2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION JAPAN

Ensuring DRAM Compliance: Novel Verification Techniques for Refresh and Refresh Management in Modern Dram Architecture

Dharini SubashChandran, Systems Verification Group, Cadence Design Systems, Bangalore, India (dharini@cadence.com)

Shyam Sharma, Systems Verification Group, Cadence Design Systems, California, USA (ssharma@cadence.com)

Gruhesh Patel, Systems Verification Group, Cadence Design Systems, Bangalore, India (gruhesh@cadence.com)

***Abstract*—DRAM data integrity is a core requirement for any of the modern SoCs NoC and PCBs where DRAM memories are used anywhere in the system. It is also one of the most difficult problems to verify in today's complex memory subsystems. Beyond the basic Refresh, Row Hammer and PRHT (Per Row Hammer Tracking) is increasing becoming an important consideration for the DRAM based systems. In the latest generation of DRAMs like DDR5 and Lpddr5, Refresh Management features are added to help designers tackle the Row Hammer challenges. This presentation talks about the innovative tools and solutions we have come up to help IP and SoC verification engineers, ensuring they can not only achieve their verification goals for the Refresh requirement that DRAMs have but also test the different aspects of Refresh Management and quantify their verification completeness by getting measurement of what all has been tested with intuitive Refresh/RFM related functional coverage.**

Keywords*— *DRAM, Refresh, Refresh Management, RAA Counters, Coverage, Callbacks, Row Hammering, Data Loss, Waveform Debugger, DRFM, Refresh Requirement Features.

I. INTRODUCTION

Dynamic Random Access Memory (DRAM) forms the core of modern computing systems, providing fast and volatile data storage crucial for program execution. However, due to its inherent design, DRAM cells require periodic refreshes to maintain data integrity. This refresh process, while essential, introduces overheads and complexities in memory management. We will analyze the challenges posed by traditional refresh mechanisms and investigate innovative approaches that aim to optimize performance and reliability in DRAM systems. This paper delves into the novel verification solutions we have developed to address DRAM refreshes and evolving landscape of refresh management techniques.

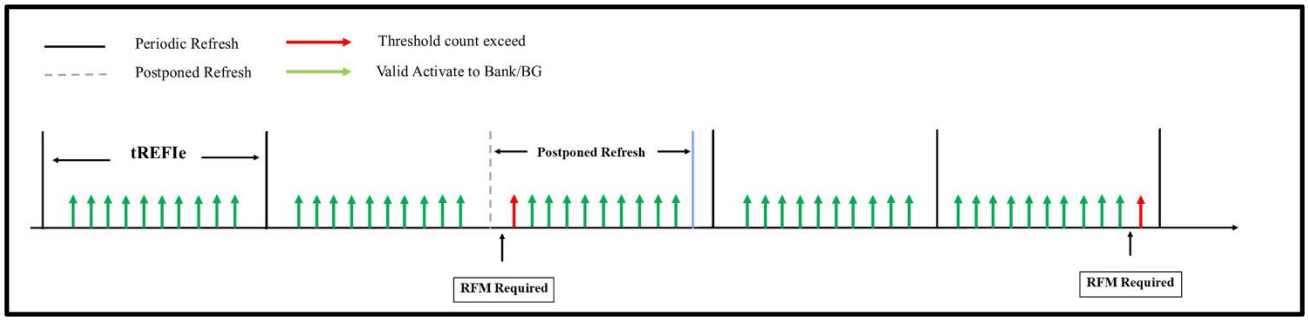

Figure 1: Periodic Refresh v/s Refresh Management

II. BACKGROUND ON THE VERIFICATION CHALLENGES

- Modeling Complexity: Accurately modeling the behavior of DRAM refresh mechanisms, within a digital simulation environment can be intricate.
- Simulation Time: Simulating the entire refresh process, especially for large DRAM capacities and over extended periods, can be time-consuming and computationally expensive.
- Corner Case Scenarios: Capturing and simulating corner case scenarios, like temperature fluctuations or rare error events, within a digital simulation environment can be challenging.

- Limited Visibility: Digital simulations may provide limited visibility into the internal state of the DRAM cells, making it difficult to pinpoint the root cause of potential issues.
- Testbench Development: Creating comprehensive testbenches that effectively exercise and verify refresh mechanisms across various operating conditions and failure modes requires significant effort and expertise.
- Keeping track of the number of row Activation during the time of high DRAM activity to ensure additional Refresh are issued to prevent potential data loss to adjacent rows to the rows that are accessed repeatedly.
- Integration with System-Level Simulation: Integrating DRAM refresh verification with system-level simulations, which include other components like processors and controllers, adds further complexity to the verification process.

These challenges highlight the need for efficient simulation techniques, accurate models, and comprehensive testbenches to effectively verify DRAM refresh mechanisms using digital simulation.

III. IMPLEMENTATION OF THE SOLUTION

1. **Core Refresh Mechanisms:** While maintaining the basic implementation of Refresh and Refresh management, we have introduced advanced refresh debug capabilities to optimize performance and efficiency.
   - **Cumulative Counter Track for Pullin-Postpone Refreshes:** The system allows for flexibility in scheduling refreshes, enabling them to be pulled in (executed earlier) or postponed (executed later) within defined limits. A dedicated counter tracks pulled-in and postponed refreshes, ensuring that the required number of refresh operations occur within the allowed timeframe to prevent data loss.
   - **Smart Optimized Refresh Mechanism**: For customized behavior and optimization users can configure dynamic turn on/off refresh checks using switch "**RefreshChecks** On/Off". Our backdoor mechanism allows modification of refresh rates which in turn can be applied at any point of the simulation using features like "**applyDerate**" giving the designers complete control.
   - **Temperature sensitive Derating Support** – While operating at high temperature, DRAM needs to derating for some of the core timings. To model this requirement in simulation, we have derating parameter **tsdramDerating** which indicates the value by which the core timings are derated**.** Parameters like **tderate** indicate when the new derated value is applied. Customized feature "r**efreshDeratingMap**" to map user defined values for derating.

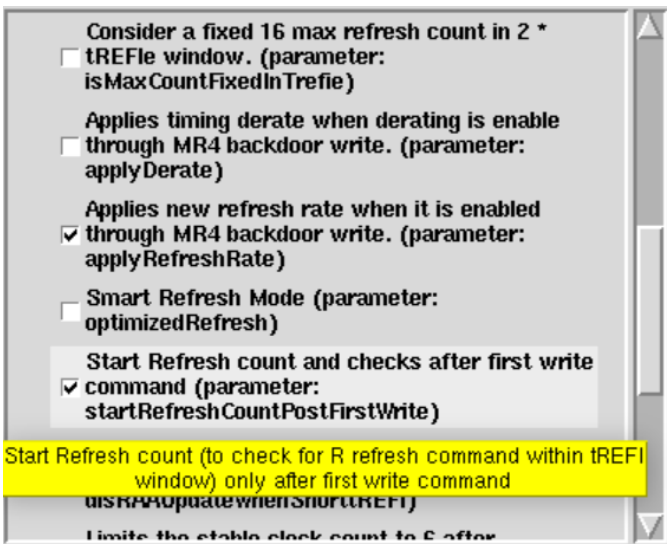

Figure 2: Configurable Refresh Features

2. **Refresh Management Mechanism:** Refresh Management is an approach used by latest generation of DRAMs like Lpddr5 to mitigate the potential loss of data from adjacent rows of a row that is activated repeatedly during time of high activity. This is an important feature to mitigate potential row hammer attacks**.** Our work introduces a novel approach to verify a flexible, secure and dynamic refresh management system using :-
   - **Row Access Counter (RAA)**: By monitoring ACT commands (Activate) on a specific bank, the RAA helps optimize refresh operations.
   - **Configurable Registers/Features**: Offers options to facilitate refresh management checks by dynamically configuring threshold values which can be vendor specific. With each vendor having different values the parameters can help test devices across multiple vendor
     - **raaimt**: To configure RAA Initial Management Threshold.
     - **raammt**: To configure RAA Maximum Management Threshold.
     - **raaCntDec**: To configure RAA counter decrement per Refresh Command.

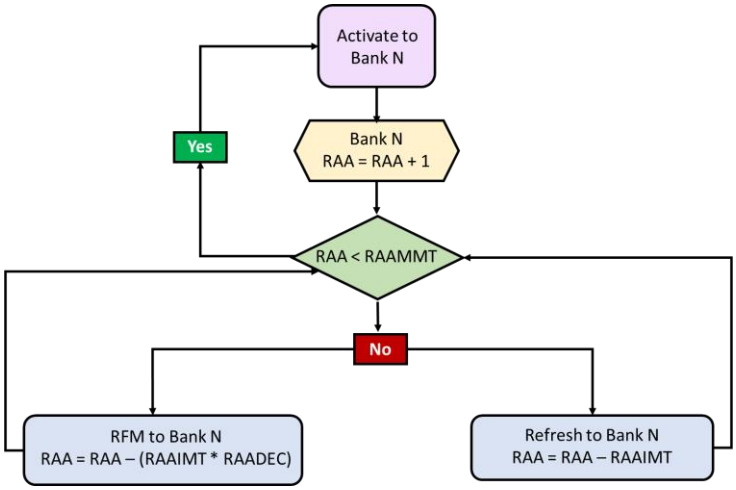

Figure 3: Flow chart depicting RAA counter update during RFM.

   - **Directed Refresh Management** (DRFM): The model tracks the victim row of a bank, allowing subsequent DRFM commands to clear that row for efficient access in future operations.

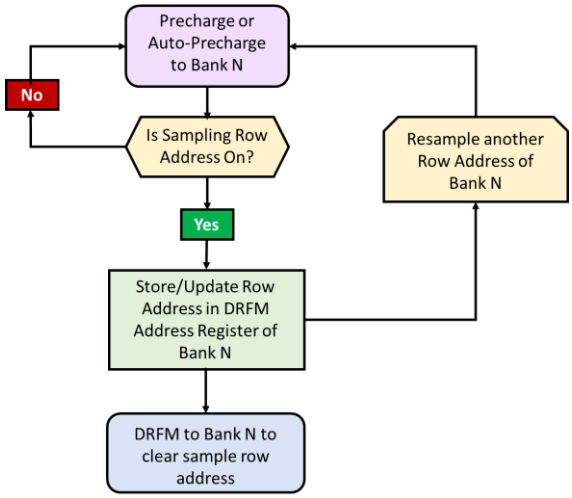

Figure 4: Flow chart for DRFM operation

3. **Comprehensive Suite for Refresh and Refresh Management Debug:**
    - **Callback Monitoring for Custom Logic**: Provides visibility into custom logic through refresh counters, allowing for the postponement or preponement of refresh cycles. Fault injection into refresh threshold counters enhances fault detection capabilities to test realistic scenarios.
    - **Visual Analysis with Debug Ports**: Utilizes debug ports for visualizing and analyzing the entire refresh process, aiding timing analysis to optimize refresh effectiveness. Counter tracking of raaimt, raammt, and raacntdec identifies sources of errors, ensuring data integrity.
    - **Metric-Driven Coverage**: Encompasses refresh mechanisms across refresh timings and intervals, data patterns, potential error scenarios. These metrics are compared against established standards or goals to ensure robust testing of refresh management.
    - **Unified Transaction Debug**: Enables efficient validation of refresh management by activating the Packet Tracker switch, tracking transactions per instance or model. Output can be viewed in text, CSV, or waveform format for comprehensive analysis.

IV. APPLICATION OF THE SOLUTION

Refresh Management- Consider a real time scenario for RFM where Bank 0 is used for multiple reads and writes. The configured values are raaimt 32 raammt 64 and raaCntDec 32 when RFM is enabled.

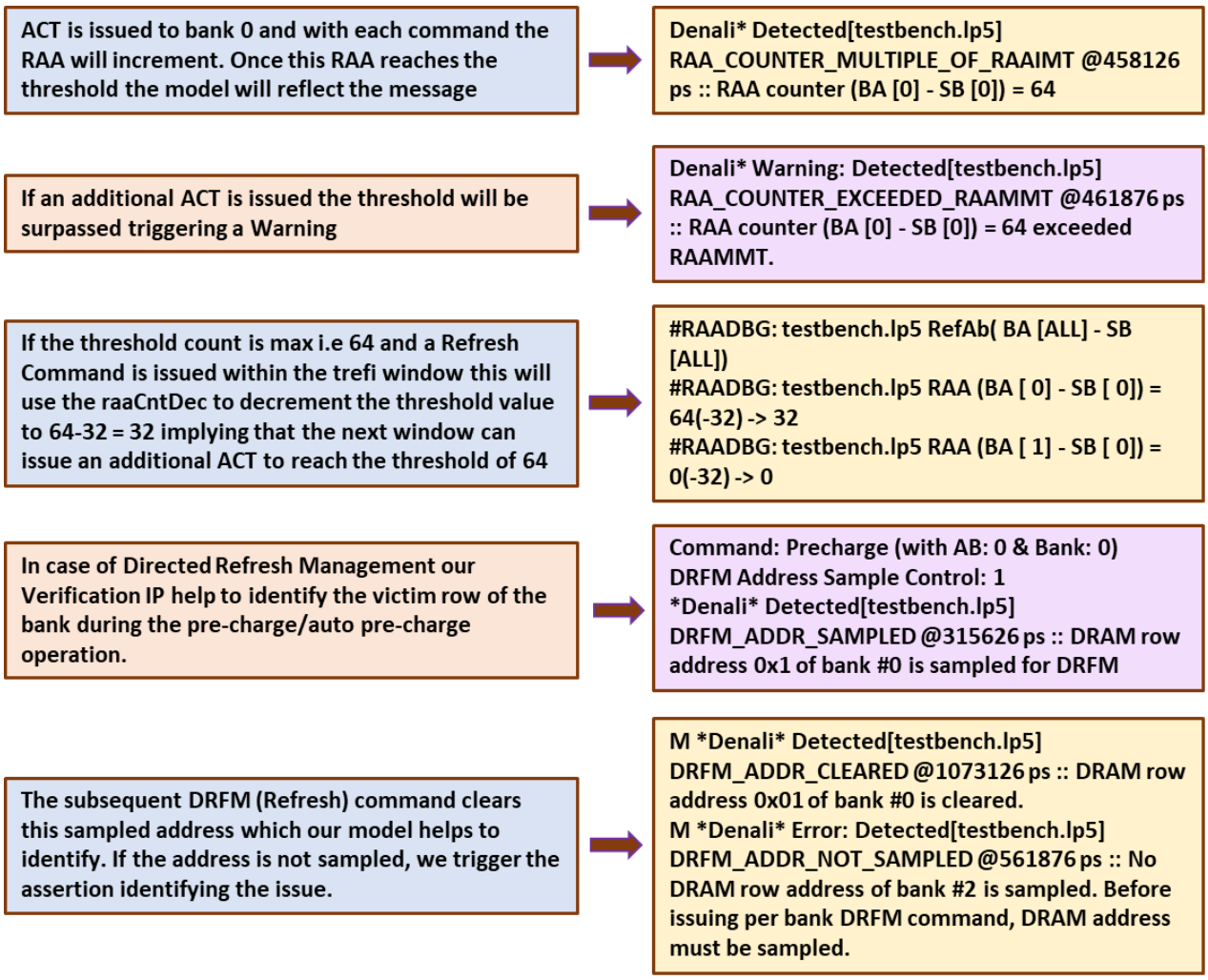

Figure 5: Real Time Scenario for RFM and DRFM Operation

Core Refresh- Model starts the Refresh Window post the first write and at every trefi window we increment the cumulative counter if there are more refreshes than needed or decrement the counter if no refresh issued.

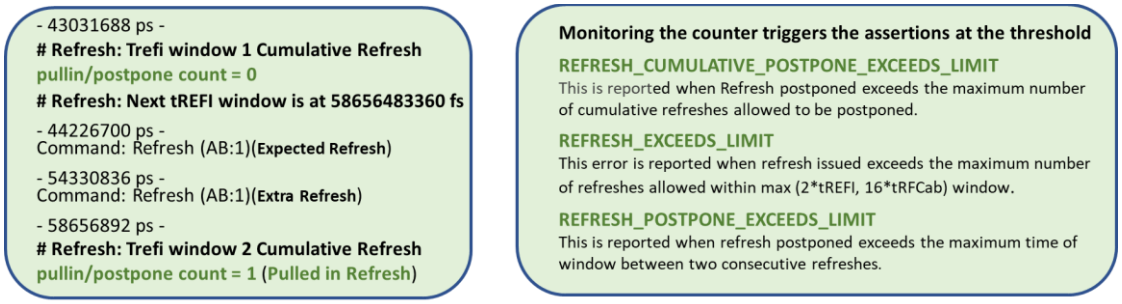

Figure 6: Trace prints for Refresh window and rolling window check.

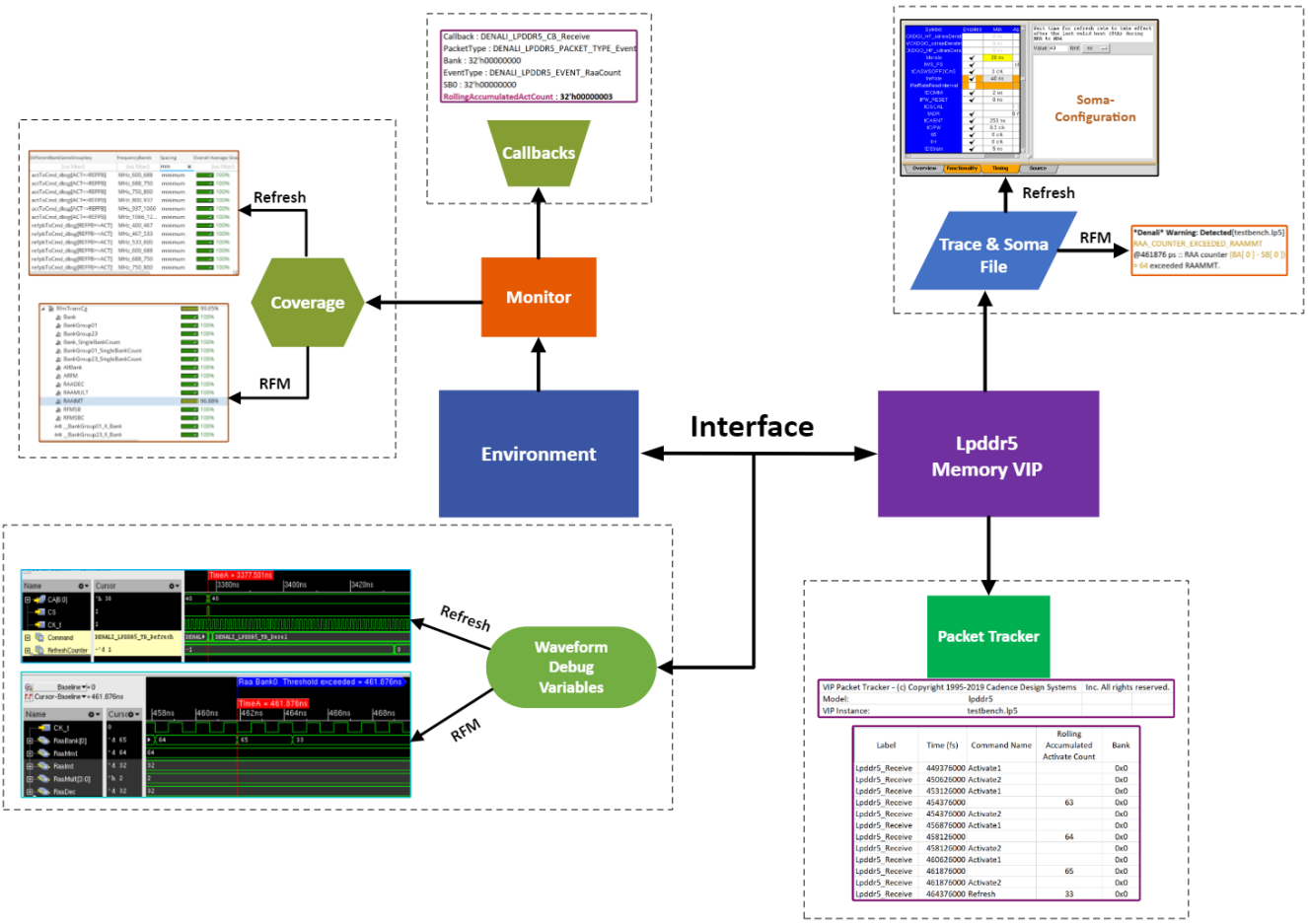

Figure 7: Comprehensive Refresh & RFM Solutions

V. RESULTS

Top accounts have provided positive feedback, highlighting the value of the extensive debug capabilities and their contribution to accelerating the verification process.

- Customers have found features like callbacks for counter monitoring and visual debug ports for observing dynamic changes throughout simulations to be particularly useful as it reduced the overall verification effort by more than 70%.
- The ease of modifying refresh rates and derating options through backdoor mechanisms has enhanced the user experience streamlining regression test suites with precision and accuracy.
- The waveform debugger reflects 100% accuracy for dram commands sampling, internal state transitions and mode register updates.
- The robust coverage and efficient timing analysis tools have enabled significant verification speed-ups, empowering customers to deliver high-quality LPDDR5-based designs with confidence and zero silicon escape.

VI. CONCLUSION

The comprehensive approach to LPDDR5 refresh and refresh management ensures data integrity while optimizing performance and efficiency. DRAM refresh and its management remain critical aspects of ensuring data integrity and system reliability in modern computing. As DRAM technology evolves and system architectures grow in complexity, the challenges associated with refresh mechanisms continue to escalate. This paper has explored the intricacies of DRAM refresh and highlighted the key verification challenges, particularly in the context of digital simulation.

Addressing these challenges requires a multifaceted approach, including the development of sophisticated simulation models, efficient verification methodologies, and comprehensive testbenches. Additionally, exploring innovative refresh management techniques and incorporating these are crucial to ensure robust and reliable DRAM operation in the future. By navigating the evolving landscape of DRAM refresh, we can pave the way for continued advancements in computing systems.

VII. REFERANCES

1. JEDEC. (2024). *DDR5 SDRAM* (JESD79-5C ed.). https://www.jedec.org/standards-documents/docs/jesd79-5c
2. JEDEC. (2023b). *LOW POWER DOUBLE DATA RATE (LPDDR) 5/5X* (JESD209-5C ed.). https://www.jedec.org/document_search?search_api_views_fulltext=lpddr5
3. Cadence Xcelium Logic simulator Xcelium Logic Simulator | Cadence
4. Cadence memory model VIPs Memory Models | Cadence