2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION JAPAN

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

Shyam Sharma, Dharini SubashChandran, Gruhesh Patel

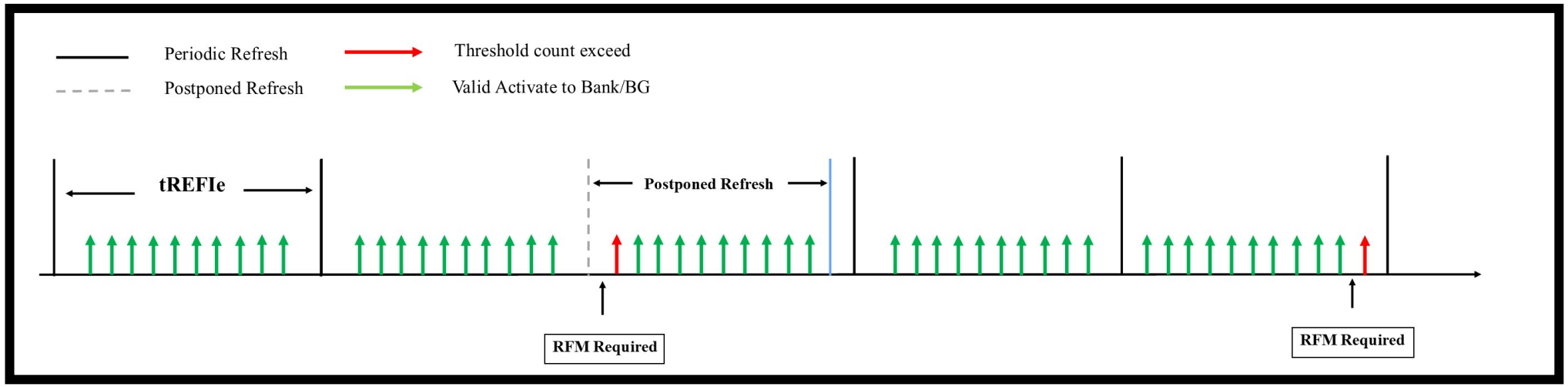
Cadence Design Systems

cadence

accellera SYSTEMS INITIATIVE

The DRAM Refresh Challenges

- **Simulation Time**: Simulating the entire refresh process can be time-consuming and computationally expensive.



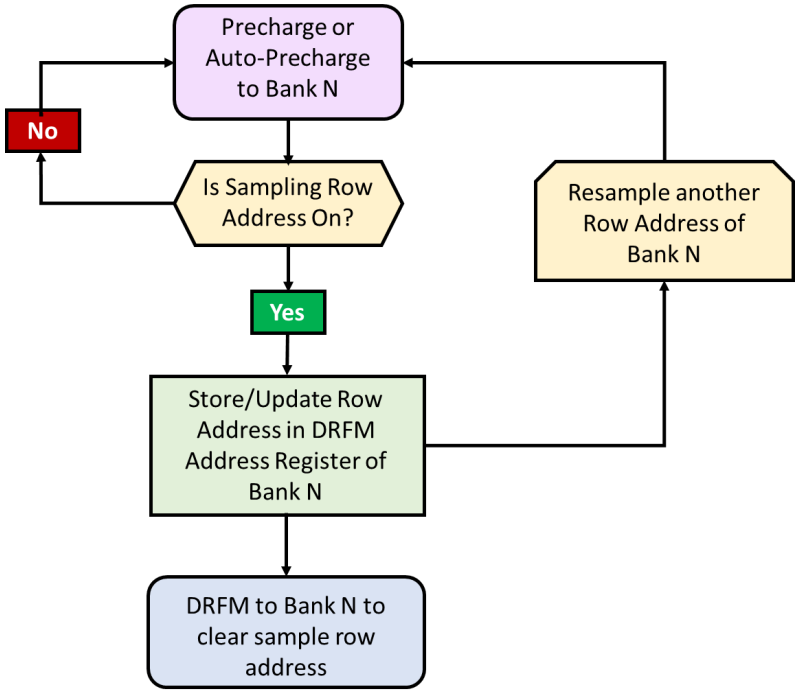
- **Integration with System-Level Simulation**: Integrating DRAM refresh verification with system-level simulations, adds complexity to the verification process.

Challenges(Con’td)

➢**Testbench Development**: Creating comprehensive testbenches that effectively exercise and verify refresh mechanisms requires significant effort and expertise

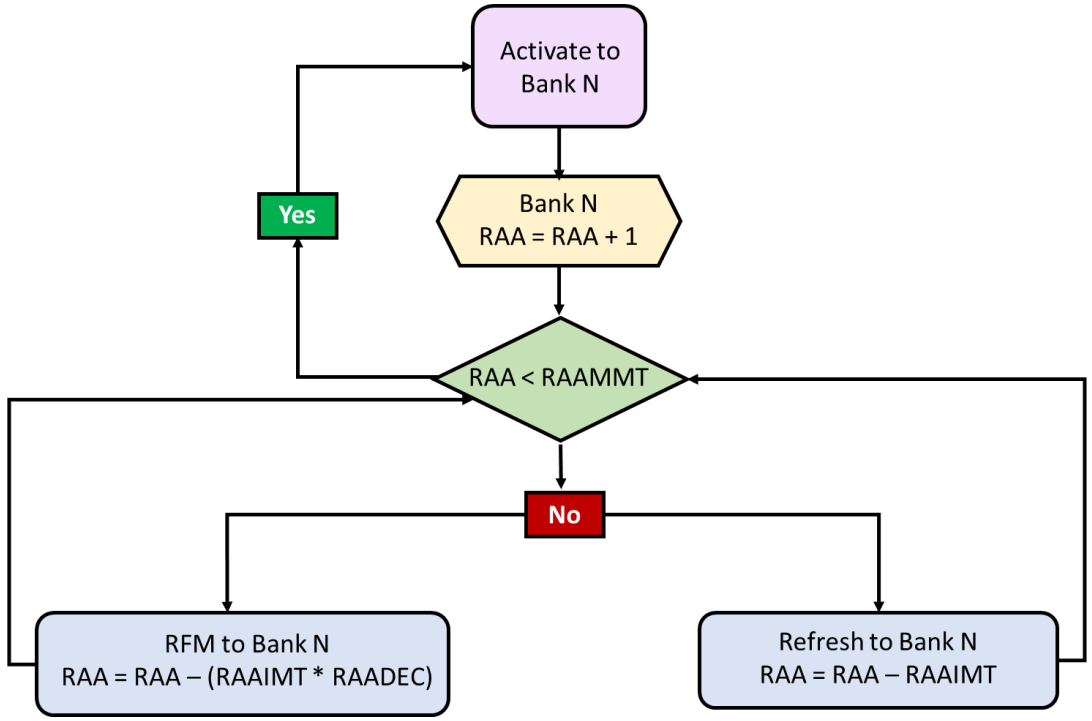
➢**Keeping track of the number of row Activation** during the time of high DRAM activity to ensure additional Refresh are issued to prevent potential data loss.

➢**Corner Case Scenarios**: Capturing and simulating temperature fluctuations or rare error events, within a digital simulation environment can be challenging.



Refresh Management: A Modern Approach

- DDR5 and LPDDR5 introduced Refresh Management (RFM) to mitigate Row Hammer risks.
- RFM uses sophisticated mechanisms to track row accesses and proactively issue refreshes.
- Goals of RFM - improved data integrity, reduced overhead, increased performance.



Cadence's Novel Verification Solutions for DRAM Refresh and RFM

➢ **Cumulative counter tracking:** monitoring pull-in and postpone refreshes.

- 43031688 ps -
 # Refresh: Trefi window 1 Cumulative Refresh
pullin/postpone count = 0
 # Refresh: Next tREFI window is at 58656483360 fs
- 44226700 ps -
Command: Refresh (AB:1)(**Expected Refresh**)
- 54330836 ps -
Command: Refresh (AB:1)(**Extra Refresh**)
- 58656892 ps -
 # Refresh: Trefi window 2 Cumulative Refresh
pullin/postpone count = 1 (**Pulled in Refresh**)

Monitoring the counter triggers the assertions at the threshold

REFRESH_CUMULATIVE_POSTPONE_EXCEEDS_LIMIT
This is reported when Refresh postponed exceeds the maximum number of cumulative refreshes allowed to be postponed.

REFRESH_EXCEEDS_LIMIT
This error is reported when refresh issued exceeds the maximum number of refreshes allowed within max (2*tREFI, 16*tRFCab) window.

REFRESH_POSTPONE_EXCEEDS_LIMIT
This is reported when refresh postponed exceeds the maximum time of window between two consecutive refreshes.

➢**Advanced refresh debug capabilities**: detailed visualization and analysis of refresh processes.

Cadence's Novel Verification Solutions for DRAM Refresh and RFM

- **Smart optimized refresh mechanism:** dynamic "RefreshChecks On/Off" and "applyDerate" features.

```
rc = $mmsomaset("testbench.lp5","refreshRateMap", "0,7,5,3,3.1, 2.2, 1.8, 1.5, 1.2,1,0.6, 0.4, 0.3");
rc = $mmsomaset("testbench.lp5","refreshDeratingMap", "0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1");
```

- **Temperature-sensitive derating support:** modeling realistic operating conditions.

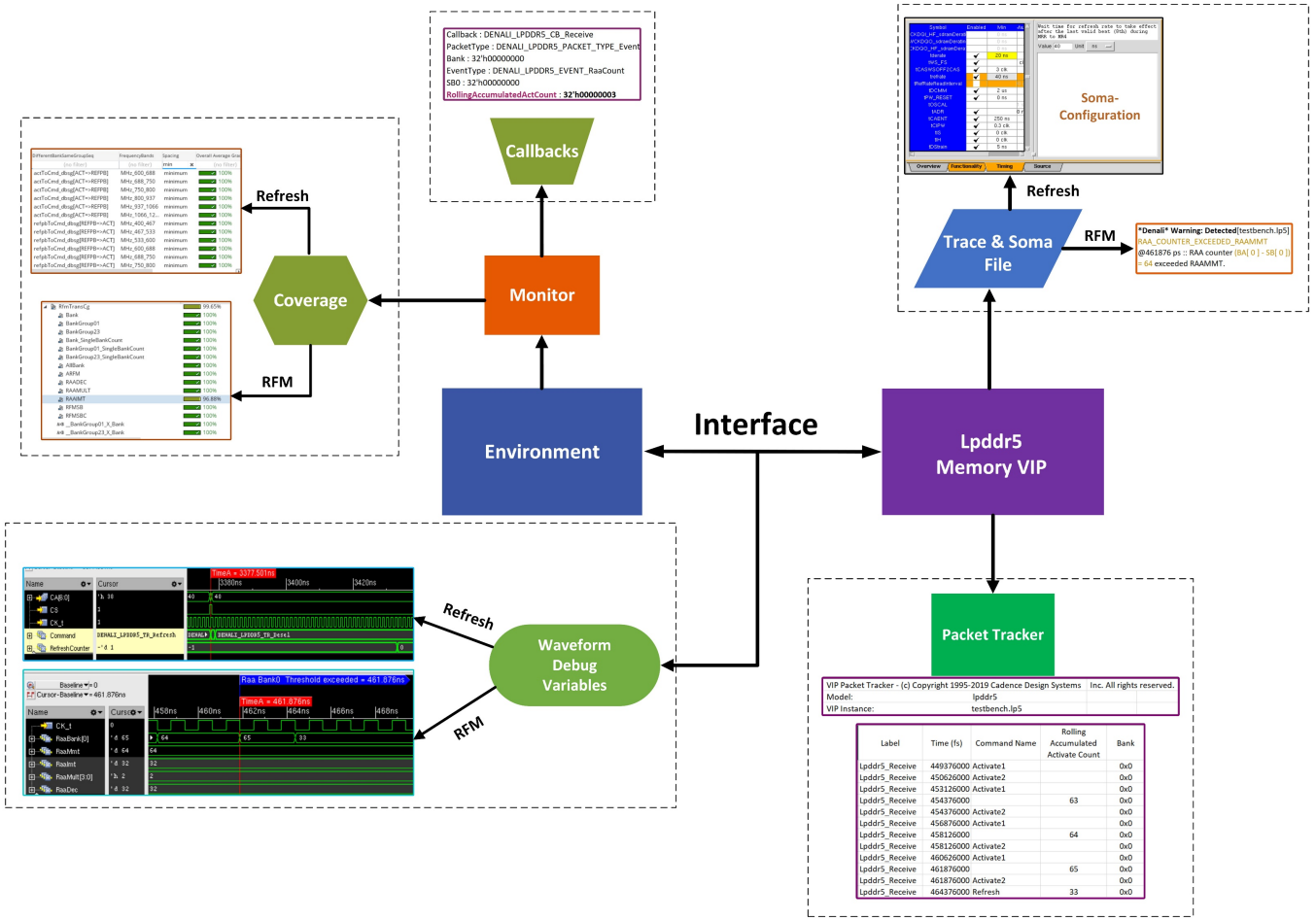
  - **Features/Knobs :**
    - **applyDerate ->** Applies timing derate when derating is enabled through MR4 backdoor WR
    - **applyRefreshRate ->** Applies new refresh rate when it is enabled through MR4 backdoor WR
    - **optimizedRefresh ->** smart refresh mode will be enabled
  - **Timings :**
    - **tderate ->** Wait time for derate to take effect after the last valid beat (8th) during MRR to MR4
    - **trefrate ->** Wait time for refresh rate to take effect after the last valid beat during MRR to MR4

- Row Access Counter (RAA) and its configuration options (raaimt, raammt, raaCntDec)

Powerful Tools for Effective Debugging

- Callbacks for custom logic monitoring: track counter changes and pinpoint errors.
- Visual analysis with debug ports: observe dynamic changes in refresh processes.
- Metric-driven coverage: ensure thorough testing of refresh mechanisms.
- Unified transaction debugging: comprehensive analysis of refresh operations.

Comprehensive Refresh and Refresh Management Solution



Label	Time (fs)	Command Name	Rolling Accumulated Activate Count	Bank
Lpddr5_Receive	449376000	Activate1		0x0
Lpddr5_Receive	450626000	Activate2		0x0
Lpddr5_Receive	453126000	Activate1		0x0
Lpddr5_Receive	454376000		63	0x0
Lpddr5_Receive	454376000	Activate2		0x0
Lpddr5_Receive	456876000	Activate1		0x0
Lpddr5_Receive	458126000		64	0x0
Lpddr5_Receive	458126000	Activate2		0x0
Lpddr5_Receive	460626000	Activate1		0x0
Lpddr5_Receive	461876000		65	0x0
Lpddr5_Receive	461876000	Activate2		0x0
Lpddr5_Receive	464376000	Refresh	33	0x0

Verifying Robust Refresh Management with RAA and DRFM

ACT is issued to bank 0 and with each command the RAA will increment. Once this RAA reaches the threshold the model will reflect the message	Denali* Detected[testbench.lp5] RAA_COUNTER_MULTIPLE_OF_RAAIMT @458126 ps :: RAA counter (BA [0] - SB [0]) = 64
If an additional ACT is issued the threshold will be surpassed triggering a Warning	Denali* Warning: Detected[testbench.lp5] RAA_COUNTER_EXCEEDED_RAAMMT @461876 ps :: RAA counter (BA [0] - SB [0]) = 64 exceeded RAAMMT.
If the threshold count is max i.e 64 and a Refresh Command is issued within the trefi window this will use the raaCntDec to decrement the threshold value to 64-32 = 32 implying that the next window can issue an additional ACT to reach the threshold of 64	#RAADBG: testbench.lp5 RefAb(BA [ALL] - SB [ALL]) #RAADBG: testbench.lp5 RAA (BA [0] - SB [0]) = 64(-32) -> 32 #RAADBG: testbench.lp5 RAA (BA [1] - SB [0]) = 0(-32) -> 0
In case of Directed Refresh Management our Verification IP help to identify the victim row of the bank during the pre-charge/auto pre-charge operation.	Command: Precharge (with AB: 0 & Bank: 0) DRFM Address Sample Control: 1 *Denali* Detected[testbench.lp5] DRFM_ADDR_SAMPLED @315626 ps :: DRAM row address 0x1 of bank #0 is sampled for DRFM
The subsequent DRFM (Refresh) command clears this sampled address which our model helps to identify. If the address is not sampled, we trigger the assertion identifying the issue.	M *Denali* Detected[testbench.lp5] DRFM_ADDR_CLEARED @1073126 ps :: DRAM row address 0x01 of bank #0 is cleared. M *Denali* Error: Detected[testbench.lp5] DRFM_ADDR_NOT_SAMPLED @561876 ps :: No DRAM row address of bank #2 is sampled. Before issuing per bank DRFM command, DRAM address must be sampled.

Proven Results: Delivering High-Quality DRAM Designs

- 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.

Questions