Towards MRAM Byte-Addressable Persistent Memory in Edge Database Systems

Luís Ferreira, Fábio Coelho, José Pereira
INESC TEC & University of Minho

August 28, 2023
Towards MRAM Byte-Addressable Persistent Memory in Edge Database Systems

Endpoints

80% of generated data by 2025

E.g., IoT devices

Why place data at the endpoints?

• Connectivity
• Privacy
• Latency
• Energy
• Scalability
Towards MRAM Byte-Addressable Persistent Memory in Edge Database Systems

Endpoints

80% of generated data by 2025

E.g., IoT devices

Why place data at the endpoints?
• Connectivity
• Privacy
• Latency
• Energy
• Scalability

Microcontrollers (MCUs)

Microprocessors (MPUs)

Very limited resources
Towards MRAM Byte-Addressable Persistent Memory in Edge Database Systems

Endpoints

80% of generated data by 2025

E.g., IoT devices

Why place data at the endpoints?
- Connectivity
- Privacy
- Latency
- Energy
- Scalability

Microcontrollers (MCUs)

Microprocessors (MPUs)

Very limited resources

Storage medium

FLASH
E.g., SD Card, embedded

MRAM Persistent Memory
Background - FLASH

FLASH

Page
Page
Page
Page

Block
0 ... 8KB

Page
Page
Page
Page

Block
0 ... 8KB

Page
Page
Page
Page

Block
0 ... 8KB

Read/Program

Erase
Background - FLASH

FLASH

0 ... 8KB

Page
Page
Page
Page

Page
Page
Page
Page

Page
Page
Page
Page

Read/Program

Erase

Split Read/Program
Background - FLASH

FLASH

[Diagram of FLASH architecture with details on page, block, and erase/program cycles]

- Page
- Block
- Read/Program
- Erase

Split Read/Program

- Many erase/program cycles remaining
- Few erase/program cycles remaining

Wear-leveling

With

Without
Background - FLASH

NAND vs NOR

Split Read/Program

Erase/program cycles remaining

With

Without

Wear-leveling

Background - FLASH

NAND vs NOR
Problem

FLASH physical limitations:
• I/O operations incur extra computational overhead
• Small I/O operations are slow
• Limited endurance

• Asymmetric performance for random and sequential accesses
# Specification comparison: MRAM vs FLASH

## Table 1
Storage devices’ characteristics comparison.

<table>
<thead>
<tr>
<th>MRAM</th>
<th>Read</th>
<th>Write</th>
<th>Erase</th>
<th>Capacity</th>
<th>Endurance</th>
<th>Energy</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1) AS3004316 [24]</td>
<td>57 MB/s</td>
<td>57 MB/s</td>
<td>N/A</td>
<td>4 Mb</td>
<td>100T</td>
<td>1.58 nJ/B</td>
<td>6.63 €/Mb</td>
</tr>
<tr>
<td>M2) MR4A16BMA35 [25]</td>
<td>57 MB/s</td>
<td>57 MB/s</td>
<td>N/A</td>
<td>32 Mb</td>
<td>Inf</td>
<td>1.58 nJ/B</td>
<td>1.99 €/Mb</td>
</tr>
<tr>
<td>M3) EMxxLx [26]</td>
<td>400 MB/s</td>
<td>400 MB/s</td>
<td>N/A</td>
<td>64 Mb</td>
<td>Inf</td>
<td>0.895 nJ/B</td>
<td>0.84 €/Mb</td>
</tr>
<tr>
<td>M4) EMD4E001GAS2 [27]</td>
<td>2.6 GB/s</td>
<td>2.6 GB/s</td>
<td>N/A</td>
<td>1 Gb</td>
<td>0.01T</td>
<td>0.523 nJ/B</td>
<td>0.098 €/Mb</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FLASH</th>
<th>Read</th>
<th>Write</th>
<th>Erase</th>
<th>Capacity</th>
<th>Endurance</th>
<th>Energy</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAND FLASH [21]</td>
<td>235 MB/s</td>
<td>23.5 MB/s</td>
<td>737 MB/s</td>
<td>128 Gb</td>
<td>60K</td>
<td>7.02 nJ/B</td>
<td>0.0012 €/Mb</td>
</tr>
<tr>
<td>NOR FLASH [28]</td>
<td>337 MB/s</td>
<td>2.5 MB/s</td>
<td>0.65 MB/s</td>
<td>512 Mb</td>
<td>100K</td>
<td>66.8 nJ/B</td>
<td>0.023 €/Mb</td>
</tr>
</tbody>
</table>
Background - MRAM

MRAM Persistent Memory

• Byte–addressable
• Maximum throughput with small I/O operations
• No need for erase before write
• No wear-leveling

• Symmetric performance for sequential and random accesses
Where does MRAM stand out?
Experiments

Raw performance

Key-value stores
- LPHT
- CLHT[1]
- Control: RocksDB

Relational databases
- SQLite

## Experimental Setup

### Table 2
Hardware specifications.

<table>
<thead>
<tr>
<th></th>
<th>STM32</th>
<th>RPi</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Model</td>
<td>STM32H743ZI</td>
<td>BCM2837</td>
</tr>
<tr>
<td>CPU frequency</td>
<td>480 MHz</td>
<td>1.2 GHz</td>
</tr>
<tr>
<td>CPU cores</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>RAM</td>
<td>1MB</td>
<td>1GB</td>
</tr>
<tr>
<td>Storage class</td>
<td>MRAM</td>
<td>SD Card</td>
</tr>
<tr>
<td>Storage size</td>
<td>4 Mb</td>
<td>32GB</td>
</tr>
<tr>
<td>Peak energy</td>
<td>66 mW</td>
<td>360-1440 mW[4]</td>
</tr>
<tr>
<td>Max. write cycles</td>
<td>$10^{14}$</td>
<td>$10^3 - 10^{4}$</td>
</tr>
<tr>
<td>Cost (Euros)</td>
<td>60</td>
<td>50</td>
</tr>
</tbody>
</table>
Raw Performance Comparison: MRAM vs FLASH

- Overall better performance
- Maximum throughput achieved at much smaller I/O operations.
- Symmetric random and sequential access performance.

Figure 1: Storage medium read and write throughput.
Data Systems: Key-value stores

LPHT

<table>
<thead>
<tr>
<th>Key hash()</th>
<th>Slot 13: Occupied</th>
<th>Slot 14: Occupied</th>
<th>Slot 15: Empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>11011000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11000000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11111110</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hit!

MRAM

Occupation status

CLHT[1]

<table>
<thead>
<tr>
<th>Bucket 1-1 (cache-line size and memory aligned)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata Slot 1: Occupied Slot 2: Occupied Slot 3: Empty</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bucket 2-1 (cache-line size and memory aligned)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata Slot 1: Occupied Slot 2: Occupied Slot 3: Occupied</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bucket 3-1 (cache-line size and memory aligned)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata Slot 1: Occupied Slot 2: Occupied Slot 3: Empty</td>
</tr>
</tbody>
</table>

MRAM

Linked-list

Code available at: https://github.com/luismeruje/Hashtables-STM32

LPHT vs RocksDB

- Big advantage in scenarios with small key/values (most common scenario)
- 134x to 3837x better for write scenario with fsync
- 1.4x to 35x better for write scenario without fsync
- 1.64x to 6.69x better for read scenario (below 32 bytes)

Figure 2: RocksDB (NAND FLASH) vs Linear Probing Hashmap (MRAM) with varying key/value size. Results for write and read operations show on the left side, and right side, respectively.
CLHT vs RocksDB

- Between 11x and 1827x more put operations per second
- Around 9x more get operations per second

Figure 3: Comparison between CLHT running on MRAM and RocksDB running on NAND FLASH in RPi.
Data Systems: SQLite

Source: https://www.sqlite.org/vfs.html

Code available at: https://github.com/luismeruje/SQLite-STM32
Data Systems: SQLite

STM32 with MRAM loses on almost all scenarios. Only outperforms RPi with FLASH in insert scenario with 2 rows/transaction.

Increased storage performance does not compensate lower computation capabilities.

Possible improvements:
- Optimized file system (remove overhead)
- Use hashing controllers, DMAs, and others to take load off the CPU.
- MRAM memory with better performance.

Figure 4: Comparison between SQLite running on: STM32’s MRAM and RPi’s NAND FLASH.
Discussion

• Big advantage in small I/O operations.
• Better endurance and performance.
• Less energy consumption.
• Less computational overhead.
• Possibly replace MPUs with MCUs.
Discussion

• Big advantage in small I/O operations.
• Better endurance and performance.
• Less energy consumption.
• Less computational overhead.
• Possibly replace MPUs with MCUs.
• Downsides: low capacity, high cost.
Discussion

• Big advantage in small I/O operations.
• Better endurance and performance.
• Less energy consumption.
• Less computational overhead.
• Possibly replace MPUs with MCUs.
• Downsides: low capacity, high cost.

• Hybrid approach could be the best solution for this moment.
• Barely grasping at the capabilities of MRAM, M3 and M4 would likely give much better results.
Towards MRAM Byte-Addressable Persistent Memory in Edge Database Systems

Luís Ferreira, Fábio Coelho, José Pereira
INESC TEC & University of Minho

August 28, 2023

Code repositories:
https://github.com/luismeruje/SQLite-STM32 (SQLite adaptation)
https://github.com/luismeruje/Hashtables-STM32 (LPHT and CLHT adaptations)