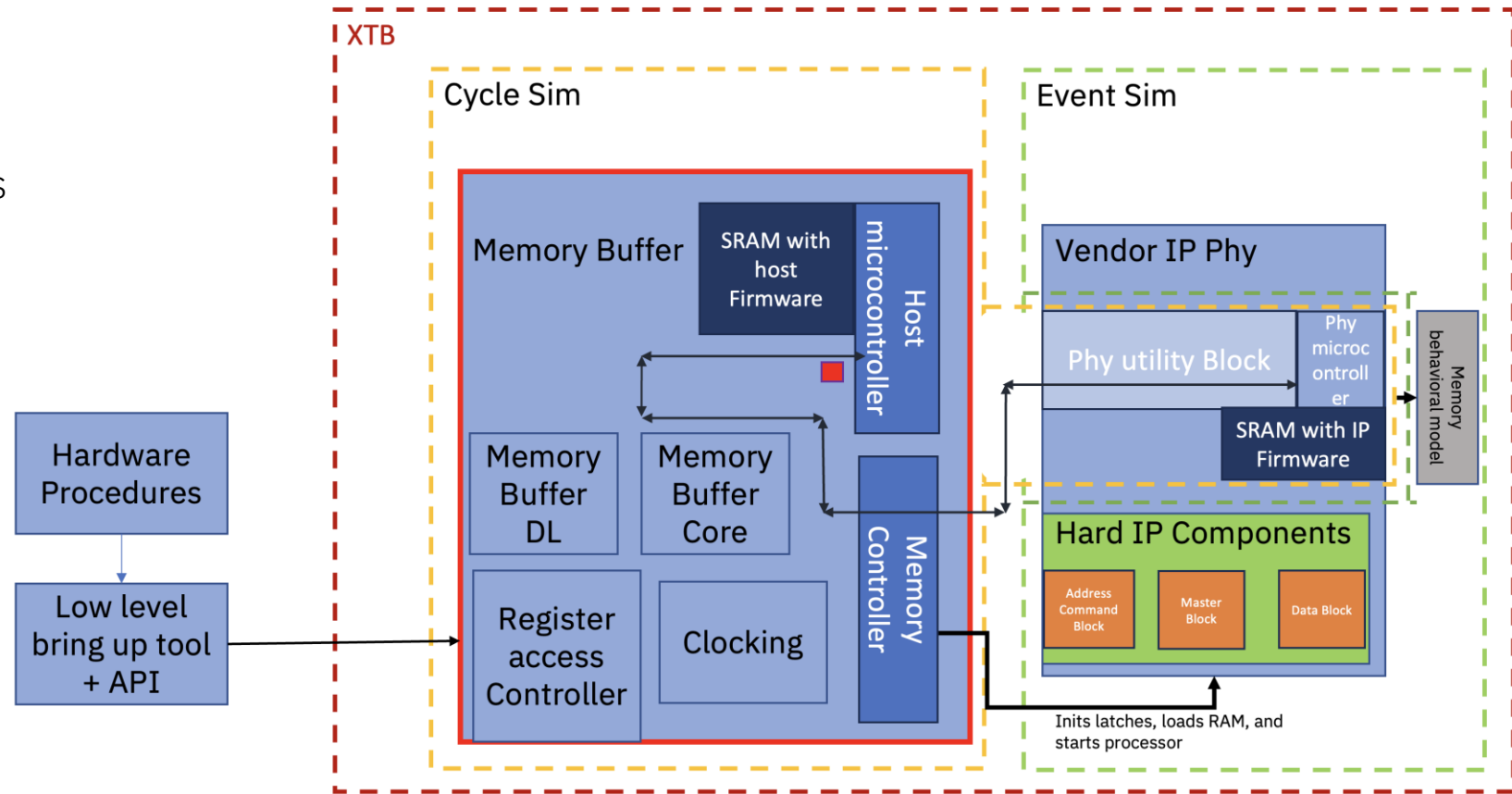
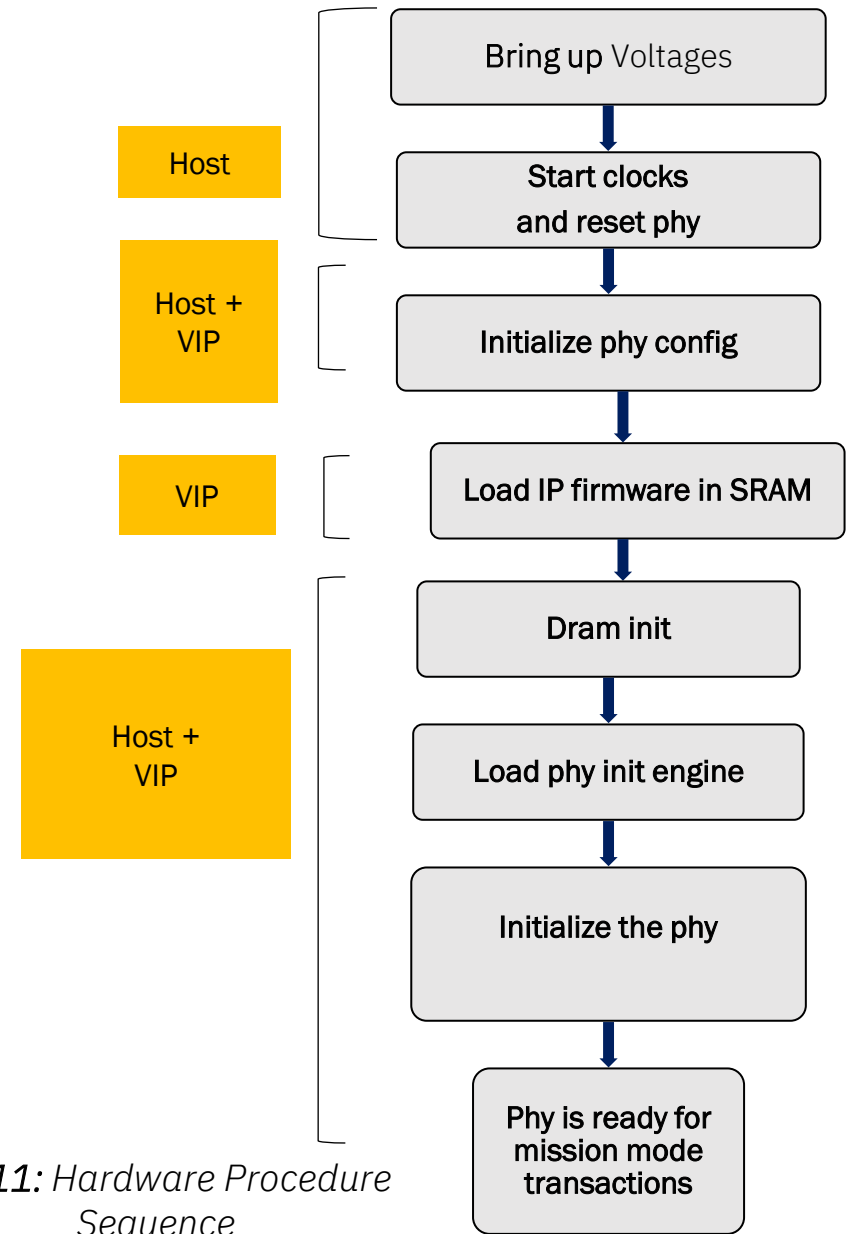


Fig. 10: Integrated Co-Simulation Model

# Evidence

- The host and Vendor IP firmware **interacts** through hardware registers
- Register sequences for boot flow as shown in Fig 11 (called as **hardware procedures**) are compiled and run on co-simulation environment
- Verification of boot flow in lab is strenuous
- Enables earlier development of firmware sequences for booting and quick firmware fixes



*Fig. 11: Hardware Procedure Sequence*

# Evidence Performance Evaluation

	Simulation Time (Hours)
Each Procedure	~20
Total Run Time	~200
Worst Restart Time Without Checkpoint (Failure in last procedure)	~180
Restart Time With X-shot	~ 0.03

Table 1: Debug and Simulation Effort

## Shift-Left Gain From X-Shot Checkpoint/Restart Enablement



- Verifying memory controller involved ~10 procedures
- Each procedure takes ~20 hours and uses model state of the last procedure
- Failure at  $n^{th}$  procedure will cause  $n \times 20$  hours of rerun to reach the failure point
- **Snapshot (X-shot) enablement was key** and **saved days** of simulation and debug effort
- Model restart within seconds (**~20-78 seconds/ 0.03hours**) from  $n - 1^{th}$  procedure state
- **Collaborative Debug ~5 seconds** with Firmware, design and verification teams on shared debug sessions <sup>[6]</sup>

# Summary

- We introduce a distributed co-simulation environment – XTB, which enables cycle and event co-simulation
  - We used it to verify and debug HW-FW interaction in industry class servers like IBM Mainframe and Power
- Additional techniques for save and restart in this cross-simulation environment enables significant reduction of debug time
  - Brings the system to last known good state in minutes instead hours
  - In HW-FW verification, saved hours to reach the failure points after debug
- Solution enables a hybrid simulation framework, important in the context of next generation hardware designs, which are increasingly made with VIP across different vendors, and such hybrid framework enables flexibility to get best of cycle and event sim, and thereby faster time to closure of such systems

# References

- [1] Janabi et.al., "An Overview of Cycle-Accurate, Event-Driven and FullSystems Simulators for Chip-Multiprocessors", International Journal of Computer Theory and Engineering, Vol. 11, No. 6, December 2019
- [2] M.C. Cogswell et. al., "A hybrid event-simulation/cycle-simulation environment for VHDL based designs". [Proceedings VHDL International Users' Forum. Fall Conference](#) 1997
- [3] R. Shanshan, L. Mingche and L. Hongyi, "Cycle-based and event-driven hybrid simulation framework for performance analysis of optical networks on-chip," Proceedings of 2012 2nd International Conference on Computer Science and Network Technology, Changchun, China, 2012.
- [4] R. Ubal, B. Jang, P. Mistry, D. Schaa and D. Kaeli, "Multi2Sim: A simulation framework for CPU-GPU computing," 2012 21st International Conference on Parallel Architectures and Compilation Techniques (PACT), Minneapolis, MN, USA, 2012, pp. 335-344.
- [5] Schubert et al., "Addressing verification challenges of heterogeneous systems based on IBM POWER9," in IBM Journal of Research and Development, 2018
- [6] Joseph et. al., "Cloud Native Hardware Logic Design in Step-wise Refinement Implementation Flows", DAC 2021
- [7] Wolfgang Roesner, "Software Methods meet Large-Scale System-on-a-Chip Design", TCE 2015
- [8] Yael et. al., "Validation of SoC Firmware-Hardware Flows: Challenges and Solution Directions" DAC 2014
- [9] S. Bergman et al., "Designer-level verification — An industrial experience story," DATE 2015
- [10] W. Starke and B. Thompto, "IBM's POWER10 Processor," 2020 IEEE Hot Chips 32 Symposium (HCS), Palo Alto, CA, USA, 2020