Fast cacheline-based data replacement for hybrid DRAM and STT-MRAM main memory

Chenji Liu¹,２, Lan Chen¹,２, a), Xiaoran Hao¹, Mao Ni¹, and Hao Sun¹, ２

Abstract The development of DRAM cannot meet the low power requirement of IoT applications due to the high refresh power. As one of new non-volatile memory, STT-MRAM has extremely low static power, high read performance and high endurance. In this paper, we build a hybrid DRAM and STT-MRAM main memory to reduce energy. Considering STT-MRAM’s high write power and high write latency, we propose a fast cacheline-based data replacement to reduce write operations of STT-MRAM. The results show that the hybrid DRAM and STT-MRAM main memory can provide comparable performance to DRAM, with an average 32% reduction in main memory energy.

Keywords: IoT, STT-MRAM, hybrid main memory, data replacement

1. Introduction

The rise of the Internet of Things (IoT) has led to an explosion of IoT devices [1, 2, 3]. IoT devices may be battery operated and do local computation on sensing data, which puts high requirements on the performance and energy of main memory [4, 5]. However, DRAM needs to be periodically refreshed, which results in high refresh power. Research shows that DRAM power in a smartphone occupied with a fraction ranging from 5% to 30% of the overall system power [6, 7]. It is difficult for DRAM to meet the low power requirement of future IoT applications.

STT-MRAM is one of the most promising alternatives to DRAM. Compared to other new Non-Volatile Memories (NVMs), STT-MRAM has advantages of low latency, high write endurance, and CMOS compatibility. Compared to DRAM, STT-MRAM has extremely low static power, comparable read performance (latency and power), but poor write performance (latency and power) and low density. However, since the STT-MRAM technology was first proposed by Hosomi et al. [8] in 2005, the capacity of STT-MRAM devices has increased rapidly. Tsuchida et al. [9] proposed a 64Mb STT-MRAM in 2010. Rizzo et al. [10] developed a 64 Mb DDR3 ST-MRAM in 2013. Rho et al. [11] presented the first 4Gb STT-MRAM in 2017. For commercial chips, EVERSPIN has introduced 256Mb DDR3 STT-MRAM [12] and 1Gb DDR4 STT-MRAM [13]. The rapid development of STT-MRAM technology offers great possibility for replacing DRAM. In this paper we build a hybrid DRAM and STT-MRAM main memory, using STT-MRAM’s non-volatility to reduce main memory energy and using DRAM to guarantee performance. We analyze internal operations of STT-MRAM device and find the actual command that triggers the write operations to the STT-MRAM cell. Then we propose a fast cacheline-based data replacement to reduce the write operations to STT-MRAM.

The rest of the paper is organized as follows. Section 2 introduces the related work. Section 3 analyzes internal operations of STT-MRAM device. Section 4 describes the fast cacheline-based data replacement. Section 5 introduces the experimental setup. Section 6 discusses the results. Section 7 concludes the study.

2. Related work

Nowadays, most of the research on STT-MRAM focus on on-chip cache. However, using STT-MRAM as main memory is not extensive, and these studies have made STT-MRAM completely replacing DRAM.

Some studies [14, 15] adjust diverse device-level parameters of STT-MRAM to make the storage capacity of STT-MRAM comparable to DRAM with better performance as well as power consumption. Meza et al. [16] show that reducing the row buffer size can greatly reduce main memory dynamic energy compared to a DRAM with large row sizes. Kultursay et al. [17] use partial write and row buffer write bypass to optimize STT-MRAM main memory. Kazi et al. [18] show that the overall system performance of large HPC clusters is not particularly sensitive to main memory latency. Kazi et al. [19] investigate the feasibility of using STT-MRAM in real-time embedded systems. Wang et al. [20] design an LPDDRx-compatible MRAM interface to solve the pin-compatibility. Kazi et al. [21] present a detailed analysis of STT-MRAM main memory timing and propose an approach to perform a reliable system level simulation of STT-MRAM. Komalan et al. [22] propose to use realistic, calibrated STT-MRAM models to better consider technologies aspects and architecture constraints.

3. Internal operations of STT-MRAM device

3.1 STT-MRAM cell structure

STT-MRAM cell structure is shown in Fig. 1. An STT-MRAM cell consists of a MOS transistor and a Magnetic...
Tunnel Junction (MTJ). The MTJ is composed of one ferromagnetic layer with fixed magnetization direction, one ferromagnetic layer with free magnetization direction, and one tunnel barrier layer. When the magnetization directions of the two ferromagnetic layers are parallel, MTJ exhibits a low resistance, representing “0”; when the magnetization directions are anti-parallel, MTJ exhibits a high resistance, representing “1”. When reading an STT-MRAM cell, a small voltage is applied between the bit line and the sense line. When writing an STT-MRAM cell, a large current and a long time are required to change the magnetization direction of the free layer. Therefore, STT-MRAM’s read performance is comparable to DRAM, but its write performance is poor.

3.2 Internal operations of STT-MRAM device
The organization of STT-MRAM device is similar to that of DRAM, as shown in Fig. 2. The difference is that STT-MRAM stores data through the MTJ, while DRAM stores data through the capacitor. Generally, an ACTIVATE command is first sent to the STT-MRAM device. This command selects a bank and row address. The data of the selected row is then transferred from the array into the row buffer. In the active state, the STT-MRAM device can perform reads and writes. A READ command decodes a specific column address. The data from the column is driven to the DQ pins. The process for the write is similar to the read except the data propagates in the opposite direction. If the row to be accessed is already in the row buffer, the row buffer hits. Whereas, if another row is in the row buffer, a PRECHARGE command needs to be sent. PRECHARGE command restores the row buffer data to the array [23].

According to the internal operations of STT-MRAM, the ACTIVATE command reads data from the array into the row buffer. The READ command reads data from the row buffer to the DQ pins. The WRITE command writes data from the DQ pins to the row buffer. The PRECHARGE command writes data from the row buffer to the array; that is, the PRECHARGE command actually triggers the write operations to the STT-MRAM cell.

4. Fast cacheline-based data replacement
According to the analysis in Chapter 3, the PRECHARGE command triggers the write operations to the STT-MRAM cell. The algorithm counts the PRECHARGE number of STT-MRAM pages. When the number reaches the threshold, the data in that page will be replaced to DRAM. In order to solve the large time overhead of the data replacement [24, 25, 26, 27, 28, 29], we propose a fast cacheline-based data replacement which can replace data during normal memory access.

4.1 Replacement granularity
To improve the replacement speed, the algorithm replaces one block data (cacheline, 32B) instead of one page data. In hybrid DRAM and PCM main memory [24, 25, 26, 27, 28, 29], one page data is always replaced. This is because the Memory Management Unit (MMU) is in unit of page. However, MMU is not necessary in embedded system, so there is no need to replace the entire page. In addition, replacing one block also solves the problem of different page sizes of DRAM and STT-MRAM. In this paper, the page size of STT-MRAM is 64B, while that of DRAM is 512B.

4.2 Replacement data
When the STT-MRAM read request causes data replacement, the read data from STT-MRAM will be directly replaced. When the STT-MRAM write request causes data replacement, the write data from the bus will be directly replaced. In DRAM, the data for replacement is initialized to the last block. Once the data replacement occurs, the block address is decremented by one. Compared to other complex selections [24, 25, 26, 27, 28, 29], this selection is more suitable for IoT devices due to the lower design complexity.

4.3 Replacement table
A PRECHARGE number table of STT-MRAM pages is needed. The table width is related to the replacement threshold. In this paper, the width is set to 8-bit. The table depth is the number of STT-MRAM pages, 512K. In addition, an address conversion table is needed. It is unrealistic to maintain a table row for each block of STT-MRAM, because the hardware overhead is huge. We only record the addresses of the two replaced blocks in the table. The table width includes three parts: 1-bit valid, 26-bit access block address, and 26-bit actual block address. The table depth is related to the data replacement numbers. In this paper, the depth is set to 1024. Once the table overflows, the data replacement will be disabled to ensure the normal operation of hybrid main memory. This paper mostly focuses on improving the data replacement speed. How to reduce the hardware overhead
caused by the data replacement is another challenge and part of our ongoing work.

4.4 Replacement parallelism
To improve the data replacement speed, the algorithm makes full use of time parallelism. When the STT-MRAM read request causes data replacement, the STT-MRAM performs the PRECHARGE command, the ACTIVATE command, and the READ command in order. Since STT-MRAM’s precharge latency and activate latency are much higher than DRAM, there is enough time for DRAM to read data. When the STT-MRAM reads data out and returns it to the cache, the read data is written back to DRAM, and the data from DRAM is written back to STT-MRAM. This way, the data has been replaced during the completion of the STT-MRAM read request. Fig. 3(a) shows the read process of STT-MRAM without data replacement:
1) STT-MRAM receives a read request and starts to read.
2) STT-MRAM reads data out and stores it in rbuf.
3) memory controller returns the rbuf data to the cache.
Fig. 3(b) shows the read process of STT-MRAM with data replacement:
1) STT-MRAM receives a read request and starts to read.
2) DRAM starts to read the data for replacing.
3) DRAM first reads data out and stores it in exbuf.
4) STT-MRAM later reads data out and stores it in rbuf.
5) memory controller returns the rbuf data to the cache, writes the rbuf data to the DRAM and writes the exbuf data to the STT-MRAM.

When the STT-MRAM write request causes data replacement, the STT-MRAM first receives the write data on the bus, and then performs the PRECHARGE command, the ACTIVATE command, and the WRITE command in order. STT-MRAM’s long precharge latency and activate latency allow DRAM sufficient time to read data out, and then write data back in advance. When the STT-MRAM performs the WRITE command, the write data is already from DRAM. Thus, the data has been replaced during the completion of the STT-MRAM write request. Fig. 4(a) shows the write process of STT-MRAM without data replacement:
1) STT-MRAM receives a write request.
2) memory controller receives the write data into wbuf.
3) DRAM starts to read the data for replacing.
4) memory controller writes the wbuf data back to DRAM.
5) memory controller writes the exbuf data to STT-MRAM.

5. Experimental setup
Due to the lack of mature NVM chips on the market, most of the research based on the software simulation. It is difficult to know the source and reliability of the simulation model parameters used in the new NVM simulator [21]. In order to obtain more reliable results, we build a hardware simulation platform based on the DRAM and STT-MRAM Verilog models. 256Mb x8 DDR3 stt-mram model of EVERSIPIN is selected. 1Gb x8 DDR3 sdram model of Micron is selected. To meet the experiment needs, we modified the capacity of the 1Gb sdram to 512Mb, 256Mb, and 128Mb. The tREFI of the 1Gb sdram is 7.8us, and the tRFC is 110ns. As DRAM capacity decreases, we keep the number of rows refreshed at a time, and increase the refresh interval. The hardware simulation platform configuration is shown in Table I. We chose embedded applications with different memory access characteristics.

We compare four different main memory structures: 256MB DRAM, 128MB DRAM + 128MB STT-MRAM, 64MB DRAM + 192MB STT-MRAM, and 256MB STT-MRAM. In the hybrid main memory, three replacement thresholds of 250, 10, and 2 are tested. The depth of the address conversion table is set to 1024. In addition to the total time, we also count the memory access time of the program to study the effect of STT-MRAM on the overall system performance and on the main memory performance. According to Micron [23], we divide the total energy into five parts: background energy, refresh energy, activate-precharge energy, read energy and write energy. At last, we study the effect of the address conversion table overflows. We reduce the table depth from 1024 to 512 and 256, and chose the program with more data replacements. The time and current parameters used in the experiment are from the EVERSIPIN and Micron data sheets [12, 30], as shown in Table II.
Table I  Hardware simulation platform system configuration

|                         | Processor 667MHz, riscv core, sequential execution |
|-------------------------|---------------------------------------------------|
| L1 I/D caches           | Private, 32KB/32KB, 2-way, 32B cache line, write-back, 2-way, LRU |
| Memory controller       | DRAM controller, STT-MRAM controller              |
| Hybrid main memory controller | DRAM + STT-MRAM: 256MB+256MB                      |
| Main memory             | DRAM: 256MB, STT-MRAM: 256MB                      |
|                         | DRAM+STT-MRAM: 128MB+128MB                        |
|                         | DRAM+STT-MRAM: 64MB+192MB                         |

Table II  Time and current parameters

| Parameters | DDR3 SDRAM | DDR3 STT-MRAM |
|------------|------------|---------------|
| tCK        | 1.5 ns     | 1.5 ns        |
| tRCD       | 15 ns      | 95 ns         |
| tRP        | 15 ns      | 66 ns         |
| tWL        | 10 cycle   | 10 cycle      |
| tWR        | 15 cycle   | 15 cycle      |
| tRFC       | 7.8 us     | N/A           |
| IDD0       | 65 mA      | 220 mA        |
| IDD2N      | 40 mA      | 90 mA         |
| IDD2P0     | 12 mA      | 55 mA         |
| IDD2P1     | 30 mA      | 60 mA         |
| IDD3N      | 40 mA      | 90 mA         |
| IDD3P      | 30 mA      | 60 mA         |
| IDD4R      | 125 mA     | 135 mA        |
| IDD4W      | 125 mA     | 165 mA        |
| IDD5       | 165 mA     | N/A           |

6. Experimental results

Figure 5 shows the performance of hybrid main memory and STT-MRAM, normalized to DRAM. It can be seen that the performance of STT-MRAM is much lower than DRAM. This is because the precharge latency and activate latency of STT-MRAM are much higher than DRAM. Hybrid main memory with two capacity ratios of 1:1 and 1:3 have the same characteristics. When the program has less access to STT-MRAM and the accesses are local (pooling, stencil_vect, fft, stringsearch), STT-MRAM’s high latency has little impact. At this time, the effect of the algorithm is not obvious. Even when the threshold is very low, the number of data replacements is few or no. Therefore, the performance does not change with thresholds. Sometimes the performance of hybrid main memory is better than DRAM (fft, stringsearch). This is because the refresh number is reduced in hybrid main memory. Refresh will block requests and increase activations of DRAM (all banks need to be precharged before refresh). In addition, hybrid main memory has better parallelism. If DRAM is being accessed and the next access is STT-MRAM or vice versa, the request can be accepted. When the program has more access to STT-MRAM and its locality is poor (cnn_layer, filter, matrixmul32), STT-MRAM’s high latency has a serious impact. At this point, the algorithm plays a significant role. The refresh number of hybrid main memory is reduced by half, so the refresh energy is halved. When the program accesses STT-MRAM more and the locality is poor (cnn_layer, filter, matrixmul32, convolution), the high activate-precharge power of STT-MRAM has a serious impact. At this point, the algorithm plays a significant role. When the threshold is not effectively reduced, the activations and precharges of STT-MRAM, and blocks some access to DRAM.

Figure 6 shows the performance of overall system with different main memory structures. It can be seen that, the performance is comparable in most cases. This is because in addition to accessing main memory, the processor also accesses peripherals and executes other instructions. All these times make up the total time. We know that peripherals work much less frequently than main memory. In this paper, main memory operates at 667MHz, while peripherals operate at 50MHz. Accessing peripherals greatly increases the total time. The longer the total time, the less obvious the effect of STT-MRAM high latency on the overall system performance. As [18] shows, the overall system performance is not particularly sensitive to the main memory latency.

Figure 7 shows the energy of hybrid main memory with a capacity ratio of 1:1. It can be seen that when the program has less access to STT-MRAM and the locality is good (pooling, stencil_vect, fft, stringsearch), the high activate-precharge power of STT-MRAM has little impact. At this time, the algorithm effect is not obvious. Even when the threshold is low, the number of data replacements is few or no. Therefore, the energy does not change with thresholds. The refresh number of hybrid main memory is reduced by half, so the refresh energy is halved. When the program accesses STT-MRAM more and the locality is poor (cnn_layer, filter, matrixmul32, convolution), the high activate-precharge power of STT-MRAM has a serious impact. At this point, the algorithm plays a significant role. When the threshold is
high, the data replacement is few. The hybrid main memory energy is higher than DRAM, even if the refresh energy is halved. As the threshold is lowered, data replacements increase. More operations are transferred from STT-MRAM to DRAM, so the energy is reduced. We can see that for convolution, the energy decreases slowly. This is because although the data is replaced, the effect is not obvious, which is closely related to the program's memory access characteristics.

Figure 8 shows the energy of hybrid main memory with a capacity ratio of 1:3. It can be seen that 1:1 and 1:3 have the same characteristics. As DRAM usage decreases, the energy further decreases. In 1:1 structure, the energy is reduced by an average of 18% in the best case (threshold = 2). In 1:3 structure, the energy is reduced by an average of 32% in the best case (threshold = 2). Figure 9 shows the energy of STT-MRAM. STT-MRAM's refresh energy is zero, but its activate-precharge power is much higher than DRAM. When the refresh energy is dominant, the energy of STT-MRAM is lower than DRAM (fft, stringsearch). Otherwise, STT-MRAM consumes more energy than DRAM. Therefore, STT-MRAM main memory has no advantages in terms of energy and performance.

Figure 10 shows the effect of the address conversion table overflow. In 1:1 hybrid main memory, when the threshold is 2, the number of data replacements of cnn_layer is 270. When the table depth is 1024, the table does not overflow. When the depth is reduced to 512, the table overflows a bit. Compared to non-overflow, the energy increases 12%.

7. Conclusion

In this paper, the hybrid DRAM and STT-MRAM main memory with the fast cacheline-based data replacement can reduce main memory energy by 32% with comparable performance to DRAM. However, STT-MRAM main memory cannot always reduce the energy, and the performance of it is much lower than DRAM. Reducing the usage of DRAM, the main memory energy may be further reduced.

Acknowledgments

This work was supported by National Key R&D Program of China (2019YFB2102400), National Science and Technology Major Project (2018ZX03001006-002).

References

[1] K.K. Sethi, et al.: “Internet of things its components, applications, issues and challenges from future perspective and scope: a review,”
A. Botti, et al.: “Integration of cloud computing and internet of things: a survey,” Future Generation Computer Systems 56 (2016) 584 (DOI: 10.1016/j.future.2015.09.021).

H. Arasteh, et al.: “IoT-based smart cities: a survey,” IEEE 16th International Conference on Environment and Electrical Engineering (2016) 1 (DOI: 10.1109/eeicee.2016.7555867).

D. Chen, et al.: “Platform choices and design demands for IoT platforms: cost, power, and performance tradeoffs,” IET Cyber-Physical Systems: Theory & Applications 1 (2016) 70 (DOI: 10.1049/iet-cps.2016.0020).

W. Shi, et al.: “Edge computing: vision and challenges,” IEEE Internet Things J. 3 (2016) 637 (DOI: 10.1109/jiit.2016.2579198).

A. Carroll and G. Heiser: “An analysis of power consumption in a smartphone,”USENIX Annual Technical Conference 26 (2010) 271.

A. Carroll and G. Heiser: “The systems hacker’s guide to the galaxy energy usage in a modern smartphone,” Asia-Pacific Workshop on Systems 45 (2013) (DOI: 10.1145/2500727.2500734).

M. Hosomi, et al.: “A novel nonvolatile memory with spin torque transfer magnetization switching: spin-RAM,” IEEE International Electron Devices Meeting (2005) 459 (DOI: 10.1109/iedm.2005.1609379).

K. Tsuchida, et al.: “A 64Mb MRAM with clamped-reference and adequate-reference schemes,” IEEE International Solid-State Circuits Conference 53 (2010) 258 (DOI: 10.1109/isscc.2010.5433948).

N.D. Rizzo, et al.: “A fully functional 64 Mb DDR3 ST-MRAM built on 90 nm CMOS technology,” IEEE Trans. Magn. 49 (2013) 4441 (DOI: 10.1109/tmag.2013.2243133).

S.W. Chung, et al.: “4Gb density STT-MRAM using perpendicular MTJ realized with compact cell structure,” IEEE International Electron Devices Meeting 27 (2016) 1 (DOI: 10.1109/iedm.2016.7838490).

EMD3D256M - 256Mb spin-transfer torque MRAM (2018) https://www.everspin.com.

EMD4E001G - 1Gb spin-transfer torque MRAM (2019) https://www.everspin.com.

Y. Jin, et al.: “Area, power, and latency considerations of STT-MRAM to substitute for main memory.” Proc. ISCA (2014).

M. Shihab, et al.: “Couture: tailoring STT-MRAM for persistent main memory.” 4th Workshop on Interactions of NVM/Flash with Operating Systems and Workloads (2016).

J. Meza, et al.: “Evaluating row buffer locality in future non-volatile main memories,” arXiv preprint (2018) arXiv:1812.06377.

E. Kulursay, et al.: “Evaluating STT-RAM as an energy-efficient main memory alternative,” ISPASS (2013) 256 (DOI: 10.1109/isspass.2013.6557176).

K. Asifuzzaman, et al.: “Performance impact of a slower main memory: a case study of STT-MRAM in HPC,” Proceedings of the second International Symposium on Memory Systems (2016) 40 (DOI: 10.1145/2989081.2989082).

K. Asifuzzaman, et al.: “STT-MRAM for real-time embedded systems: performance and WCET implications,” Proceedings of the International Symposium on Memory Systems (2019) 195 (DOI: 10.1109/3357526.3357531).

J. Wang, et al.: “Enabling high performance LPDDRx-compatible MRAM,” Proceedings of the 2014 International Symposium on Low Power Electronics and Design (2014) 339 (DOI: 10.1109/2627369.2627610).

K. Asifuzzaman, et al.: “Enabling a reliable STT-MRAM main memory simulation,” International Symposium on Memory Systems (2017) 283 (DOI: 10.1145/3132402.3132416).

M.P. Komalan, et al.: “Main memory organization trade-offs with DRAM and STT-MRAM options based on gem5-NVMain simulation frameworks,” 2018 Design, Automation & Test in Europe Conference & Exhibition (2018) (DOI: 10.23919/date.2018.8341987).

Calculating memory system power for DDR3: TN-41-01 (2007) https://www.micron.com.

L.E. Ramos, et al.: “Page placement in hybrid memory systems,” Proceedings of the International Conference on Supercomputing (2011) 85 (DOI: 10.1145/1995896.1995911).

S. Lee, et al.: “CLOCK-DWF: a write history-aware page replacement algorithm for hybrid PCM and DRAM memory architectures,” IEEE Trans. Comput. 63 (2014) 2187 (DOI: 10.1109/tc.2013.98).

M. Lee, et al.: “M-CLOCK: migration-optimized page replacement algorithm for hybrid DRAM and PCM memory architecture,” Proceedings of the 30th Annual ACM Symposium on Applied Computing (2015) 2001 (DOI: 10.1145/2695664.2695675).

S. Kim, et al.: “Adaptive-classification CLOCK: page replacement policy based on read/write access pattern for hybrid DRAM and PCM main memory,” Microsys. 57 (2018) 65 (DOI: 10.1016/j.micropro.2018.01.003).

K. Chen, et al.: “A novel page replacement algorithm for the hybrid memory architecture involving PCM and DRAM,” IFIP International Conference on Network and Parallel Computing (2014) 108 (DOI: 10.1007/978-3-662-44917-2_10).

Z. Wu, et al.: “APP-LRU: a new page replacement method for PCM/DRAM-based hybrid memory systems,” IFIP International Conference on Network and Parallel Computing (2014) 84 (DOI: 10.1007/978-3-662-44917-2_8).

1Gb: x4, x8, x16 DDR3 SDRAM (2018) https://www.micron.com.