The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions

Minesh Patel, Jeremie S. Kim, Onur Mutlu

Presented at ISCA, June 2017
Executive Summary

• **Motivation:** DRAM refresh energy/performance overhead is high
• **Problem:** DRAM retention failure profiling is hard
  – Complicated by cells *changing* retention times *dynamically*
  – Current profiling methods are *unreliable* or *too slow*
• **Goals:**
  1. Thoroughly analyze tradeoffs in retention failure profiling
  2. Develop a *fast* and *reliable* profiling mechanism
• **Key Contributions:**
  1. *First* detailed characterization of 368 LPDDR4 DRAM chips
  2. *Reach profiling:* Profile at a *longer refresh interval* and/or *higher temperature*, where cells are more likely to fail
• **Evaluation:**
  – 2.5x faster profiling with *99%* coverage and *50%* false positives
  – Enables longer refresh intervals that were previously unreasonable
DRAM Cell Leakage

DRAM encodes information in *leaky* capacitors

Stored data is *corrupted* if too much charge leaks (i.e., the capacitor voltage degrades too far)
Retention failure – when leakage corrupts stored data
Retention time – how long a cell holds its value
DRAM is Much More Than Just One Cell!

8 GiB DRAM = $6.4 \times 10^{10}$ cells
DRAM Refresh

DRAM refresh periodically restores leaked charge
- Every cell every refresh interval (default = 64ms)
- Significant system performance/energy overhead

![Average System Performance Overhead vs DRAM Chip Size (Gb)]
Decreasing Refresh Overhead

Most cells do not fail at a longer refresh interval

< 100 failures
Retention Failure Mitigation

• Prior works handle these few failures to allow **reliable** operation at a longer refresh interval
  • RAIDR [Liu+, ISCA’12]
  • SECRET [Lin+, ICCD’12]
  • ArchShield [Nair+, ISCA’13]
  • DTail [Cui+, SC’14]
  • AVATAR [Qureshi+, DSN’15]

**Need a fast and reliable profiling mechanism to find the set of retention failures!**

• However, they **assume** they can **perfectly** identify the set of failing cells to handle
REAPER Outline

1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. LPDDR4 Characterization
5. Reach Profiling
6. End-to-end Evaluation
Unfortunately, real DRAM cells have variation in retention times.

- Here, all cells have identical retention times.
- All cells require the same short refresh interval.
Sources of Retention Time Variation

• Process/voltage/temperature

• Data pattern dependence (DPD)
  • Retention times change with data in cells/neighbors
  • e.g., all 1’s vs. all 0’s

• Variable retention time (VRT)
  • Retention time changes randomly (unpredictably)
  • Due to a combination of various circuit effects
Heterogeneous Retention Times

Row Decoder

SA  SA  SA  SA  SA

Long  Moderate  Short

Can handle a longer refresh interval
Requires short refresh interval
Extended Refresh Interval (128ms)

How can we **quickly** and **reliably** determine the failing cells at an increased refresh interval $T$?
REAPER Outline

1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. Individual Bit Failures
5. Reach Profiling
6. End-to-end Evaluation
Solution #1: ECC-Scrubbing

Key idea: leverage error-correcting codes (ECC) by periodically accessing all ECC words to continuously detect new failures (e.g., AVATAR [Qureshi+, DSN’15])

• Pros
  • **Simple:** read accesses to all DRAM locations
  • **Low overhead:** DRAM is available during scrubs

• Cons
  • **Unreliable:** does not account for changes in data pattern, which changes cell retention times
    • Can potentially miss failures between scrubs
Solution #2: Brute-force Profiling

Key idea: for \{N\} data patterns * \{M\} test rounds:

1) Write data pattern to DRAM
2) Wait for the refresh interval

Our goals:

1) study profiling tradeoffs
2) develop a fast and reliable profiling mechanism

- Slow: Many test rounds required for reliability
- High overhead: DRAM is unavailable for a long time

Key idea:

for \{N\} data patterns * \{M\} test rounds:

1) Write data pattern to DRAM
2) Wait for the refresh interval

Our goals:

1) study profiling tradeoffs
2) develop a fast and reliable profiling mechanism

- Slow: Many test rounds required for reliability
- High overhead: DRAM is unavailable for a long time
1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. LPDDR4 Characterization
5. Reach Profiling
6. End-to-end Evaluation
Experimental Infrastructure

• 368 2y-nm LPDDR4 DRAM chips
  • 4Gb chip size
  • From 3 major DRAM manufacturers

• Thermally controlled testing chamber
  • Ambient temperature range: \{40°C – 55°C\} ± 0.25°C
  • DRAM temperature is held at 15°C above ambient
LPDDR4 Studies

1. Temperature
2. Data Pattern Dependence
3. Retention Time Distributions
4. Variable Retention Time
5. Individual Cell Characterization
New failing cells continue to appear over time
- Attributed to variable retention time (VRT)

The set of failing cells changes over time

Error correction codes (ECC) and online profiling are necessary to manage new failing cells

- New failing cells continue to appear over time
  - Attributed to variable retention time (VRT)

- The set of failing cells changes over time
Single-cell Failure Probability (Cartoon)

Probability of Read Failure vs. Refresh Interval (s)

- **Idealized cell** (retention time = 3s)
- **Actual cell** $N(\mu, \sigma) \mid \mu = 3s$
Any cell is more likely to fail at a *longer* refresh interval OR a *higher* temperature.
## REAPER Outline

1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. LPDDR4 Characterization
5. Reach Profiling
6. End-to-end Evaluation
Reach Profiling

**Key idea:** profile at a longer refresh interval and/or a higher temperature

**Pros**
- **Fast + Reliable:** reach profiling searches for cells where they are most likely to fail

**Cons**
- **False Positives:** profiler may identify cells that fail under profiling conditions, but not under operating conditions
A Complex Tradeoff Space

- **Profile here?**
  - Slower
  - Less reliable
  - Fewer false positives

- **Target**
  - Faster
  - More reliable
  - More false positives

- **Profile here?**

Temperature vs. refresh interval graph.
Towards an Implementation

Reach profiling is a general methodology

3 key questions for an implementation:

What are desirable profiling conditions?

How often should the system profile?

What information does the profiler need?
Three Key Profiling Metrics

1. **Runtime**: how long profiling takes

2. **Coverage**: portion of all possible failures discovered by profiling

We explore how these metrics change under **many** different profiling conditions
Q1: Desirable Profiling Conditions

• Similar trends across chips and vendors!

• For 99% coverage, we find on average:
  • \textbf{2.5x speedup} by profiling at \textbf{+250ms} at a cost of a \textbf{50\% false positive rate}
  • \textbf{>3.5x speedup} by profiling at \textbf{+ >500ms} at a cost of a \textbf{>75\% false positive rate}

• More examples and detail in the paper
Q2: How Often to Profile

• Estimation using a **probabilistic model**
  • Can use our empirical data for estimates
  • Details are in the paper

• e.g., Need to reprofile every **2.3 days** for a:
  • 2GB ECC DRAM
  • 1024ms refresh interval at 45°C
  • Profiling with 99% coverage
Q3: Necessary Information

• The cost of handling identified failures
  • Determines how many errors we can mitigate
  • e.g., error-correction codes (ECC)

• Empirical per-chip characterization data
  • Used to reliably estimate profiling parameters
  • Details are in the paper
1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. LPDDR4 Characterization
5. Reach Profiling
6. End-to-end Evaluation
Our Mechanism: REAPER

• Simple implementation of reach profiling

• Pessimistic assumptions
  • Whole system pauses during profiling
    • Firmware executes profiling routine
  • Exclusive DRAM access
  • Only manipulates refresh interval, not temperature
Evaluation Methodology

• Simulators
  • Performance: Ramulator [Kim+, CAL’15]
  • Energy: DRAMPower [Chandrasekar+, DSD’11]

• Configuration
  • 4-core (4GHz), 8MB LLC
  • LPDDR4-3200, 4 channels, 1 rank/channel

• Workloads
  • 20 random 4-core benchmark mixes
  • SPEC CPU2006 benchmark suite
Simulated End-to-end Performance

On average, REAPER enables:
- **16.3% system performance improvement**
- **36.4% DRAM power reduction**

REAPER enables longer refresh intervals, which are unreasonable using brute-force profiling.

Reprofile rarely  Reprofile often
Other Analyses in the Paper

• Detailed LPDDR4 characterization data
  • Temperature dependence effects
  • Retention time distributions
  • Data pattern dependence
  • Variable retention time
  • Individual cell failure distributions

• Profiling tradeoff space characterization
  • Runtime, coverage, and false positive rate
  • Temperature and refresh interval

• Probabilistic model for tolerable failure rates

• Detailed results for end-to-end evaluations
Executive Summary

• **Motivation**: DRAM refresh energy/performance overhead is high

• **Problem**: DRAM retention failure profiling is hard
  – Complicated by cells changing retention times *dynamically*
  – Current profiling methods are *unreliable* or *too slow*

• **Goals**:
  1. Thoroughly analyze tradeoffs in retention failure profiling
  2. Develop a *fast* and *reliable* profiling mechanism

• **Key Contributions**:
  1. *First* detailed characterization of 368 LPDDR4 DRAM chips
  2. *Reach profiling*: Profile at a *longer refresh interval* and/or *higher temperature*, where cells are more likely to fail

• **Evaluation**:
  – *2.5x* faster profiling with *99%* coverage and *50%* false positives
  – Enables longer refresh intervals that were previously unreasonable
The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions

Minesh Patel, Jeremie S. Kim, Onur Mutlu

Presented at ISCA, June 2017

SAFARI

ETH Zürich Carnegie Mellon

P&S SoftMC - 13 October 2021