2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Breaking Barriers: Formal Verification in Complex Compressor Controller Architecture

Namita Rawat
Formal Verification Engineer
Intel Corporation
Bangalore, India
namita.rawat@intel.com

Usha Rani Bagadi
Formal Verification Engineer
Intel Corporation
Santa Clara, CA, USA
usha.rani.bagadi@intel.com

Rahul Dabur
Formal Verification Engineer
Intel Corporation
Bangalore, India
rahul.dabur@intel.com

Sarsij Saurabh
Formal Verification Engineering Manager
Intel Corporation
Bangalore, India
sarsij.saurabh@intel.com

Abstract - ***Formal verification is a pivotal methodology for ensuring the accuracy and dependability of complex digital designs. Its capacity to exhaustively and swiftly verify the design-under-test (DUT) offers a distinct advantage over conventional simulation-based methods. As intellectual property (IP) blocks grow in complexity, applying formal verification to these designs becomes increasingly challenging. A primary obstacle for formal verification engineers is achieving convergence, a difficulty that escalates with design intricacy. This paper examines the deployment of formal verification on a complex graphics block the compression controller. We elucidate our strategy, employing diverse abstraction techniques to manage complexity and enhance convergence. By systematically navigating the intricacies of compression logic, we demonstrate the efficacy of formal verification in ensuring design compliance and optimizing the verification process. This study highlights the critical role of formal verification in contemporary hardware development and offers insights into refining verification strategies for sophisticated systems.***

I. INTRODUCTION

In the era of artificial intelligence and big data, the efficient processing, storage, and management of vast volumes of information are critical to sustaining innovation and maintaining a competitive edge across industries. Data serves as the foundational asset driving intelligent decision-making and operational efficiency. However, with the exponential growth of data, ensuring its integrity, availability, and secure handling has become increasingly challenging. High-profile incidents, such as the 2017 Equifax data breach, underscore the devastating consequences of compromised data integrity—highlighting the urgent need for robust data management strategies.

To address the growing demands of data-intensive applications, **data compression** has emerged as a vital technique, enabling the reduction of data size for efficient storage and faster retrieval without sacrificing information fidelity. Implementing such compression in real-world AI systems involves complex architectural designs, which must be rigorously verified to guarantee correctness and reliability. This paper focuses on the formal verification of a sophisticated design integral to AI-based compression controller. While formal methods offer exhaustive and mathematically sound validation, they often encounter significant convergence challenges in large, intricate systems. We address these limitations through a structured verification approach that integrates advanced convergence strategies and utilizes an abstracted compressor model verified via C2RTL methodologies. Our approach not only exposes hard-to-detect corner-case bugs but also achieves high convergence rates, leading to more robust and

optimized system designs. This work contributes to the advancement of formal verification in complex AI infrastructures, offering practical insights and methodologies for ensuring design integrity, performance, and scalability.

II. DESCRIPTION

A. Design Overview

As industries generate vast amounts of data, efficient storage becomes critical. Big data enterprises face challenges in managing large datasets, necessitating expansive storage solutions. Data compression strategy enables storage in reduced formats to optimize memory usage and enhance retrieval efficiency. Compressed data formats are essential for maintaining system performance and scalability, ensuring businesses can effectively leverage their data assets.

Central to the compression process is the compression controller, which orchestrates transaction management to ensure effective data handling and storage. It aligns command and data streams, identifies transactions needing compression, and routes them to the compressor. The controller maintains a lot of transactions, preventing data loss or corruption that could compromise system performance and security. The compression cluster, comprising the controller and compressor, handles read and write transactions, ensuring swift access and retrieval of compressed data while upholding data integrity and enhancing system performance.

The compressor uses sophisticated algorithms to transform raw data into compressed formats, balancing compression ratio and data integrity. It applies to lossless or lossy techniques based on requirements, enabling faster transmission and storage.

The Design Under Test (DUT) is tasked with processing write transactions originating from the compression frontend unit, subsequently channeling them to the controller for evaluation. The controller's primary function is to determine the necessity of compressing data prior to its storage in memory. This involves a meticulous segregation of transactions, identifying those that require compression. These selected transactions are then routed to the compressor, which employs advanced algorithms to perform the compression operation. The compressor outputs compressed data accompanied by compression ratio metrics, essential for accurate future decompression. Following this, the compressed data is efficiently transferred to memory, while the compression metadata is systematically stored in the buffer for subsequent reference and validation.

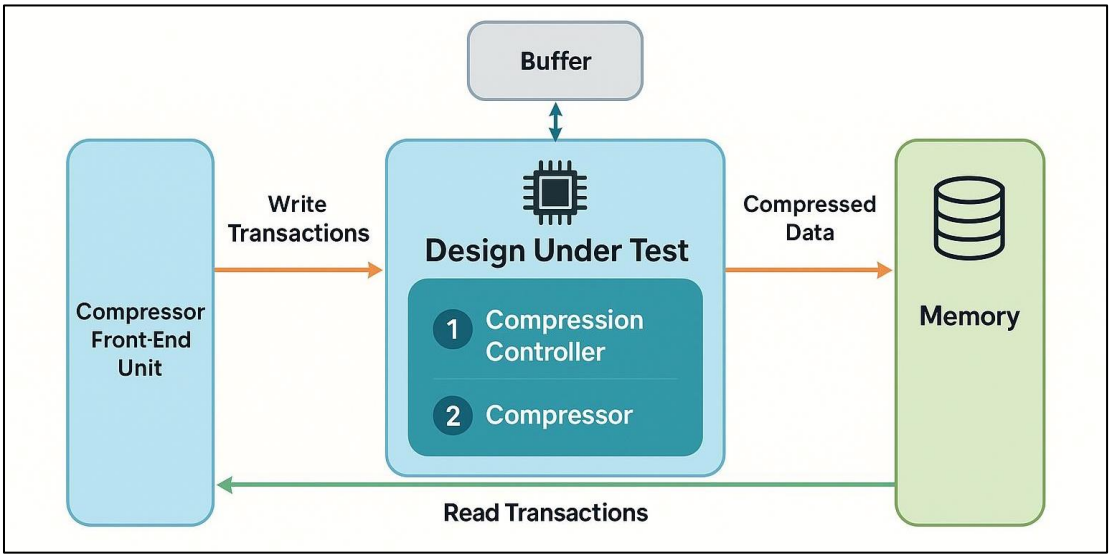

Figure 1: Overview of Compression Cluster

B. Design Complexities causing Verification Challenges

The verification of compression logic is inherently challenging due to its intricate control flow and complex data management requirements. The complexity of design is further increased by features intended to enhance the performance of the compression unit.

These features include **Multi-Transaction Model** introduces layer of complexity by handling multiple transaction types simultaneously, necessitating sophisticated management strategies. Handling **Unaligned Command and Data Input Interface** requires additional processing to ensure coherent data flow and accurate transaction handling. **Dedicated Compressor Module** to accelerate data compression, situated outside the compression controller. This configuration necessitates additional command and data alignment, as the compression unit primarily handles data and limited command information essential for the compression process. The functional path for **Zero-Length Transactions** varies based on configuration settings, requiring adaptive handling mechanisms.

The combination of compression controller features and the enhancements results in the extensive use of storage units and multiple internal logics, further complicating the verification process. These complexities demand a meticulous approach to verification, ensuring that all elements function harmoniously to maintain system integrity and performance.

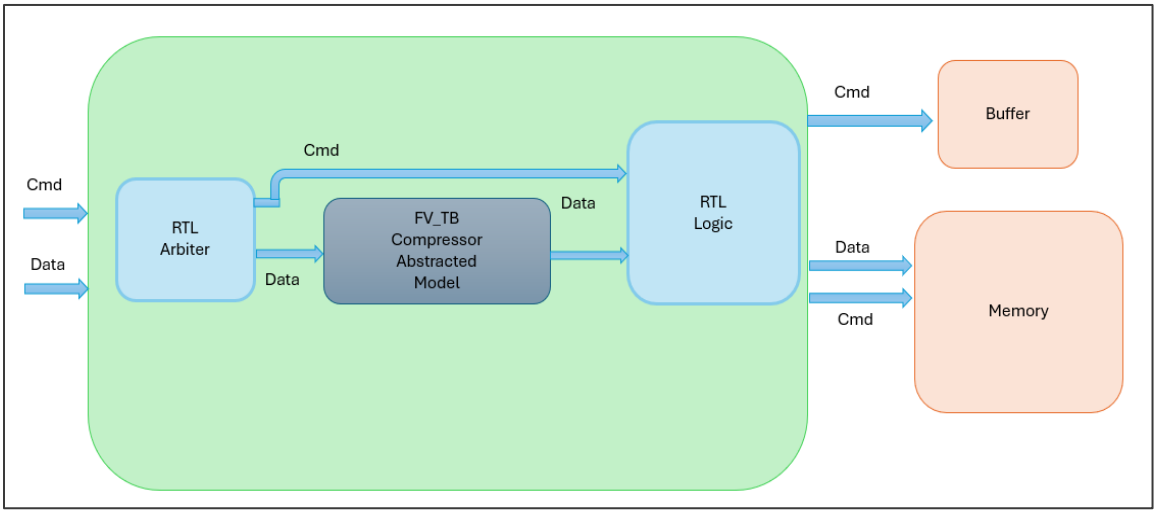

Figure 2: Compression Control with Abstracted Model

C. Verification Strategy

To strategically develop our verification testbench, several challenges that complicated the implementation of fundamental checks were first addressed. An abstraction model for the compressor was employed, capturing its behaviour within a few cycles. It was verified using C2RTL techniques. For command and data alignment, a separate model was utilized to synchronize them at the input interface. Additionally, to achieve alignment at the compressor's output, fields that could be passthrough and remain consistent across both command and data were identified. A color-coding algorithm was applied to these passthrough fields, aiding in the distinction of transactions.

This setup allows us to leverage FPV for

- **Basic Checkers**: Fundamental checks were implemented to ensure the basic functionality and correctness of the design, serving as the first line of defense against potential errors.
- **Data Integrity Assessment**: Rigorous checks were conducted to verify that data remained accurate and uncorrupted throughout processing, maintaining consistency across transactions.

- **Transaction Progress and Starvation (Latency Analysis)**: Transaction progress was analyzed to identify any potential starvation issues, ensuring that all transactions were processed in a timely manner and that latency was minimized.
- **Erroneous Transaction Drop**: Mechanisms were put in place to detect and discard transactions containing errors or invalid data, preventing them from affecting system performance.
- **WAW (Write After Write) Checks**: Verification was performed to ensure that write operations did not prematurely overwrite previous writes, maintaining data consistency and correctness.
- **Multi-Cache line Exclusivity**: Checks were implemented to ensure that transactions involving multiple cache lines were handled exclusively, preventing conflicts and maintaining data integrity.
- **In-Order Transaction to Memory**: Transactions were verified to ensure they were written to memory in the correct sequence, preserving the intended order of operations.
- **In-Order/Out-Order Transaction to Buffer**: Transactions were sent to the compressor in order, but from the compressor to the buffer, they could be processed out of order based on the specific input requirements of certain transactions. For this path, distinct checks were implemented to accurately identify and fulfill each transaction's requirements, ensuring proper handling and storage.

D. Optimization Techniques

Achieving convergence in complex designs, characterized by numerous gates and storage elements, was challenging despite FPV checks. Common obstacles included state space explosion and unbounded proofs. To address these, strategic approaches were employed:

- **Abstraction**: The compressor block inherently possesses a high level of complexity, which poses challenges in achieving verification bounds. This complexity can lead to convergence issues for the controller. To address this, an abstracted model of the compressor is utilized, which simplifies the representation while preserving essential behaviors. This model is verified using C2RTL techniques, effectively limiting the verification bounds and facilitating a more manageable scope for formal analysis.

- **Parameter Reduction**: The compressor controller interacts with substantial memory storage elements, accommodating multiple transactions simultaneously. This interaction exponentially increases the complexity of verification tools. By optimizing the depth of storage elements, the scope of transactions that the tool can handle is reduced, thereby streamlining the verification process and enhancing tool efficiency.

- **Incremental Verification**: Given the design's multifaceted features and components, segmenting verification paths enhances the scope of analysis at higher bounds. This approach mitigates the risk of overlooking corner cases that might be missed if the entire block is considered in a single context. The segmented paths include verification from input to compressor, compressor to memory, and compressor to buffer, with each path being verified independently to ensure comprehensive coverage.

- **Symbolic Coding**: Symbolic coding focuses on verifying transactions by tracing a single transaction from input to output. This technique reduces the complexity that the verification tool must manage, allowing for more precise analysis. It is particularly beneficial in end-to-end checks, where the focus is on ensuring the integrity and correctness of individual transaction flows.

- **Heuristics and Pruning**: These techniques are employed to minimize redundancies, enabling the verification tool to handle fewer elements. By applying over-constraints to prioritize specific transaction types, the design is subjected to multiple practical scenarios, ensuring thorough verification. This approach allows for the identification of potential issues in diverse operational contexts, enhancing the

robustness of the design. Additionally, it further decreases complexity by streamlining the verification process.

These techniques streamline complexity by reducing state elements, enabling comprehensive analysis of the Compression Controller's behavior and identifying elusive bugs.

III. APPLICATION

Handling these complex designs requires a precise understanding of designer requirements and the limitations of formal verification tools. This paper strategically outlines how to break down design complexity and apply formal techniques. These techniques are easily applicable to other complex designs like memory controllers and EUs, providing convergence with properties. For complex designs with multiple algorithmic paths and corner cases, relying solely on simulation does not guarantee a bug-free environment.

IV. RESULTS

The formal verification for the compression controller identified several bugs. Through the application of multiple convergence techniques and sustained efforts, we uncovered eight pivotal corner-case anomalies that would be challenging to detect through traditional simulation methods. Few of them are mentioned below:

Bug Description

1. **Transaction Mismanagement Leading to Data Corruption**: The failure observed stems from the input stage, where two distinct transactions with identical addresses are being processed. Ideally, these transactions should be handled sequentially, ensuring that each is processed independently and in the correct order. However, the design's current implementation lacks the capability to differentiate between transactions with the same address, resulting in the premature dispatch of the second transaction while inadvertently dropping the first. This issue underscores the critical need for precise address handling mechanisms within the design architecture.

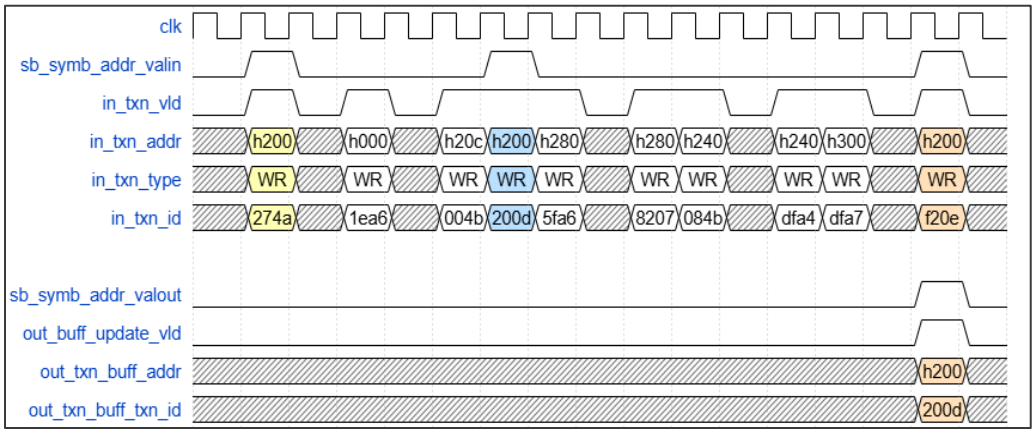

Figure 3: Write after Write (WAW) Hazard results in data corruption

2. **Premature Discarding of Zero-Length Command Transactions Due to Data Unavailability:** The observed failure occurs when the initial transaction is a zero-length command, leading to its premature discard due to an incorrect condition in the design. The current design logic mandates that the data FIFO must not be empty for the transaction to progress to the next stage. This condition is inherently flawed for zero-length commands, which are expected to arrive without accompanying data. Consequently, the design erroneously drops these transactions, failing to recognize their validity.

Addressing this flaw is essential to prevent the loss of valid transactions and maintain the integrity of the processing pipeline.

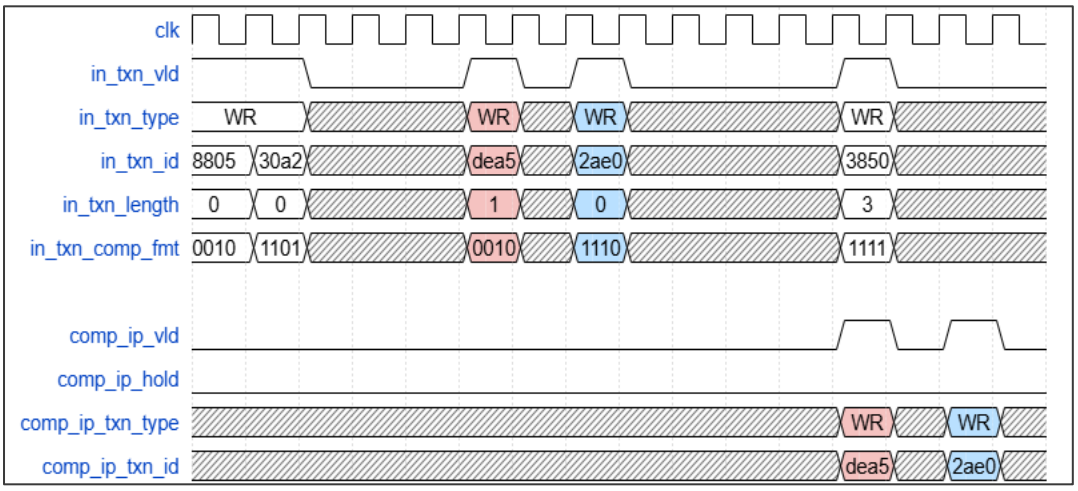

Figure 4: Mismanagement of zero length transaction leads to transaction drop

By implementing advanced convergence techniques, we significantly enhanced proof convergence, elevating convergence from a mere ~10% to an impressive ~89%. This substantial improvement demonstrates the efficacy of our verification strategy in ensuring comprehensive validation and reliability of the design, ultimately fortifying its integrity and operational robustness.

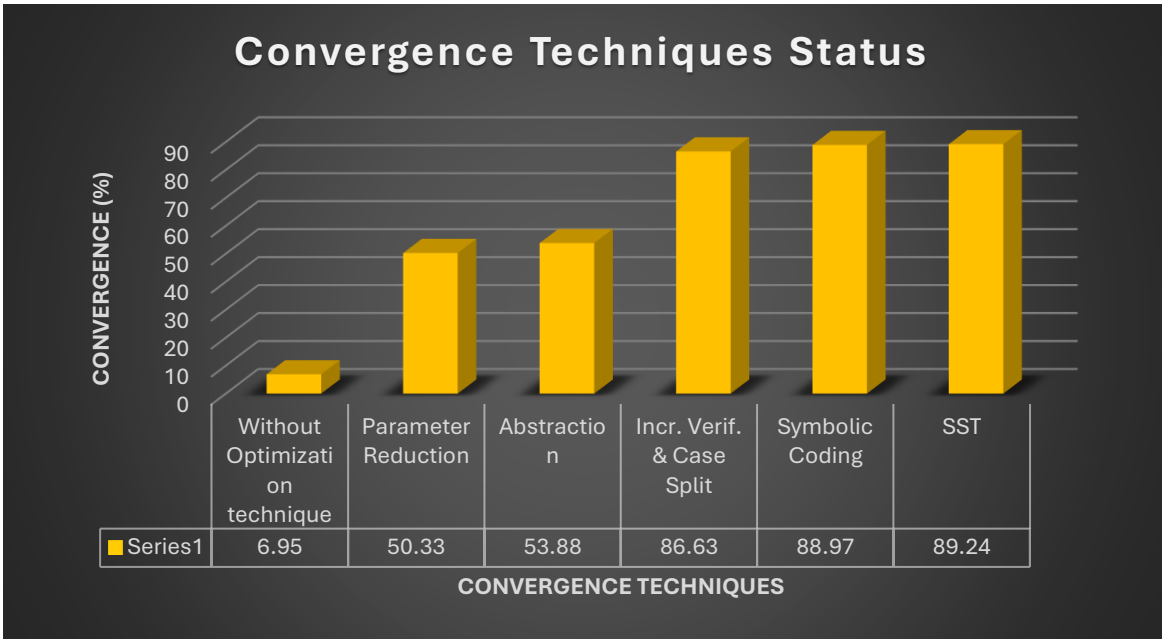

Figure 5: Evaluating Convergence Efficiency

V. CONCLUSION

The application of formal property verification (FPV) and optimization techniques to complex designs, such as the Compressor Logic Controller, has resulted in significant advancements in identifying and resolving intricate bugs. These methodologies substantially enhance design reliability, offering scalable solutions that are applicable to a wide range of complex systems. Through comprehensive validation and operational robustness, formal verification underscores its critical role in modern hardware development. The insights gained from these processes not only refine current design practices but also pave the way for future innovations, emphasizing the importance of meticulous testing and optimization in achieving high-quality, dependable systems

ACKNOWLEDGMENT

I would like to express my gratitude to my lead, Usha and Rahul, for their invaluable support throughout the project. Special thanks to my manager Sarsij Saurabh for assigning this task and providing opportunities for learning and development. I would like to deeply appreciate the guidance provided by Bathri Narayanan Subramanian. Finally, heartfelt thanks are due to colleagues for their continuous support.

REFERENCES

[1]. E. Seligman, T. Schubert, and M. V. A. Kiran Kumar, *Formal Verification: An Essential Toolkit for Modern VLSI Design*. Morgan Kaufmann, 2015.

[2]. Rahul Dabur, Tushar Agarwal, and Sarsij Saurabh, "Deciphering Complexity: Formal Verification of Systolic Controllers for Robust System Performance,", DVCON, India.

[3]. K. L. McMillan, "Symbolic Model Checking: An Approach to the State Explosion Problem," *IEEE Transactions on Software Engineering*, vol. 19, no. 2, pp. 133-147, Feb. 1993.

[4]. S. Ray, A. K. Singh, and S. R. Sarangi, "Formal Verification of Cache Coherence Protocols Using Incremental Induction," in *Proceedings of the 54th Annual Design Automation Conference (DAC)*, Austin, TX, USA, 2017, pp. 1-6.

[5]. Usha Rani Bagadi, Mohit Choradia, Ajay Kumar Kolluri, and Vichal Verma, "Enhancing Arbitration Integrity: Formal Verification of Weighted Round Robin Arbiter in High-Performance Graphics,"

[6]. presented at DVCon, Intel, 2023.

[7]. Vedprakash Mishra and Keerthi B, "Towards Rigorous Fairness: Formal Verification of Multi-Level Arbitration through Hierarchical Family Chains," presented at DVCon, Intel, 2023.

[8]. Disha Puri, Madhurima E, and Shravya Jampanna, "Raising the Bar: Achieving Formal Verification Sign-Off for Complex Algorithmic Designs, with a Dot Product Accumulate Case Study," in *Proceedings of DVCON*, India, 2023.

[9]. J. Doe and A. Smith, "A Case Study for Formal Verification of a Timing Co-Processor," in *Proceedings of the IEEE International Conference on Computer Design*, New York, NY, USA, 2023.