Speeding Up Crossbar Resistive Memory by Exploiting In-memory Data Patterns

Wen Wen
Lei Zhao, Youtao Zhang, Jun Yang
Executive Summary

• **Problems: performance and reliability of write operations**
  • The large sneaky currents and IR drop issues in crossbar ReRAM

• **Proposed solutions: speeding up RESET operation based on data pattern**
  • Profiling the number of bitline LRS cells by exploiting intrinsic in-memory processing capability of crossbar ReRAM
  • Data compression and row address dependent layout to reduce bitline LRS cells

• **Contributions**
  • Correlation between the RESET latency and the number of LRS cells on selected bitlines
  • A novel profiling technique to dynamically track the bitline data patterns

• **Results**
  • Performance: 20.5% over baseline, 14.2% over state-of-the-art
  • Dynamic energy: 15.7% less than baseline, 7.6% less than state-of-the-art
Outline

• Background and Motivation
• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells
• Evaluation
  • Methodologies
  • Experimental Results
• Conclusion
• Background and Motivation

• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells

• Evaluation
  • Methodologies
  • Experimental Results

• Conclusion
ReRAM Cell

ReRAM cell structure and two resistance states

High Resistance State (HRS, Logic “0”)  Low Resistance State (LRS, Logic “1”)
ReRAM Crossbar

ReRAM array structures

✓ Smallest $4F^2$ planar cell size, low fab cost and better scalability.

✗ Sneak currents and IR drop
Sneak Currents in Crossbar ReRAM

- Diode selectors help but cannot eliminate sneak currents
- Sneaky currents lead to serious IR drop issue
  - Hurt energy efficiency, performance and write reliability
- The Slower RESET operation is the performance bottleneck
  - SET takes shorter time than RESET [Xu et al’HPCA15, Zhang et al’DATE16]
How does IR drop affect RESET latency?

\[ t \times e^{kV_d} = C \]

- RESET latency is highly sensitive, i.e., exponentially inverse correlation, to voltage drop
  - \( t \): RESET switching time; \( V_d \): voltage drop
  - \( C \) and \( k \) are experimental fittings constants
  - A voltage drop of 0.4V results in 10x RESET latency increase [Govoreanu et al’ IEDM11]
1. During RESET, half-selected cells exhibit as resistive devices.
2. With same voltage stress, a half-selected cell in LRS would have larger sneak current than the one in HRS.
3. RESET operations conservatively use the worst-case access latency of all cells in ReRAM arrays.

Facts

1. The number of half-selected cells in LRS affects RESET latency.
2. Dynamically profile and track runtime data patterns may avoid using worst-case RESET latency for all cells.

Observations

1. Explore correlation between RESET latency and the number of bitline LRS cells.
2. Need a runtime profiler to dynamically track bitline data patterns.
3. Need to reduce the number of LRS cells on bitlines.

TODO List
More LRS cells there are in the bitline, the larger IR drop the sneak current brings, and the longer time the RESET operation takes.
This impact **diminishes** as the row becomes **closer** to the write driver.
Outline

• Background and Motivation
• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells
• Evaluation
  • Methodologies
  • Experimental Results
• Conclusion
Low Overhead Runtime Profiling(1)

• The worst-case bitline data pattern within one bitline-sharing-set determines the optimal RESET latency
The worst-case bitline data pattern within one bitline-sharing-set determines the optimal RESET latency
• Aggregated currents firstly are converted into digital counters, which represent LRS cell percentages.
• W-Flag is updated by comparing all counters in one bit-sharing-set to decide the worst-case bitline.
• Until now, we’ve talked about how to profile bitline data patterns, but we have not exploit the impact of row address on RESET latency yet!
• The rows with different addresses are mapped to 8 groups with different worst-case RESET latencies.
A Summary for Profiling Technique

• Finding out the worst-case bitline: 3-bit W-Flag
  • Recording the percentage of LRS cells in worst-case bitline
  • Periodically detecting in each mat

• Tracking the worst-case: 6-bit W-Cnt
  • W-Cnt is cleared when W-Flag is updated
  • Increment the counter of W-Cnt for each write
  • W-Cnt overflow triggers increment of the W-Flag

• RESET latency optimization
  • W-Flag, W-Cnt and row address are used to determine tWR for RESET
RESET latency depends on bitline **data patterns (W-Flag, W-Cnt)** and row address.

<table>
<thead>
<tr>
<th>W-Flag, W-Cnt</th>
<th>Row address</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRS Cell Ratio</td>
<td>Row Address Group</td>
</tr>
<tr>
<td>111</td>
<td>0</td>
</tr>
<tr>
<td>202.4</td>
<td>197.7</td>
</tr>
<tr>
<td>110</td>
<td>0</td>
</tr>
<tr>
<td>202.4</td>
<td>197.7</td>
</tr>
<tr>
<td>101</td>
<td>0</td>
</tr>
<tr>
<td>199</td>
<td>194</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>189</td>
<td>184.3</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
</tr>
<tr>
<td>173.8</td>
<td>169.7</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
</tr>
<tr>
<td>154.6</td>
<td>150.9</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
</tr>
<tr>
<td>132.9</td>
<td>129.3</td>
</tr>
<tr>
<td>000</td>
<td>0</td>
</tr>
<tr>
<td>109.7</td>
<td>106.9</td>
</tr>
</tbody>
</table>

RESET latency conservatively uses the **upper limit number**, which is the **worst-case in next LRS cell ratio range**.
Determine RESET Timing (ns)

- RESET latency depends on bitline data patterns (W-Flag, W-Cnt) and row address

<table>
<thead>
<tr>
<th>W-Flag, W-Cnt</th>
<th>Row address</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRS Cell Ratio</td>
<td>Row Address Group</td>
</tr>
<tr>
<td>111</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>202.4</td>
</tr>
<tr>
<td>101</td>
<td>202.4</td>
</tr>
<tr>
<td>100</td>
<td>199</td>
</tr>
<tr>
<td>011</td>
<td>189</td>
</tr>
<tr>
<td>010</td>
<td>173.8</td>
</tr>
<tr>
<td>001</td>
<td>154.6</td>
</tr>
<tr>
<td>000</td>
<td>132.9</td>
</tr>
<tr>
<td>010</td>
<td>109.7</td>
</tr>
</tbody>
</table>

LRS: 62.5%~75%, but no exceeding 87.5% before W-Cnt overflows

- RESET latency conservatively uses the upper limit number, which is the worst-case in next LRS cell ratio range
• Background and Motivation

• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells

• Evaluation
  • Methodologies
  • Experimental Results

• Conclusion
Naïve data compression cannot help since:

- The RESET latency depends on the worst-case of all 512 bitlines
- Naïve data compression may not help the worst-case bitline
- Row-address biased data layout
  - Evenly distribute extra 0s after compression
Outline

• Background and Motivation
• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells
• Evaluation
  • Methodologies
  • Experimental Results
• Conclusion
Experimental Methodologies

- **Chip-multiprocessor system simulator**
  - CPU, multi-level cache and ReRAM memory

- **ReRAM**
  - Performance/energy numbers from HSPICE and NVSim
  - Memory: 8GB, 1 channel, 2 ranks, 8 chips/rank, 2Gb x8 ReRAM Chip, 8 banks/chip, 1024 mats/bank
  - ReRAM Timing: **Read**: 18ns@1.5V, **SET**: 10ns@3V, **RESET is dynamically determined**@-3V, 88µA

- **Workloads**
  - SPEC2006, BioBench and PARSEC
  - WPKI/RPKI: High, Medium, Low

- **Compared schemes**
  - **BL**: Conventional ReRAM crossbar design with DSGB
  - **RA**: The state-of-the-art row address awareness technique [Zhang et al’ DATE’16]
  - **LRS**: Naïve data pattern profiling technique
  - **CMP**: LRS + data compression + row-address biased shifting technique
  - **ALL**: Proposed design with all enhancements
• Background and Motivation
• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells
• Evaluation
  • Methodologies
  • Experimental Results
• Conclusion
System Performance

Normalized Cycle Per Instruction Comparison (The lower, the better)

- More performance improvements on high memory intensity benchmarks
Memory Dynamic Energy and EDP

- 15.7% and 7.6% dynamic energy reduction, and 31.9% and 19.5% EDP improvements over baseline and state-of-the-art, respectively.
Outline

• Background and Motivation
• Design Details
  • Low Overhead Runtime Profiling
  • Reduce Bitline LRS Cells
• Evaluation
  • Methodologies
  • Experimental Results
• Conclusion
Conclusion

- **Problems: performance and reliability of write operations**
  - The large sneaky currents and IR drop issues in crossbar ReRAM

- **Proposed solutions: speeding up RESET operation based on data pattern**
  - Profiling the number of bitline LRS cells by exploiting intrinsic in-memory processing capability of crossbar ReRAM
  - Data compression and row address dependent layout to reduce bitline LRS cells

- **Contributions**
  - Correlation between the RESET latency and the number of LRS cells on selected bitlines
  - A novel profiling technique to dynamically track the bitline data patterns

- **Results**
  - Performance: 20.5% over baseline, 14.2% over state-of-the-art
  - Dynamic energy: 15.7% less than baseline, 7.6% less than state-of-the-art
Thank you!
Speeding Up Crossbar Resistive Memory by Exploiting In-memory Data Patterns

Wen Wen
Lei Zhao, Youtao Zhang, Jun Yang
Backup Slides
# Prior Arts

## Performance of RESET operation
- Double-sided ground biasing and multi-phase RESET [Xu et al’HPCA15]
- Row address dependent logical regions [Zhang et al’DATE16]
- No studies on impact of bitline data patterns

## Current accumulation feature of ReRAM crossbar
- Using ReRAM crossbar to implement dot-product analogy calculations [Bojnordi et al’HPCA16, Chi et al’ISCA16, Shafiee et al’ISCA16, Song et al’HPCA17]
- Few studies on exploiting this feature to accelerate memory access

## Data pattern in ReRAM crossbar
- Correlation between voltage drop and data pattern in ReRAM crossbar [Liang et al’TED2010]
- Correlation between Read latency and bitline data pattern [Zhang et al’JETCAS15]
- Correlation between RESET latency and the number of RESET bits [Xu et al’HPCA15]
- No observations on correlation between RESET latency and bitline data pattern
• Build and simulate crossbar ReRAM HSPICE model
  • Key parameters are summarized below

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Mat Size: A wordlines A bitlines</td>
<td>512 ×512</td>
</tr>
<tr>
<td>n</td>
<td>Number of bits to read/write</td>
<td>8</td>
</tr>
<tr>
<td>Rwire</td>
<td>Wire resistance between adjacent cells</td>
<td>2.82Ω</td>
</tr>
<tr>
<td>Kr</td>
<td>Nonlinearity of the selector</td>
<td>200</td>
</tr>
<tr>
<td>Vw</td>
<td>Full selected voltage during write</td>
<td>3V</td>
</tr>
<tr>
<td>-</td>
<td>Voltage biasing Scheme</td>
<td>DSGB</td>
</tr>
</tbody>
</table>
## Overhead Analysis

### Profiling
- Power consumption/area estimation by HSPICE and NVSim at 32nm
- **Profiling overhead is small: 3.7x read energy, negligible area overhead**
- Profiling overhead of one bank is summarized as below

<table>
<thead>
<tr>
<th>Comp.</th>
<th>Params</th>
<th>Spec.</th>
<th>Power/Energy</th>
<th>Area (mm$^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>Sampling speed, Resolution</td>
<td>1.28GS/s</td>
<td>24.48mW</td>
<td>0.012</td>
</tr>
<tr>
<td></td>
<td>Number, 8-bit</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S+H</td>
<td>Number</td>
<td>8 ×64</td>
<td>5uW</td>
<td>0.00002</td>
</tr>
<tr>
<td>ReRAM Array</td>
<td>Mat number, Mat size</td>
<td>1024 ×512</td>
<td>Profiling: 267.178pJ</td>
<td>2.078</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read: 72.842pJ</td>
<td>Leakage: 255.233mW</td>
<td></td>
</tr>
</tbody>
</table>

### Counters storage and RESET adjustment
- 3-bit W-Flag and 6-bit W-Cnt for each bitline-sharing-set
- 288KB storage overhead of all flags for a 8GB memory
Sensitivity Study

- Trivial improvement by doubling the number of ADC units
  - Only 1.1% performance improvement observed

- The proposed scheme is slightly worse (only 1.6%) than RA for 256 x 256 mat size
  - Profiling overhead is independent on mat size, and has larger impact on smaller mats