Architecting Phase Change Memory as a Scalable DRAM Alternative

Benjamin C. Lee      Engin Ipek      Onur Mutlu      Doug Burger

June 2009 ISCA

Presented by Moritz Herting

19.03.2020
Summary

- DRAM is hard to scale down
  - Scaling down decreases Power Consumption and increases Capacity
- Can we replace DRAM with PCM?
  - PCM is easy to scale down
- Get Latency, Power Consumption and Area onto the same Level
- Rearrange Buffer and introduce Partial Writes
- Evaluate different Configurations which use the same Area and compare Latency, Power draw and Endurance
- First to show how to use PCM Technology to architect main Memory that is close to DRAM Performance and has other Advantages as well
Outline

- Basics of DRAM and PCM
- Experimental Methodology
- Architectural Changes
- Process Scaling Improvements
- Conclusion
- Discussion
DRAM Structure

- **DRAM Cell consists of one Capacitor and one Transistor**
- **Store a Bit**
  - Charge/Discharge the Capacitor
- **Read a Bit**
  - The Charge of the Capacitor gets directly to the Buffer via the Bitline
- **Not easily scalable**
  - Smaller Capacitors have smaller Charge Capacity
  - Smaller Access Transistors increase Charge Leakage
  - Harder to store Charge for a long Time
PCM Structure

- Similar general Structure as DRAM
- Different Storage Elements
- Needs a special Sense Amplifier and can’t connect directly to the Buffer
- Data gets stored by changing the physical Property of a Material
PCM Storage Element

- Cool down Time determines the Resistance of the Chalocogenide
- Heat up the Chalcogenide to 650°C
- Induce Current onto the Heating Element

No size limiting element!
Writing to a Cell

- Fast Temperature Drop induces amorphous State → High Resistance

- Slow Temperature Drop induces crystalline State → Low Resistance

- Intermediate States are possible → Multi Level Cells
Reading from a Cell

• **Single Level Cells**
  - Low Resistance => 0
  - High Resistance => 1

• **Multi Level Cells**
  - Lowest Resistance => 00
  - Higher Resistances => 01, 10 or 11
# DRAM vs PCM Cell

<table>
<thead>
<tr>
<th>Feature</th>
<th>DRAM [DDR2]</th>
<th>PCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Latency</td>
<td>5 cycles</td>
<td>4.4x 22 cycles</td>
</tr>
<tr>
<td>Write Latency</td>
<td>5 cycles</td>
<td>12x 60 cycles</td>
</tr>
<tr>
<td>Read Energy</td>
<td>1.17 pJ/bit</td>
<td>2.11x 2.47 pJ/bit</td>
</tr>
<tr>
<td>Write Energy</td>
<td>0.39 pJ/bit</td>
<td>43.12x 16.82 pJ/bit</td>
</tr>
<tr>
<td>Area</td>
<td>6 F²/cell</td>
<td>Multi level cells</td>
</tr>
<tr>
<td>Scalable</td>
<td>Not easily</td>
<td>Yes</td>
</tr>
<tr>
<td>Volatile</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Refreshes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Outline

- Basics of DRAM and PCM
- Experimental Methodology
- Architectural Changes
- Process Scaling Improvements
- Conclusion
- Discussion
Methodology

- **Impact on Applications**
  - Latency
  - Power Consumption

- **Endurance**

- **Simulation using SESC**

- **4 Core Superscalar, Out-of-Order CPU @4GHz**

- **Parallel Workloads**

- **Memory intense Workloads**
Performance/Energy Baseline of PCM

- **Latency**
  - 1.2x up to 2.2x
  - 2.16x on Average

- **Energy**
  - 1.4x up to 3.4x
  - 2.2x on Average
Outline

- Basics of DRAM and PCM
- Experimental Methodology
- Architectural Changes
- Process Scaling Improvements
- Conclusion
- Discussion
Problems to solve

- High Latency
- High Energy Usage
- Limited Endurance
- Increased Area Usage

Buffer Reorganization

Partial Writes
Buffer Reorganization

Narrower Rows

Multiple Buffers
Narrow Rows

- Decrease Amount of simultaneous Writes
  - Decrease Power Draw per Array write
  - Improve Endurance
  - Less write Coalescence

- Decrease Amount of simultaneous Reads
  - Decrease Power Draw per Array read
  - Less spacious Coherence

- Fewer Latches per Buffer
  - Decrease Power Draw
  - Decreased Area
Multiple Rows

- **Less Conflict Misses**
  - More read and write Coalescence
    - Lower Latency
- **Less frequent Reads and Writes**
  - Lower Power Usage
  - Improves Endurance
- **Increased Area**
Evaluating Buffer Reorganizations
Performance and Energy

Possible Approaches within the same Area as DRAM

- Huge Difference in Performance and Energy Usage depending on the Approach
- Smaller 9F^2 Cells wouldn’t enable us better Approaches
- 4 x 512B seems like a good Approach
  - Reduced Latency from 1.6x times (Baseline) to 1.16x
  - Reduced Power Consumption from 2.2x to about the same Level

(PCM Baseline would use too much Area)
Performance of Row Buffer Configurations

- Single buffer
  - Not much spatial Locality
  - Will get evicted to fast for temporal Locality
- Multiple decently sized Buffers
  - Able to use temporal Locality
- 4x512B Buffer lead to 66% as much Delay as one 2048B Buffer
Write Coalescing different Approaches

- Multiple not to small Buffers significantly decrease the Number of Writes
- More/bigger Buffers won’t significantly decrease the Number of Writes
- 4x512B Buffer lead to 53% less Writes compared to one 2048B Buffer
Energy Usage of different Approaches

- One and two Buffers use a similar Amount of Energy
  - Fewer Reads and Writes impact Energy Consumption way more than doubling the Row Buffer
- 4 Buffers won’t use way more Energy than just two
- Increasing the Width of small Buffers won’t use much more Energy than they save from less Cell Accesses
- Increasing the Width of big Buffers won’t save us enough Cell Accesses to justify the additional Energy Consumption of the Buffer itself
- 4x512B Buffer is a good Middle Ground
Problems to solve

- High Latency
- High Energy Usage
- Limited Endurance
- Increased Area Usage

Buffer Reorganization

Partial Writes
Partial Writes Idea

![Diagram of partial writes idea with binary numbers and highlighted dirty areas.](image-url)
Partial Writes Functionality

- Decreases the average Amount of written Bits per Array Write
- Reduce total number of Cell Writes
  - Enhance Endurance
  - Decrease Power Consumption
- Store one dirty Bit per Block
- Buffer Reordering will Accommodate for Area Overheads
- Requires very small Changes in CPU Cache Structure to include those dirty Bits
- 64B and 6B Approach
Partial Writes required Changes

• **64B Blocks**
  - Tracking begins at L2 Cache
  - Requires one Bit per L2 Cache Line
  - 0.2% Overhead in L2 Cache
  - No Change in L1 Cache needed

• **4B Blocks**
  - Tracking begins at L1 Cache
  - Requires 16b per L2 Cache Line
  - Requires 6b per L1 Cache Line
  - 3.1% Overhead in each Cache
Partial Writes Endurance Evaluation

- 0.7 Years with 64B Blocks
- 5.6 Years with 4B Blocks
- Would increase by 4 Orders of Magnitude with 32nm Process Size
  - ~700 Years with 64B
  - ~5'600 Years with 4B
Outline

- Basics of DRAM and PCM
- Experimental Methodology
- Architectural Changes
- Process Scaling Improvements
- Conclusion
- Discussion
Process Scaling Benefits

● Can further reduce PCM Energy Costs
● Improve Endurance further
● Increase Density
  - Increase total Capacity
  - Decrease Price/Capacity Ratio
● Won’t decrease Latency
Scaling Improvements from 90nm to 40nm

- PCM will use 61.3% of Energy compared to DRAM
- Will decrease Power by another 2.4x
- DRAM only decreases Power by 1.5x
- PCM scales 1.6x faster
Conclusion

• DRAM is hard to scale down
  – Scaling down decreases Power Consumption and increases Capacity

• Can we replace DRAM with PCM?
  – PCM is easy to scale down

• Get Latency, Power Consumption and Area onto the same Level

• Rearrange Buffer and introduce Partial Writes

• Evaluate different Configurations which use the same Area and compare Latency, Power draw and Endurance
Strengths and Weaknesses

- **The Good:**
  - Good Structure
  - Most of the important Numbers and Assumptions are clear and Sources are easily retractable
  - Almost all important Aspects are evaluated

- **The Bad:**
  - No Energy Evaluation with Partial Writes
  - Only Memory intensive Workloads have been looked at
    - Maybe some unforeseen Behavior
  - Some more Numbers would have been nice
    - DRAM Scaling, presumably 90nm as well
    - Expected Run Time per Year for the Endurance Evaluation, presumably 24/7/365
    - IPC for CPU Simulation
  - Rather exact Numbers regarding they are extrapolated from Simulations/Predictions
    - Maybe some expectable Derivations
Situation Today

• DDR4
  - Can be produced in 12nm Process Node Size
  - Similar Cell Latency compared to DDR2
  - Less than half the Power Consumption of DDR2
  - Higher Data Rate

• Buffer Reorganization for DRAM has been proposed in 2011
  - 35.8% improved Performance (4 core)
  - 42% Reduction in Energy (4 core)

• Low-Latency PCM
  - 119% higher Performance than normal PCM
  - 43% less Energy

• PDRAM
  - Hybrid System
  - 30% Energy Savings

• Optane/3D XPoint
  - Might be based on PCM
  - Hard to find exact Numbers
  - Similar Latency to DDR4
  - 1/3 of the Bandwith
  - Already in use in Enterprise Solutions as an Addition to DRAM
  - In use by consumers as an HDD Cache to cheaply bring Performance to a similar Level as an SSD
Situation Today – Further Readings

- **Samsung 12nm DDR4 Chip**

- **DRAM Buffer Reorganization**

- **Low-Latency PCM**
  - https://dl.acm.org/doi/10.1145/3316781.3317853

- **PDRAM (Hybrid System)**

- **Optane**

- **NVRAM Standard Proposal (video)**
  - https://youtu.be/xxpF5oVZsrA
Scalable Alternatives

- **Flash (also mentioned in the Paper)**
  - Slow
  - https://de.wikipedia.org/wiki/Flash

- **Static RAM**
  - Only volatile Alternative
  - https://de.wikipedia.org/wiki/Static_random-access_memory

- **Optane/3D XPoint**
  - Not available at the Time of Publication

- **Optical PCM**
  - Not yet available
  - https://www.youtube.com/watch?v=UWMEKex6nYA (video)
  - https://www.osapublishing.org/ol/abstract.cfm?uri=ol-44-7-1821

- **Ferroelectric RAM**
  - https://de.wikipedia.org/wiki/Ferroelectric_Random_Access_Memory

- **Resistive RAM**
  - https://de.wikipedia.org/wiki/Resistive_Random_Access_Memory

- **Magnetoresistive RAM**
  - https://de.wikipedia.org/wiki/Magnetoresistive_Random_ACCESS_Memory

- **Nanotube based RAM**
  - https://de.wikipedia.org/wiki/NRAM
  - https://www.youtube.com/watch?v=V1HN0w_algg
Own Thoughts

- Good Addition to volatile DRAM
- No full replacement in Performance oriented Devices
- Maybe a suitable replacement in Business oriented Ultrabooks/Laptops
- Probably coming more to Consumer Products soon
  - Optane
- **Hybrid Systems**
  - Power Consumption
  - Security
  - Performance
- **Security Implications**
  - Encrypted Hard Drive
    - Volatile Encryption/Decryption Unit
    - Volatile Accessor on the same Die which has to be cryptically unlocked after each Power Loss

- Others say Holy Grail is Persistence up until CPU Registers
Discussion Topics

• What are some use Cases for persistent main Memory in Applications?
• What could be the Place of PCM within Today's Computers?
• If we could have PCM good enough to replace Memory up until CPU Registers
  - Would there still be a Reason for Volatile Memory?
  - Which Applications and Use cases could profit from this and how?
• Can you come up with some Workloads where Persistence could be important/valuable enough to justify some Loss in Performance and/or Power Consumption?
• Do you think PCM would have better Chances when looking at Applications which are less Memory intensive
• Do you think there is a Difference between Workloads who have a relative higher Amount of
  - Reads
  - Writes
• Do you think there would be a Difference when looking at Consumer Workloads and Usage instead of Enterprise Usage
  - Gaming
  - Operation System Performance
  - Browsing the Eeb
  - Office Work (PowerPoint, Word, Excel)
• Do you think it would be easy/quick to change Applications to better make use of persistent Memory
• Can you come up with some Workloads where Persistence could be a Disadvantage and why?
  - Security
    • Drive Encryption
Architecting Phase Change Memory as a Scalable DRAM Alternative

Benjamin C. Lee      Engin Ipek      Onur Mutlu      Doug Burger

Discussion Link: https://moodle-app2.let.ethz.ch/mod/forum/discuss.php?d=45327#p89471

June 2009 ISCA

Presented by Moritz Herting

19.03.2020
### Appendix

<table>
<thead>
<tr>
<th></th>
<th>PCM</th>
<th>DRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Array</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A$</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>$C$</td>
<td>9MLC, 12MLC</td>
<td>6</td>
</tr>
<tr>
<td>cell size ($F^2$)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|                |              |              |
| **Periphery**  |              |              |
| $S$            | 44           | 14           |
| sense amplifier (T @ $250\lambda^2/\text{T}$) | 2750        | 875         |
| sense amplifier ($F^2$) |              |              |
| $L$            | 8            | 0            |
| latch (T @ $250\lambda^2/\text{T}$) | 500         | 0           |
| latch ($F^2$)  |              |              |
| $D$            | 6            | 0            |
| decode 2-AND (T @ $1000\lambda^2/\text{T}$) | 250         | 0           |
| decode 2-AND ($F^2$) |              |              |

|                |              |              |
| **Buffer Organization** |              |              |
| $W$            | 64::2x::2048 | 2048         |
| buffer width (B) |              |              |
| $R$            | 1::2x::32   | 1            |
| buffer rows (ea) |              |              |

\[
\hat{A}_D = \underbrace{A \cdot C_D}_{\text{array}} + \underbrace{W_D \cdot S_D}_{\text{sense}}
\]

\[
\hat{A}_P = \underbrace{A \cdot C_P}_{\text{array}} + \underbrace{W_P \cdot S_P}_{\text{sense}} + \underbrace{R_P \cdot W_P \cdot L_P}_{\text{latch}} + \underbrace{R_P \cdot G(\log_2 R, 2) \cdot D_P}_{\text{decode}}
\]
### Appendix

**Endurance**

<table>
<thead>
<tr>
<th>(W)</th>
<th>writes per second per bit</th>
<th>calc</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\hat{L})</td>
<td>memory module lifetime (s)</td>
<td>calc</td>
</tr>
<tr>
<td>(E)</td>
<td>write endurance</td>
<td>(1E+08)</td>
</tr>
</tbody>
</table>

**Memory Module**

| \(C\) | logical capacity (Gb) | 2 |

**Memory Bus Bandwidth**

| \(f_m\) | memory bus frequency (MHz) | 400 |
| \(M_f\) | processor frequency multiplier | 10 |
| \(B\) | burst length (blocks) | 8 |

**Application Characteristics**

| \(N_w, N_r\) | number of writes, reads | sim |
| \(T\) | execution time (cy) | sim |

**Buffer Characteristics**

| \(W_P, R_P\) | buffer width (B), rows | 512, 4 |
| \(N_{wb}, N_{wa}\) | buffer, array writes | sim |
| \(\delta\) | fraction of buffer written to array | sim |

\[
\hat{W} = \frac{f_m \cdot (N_w + N_r) \cdot (B/2) \cdot M_f}{B/2} \times \frac{N_w}{\frac{N_w + N_r}{\text{writeIntensity}}} \times 8W_P \cdot \left(\frac{N_{wa}}{N_{wb}}\right) \cdot \delta \times \frac{1}{C/2}
\]