Sarcouncil Journal of Engineering and Computer Sciences

ISSN(Online): 2945-3585

Volume- 04| Issue- 10| 2025

Research Article **Received:** 21-08-2025| **Accepted:** 21-09-2025 | **Published:** 17-10-2025

Enabling Shift-Left Firmware Validation through High-Fidelity Simulation for IBM Power Systems

Vijay Francis Gregary Lobo

Independent Researcher, USA

Abstract: The development of advanced computing systems has become a complex process with the evolution of architectures to chiplet-based designs and the integration of AI into hybrid cloud platforms. This article discusses how IBM Power Systems is adopting a shift-left strategy for firmware validation, leveraging high-fidelity simulation frameworks such as WindRiver Simics and QEMU. It can perform thorough verification of firmware components months in advance of even having physical hardware; this approach completely changes the nature of the development cycle. Developers can use detailed digital twins of the critical system components, including management controllers, storage interfaces, and power delivery subsystems, to discover and eliminate integration mistakes, previously only heard of during integration activity. The article discusses the simulation strategy featured in various firmware fields with references to the cross-team collaboration and practice of systematic modeling that allows developing the interdependent components in parallel. Not only does this allow for a faster development schedule, but it is also capable of improving the quality of the product through an increase in the amount validated. The simulation-based approach conforms to Critical and Emerging Technology objectives and provides a framework that would maintain ongoing job creation in the enterprise computing platforms.

Keywords: Firmware Validation, Simulation-Driven Development, Digital Twins, Shift-Left Methodology, Enterprise Computing Platforms.

INTRODUCTION

Firmware acts as the gap between the innovation of hardware and the software performance. The complexity of firmware development has grown exponentially in current computing platforms, particularly the ones that are aimed at hybrid clouds, AI as well as enterprise. Firmware complexity has increased exponentially due to higher levels of system integration, the proliferation of specialized accelerators, diverse computing elements, and advanced power management functionalities. Recent analysis of firmware evolution patterns across multiple generations of enterprise systems shows this complexity manifests in both code size expansion and increasing interdependencies between firmware components (Zhang, Y. *et al.,* 2024). IBM's Power Systems have embraced simulation-first strategies to mitigate hardware dependencies during development. This method has changed the traditional development cycles because firmware can be validated significantly before the availability of a physical piece of hardware available to the architecture leading to the possibility of architectural optimizations and defect avoidance as opposed to detection.

The shift-left process aims to verify and refine firmware significantly earlier in the product lifecycle, sometimes even before silicon chips are manufactured. The changes in methodology through this approach essentially transform the working nature of firmware teams, whereby the teams are able to spot and resolve integration challenges that would still go undetected until later stages of hardware integration. Comprehensive studies of verification methodologies demonstrate that simulation environments create a controlled space where engineers can test boundary conditions, error recovery mechanisms, and performance characteristics that would be difficult to reproduce in physical testing environments (Nahir, A., Ziv, A. *et al.,* 2010). The simulation frameworks deployed across IBM's Power development ecosystem incorporate detailed models of component behaviors and protocol implementations. These models undergo continuous refinement as hardware specifications evolve, ensuring simulation fidelity remains aligned with expected silicon behavior. WindRiver Simics and QEMU serve as the foundational simulation engines, handling system-level and processor-specific modeling, with the ability to create comprehensive digital twins of complex subsystems, including BMCs, VRMs, and high-speed storage interfaces.

This article examines how these simulation frameworks establish a robust pre-silicon validation ecosystem that has become essential to maintaining quality metrics while addressing ever-increasing firmware complexity in enterprise computing platforms. By implementing

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

***Corresponding Author:** Vijay Francis Gregary Lobo

DOI- https://doi.org/10.5281/zenodo.17377972

simulation-driven development practices, firmware teams gain the ability to validate critical system behaviors—from power sequencing to boot flow validation—months before hardware availability, fundamentally transforming the reliability and maturity of enterprise systems at launch.

SYSTEM MODELING

This section consolidates the system modeling components into a unified framework. It highlights that the subsections below each represent a distinct model within the broader system, with the focus on only a few critical and illustrative modeling efforts. Together, these subsections form the core of the system modeling efforts discussed in this document—each is essential for high-fidelity simulation-driven firmware validation, yet represents only a subset of the broader modeling portfolio required for comprehensive system emulation.

Aspeed AST2700 BMC Model Development

Baseboard Management Controllers (BMCs) serve as foundational components for server manageability in modern enterprise computing platforms. These specialized System-on-Chip (SoC) solutions deliver comprehensive out-of-band management capabilities, including remote monitoring, diagnostics, and recovery functions essential for data center operations. IBM's approach to modeling BMCs for next-generation Power systems implements sophisticated hardware abstraction techniques that create well-defined boundaries between hardware and software components, enabling concurrent development across multiple teams while maintaining system coherence (Yoo, S., & Jerraya, A. A. 2003).

The BMC simulation framework incorporates comprehensive peripheral modeling for critical interfaces including I2C, SPI, GPIO, UART, and Ethernet. Each interface model captures not only the functional behavior of the corresponding hardware but also error conditions and protocol-specific nuances. The simulation environment includes detailed models of secure boot path logic, reflecting the increasing emphasis on hardware-based security foundations in enterprise computing platforms. This includes simulation of cryptographic verification sequences, key management infrastructure, and protection against unauthorized firmware modifications.

Integration with HostBoot and Redfish firmware layers represents a particularly valuable aspect of the BMC simulation. The model implements the complete API surface required for host-BMC interactions, including the proprietary Fru Serial Interface (FSI), a two-wire IBM proprietary interface and protocol that enables low-level communication between the main processor and the service processor. This approach to comprehensive interface modeling builds upon established research in hardware/software co-simulation, where modeling the interactions between heterogeneous components presents unique verification challenges that benefit from systematic simulation methodologies (Aversa, R. *et al.*, 1996).

The BMC simulation model enabled full firmware stack bring-up in a simulated environment, even before BMC silicon samples were available from vendors. This capability delivered significant schedule advantages by decoupling firmware development from hardware availability constraints. The key outcome was early validation of host-to-BMC interactions, including FSI transactions, boot sequencing, and telemetry systems. This approach enabled firmware teams to identify and resolve integration issues that would traditionally only be discovered during hardware bring-up, significantly reducing validation cycles and accelerating platform readiness.

Table 1: Aspeed AST2700 BMC Simulation Components and Their Benefits (Yoo, S., & Jerraya, A. A. 2003; Aversa, R. *et al.*, 1996)

Component Category	Simulation Elements	Benefits
Peripheral Interfaces	I2C	Pre-silicon interface validation
	SPI	Controlled error injection
	GPIO	Protocol conformance testing
	UART	Timing analysis without hardware
	Ethernet	Early protocol interoperability testing
Security Features	Secure Boot Path Logic	Security vulnerability detection pre-silicon
	Cryptographic Verification	Chain-of-trust validation in simulation
	Key Management	Cryptographic implementation verification
Firmware Integration	HostBoot Interface	API boundary testing without hardware

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

	Redfish API	Cross-team parallel development
	Communication Protocol	Protocol edge case discovery
Boot Process	Boot Sequencing	Boot path coverage analysis
	Telemetry	Telemetry data verification
	System Monitoring	Error recovery scenario testing

UFS Simulation for Firmware Consumption

Universal Flash Storage (UFS) has become a critical component in high-speed, low-latency storage infrastructures, offering superior performance characteristics compared to legacy storage technologies. For IBM's Power development ecosystem, a comprehensive UFS model was designed using WindRiver Simics to reflect real-world device behavior with high fidelity. This simulation approach leverages concepts from storage virtualization research, where the abstraction of physical storage characteristics enables more flexible and resilient implementations across diverse computing environments (Badshah, A. 2025). The UFS simulation strategy implements many of these virtualization principles by creating a software-defined representation of storage hardware that can be validated independently of physical components.

The UFS simulation model implemented for IBM's Power development ecosystem incorporates detailed representations of SCSI command sets and descriptors, ensuring that all mandatory and optional commands are properly recognized and processed according to specification. Particular attention was given to endian handling and status reporting mechanisms, which represent common sources of integration challenges when implementing UFS controllers. The simulation environment provides complete visibility into command processing sequences, enabling firmware developers to trace execution paths that would be difficult to instrument in physical implementations.

A notable advancement in the UFS simulation framework is its ability to support simulated boot operations from UFS for the service processor. This capability enables validation of the complete initialization sequence, including firmware loading, descriptor table processing, and transition to operational states. The simulation environment also supports error injection capabilities, allowing firmware developers to validate recovery mechanisms for scenarios that would be difficult to reproduce with physical hardware. These capabilities reflect emerging approaches to fault-tolerant system design, where proactive error detection and handling mechanisms are integrated throughout the storage stack to enhance system resilience (Agarwal, V. 2025).

The UFS model's key outcome has been ensuring bug-free storage initialization across boot flows, accelerating time-to-validation and reducing post-silicon issues. By enabling complete firmware validation in simulation, the model has eliminated hardware-dependent debugging cycles and provided a stable platform for iterative firmware development. This approach has demonstrated particular value for handling edge cases in protocol implementations and ensuring robust operation across diverse operating conditions.

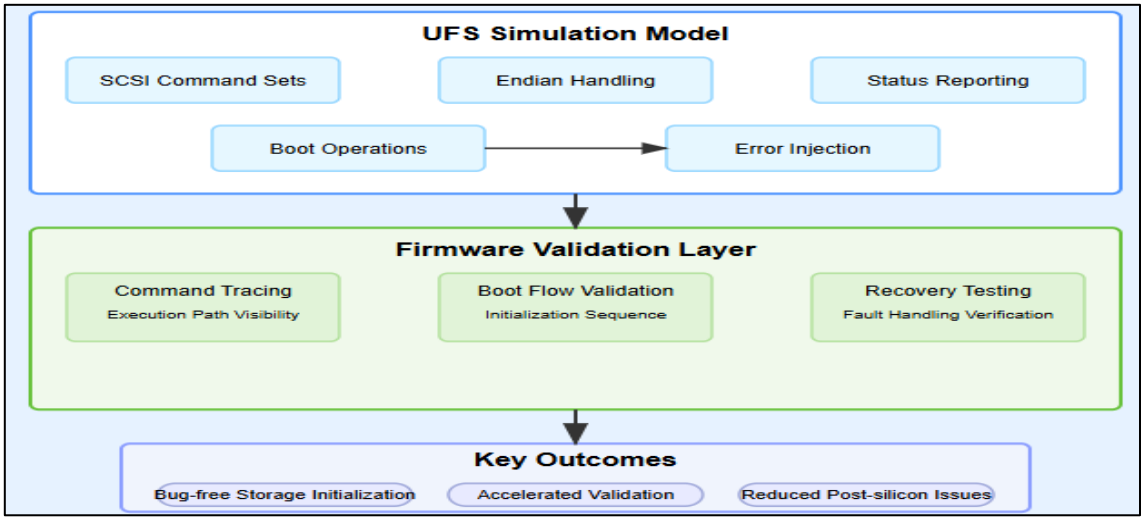


Fig 1: UFS Simulation Framework for Power12 Firmware Validation (Badshah, A. 2025; Agarwal, V. 2025)

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

PMBus and VRM Controller Modeling

To model dynamic power scaling and system health telemetry in IBM's Power systems, comprehensive simulation models of PMBus-based Voltage Regulator Modules (VRMs) were developed and integrated into the broader simulation environment. This approach recognizes the critical role of power management in modern computing platforms, where energy efficiency and thermal characteristics directly impact both performance capabilities and operational costs. The simulation models capture the complex behavioral characteristics of power delivery subsystems, including detailed implementation of the Power Management Bus (PMBus) protocol specifications that define communication between system controllers and power conversion devices.

The VRM simulation framework implements sophisticated voltage and current rail monitoring capabilities, modeling the telemetry mechanisms that provide real-time visibility into power delivery subsystems. This includes accurate representation of analog-to-digital conversion processes, filtering algorithms, and reporting mechanisms defined in the PMBus specification. These capabilities build upon established research methodologies in power-aware computing, where detailed power measurement instrumentation enables systematic analysis of consumption patterns and informs development of optimized power management policies across varied computational workloads (Haidar, A. *et al.*, 2017). The simulation environment also incorporates comprehensive fault injection capabilities, allowing firmware developers to validate detection and recovery mechanisms for scenarios such as over-voltage, under-voltage, over-current, and thermal excursions.

A particularly valuable aspect of the VRM simulation is its implementation of dynamic Voltage Identification (VID) response mechanisms. These features model how voltage regulators respond to changing VID commands from the processor, including transient behaviors during voltage transitions. The simulation captures timing characteristics, slew rate limitations, and stability considerations that impact system behavior during dynamic voltage scaling operations. This approach to power subsystem modeling draws inspiration from datacenter-scale power simulation methodologies, where high-fidelity modeling of power conversion efficiencies and thermal characteristics enables more accurate prediction of system behavior under diverse operating conditions (Sankar, S. *et al.*, 2013).

Firmware development teams leveraged these simulation capabilities to validate power-up sequencing logic, ensuring that voltage rails activate in the correct order and with appropriate timing to prevent damage to silicon components. The simulation environment also enabled integration and testing of real-time voltage scaling policies, which dynamically adjust operating voltages based on workload characteristics and thermal conditions. Additionally, teams used the simulation to develop and validate telemetry collection mechanisms for system health analysis, ensuring that power-related data is accurately captured and presented to management software.

HostBoot Enablement on Simulation

IBM Power systems' simulation framework provides robust support for HostBoot firmware - the essential initialization software that transitions Power processor cores from reset to operational status. This capability marks a genuine breakthrough in firmware validation approaches, permitting thorough end-to-end verification of boot sequences without depending on physical hardware. The environment effectively creates a functional digital replica of processor initialization, capturing the intricate interactions between various hardware subsystems and firmware elements throughout the entire boot process.

At the heart of HostBoot simulation lies detailed modeling of the Fru Serial Interface (FSI) master/slave architecture. This IBM-proprietary communication protocol facilitates vital exchanges between service processors and main processors during system initialization. The simulation captures FSI transaction mechanics, allowing accurate representation of communication pathways that orchestrate platform initialization. Such interface simulation approaches borrow concepts from mechanism and machine theory, where meticulous modeling of kinematic relationships forms the foundation for understanding complex system behaviors in engineering contexts (Kakavand, M., & Nikoobin, A. 2020).

The framework additionally implements thorough modeling of the Self Boot Engine (SBE) - a specialized processor component responsible for early platform initialization tasks, including memory subsystem training and configuration.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

Hostboot enables validation of memory initialization sequences before hardware becomes available - traditionally a challenging aspect of platform bring-up that frequently reveals integration issues between memory controllers, DRAM devices, and firmware. Hostboot simulation encompasses detailed modeling of memory training algorithms, configuration register programming sequences, and error detection mechanisms.

Hypervisor handoff modeling represents another crucial simulation aspect, capturing the sophisticated transition from firmware control to operating system execution. This includes simulation of hardware state transitions, register configurations, and memory map preparations needed for successful execution control transfer. The complete environment enables comprehensive boot simulations, allowing firmware teams to verify entire flows from initial power-on through service processor initialization, HostBoot execution, and final hypervisor handoff. Such capabilities dramatically accelerate firmware validation cycles by providing extensive visibility into boot sequences and enabling systematic validation of platform initialization flows, comparable to approaches described in secure boot methodology research where each boot stage's integrity requires verification before transferring execution control (Sood, K. *et al.*,).

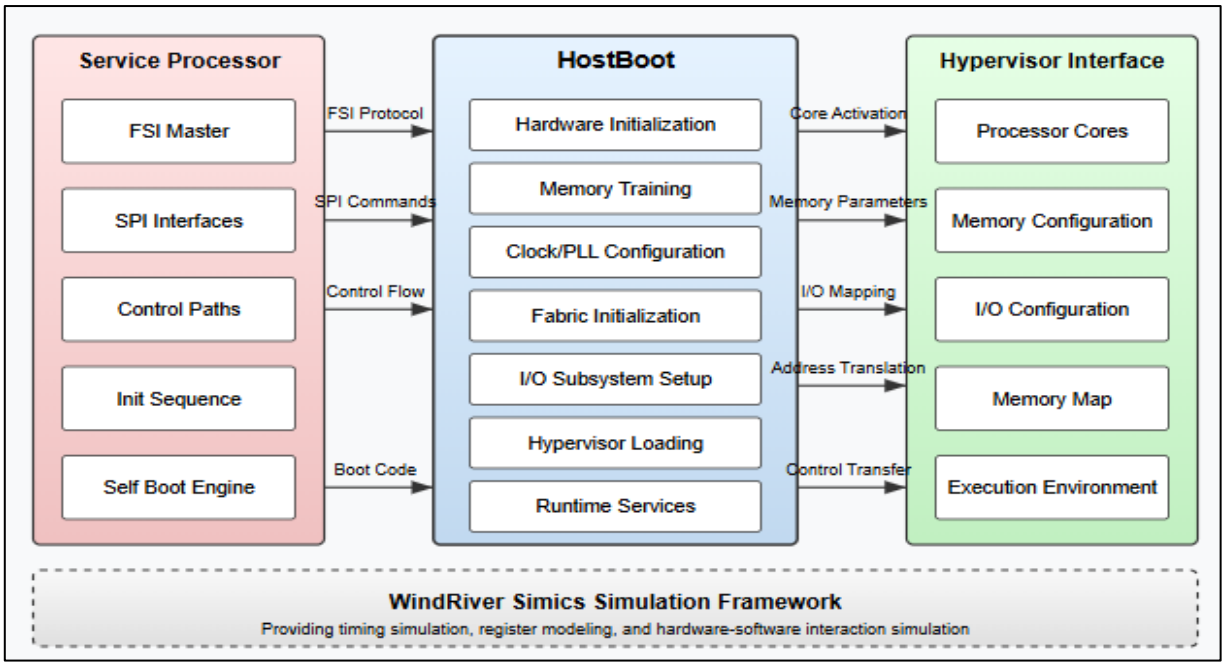


Fig 2: HostBoot Enablement Process in IBM Power Development Ecosystem Simulation Framework (Kakavand, M., & Nikoobin, A. 2020; Sood, K. *et al.*,).

CROSS-TEAM COLLABORATION

Simulation-driven firmware development is achievable in success when there is daily, rigorous interaction among many engineering disciplines. Beyond serving as a technical validation platform, the IBM Power simulation environment functions as a collaborative framework enabling diverse teams to work simultaneously on interdependent components. This collaborative approach acknowledges that modern computing platforms integrate specialized subsystems requiring coordinated development efforts across organizational boundaries. Studies on boundary objects in collaborative design reveal how shared simulation environments can facilitate coordination between different domains by providing common representations flexible enough to accommodate specialized perspectives while maintaining coherent shared understanding (Krakowski, A. *et al.*, 2023).

The development process incorporates frequent check-ins with HostBoot and On-Chip Controller (OCC) teams. These cross-functional touchpoints ensure simulation models accurately reflect evolving firmware requirements and implementation details. The OCC, which handles sophisticated power management and thermal control functions, represents a particularly complex firmware component with intricate hardware dependencies. Regular collaboration ensures simulation models capture nuanced

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

interactions between OCC firmware and underlying hardware platforms.

Peer code reviews and validation walkthroughs constitute another essential collaborative practice within the simulation ecosystem. These structured knowledge-sharing sessions enable expertise cross-pollination and ensure comprehensive validation coverage. Walkthroughs often reveal subtle integration issues that might otherwise remain undetected until hardware integration phases. This collaborative validation approach aligns with research on verification processes in complex systems, where integrating verification activities across system development lifecycles enables more effective risk mitigation and quality assurance compared to traditional stage-gated approaches (Skreddernes, O. *et al.*, 2023).

Accelerated feature coverage and system readiness emerge as key outcomes from these collaborative practices. By enabling concurrent development across multiple firmware components, the simulation-driven approach reduces sequential dependencies and compresses overall development timelines. The collaborative framework ensures identification and resolution of integration issues early in development cycles, reducing costly rework during hardware validation phases.

Table 2: Collaborative Framework Components for Firmware Simulation (Krakowski, A. *et al.*, 2023; Skreddernes, O. *et al.*, 2023)

Team	Primary Contribution	Key Activities	Benefits
Memory Validation	System fidelity models	Memory subsystem validation, configuration, training algos, etc	Consistent behavior across simulation levels
HostBoot	Boot sequence firmware	Implementation updates, interface specifications	Early validation of boot sequences
OCC	Power/thermal control	Hardware-firmware interface definition	Validation of complex dependencies
Peer Review	Cross-functional expertise	Code reviews, validation walkthroughs	Detection of subtle integration issues

ADVANTAGES OF SIMULATION-DRIVEN DEVELOPMENT

The simulation-driven methodology implemented for IBM Power Systems offers substantial advantages throughout firmware development lifecycles. By creating detailed digital replicas of hardware components, this approach fundamentally transforms traditional development paradigms previously dependent on physical hardware availability. Benefits span technical, operational, and organizational dimensions that collectively enable more efficient, higher-quality firmware delivery.

A primary advantage comes from achieving firmware delivery before hardware arrival. This capability disconnects software development schedules from hardware manufacturing timelines, enabling parallel development paths that dramatically compress overall product delivery schedules. Studies on simulation in manufacturing processes demonstrate how virtual prototyping can dramatically reduce development time and costs by enabling concurrent engineering activities across traditionally sequential development phases (Wiendahl, H. P., & Worbs, J. 2003). For IBM Power systems, this approach allowed firmware teams to deliver functional code ready for integration testing immediately when initial silicon samples became available, eliminating traditional delays between hardware arrival and firmware readiness.

The approach substantially reduces dependency on lab hardware and debug tools - particularly valuable during early development phases when hardware availability remains limited and access to specialized debug equipment may be constrained. By enabling developers to reproduce and analyze complex scenarios in simulation environments, teams can diagnose and resolve issues without competing for limited hardware resources. The simulation environment provides extensive visibility into system states otherwise difficult or impossible to observe in physical hardware, enabling more efficient root cause analysis and issue resolution.

Creating repeatable, automatable validation scenarios represents another significant advantage. The simulation environment enables precise control over initial conditions, and error injection capabilities, otherwise challenging to reproduce consistently with physical hardware. This reproducibility proves essential for systematic

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

validation and regression testing, ensuring previously resolved issues do not reappear in subsequent firmware releases. The deterministic nature of simulation environments facilitates automation of test sequences, enabling more comprehensive validation coverage with reduced manual effort.

The simulation framework additionally enables continuous integration practices for firmware development. With simulation-based validation that is part of automated build and test pipelines, teams will get instant feedback to changes in the code, identifying any problem in the integration before diffusing into large development ecosystems. This methodology conforms with the literature on embedded devops practices, whereby organizations implementing embedded devops practice in the form of adoption of automated testing and continuous integration techniques in firmware development record significant improvement in the time to market and product quality despite the complexity of embedded systems (Katapara, P., & Sharma, A. 2025). For IBM Power systems, CI integration enabled daily validation of complete firmware stacks, ensuring prompt identification and resolution of integration issues throughout development cycles.

CHALLENGES AND MITIGATIONS

Concluding that simulation-driven development could bring substantial benefits of validating firmware, it produces certain barriers that must be negated through systematic mitigation strategies. These challenges stem from inherent complexity in accurately representing hardware behavior within software-based simulation environments, particularly for sophisticated systems like IBM Power platforms. Development teams encountered several significant obstacles throughout the next-generation Power simulation implementation and deployed structured solutions addressing each challenge.

Simulating asynchronous behavior and real-time delays with sufficient fidelity to validate timing-sensitive firmware operations presents a primary challenge. Modern computing platforms incorporate multiple clock domains, asynchronous interfaces, and complex state machines interacting in ways difficult to model accurately. Firmware depending on precise timing behavior—particularly for operations like memory training, bus initialization, and power sequencing—may function correctly in simulation yet fail on physical hardware if timing models lack adequate fidelity. Research on hardware-software co-design methodologies emphasizes how accurately modeling asynchronous interactions between hardware and software components represents a fundamental challenge in embedded system validation, requiring sophisticated approaches to timing representation and event synchronization (Raza, H. M., & Patrikar, R. M).

Keeping simulation models synchronized with evolving hardware designs presents another significant challenge. As hardware specifications evolve throughout development cycles, simulation models require corresponding updates to maintain fidelity. Without systematic synchronization processes, simulation models can diverge from hardware implementations, leading to false positive or negative validation results. This challenge becomes particularly acute for new hardware features or interface modifications that may lack full documentation in preliminary specifications. Research on bidirectional model transformation approaches highlights complexity in maintaining consistency between different model representations in evolving systems, particularly when changes originate from either direction in development processes (Ahmad, H. G. *et al.*, 2021).

To address model synchronization challenges, the IBM Power development process incorporates regular, structured synchronization sessions with hardware design teams. These cross-functional meetings ensure simulation models accurately reflect current hardware specifications, including recent changes and design refinements. The process includes systematic reviews of hardware specifications, register definitions, and protocol implementations to identify and address potential discrepancies. By establishing clear ownership boundaries and communication channels, the approach ensures simulation models evolve alongside hardware designs, maintaining validation fidelity throughout development lifecycles.

FUTURE DIRECTIONS

Much of simulation-driven firmware development remains to be discovered, but as it evolves, there is potential for research and development in a number of areas to expand the possibilities and increase the usefulness of simulation frameworks on the IBM Power Systems. All these future directions are continuations of the previously existing simulation infrastructure, but also integrate the latest technologies and approaches in

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

order to meet the challenge of an even more sophisticated type of validation. It will also be strategically superior to invest in these fronts, which would greatly enhance the shift-left concept and provide other boosts during the lifecycle of products.

One significant future direction involves the tighter integration of artificial intelligence and machine learning techniques for telemetry analysis within simulation environments. Existing simulation systems produce large amounts of telemetry data during validation simulations, such as power consumed, thermal characteristics, performance counters, and error reports. Although this data offers invaluable insight into the way the system behaves, scale and complexity level are beyond what conventional analysis methods can achieve more than usual. The method using AI brings with itself promising opportunities to detect anomalous behavior patterns, anticipate and identify possible problem occurrences before they become failures and to establish optimized system configurations based on actual observed behavior. Research on machine learning methods for anomaly detection in telemetry data demonstrates how techniques like isolation forests, one-class SVMs, and deep learning approaches can effectively identify abnormal patterns in complex system data, potentially enabling more proactive identification of firmware issues during simulation (Fejjari, A. 2025).

Another promising direction involves expanding open-source collaboration for developing reusable storage and BMC models. While the current simulation environment incorporates sophisticated models of these components, broader industry collaboration could accelerate model development and enhance model fidelity through diverse usage scenarios. Open-source approaches would enable the sharing of simulation assets across organizations while preserving proprietary implementations where necessary. This collaborative approach could be particularly valuable for components like storage controllers and management processors that follow industry standards and exist across multiple platforms. By establishing common simulation interfaces and reference implementations, the industry could reduce duplicate development efforts and focus resources on platform-specific innovations.

A third emerging direction involves leveraging simulation environments for security validation and fuzz testing. As security considerations become increasingly central to firmware development, simulation provides a controlled environment for systematically evaluating security properties and identifying potential vulnerabilities. Simulation-based security validation can include verification of secure boot implementations, testing of access control mechanisms, and validation of cryptographic operations. Fuzz testing—the practice of providing invalid, unexpected, or random inputs to detect unhandled conditions—becomes particularly valuable in simulation environments where failures can be analyzed without risking physical hardware damage. Research on simulation validation methodologies emphasizes the importance of systematic approaches to model verification and validation, ensuring that simulation results accurately predict real-world system behavior across diverse operating conditions and threat scenarios (Simpkins, S. D. *et al.*, 2001).

All these future directions are designed around the existing simulation infrastructure but solve the emerging issues on firmware development. Through ongoing advancement of simulation skills in these directions, IBM Power Systems will continue to reinforce the shift-left strategy, reduce time-to-market, and perhaps most importantly, increase the quality of the products of the following generations of the platform.

CONCLUSION

The importance of high-fidelity simulation as an accelerator of firmware validation in IBM Power Systems is changing the way enterprise platforms undergo development and verification in a highly complex environment. The simulation framework would allow validation efforts to commence months in advance of hardware availability, through the development of complete digital twins of key elements of hardware to be available. The simulation-driven development process that is explored in this article exemplifies how simulation technology can speed time-to-market as well as improve the quality of products. The strategy provides several advantages, such as less reliance on physical hardware, the possibility of replicating validation scenarios, and the possibility of unleashing continuous integration tactics in the development of firmware. Although problems such as problems with the accurate modeling of timing-sensitive behaviors, as well as keeping an adequate model synchronized to the changing hardware requirements, may have been encountered, systematic mitigation methods have been able to

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher

counter such fears. With the increasing evolution of simulation technologies, the shift-left methodology is further reinforced by the incorporation of artificial intelligence in telemetry analysis, broader open-source collaboration engagement, and a widespread increase in security validation capacity. The simulation-based acceleration not only fulfills the industry demand of complex computing and microelectronics but also provides a path to further innovations, even on the newer platform generations.

REFERENCES

1. Zhang, Y., Yu, B., Zhou, L., & Yang, Q. "Comprehensive IoT Firmware Characterization Based on a Large-Scale Firmware Dataset." *Proceedings of the 2024 9th International Conference on Cyber Security and Information Engineering*. (2024).
2. Nahir, A., Ziv, A., Galivanche, R., Hu, A., Abramovici, M., Camilleri, A., ... & Kapoor, S. "Bridging pre-silicon verification and post-silicon validation." *Proceedings of the 47th Design Automation Conference*. (2010).
3. Badshah, A. "Storage Virtualization: Concepts, Components, and Applications." (2025).
4. Agarwal, V. "Designing Resilient Distributed Systems: Fault Tolerance Strategies And Insights." *Technology (Ijrcait)* 8.1 (2025).
5. Yoo, S., & Jerraya, A. A. "Introduction to hardware abstraction layers for SoC." *2003 Design, Automation and Test in Europe Conference and Exhibition*. IEEE, (2003).
6. Aversa, R., Mazzocca, N., & Villano, U. "Design of a simulator of heterogeneous computing environments." *Simulation Practice and Theory* 4.2-3 (1996): 97-117.
7. Haidar, A., Jagode, H., YarKhan, A., Vaccaro, P., Tomov, S., & Dongarra, J. "Power-aware computing: Measurement, control, and performance analysis for Intel Xeon Phi." *2017 IEEE high performance extreme computing conference (HPEC)*. IEEE, (2017).
8. Sankar, S., Kansal, A., & Liu, J. "Towards a holistic data center simulator." *Microsoft Res* 3 (2013).
9. Kakavand, M., & Nikoobin, A. "High-fidelity hardware-in-loop simulation of tethered-wings based on discrete-time mechanics." *Mechanism and Machine Theory* 151 (2020): 103909.
10. Sood, K. *et al.*, "Secure the Network Infrastructure – Secure Boot Methodologies." *Intel.*
11. Krakowski, A., Bishop, L., Greenwald, E., & Yun, S. "The role of boundary objects in collaborative design." *Curator: The Museum Journal* 66.4 (2023): 561-568.
12. Skreddernes, O., Haugen, R. A., & Haskins, C. "Coping with Verification in Complex Engineered Product Development." *INCOSE International Symposium*. Vol. 33. No. 1. (2023).
13. Wiendahl, H. P., & Worbs, J. "Simulation based analysis of complex production systems with methods of non-linear dynamics." *Journal of Materials Processing Technology* 139.1-3 (2003): 28-34.
14. Katapara, P., & Sharma, A. "Embedded DevOps: A Survey on the Application of DevOps Practices in Embedded Software and Firmware Development." *arXiv preprint arXiv:2507.00421* (2025).
15. Raza, H. M., & Patrikar, R. M. "Verification of Multiprocessor system using Hardware/Software Co-simulation." *Proceedings of SPIT-IEEE Colloquium and International Conference.*
16. Ahmad, H. G., Tahir, I., & Abbas, N. N. "Implementation of Novel Approaches in Bidirectional Model Transformation: a Systematic Literature Review." *Azerbaijan Journal of High Performance Computing* 4.1 (2021): 91-112.
17. Fejjari, A., Delavault, A., Camilleri, R., & Valentino, G. "A Review of Anomaly Detection in Spacecraft Telemetry Data." *Applied Sciences* 15.10 (2025): 5653.
18. Simpkins, S. D., Paulo, E. P., & Whitaker, L. R. "Case study in modeling and simulation validation methodology." *Proceeding of the 2001 Winter Simulation Conference (Cat. No. 01CH37304)*. Vol. 1. IEEE, (2001).

Source of support: Nil; **Conflict of interest:** Nil.

Cite this article as:

Lobo, V. F. G. "Enabling Shift-Left Firmware Validation through High-Fidelity Simulation for IBM Power Systems." *Sarcouncil Journal of Engineering and Computer Sciences* 4.10 (2025): pp 111-119.

Copyright © 2022 The Author(s): This work is licensed under a Creative Commons Attribution- NonCommercial-NoDerivatives 4.0 (CC BY-NC-ND 4.0) International License

Publisher: SARC Publisher