IMPROVING PHASE CHANGE MEMORY (PCM) AND SPIN-TORQUE-TRANSFER MAGNETIC-RAM (STT-MRAM) AS NEXT-GENERATION MEMORIES: A CIRCUIT PERSPECTIVE

by

Bo Zhao

B.S., Electrical Engineering, Beihang University, 2007

M.S., Electrical Engineering, University of Pittsburgh, 2009

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy in Computer Engineering

University of Pittsburgh

2013

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Bo Zhao

It was defended on

November 8th 2013

and approved by

Jun Yang, Ph.D., Associate Professor

Department of Electrical and Computer Engineering

Steven P. Levitan, Ph.D., John A. Jurenko Professor

Department of Electrical and Computer Engineering

Hai (Helen) Li, Ph.D., Assistant Professor

Department of Electrical and Computer Engineering

Youtao Zhang, Ph.D., Associate Professor

Department of Computer Science

Paul W. Leu, Ph.D., Assistant Professor

Department of Industrial Engineering

Dissertation Director: Jun Yang, Ph.D., Associate Professor

Department of Electrical and Computer Engineering

IMPROVING PHASE CHANGE MEMORY (PCM) AND SPIN-TORQUE-TRANSFER MAGNETIC-RAM (STT-MRAM) AS NEXT-GENERATION MEMORIES: A CIRCUIT PERSPECTIVE

Bo Zhao, PhD

University of Pittsburgh, 2013

In the memory hierarchy of computer systems, the traditional semiconductor memories Static RAM (SRAM) and Dynamic RAM (DRAM) have already served for several decades as cache and main memory. With technology scaling, they face increasingly intractable challenges like power, density, reliability and scalability. As a result, they become less appealing in the multi/many-core era with ever increasing size and memory-intensity of working sets.

Recently, there is an increasing interest in using emerging non-volatile memory technologies in replacement of SRAM and DRAM, due to their advantages like non-volatility, high device density, near-zero cell leakage and resilience to soft errors. Among several new memory technologies, Phase Change Memory (PCM) and Spin-Torque-Transfer Magnetic-RAM (STT-MRAM) are most promising candidates in building main memory and cache, respectively. However, both of them possess unique limitations that preventing them from being effectively adopted.

In this dissertation, I present my circuit design work on tackling the limitations of PCM and STT-MRAM. At bit level, both PCM and STT-MRAM suffer from excessive write energy, and PCM has very limited write endurance. For PCM, I implement Differential Write to remove large number of unnecessary bit-writes that do not alter the stored data. It is then extended to STT-MRAM as Early Write Termination, with specific optimizations to eliminate the overhead of pre-write read. At array level, PCM enjoys high density but could not provide competitive throughput due to its long write latency and limited number of read/write circuits. I propose a Pseudo-Multi-Port Bank design to exploit intra-bank parallelism by recycling and reusing shared peripheral circuits between accesses in a time-multiplexed manner. On the other hand, although STT-MRAM features satisfactory throughput, its conventional array architecture is constrained on density and scalability by the pitch of the per-column bitline pair. I propose a Common-Source-Line Array architecture which uses a shared source-line along the row, essentially leaving only one bitline per column.

For these techniques, I provide circuit level analyses as well as architecture/system level and/or process/device level discussions. In addition, relevant background and work are thoroughly surveyed and potential future research topics are discussed, offering insights and prospects of these next-generation memories.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

Completing a Ph.D. degree is an individual achievement, yet it is obviously not possible without the personal and practical supports of numerous people. One of the joys of completion is to look over the journey past and remember all these individuals who have helped and encouraged me in one way or the other during this rewarding and amazing journey.

First and foremost, it is difficult to overstate my gratitude to my advisor, Professor Jun Yang, and my "secondary" advisor, Professor Youtao Zhang. I could not have asked for better role model of researchers, each enthusiastic, diligent, and insightful. The joy and perseverance they have for academic pursuits were contagious and motivational for me. They have taught me, both consciously and unconsciously, how conscientious research is conducted. In addition, Professor Jun Yang and Professor Youtao Zhang are inspirational, supportive, patient, and the true embodiment of a mentor. I am deeply grateful to them for giving me the confidence to explore my research interests and the guidance to avoid getting lost in my exploration. Beyond knowledge, their special attention to details trained me to schedule, to organize, to illustrate, to present, all are invaluable skills for whatever I am and will be working on. It has always been my honor and pleasure working with Professor Jun Yang and Professor Youtao Zhang.

In combination with the mentorship of my advisor, I am blessed to work with my active committee members and taught by many intelligent faculties. Professor Steven Levitan is not only a fabulous guide but a dear friend: sharp, cheery, perceptive, and thankfully, readily available for me. He was very generous with his time and knowledge, and we had numerous useful discussions and brainstormings throughout my graduate study. He also led me into the door of integrated circuit design, and I have never expected my three-year Teaching Assistant service for his introductory and advanced VLSI courses to be so fun and rewarding. My sincere gratitude is reserved for Professor Hai (Helen) Li and Professor Yiran Chen for their constructive insights and advices, which were substantially helpful to my research and study. I would like to thank Professor Paul Leu from Industrial Engineering for his time, interest, helpful comments, and a welcome diversity of perspectives. I also appreciate Professor Ken Mai, Professor Larry Pileggi, and Professor Rick Carley at ECE department of Carnegie Mellon University, for their informative course works that provided me with thorough and in-depth knowledge.

Also, I have been very fortunate with my brilliant research group comrades: Ping Zhou, Yu Du, Lei Jiang, Yi Xu, Lin Li. They have contributed immensely to my personal and professional time at Pittsburgh. The group has been a constant source of friendships as well as inspiration, feedback, suggestion, assistance, and collaboration. I am indebted to them for helping me whole-heartedly on many of my research projects, and for providing a stimulating and fun environment in which to learn and grow. Moreover, heartfelt thanks goes to my roommates and all my friends and classmates in University of Pittsburgh and Carnegie Mellon University, for their friendship and company, for each and every day and night and meal and game we spent, enjoyed, and suffered together.

Despite the geographical distance, my family was always nearby. Words cannot express the feelings I have for my mother Zheng Liu, my father Zhongxiang Zhao, and all my grandparents who raised me with unconditional love and supported me in all my pursuits. I would not be here if not for you. My dear daughter, Ziyue Zhao, who joined my life sixteen months ago, thank you for re-energizing me with laughters every evening after a long day. I wish to thank my entire extended family as my constant harbor, spiritually, emotionally, and physically, during the hard and easy times. And also to my in-laws for their faithful love, help and encouragement over the past ten years. Lastly, to my beloved, Di Wang, for always being there. Wherever I am, all of you are always in my heart.

To My Family

1.0 INTRODUCTION

The two most essential functions of computer systems are computation and storage. While computations are mostly centralized in the Central Processing Unit (CPU), storages take place ubiquitously, from temporary buffering of intermediate computation results, to permanent massive file backups that can be carried in our pockets.

In computer systems, the vast existence of storage is organized in a hierarchical manner known as the *memory hierarchy* illustrated in Figure 1. A memory hierarchy can be seen as a hardware optimization that takes the benefits of spatial and temporal localities to speed up data accesses. It distinguishes each level in the hierarchy by speed and density. The higher level memories (closer to CPU) are faster to serve the high frequence CPU, but smaller to hold only the most active data. The lower level memories (further from CPU) tend to be slower, but larger to store the large working set. Thus, a program will achieve greater performance if it uses data while it is cached in the higher levels of the memory hierarchy and avoids bringing other data into the higher levels of the hierarchy that will displace data that will be used shortly in the future.

The unique speed and density requirements in turn determine the choices of memory technologies in each level. In classic implementations that still dominate nowadays, Static RAM (SRAM), Dynamic RAM (DRAM), and mechanical Hard Disk Driver (HDD) are employed in the hierarchy from top to bottom, as shown in Figure 1. SRAM is the fastest memory that can match the GHz operation of CPU pipeline. It can also take advantage of its multi- to many-port capability to achieve extremely high bandwidth which is crucial to Register Files (RF) and L1 caches. Moreover, it is built out of standard logic transistors that can be seamlessly integrated with CPU components on-chip. Therefore, SRAM is the natural choice of all kinds of on-chip storage, as well as the large off-chip L4 cache due to its faster speed than DRAM. Although DRAM is also a solid-state memory, it is generally not process-compatible with CPU due to its specially optimized fabrication processes for the dedicated capacitor and high density. DRAM is used as large main memory with adequate speed and much lower power than SRAM. It features much higher density and thus capacity to hold the entire working sets of multiple programs. HDD provides even higher density and capacity than DRAM with non-volatile storage for the whole file system. However, data accesses are carried out by mechanical mechanisms of rotating the disks and moving the magnetic heads for address searching, resulting in very high latency.

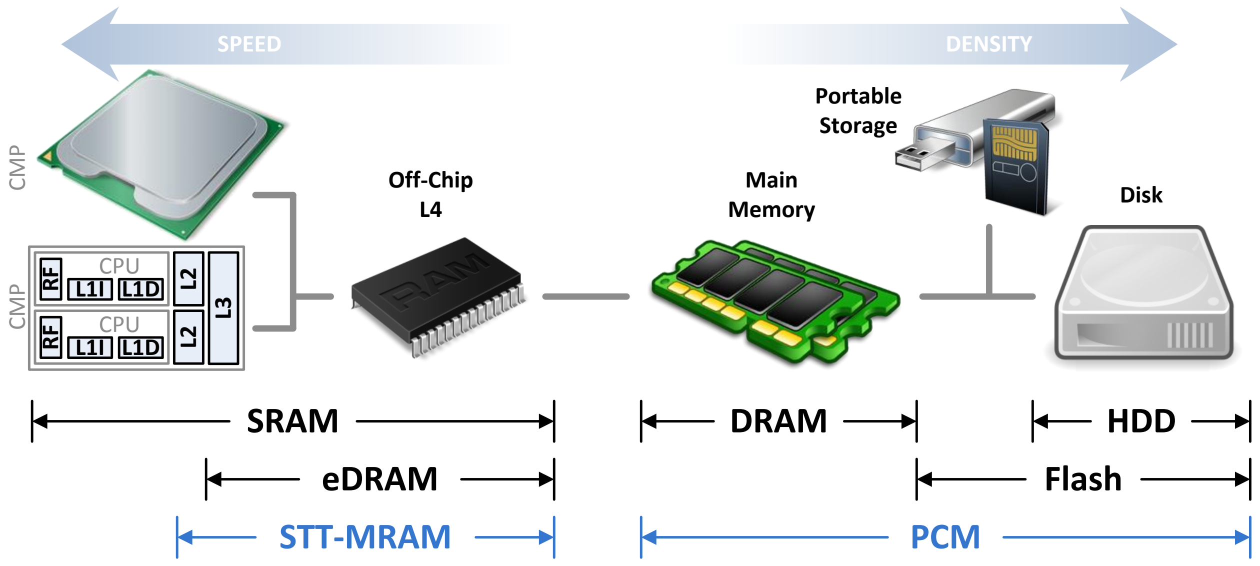


Figure 1: Memory hierarchy and application range of memory technologies.

As more and more processor cores are integrated in a Chip Multi-Processor (CMP) to exploit computational parallelism, the capacity demand for on-chip storage, especially lower level caches, increases rapidly. As a result, large caches usually occupy >50% die area in modern processors, giving rise to the leakage concern of SRAM in deep sub-micron technologies. SRAM cell is particularly leaky due to its multi-transistor nature. This is less a problem in small and very active higher-level caches, but becomes prohibitive given the large size and low activity of L3 caches. Besides, the low density of SRAM, also a result of its multi-transistor cell, leads to too large cache area and thus large die size that ultimately increases cost and decreases yield. In the other end of the hierarchy, although the density of HDD kept growing constantly, its speed was not improved fast enough that lagged behind all upper-level memories further and further. Also, its mechanical movements based operations are not suitable for the fast-growing portable applications.

Consequently, in the past decade, we have seen the rise of eDRAM and Flash. eDRAM is essentially DRAM "embedded" with processors. It is fabricated with logic-compatible processes, yielding higher speed than commodity DRAM, at the expense of shorter retention time and lower density. Nevertheless, eDRAM still offers $\sim 3\times$ density over SRAM, with much lower leakage. Therefore, successful adoptions of eDRAM as up to 80MB on-chip L3 cache [59] and 192MB off-chip L4 cache [65] have been demonstrated. Because its latency is still relatively large compared to SRAM, eDRAM is not yet utilized in the more delay-sensitive L2 caches. In the other end of the hierarchy, Flash memory based Solid State Disks (SDD) easily outperform HDD by orders of magnitudes, with lower power and heat dissipation. While in the middle, commodity DRAM kept scaling and doubling its capacity. Up to this point, we have got a pure solid-state memory hierarchy.

However, as the integrated circuit technology heading into the deep sub-micron territory and approaching the $\sim 10nm$ era, this entire pool of memories face crisis again. The refresh problem of DRAM/eDRAM worsens rapidly with technology advancement and capacity increase, as well as temperature and process variations. This is because the cell transistor becomes leakier and more vulnerable to variations with shrinked footprint, and there are more rows of cells to be refreshed within the same time interval in a larger memory with larger number of rows. Such refresh problem will place more negative effects on system performance in future generations of DRAM/eDRAM. To maintain reasonable retention time and sensing margin, adequate cell-to-bitline capacitance ratio must be maintained in DRAM/eDRAM, this in turn limits the size shrinking of cell capacitor and ultimately hurts the DRAM scaling in general. Furthermore, DRAM main memories consume >40% power of modern computer systems, and such percentage will keep growing with faster and larger DRAMs. On the other hand, Flash also faces scaling problem because of its limited footprint of the floating gate and thus limited number of stored electrons that is very vulnerable to process variation. As a result, although people are pushing Flash's multi-level capability to an extreme, very sophisticated read and write schemes must be developed to make it viable. Flash also has very limited lifetime of only $\sim 10^5$ writes. So SSDs are usually equipped with up to 50% user-invisible redundancy (known as *over-provisioning*) to maintain the nominal capacity in its lifecycle (e.g. 10 years). Its block-level erase and program is not only slow but also too inefficient from an energy point of view. Moreover, generating the very high voltage ($\sim$20V) for erase/program out of the decreasing supply voltage is becoming harder. Finally, both DRAM/eDRAM and Flash suffer from worsening reliability. They become less immune to soft error mechanisms like alpha particle strike due to the decreasing number of electrons held in their storage nodes.

Recently, there is an increasing interest in introducing new non-volatile memory technologies such as Phase Change Memory (PCM), Spin-Torque-Transfer Magnetic-RAM (STT-MRAM), Memristor, Ferroelectric RAM (FeRAM), Conductive-Bridging RAM (CB-RAM), etc., to the memory hierarchy in replacement of existing memories. Among these emerging memory technologies, Phase Change Memory (PCM) and Spin-Torque-Transfer Magnetic-RAM (STT-MRAM) are two of the most promising candidates as the next-generation memories. Table 1 compares PCM and STT-MRAM with existing solid-state memory technologies SRAM, eDRAM, DRAM and Flash. PCM and STT-MRAM share many common characteristics like resistance-based storage, non-volatility, high density, good scalability, low leakage, immunity to soft errors, etc. Nevertheless, they possess unique features that distinguish themselves in the memory hierarchy.

PCM features similar cell density to DRAM and Flash and multi-level storage capability, which naturally place it at the lower levels of the memory hierarchy, as shown in Figure 1. Due to its clear advantages like byte-addressability, lower latency, better endurance, better scalability and better reliability, PCM is an inherent replacement of Flash and HDD on disk storages and portable storages. When compared to DRAM, PCM still exhibits many benefits on density, leakage, non-volatility, scalability and reliability. However, disadvantages also exist as long latencies, high write energy, low throughput and limited endurance. Therefore, special efforts must be paid to mitigating these drawbacks for successful application of PCM as DRAM replacement in main memory.

STT-MRAM usually have similar or better density than eDRAM with advantages like lower leakage, non-volatility, better scalability and better reliability. Although its 10^{15} write endurance is worse than eDRAM, this is generally good enough to be considered unlimited.

Table 1: Comparison of solid-state memory technologies. [a]

	SRAM	eDRAM	DRAM	Flash	PCM	STT-MRAM
State-of-the-art cell size (F^2)	135	35	6	4	4	14
Storage type	latch	charge	charge	trapped charge	resistance	resistance
Cell read speed	$100ps$	$1ns$	$10ns$	$10\mu s$	$30ns$	$1ns$
Cell write speed	$100ps$	$1ns$	$10ns$	$100\mu s$	$150ns$	$3\sim10ns$
Read energy	Low	Low	Medium	Medium	Low	Low
Write energy	Low	Low	Medium	High	High	Medium
Leakage	High	Medium	Medium	Medium	Low	Low
Throughput	Very High	High	Medium	Very Low	Low	High
Retention time (volatility)	∞ with power	$50\mu s \sim 1ms$	$>64ms$	non-volatile	non-volatile	compromisable non-volatile
Write endurance	10^{16}	10^{16}	10^{16}	10^{5}	$10^{8} \sim 10^{9}$	10^{15}
Multi-level storage	No	No	No	4~6b/cell	2b/cell	Possible
Multi-port access	Many-port	2-port	No	No	No	No
Scalability	Good	Poor	Poor	Poor	Good	Good
Byte addressability	Yes	Yes	Yes	No	Yes	Yes
Sofe error	Low	High	High	High	No	No
Logic-process compatibility	Yes	Yes	No	No	Possible	Possible

[a]Based on state-of-the-art and most-aggressive data from published prototypes.

Hence, STT-MRAM covers the entire application range of eDRAM, as shown in Figure 1. Furthermore, STT-MRAM can go higher than eDRAM in the hierarchy to L2 cache thanks to its short read latency and shorten-able write latency with non-volatility compromise technique of volatile write. Notice that although STT-MRAM cell read latency is much larger than SRAM, the read access latency at memory module level can be comparable or smaller with same capacity, resulting from the much higher density and thus reduced footprint and interconnect delay. Above L2 cache, SRAM is irreplaceable due to its unparalleled speed and throughput, and the multi/many-port capability as the necessity of Register Files and L1 caches. However, the much large write energy of STT-MRAM, even with volatile write, is a major hurdle of using it in higher level caches which have relatively high write activity [5]. Furthermore, although the density and scalability of STT-MRAM device/cell are excellent, such advantages are suppressed by the domination of wiring in its cell array. Therefore, improvements are also necessary for STT-MRAM as a replacement of eDRAM and SRAM in large caches.

1.1 RESEARCH OVERVIEW

As discussed above, although considered promising, both PCM and STT-MRAM face unique limitations that prevent them from being effectively adopted as main memory and large cache, respectively. Here I further identify these challenges at bit and array levels.

Bit Level Challenge : PCM. PCM write is a thermal-driven process that involves heating (melting) and cooling (crystallizing) the phase change material to change its crystal structure and thus resistance (details in Chapter 2.3). These procedures incur high current injection in the range of $0.1{\sim}1mA$ [9, 29]. Such high currents are supplied from high voltage sources of $2{\sim}5V$ [7, 29]. Moreover, the currents keep flowing through the phase change material for $50{\sim}400ns$ [7, 29] to fully finish melting or crystallizing. Therefore, the per-bit write energy of PCM is quite high. For instance, assuming the conservative write current, voltage, and pulse width to be $100\mu A$ [9], $2V$ [7], and $50ns$ [29] respectively, this leads to $100\mu A \times 2V \times 50ns = 10pJ$ per bit, which is much higher than DRAM's $\sim 1.5pJ$ per bit [63]. What makes things worse is the fact that the high write voltages are generated by charge pump circuits from regular power supplies with *limited power efficiency*. So the actual energy/power consumption is even higher at chip level. On the other hand, due to the repeated heat stress in melting and crystallizing processes, a PCM cell can be written for a limited number of times, typically $10^8 \sim 10^9$ [81]. While this is better than the 10^5 write endurance of Flash, it is much worse than that of a DRAM cell (10^{16}) and is a big concern when PCM is used in main memory.

Bit Level Challenge : STT-MRAM. STT-MRAM write uses spin-polarized current flowing through the magnetic device to disturb its magnetic torque in one stable direction, turn the torque, and let it settle in the other stable direction with different resistance (details in Chapter 2.4). Similar to PCM, such procedure also requires high current in the range of $50{\sim}500\mu A$ [28, 19], as while as long period of $10{\sim}100ns$ [19, 16, 36]. Therefore, STT-MRAM also suffers from high write energy of conservatively $50\mu A \times 1.2V \times 10ns = 0.6pJ$ per bit, much higher than that of SRAM and eDRAM. Several recent studies proposed to relax the non-volatility requirement from the typical ten year storage-class retention time, to reduce the write pulse width and thus write energy [52]. However, even with such volatile writes, the write energy still dominates total energy of an STT-MRAM based cache, offsetting the energy savings from low leakage [5].

Array Level Challenge : PCM. Besides write energy and endurance, one major challenge for PCM to replace DRAM is its low throughput. For example, a state-of-the-art PCM chip can achieve 40MB/s program throughput [9], while the throughput of even an old DDR2-800 DRAM is 100MB/s per chip. PCM's throughput is constrained by three factors. First, given the large write power per bit, large number of concurrent bit-writes can raise concerns of voltage droop and power supply noise [14]. Hence the number of cells written in parallel has to be restricted [26], which is already constrained by the chip power budget. However, as will be shown in later chapters, our bit level solution can effectively remove large portion of bit-writes and thus provide power headroom for throughput improvement. Also, good scalability of phase change materials implies that aggressive write current scaling can be expected. Therefore, the write power/energy factor is less a concern here [9]. Second, PCM's write operation is not only slow but much slower than its read operation, which is determined by the device characteristics. A typical set (crystallizing) procedure takes at least $120ns$ [7], and the write latency of multi-level PCM is much worse due to the multi-iteration program-and-verify procedure [4]. Moreover, because of the limited number of concurrent bit-writes, writing a line (e.g. 512 bits) is usually completed in several iterations, with each iteration writing part of the line, incuring $\sim 1000ns$ page write latency [85]. Obviously, such long writes increase memory bank occupation time and thus block the subsequent accesses. Third, read and write circuits of PCM are usually quite large as a result of high-current, high-voltage operations and complex control. This is even more pronounced considering the extremely small and dense PCM cells of down to $4F^2$. To keep the overhead of peripheral circuits low for high area efficiency and high effective density, PCM chips usually utilize limited number of read and write circuits that are globally shared among all arrays in a PCM bank [7, 26, 29, 47, 14, 9], ultimately limiting achievable throughput.

Array Level Challenge : STT-MRAM. Unlike the uni-polar PCM which utilizes the same current direction for read, write-1 and write-0, the magnetic device of STT-MRAM is a bipolar device that changing its state from '0' to '1' requires a different current direction than from '1' to '0'. To provide bi-directional currents to an STT-MRAM cell, a classic array structure utilizes bitline pairs to control voltages on two ends of a cell [19, 27, 13, 61, 57, 70]. However, with aggressive scaling of the magnetic device and the access transistor [51, 1], the wire width and spacing of bitline pair determine the actuall cell size in an array [27, 13, 57] and become the bottleneck to further shrinking memory area, suppressing the benefits of device scaling. In other words, the device/cell level advantages cannot be translated into array level benefits. This problem is usually overlooked due to the illusion of excellent density and scalability of STT-MRAM device/cell.

In summary, the challenges of PCM and STT-MRAM are at bit and array levels, as shown in Figure 2. At bit level, both PCM and STT-MRAM suffer from excessive write energy, and PCM has very limited write endurance. At array level, PCM enjoys high density but could not provide competitive throughput due to its long write latency and limited number of read/write circuits. Although STT-MRAM owns satisfactory throughput, its conventional array architecture is constrained on density and scalability by the pitch of per-column bitline pair.

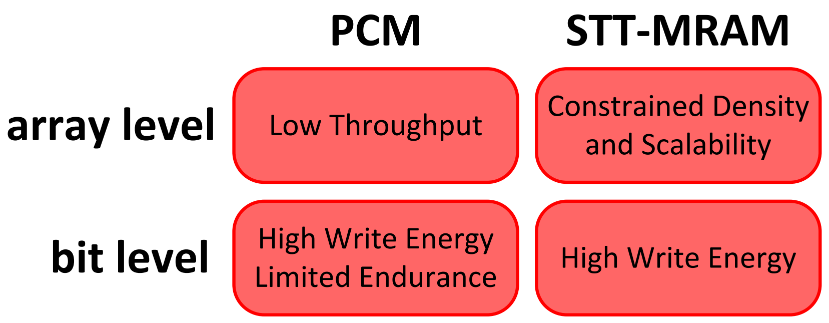


Figure 2: Challenges of PCM and STT-MRAM to be addressed.

As responses to these challenges, in my research work I develop multiple circuit solutions at corresponding bit or array levels, as listed below.

Bit Level Solution : PCM. In memory write operations, a great portion of bit-writes are redundant. That is, a write into a cell did not change its value. Based on this observation, I implement a circuit-level technique *Differential Write* [75, 31] to remove these unnecessary bit-write operations in PCM. Before each write, Differential Write performs a read first, and compares the stored data with to-be-written data. Then, only the cells that are actually changed are written, all redundant bit-writes are suppressed. As a result, the write energy and thermal stress on these redundant cells are removed, leading to significant energy reduction and lifetime improvement. In addition, Differential Write helps reduce write power and opens new opportunities to throughput enhancement and power-budget based memory scheduling [78].

Bit Level Solution : STT-MRAM. Similar to main memory, redundant bit-writes also widely exist in caches. For the same purpose of removing redundant bit-writes, I extend Differential Write to STT-MRAM with specific optimizations to remove the overhead of pre-write read. This is possible because the state change of the magnetic device is not a gradual procedure. Instead, resistance changes abruptly near the end of a write cycle. This means that at the early stage of a write operation, STT-MRAM cell still holds its valid old value. On the other hand, a read operation is performed through applying a voltage on the cell and then sensing the resulting current. A write operation follows exactly the same scenario of flowing current through the cell, except its higher voltage and longer pulse. Therefore, by sensing the write current at early stage of a write, the original cell data can be known, followed by throttling the write current if it is redundant. Such an *Early Write Termination* [76] technique achieves significant energy reduction with no performance penalty.

Array Level Solution : PCM. The slow write operation of PCM can hold a bank for a long time and blocks subsequent read operations. This is quite harmful to system performance as reads are on the critical path of CPU execution while writes are not. In fact, there is possibility to parallel a read with the on-going write because only the write circuits are in use and the read circuits are idle. However, the write operation occupies peripheral circuits like row decoder that are necessary for a read to start. Also, a second access to the same bank may interfere with the existing one and destroys both accesses. I propose a *Pseudo-Multi-Port Bank* design [78] that allows write and read to happen concurrently in different arrays of the bank, and thus the limited number of read and write circuits are fully utilized to provide parallelism. By leveraging the hierarchical wordline and bitline structures, the decoder result is latched on local wordlines so that the row decoder and global wordlines are released for other accesses to use in a time-multiplexed manner. Besides intra-bank parallelism, the Pseudo-Multi-Port Bank design provides potential for novel memory scheduling enhancements to fully take advantage of intra-bank parallelism and further improve throughput.

Array Level Solution : STT-MRAM. In a conventional dual-bitline array structure, every column of cells has a bitline pair consists of one bitline and one source-line which are identical metal wires. To eliminate the dominance of the bitline pair on cell area, I propose a *Common-Souce-Line Array* structure [72, 73] in which the per-column source-lines are removed and replaced by per-row source-lines that are shared by cells in the same row. A common-source-line array retains the original read scheme of a dual-bitline array but requires new method to write because writing different cells in a row are no longer independent. I develop novel write schemes and refine the array design to cope with the cell interference problem during a write. Compared to a dual-bitline array, the common-source-line array achieves significant area saving and liberates the scaling potential of future STT-MRAMs.

In summary, my circuit level solutions to the challenges of PCM and STT-MRAM are shown in Figure 3. At bit level, for PCM I implement Differential Write to remove large number of unnecessary bit-writes that do not alter the stored data. It is then extended to STT-MRAM as Early Write Termination, with specific optimizations to remove the overhead of pre-write read. At array level, I propose a Pseudo-Multi-Port PCM bank design to exploit intra-bank parallelism by recycling and reusing shared peripheral circuits between accesses in a time-multiplexed manner. For STT-MRAM I propose a Common-Source-Line Array architecture which uses a shared source-line along the row, essentially leaving only one bitline per column.

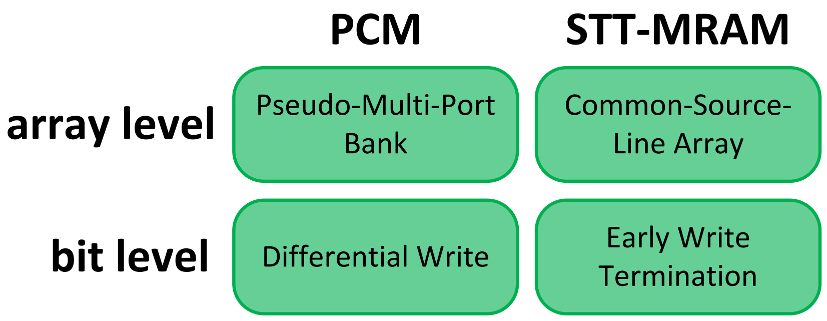


Figure 3: Circuit level solutions to the challenges of PCM and STT-MRAM.

Although the proposed solutions are circuit level designs, they also bring forth new potentials and opportunities to other design levels. Therefore in addition to circuit level analyses, for each technique I provide adequate architecture/system level and/or process/device level discussions for more comprehensive insights of my research work.

1.2 DISSERTATION ORGANIZATION

The remainder of this dissertation is organized as follows. Chapter 2 introduces the background knowledge within the scope of this work. The proposed techniques are then detailed in Chapter 3, 4, 5, 6. Chapter 7 thoroughly surveys related work. Finally Chapter 8 discusses potential future research topics and concludes this dissertation.

2.0 BACKGROUND

This chapter first briefly reviews the popular SRAM and DRAM/eDRAM, and then describes the essential characteristics of the emerging PCM and STT-MRAM.

2.1 SRAM

As the name suggests, the storage nodes of Static RAM (SRAM) are "statically" held to power or ground to retain their values as long as power is applied. To achieve this, a standard SRAM cell uses internal feedback based on a symmetric, bi-stable structure as shown in Figure 4. It contains a latch formed by a pair of cross-coupled inverters $\mathrm{PU_L}/\mathrm{PD_L}$ and $\mathrm{PU_R}/\mathrm{PD_R}$ holding the state, and a pair of access transistors $\mathrm{PG_L}$ and $\mathrm{PG_R}$ controlled by the wordline to read or write the state through the complementary bitlines. The positive feedback ensures the complementary logic values on the two storage nodes D and $\overline{\mathrm{D}}$, and corrects disturbances caused by leakage or noise.

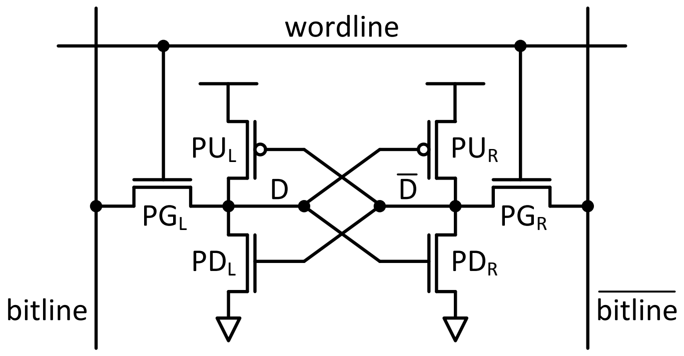
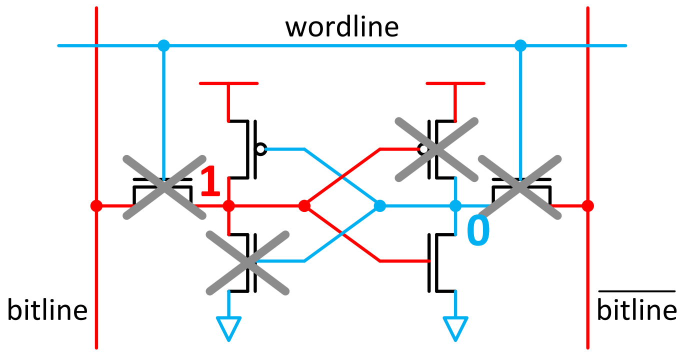
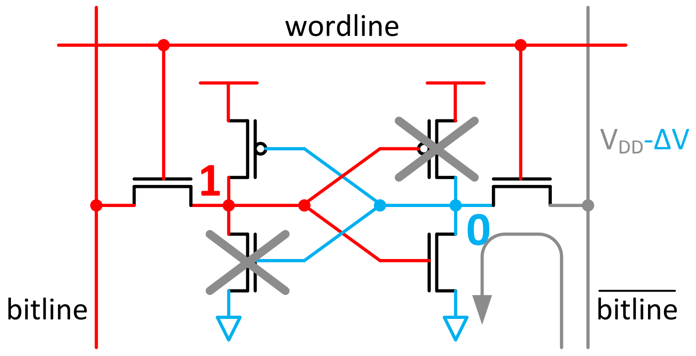


Figure 4: Schematic of an SRAM cell. PU – Pull Up; PD – Pull Down; PG – Pass Gate.

SRAM cell read operation is illustrated in Figure 5. The two bitlines are precharged high at V_{DD} and then left floating. When the wordline is raised and PGs are turned on, the 0-side bitline is discharged to $V_{DD} - \Delta V$ through the PD and PG, while the other bitline remains high. In case of small signal sensing, such ΔV is in the range of 100~200mV which can be captured by the sense amplifier circuit connected to bitlines to determine the logic value. In case of large signal sensing, $\Delta V = V_{DD}$ so bitlines swing full-rail and can drive skewed inverters directly. In both cases, the cell read latency is only in the order of 100ps.



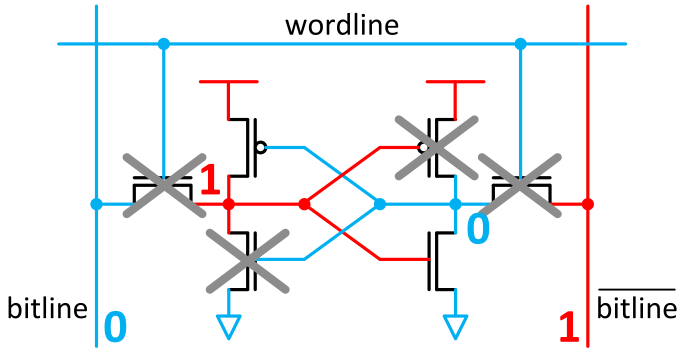
(a) Idle



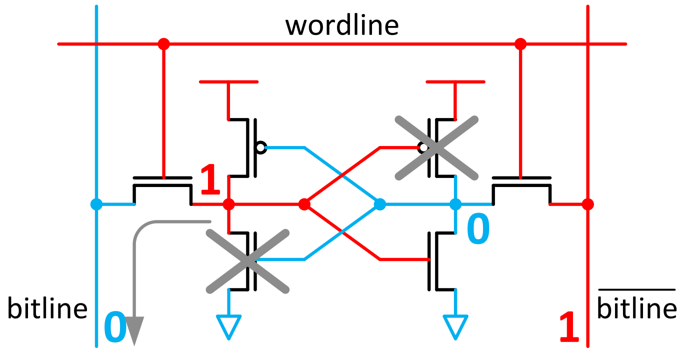
(b) Turn on access transistors and pull down the 0-side bitline

Figure 5: SRAM cell read operation.

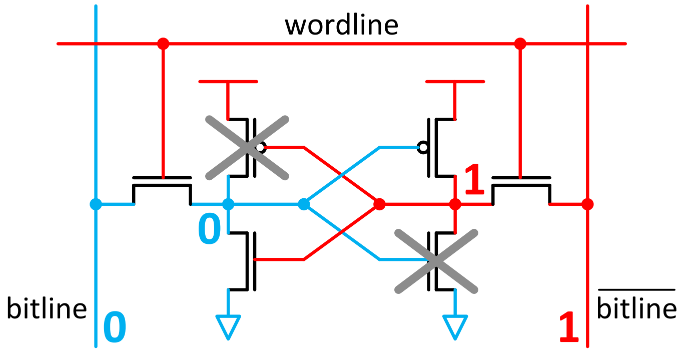
The write operation is shown in Figure 6. According to the write data, one bitline is held low by the write driver circuit while the other floats high. When the wordline is turned on, the 1-node is discharged by the write driver through PG, and the inverter on the 0-node will be eventually triggered once the 1-node is low enough, which ultimately flip the cell state using positive feedback. Thanks to such feedback mechanism, SRAM cell write is also very fast.



(a) Set bitline voltages



(b) Turn on access transistors and discharge the 1-node



(c) 0-node inverter triggered by the discharged 1-node and cell flips to the new state

Figure 6: SRAM cell write operation.

Although such multi-transistor cell structure provides stable storage and fast read/write, it also leads to large cell area ($>130F^2$) and thus low density. The other resulting drawback is high leakage. Figure 7 illustrates the subthreshold and gate leakage paths in an SRAM cell (junction leakages are not shown). As shown in the figure, each cell possesses multiple leakage paths. When many SRAM cells are organized into memory arrays and blocks, the total leakage becomes significant. This is especially problematic in large caches with millions of SRAM cells even with advanced leakage control schemes. Therefore, leakage powers of SRAM memories are usually dominated by cell leakage.

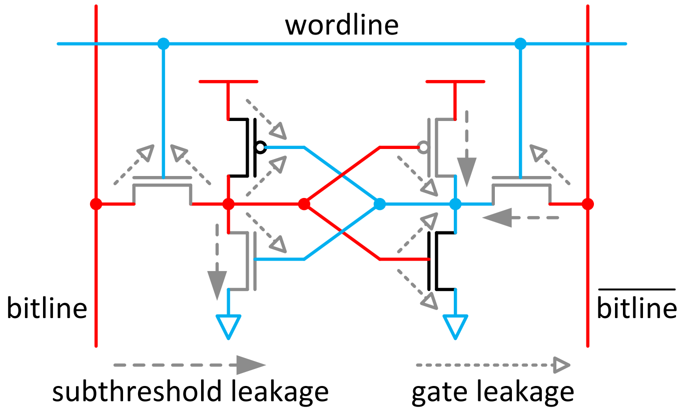


Figure 7: Leakage currents in an SRAM cell.

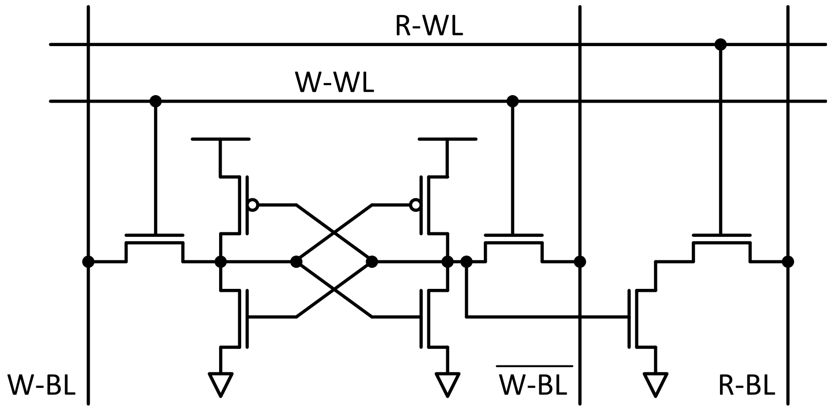


Figure 8: An 8T dual-port SRAM cell with one read port and one write port.

On the other hand, SRAM cell features unparalleled flexibility for function extensions. For example, the 8-transistor dual-port cell shown in Figure 8 provides a separate read port by adding a duplicated read path (PD and PG) and the corresponding wordline and bitline to the standard 6-transistor cell. Thanks to its decoupled structure, both read and write portions of the cell can be independently optimized, leading to better margins and therefore lower V_{DD-min}, which is crucial for L1 caches to sustain aggressive DVFS of the processor. To achieve higher throughput as required by register files, more read ports (PD and PG) and write ports (a pair of PGs) can be added at the expense of larger area and higher leakage. Furthermore, other structural extensions are also available for two bi-directional ports, Content-Addressable Memory (CAM), subthreshold memory, etc.

In summary, thanks to its fast operations, multi-port capability, and process compatibility, SRAM is widely used in all kinds of on-chip storage. On the other hand, due to its low density and high leakage, SRAM is becoming less appealing in large on-chip caches.

2.2 DRAM & EDRAM

As opposed to SRAM, Dynamic RAM (DRAM) stores data as charge on a capacitor that is "dynamically" floating. Thus, the basic cell is substantially simpler and smaller than SRAM, as shown in Figure 9. It only consists of a capacitor to store charges, and a pass gate transistor controlled by the wordline (WL) to read or write the cell through the single bitline (BL).

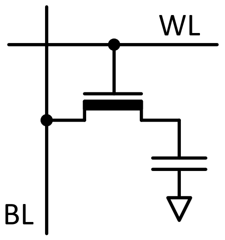
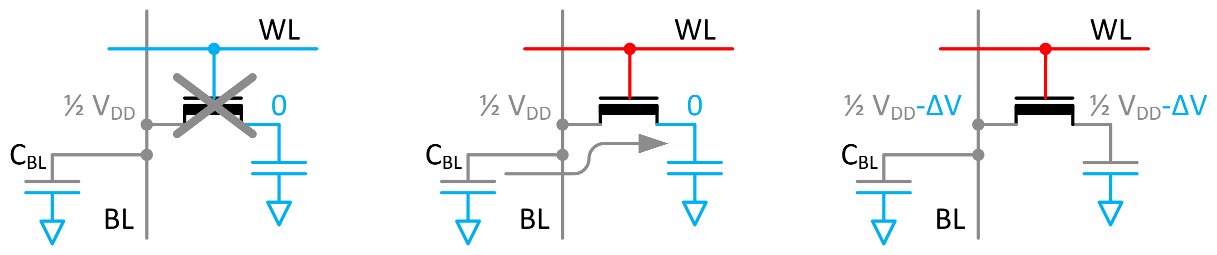
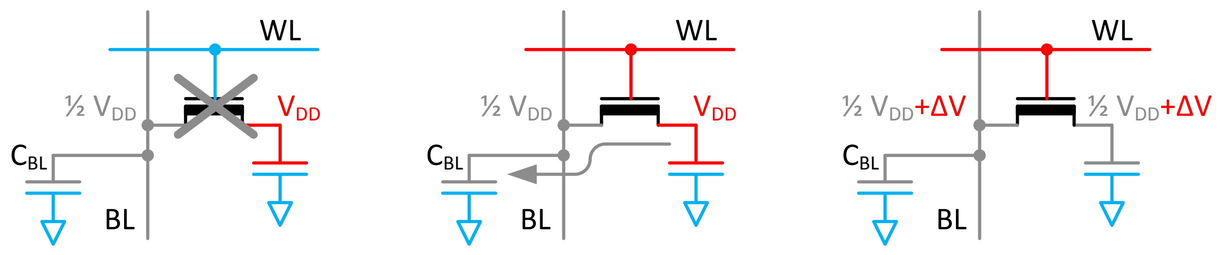


Figure 9: Schematic of a DRAM cell.

DRAM cell read operation is illustrated in Figure 10. The BL is precharged to the mid-point voltage $\frac{1}{2}V_{DD}$ and then left floating. When the WL is raised and the access transistor is turned on, charge sharing happens automatically between cell capacitor and BL capacitance. At the end of charge sharing, the BL and the cell reach a common voltage which is ΔV higher or lower than $\frac{1}{2}V_{DD}$, and such ΔV can be used by sense amplifier (SA) circuit to determine the logic value. However, at the same time the voltage stored on the cell capacitor is destroyed. This requires to equipe every BL with a SA which can amplify the ΔV to a logic level on BL and cell, so the full-rail voltage is *restored* into every cell. Therefore, the number of SAs in a DRAM memory equals to the number of cells on one row. This entire row of SAs also acts as the row buffer: WL and SAs remain active after data sensing, and SAs hold the sensed data as full-rail voltages on BLs which can be accessed by I/O using column address. Because DRAM cell and thus BL pitch is small, a 1:1 ratio of BL:SA is challenging. Fortunately, a DRAM SA is quite simple to consist of only 2 cross-coupled inverters, and layout tricks can loose the SA pitch to be 2 $\sim$ 4$\times$ the BL pitch. DRAM read operation is much slower than SRAM due to the RC delay of charge sharing and the weak SA driving long BL. Also, such destructive read is extremely energy inefficient because the mandatory restore must happen on all BLs and cells even if only a very small portion of the row is accessed by I/O.



(a) Read 0

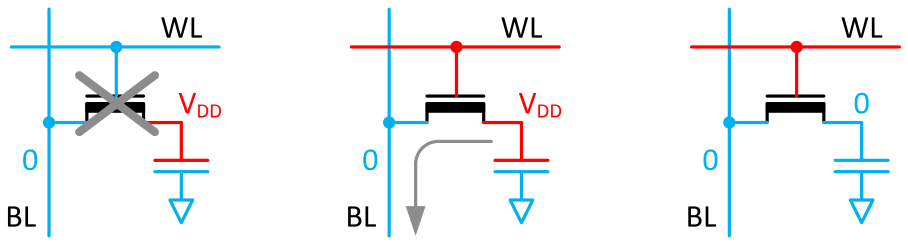


(b) Read 1

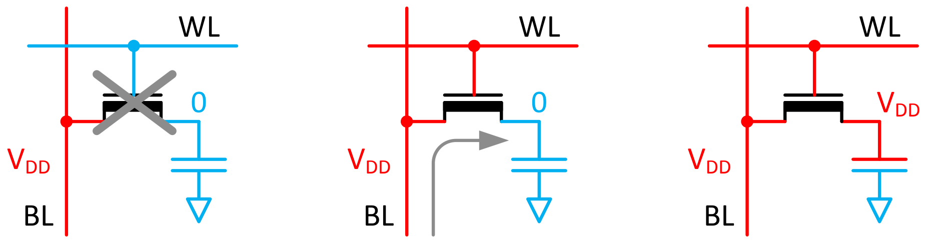
Figure 10: DRAM cell read operation.

The write operation is shown in Figure 11. According to the write data, the write driver (WD) circuit overpowers the SA which then pulls the BL to V_{DD} or ground. The BL voltage is then gradually forced onto the cell capacitor by charging or discharging through the access transistor. Therefore in case of open-page access (row buffer is active), a DRAM write is always “write-through”: the write data overdrives SA row buffer as well as the cell. Such write operation, and similarly the restore operation after a read, is quite slow as a result of the RC behavior on BL and cell.

Because the dynamic storage node in DRAM cell is not held to power or ground, charges leak into or out of the cell over time which gradually weakens the voltage margin between



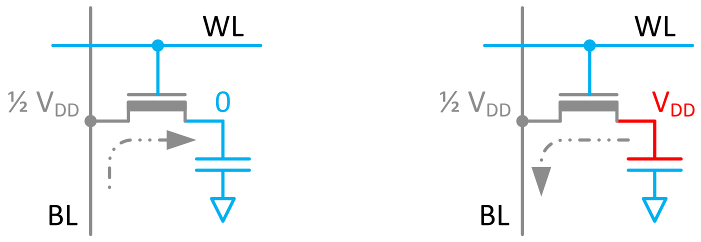
(a) Write 0



(b) Write 1

Figure 11: DRAM cell write operation.

cell and BL, as shown in Figure 12. The stored value is lost when the margin becomes too low to be distinguished by the read circuit. Therefore, DRAM cells must be periodically read and refreshed to restore the voltage on the cell capacitor. The time between the value is fully restored/written to the cell to the value expires is defined as the *retention time*. Obviously, longer retention time is favorable to mitigate the energy consumption and latency/throughput impacts of refresh operations.

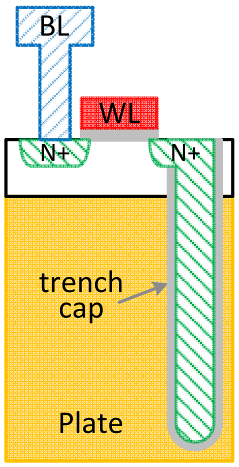


(a) 0-state (b) 1-state

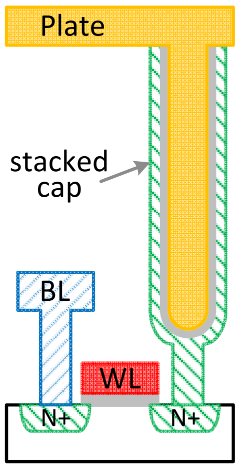
Figure 12: DRAM cell leakage.

Suppressing cell leakage is one of the most effective ways to prolong retention time. In current commodity DRAM technologies, the cell access transistor has very high V_{th} to reduce subthreshold leakage that a V_{DD} at its gate can barely turn it on. So DRAM wordlines are at a boosted voltage domain V_{PP} [81, 83] to 1) fully turn on access transistor for read and write, and 2) offset the V_{th} drop when passing a high voltage (logic 1) throught the nMOS access transistor. Also, inactive wordlines are pulled to a negative voltage below ground so the gate of the access transistor is more inversely biased to further suppress leakage [81, 2]. Furthermore, the structure, material, and fabrication of the access transistor are actually specially engineered for leakage control, and more efforts are needed in deeper scaled technologies.

Building large cell capacitor is also critical for retention time. This is usually implemented as trench or stacked capacitor [81]. A trench capacitor is essentially a deep hole etched into the silicon substrate, as illustrated in Figure 13(a). In contrast, the stacked implementation resides above the cell transistor, as shown in Figure 13(b). By exploiting vertical dimension spaces, both kinds of capacitor can achieve very high capacitance in small footprint. Because capacitance is directly proportional to physical dimensions of a capacitor device [37, 60], modern implementations of both device types tend to have very high and growing aspect ratio with technology scaling [81], i.e. they become deeper/taller to maintain capacitance with thrinking footprint. This creates fundamental challenges in the fabrication process and circuit integration, which ultimately limit the DRAM scaling.



(a) Trench capacitor



(b) Stacked capacitor

Figure 13: Cross-section comparison of DRAM cells with trench and stacked capacitors.

Commercial stand-alone DRAMs are built in specialized processes optimized for dense capacitor structures to offer a cell size down to $6F^2$, but they also have much higher latency and are not compatible with micro-processor/SoC/ASIC fabrication processes. Embedded DRAMs (eDRAM) share the same fundamentals with stand-alone DRAMs but trade density and retention time for logic process compatibilty and lower latency, so that it can be integrated on-chip as an SRAM alternative in large caches to provide higher density and lower power [2]. Some types of eDRAM use capacitor-less multi-transistor cell to totally eliminate the extra complexity and cost of capacitor integration [11].

In summary, DRAM's high density and low power earned it the position in high capacity main memory. Its embedded variants are also preferable to SRAM in large caches. However, the worsening refresh problem, energy inefficiency, and scaling difficulty are unique challenges in future DRAM generations.

2.3 PCM

Analogous to DRAM that uses a capacitor device to store a bit data, Phase Change Memory (PCM) utilizes a special device made of phase change material to remember information. The phase change material is one type of chalcogenide alloy, such as $Ge_2Sb_2Te_5$ (or GST in short), which can exist in two stable physical states: amorphous and crystalline. In the amorphous state, the material structure is highly disordered and thus highly resistive. In the crystalline state, the material has a regular crystalline structure and exhibits low resistivity. PCM exploits the significant difference in resistivity ($>10^2$) between these two states to store data, dissimilar from SRAM and DRAM that stores data as voltage levels. Typically, a cell in the amorphous state (high resistance) is regarded as a RESET state (or logic '0'), and a cell in crystalline state (low resistance) is regarded as a SET state (or logic '1'). The state of a GST device is preserved even after the cell is powered off, meaning that PCM is non-volatile. PCM also has good data retention time around ten years in general [81, 26, 4].

Figure 14 illustrates a PCM cell showing the structure of a typical storage node. The device usually consists of a thin layer of GST and a joule heater (or Bottom Electrode Contact (BEC)), sandwiched between top and bottom electrodes (TE and BE) that provide electronic contacts with the bitline (BL) and the access device (an nMOS in this example).

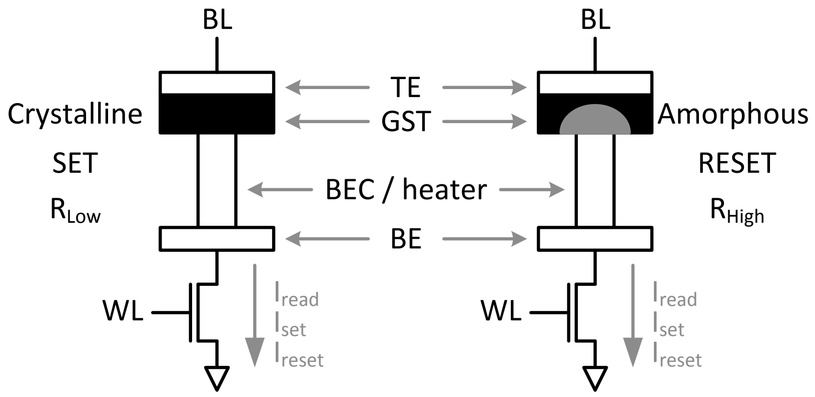


Figure 14: Conceptual view of a PCM cell showing phase change device structure.

Once the access transistor is enabled by the wordline (WL), read (I_{read}) or write (I_{SET}, I_{RESET}) currents from the BL can flow through the cell. These are essentially different from the voltage-mode/charge-based read/write operations of SRAM and DRAM.

Writing a PCM cell requires changing the physical state of its GST material. This is performed by injecting write currents for the heater to heat the GST at their junction. There are two write operations controlled by the applied voltage/current and duration, as shown in Figure 15. The SET operation heats GST above crystallization temperature (~300°C) but below melting temperature (~600°C) over a period of time with gradual quenching process. This places the entire GST in the low-resistance ($1K{\sim}2K\Omega$ [40, 7]) crystalline state (black in Figure 14). The RESET operation heats GST above melting temperature with fast quenching. This turns part of the GST near the junction into high resistance ($100K{\sim}700K\Omega$ [40, 29]) amorphous state (the gray "mushroom cap" in Figure 14).

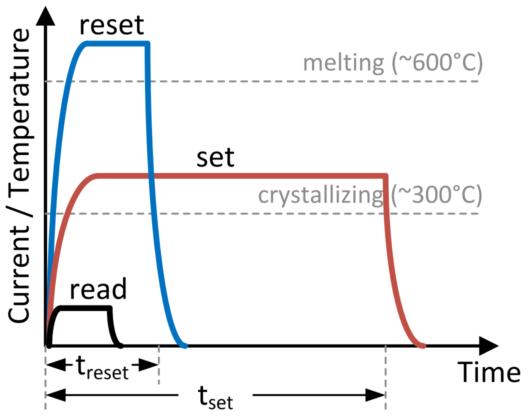


Figure 15: Read and write voltage/current/temperature pulses.

Such write operations contribute to the main limitations of PCM. Due to the heating and cooling processes, both SET and RESET take relatively long time to complete, and SET (150∼400ns [9, 29]) is slower than RESET (50∼100ns [7, 9]). These procedures incur high current injection in the range of 0.1∼1mA [9, 29] which are supplied from high voltage sources of 2∼5V [7, 29]. Therefore, the write power and energy per PCM cell are quite high. Given limited power budget, the number of cells written concurrently has to be restricted [26] to prevent excessive voltage droop and power supply noise [14], leading to multi-iteration operation with extremely long total latency in writing a wide line/page [26, 29, 85]. Such long writes increase memory bank occupation time and thus block subsequent accesses, ultimately hurt memory throughput and system performance, as will be discussed in Chapter 5. Also as a result of the repeated heat stress in melting and crystallizing, a PCM cell can be written for a limited number of times, typically 10^8∼10^9 [81], which is orders of magnitude lower than SRAM and DRAM (10^{16}). As will be demonstrated in Chapter 3, a PCM main memory without any lifetime improvement technique may last only ∼200 days running a typical `SPEC CPU` program.

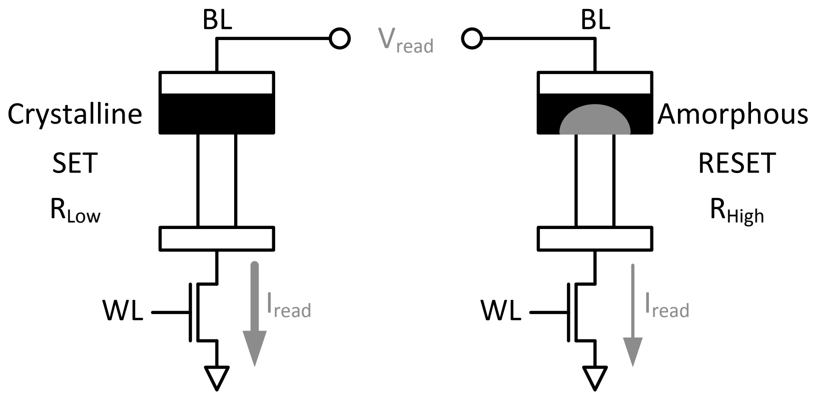


Figure 16: PCM cell read operation.

Reading data from a PCM cell involves sensing the resistance of the GST, as shown in Figure 16. When a regulated low voltage V_{read} is applied on BL, the amount of current I_{read} that flows from BL to the cell is determined by the cell resistance, i.e. larger with a SET cell and smaller with a RESET cell. The sense amplifer (SA) then compares I_{read} with a reference current I_{ref}, which is $I_{read-RESET} < I_{ref} < I_{read-SET}$, to determine the logic value. Notice that from a cell's perspective, the only differences between read and write are the amount and duration of applied current. Because V_{read} and I_{read} are intentionally upper limited, PCM read operation is non-destructive and has negligible heat stress compared to write operations. And read is much faster than write with $\sim 60ns$ random access latency at chip level [7, 26].

One distinguished benefit of such current-mode read/write operations is the efficient column multiplexing. In DRAM, charge sharing between cell and BL happens automatically once WL turns on, which destroys the entire row of data. This requires to restore every cell and drive/precharge every BL on the row, even if only a very small portion of the row is selected by column multiplexer for read/write. Similarly in SRAM, one BL per cell is discharged automatically once WL turns on, no matter the created voltage margin is used or not by SA, bringing forth stability concerns. And obviously all the BLs has to be precharged after the access. In other words, in both SRAM and DRAM the read process takes place unconditionally on *every* cell of an active row, regardless of the column multiplexer configuration. Therefore, with voltage-mode operations, a significant portion of energy is wasted on these "half-selected" (row-selected but not column-selected) cells and their BLs. With current-mode operations, cells are always in a passive position, and currents from read or write circuits are only directed to column-selected BLs and cells. Therefore on half-selected cells and BLs, neither the storage is violated nor any energy is wasted.

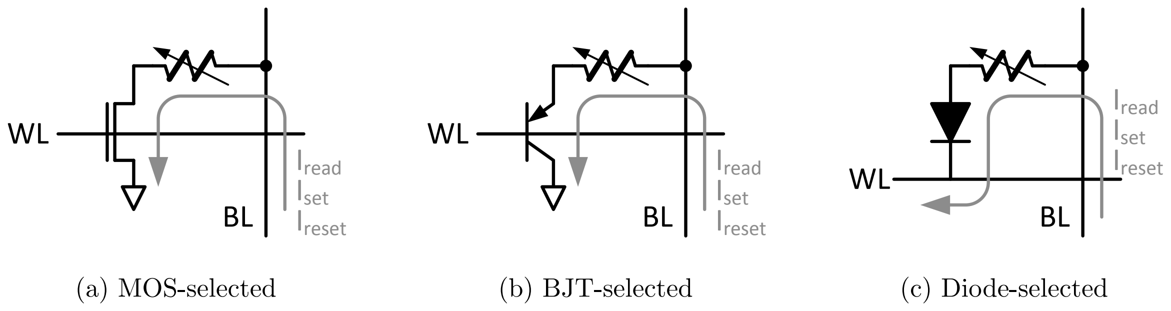


Figure 17: Three types of PCM cell using different access devices. The storage node is depicted as the alterable resistor.

There are three main options for the access device in a PCM cell: MOS, BTJ, and diode. The MOS-selected [7, 3, 40, 26, 17, 47] and BJT-selected [15, 4, 62] cells in Figure 17(a) and 17(b) are quite similar: WL controls the conductivity of the access device, and read and write currents flow through the cell to ground. The cell sizes are dominated by the access devices that are larger than the storage node. The diode-selected cell [29, 14, 9] in Figure 17(c) is fundamentally different that the active WL is grounded to drain read and write currents. Thanks to its simplest structure and the tiny diode device that can be stacked with the GST device, a diode-selected cell can achieve the smallest cell size of $4F^2$ [14, 9], which is just a cross-point of WL and BL. However, because turning on the diode requires an applied voltage which consumes the operation voltage headroom, the read and write operation voltages should be higher than the MOS-selected cell by the built-in potential (or threshold voltage) of a diode [29].

In PCM chips, boosted voltages are widely used mainly for two reasons. First of all, because GST resistance is quite large, high voltage is required to generate the high write currents. Secondly, because of the current-driven read and write operations, it is important to mitigate signal IR drop by minimizing the serial parasitic resistance in the read/write current path. So high voltages are also applied to the switch devices in the path, e.g. cell access transistors and column multiplexers, to make them more conductive [26, 9]. The huge current load, the coexistence of multiple boosted voltage levels, and the fine-grain controls for on/off scheduling, position compensation [40], pulse shaping [29], etc. lead to complex and large voltage regulation systems, mainly consist of charge pumps, that usually occupy considerable silicon area in high density PCM chips [14, 9]. Also as a result of the high current/voltage operations, read and write circuits of PCM are usually quite large in size. Hence, when pairing them with the extremely dense cells, their numbers are limited, and they are highly shared through hierarchical connections, for the purposes of pitch matching and density conservation.

The non-volatility of PCM is a new opportunity to exploit. Because the non-volatile, zero-leakage cells can preserve data without power, it is possible to power down an entire memory bank or chip during idle phases to eliminate leakage power on peripheral circuits [75, 31], which is crucial to meet the low-power requirements of future memory systems. Moreover, the physical state based storage is immune to soft errors caused by alpha particle or cosmic radiation usually seen in voltage/charge based storages [85]. PCM also offers much better scalability that the write currents reduce with the shrinking of GST device [81, 7], which is also a substantial solution to the power-bounded throughput problem [9]. Hence PCM provides a truly scalable solution compared to conventional DRAM.

Also, given the large resistance contrast between crystalline and amorphous states, it is possible to exploit partial crystallization states to store two or more bits per cell, forming a multi-level cell (MLC) PCM [4]. The MLC write typically uses an iterative Program-and-Verify (P&V) scheme. Here, a RESET operation is always done first to put the cell in an initial state. A series of SET and verify (read) operations then follow until the target resistance level is reached. This achieves precise control of the smaller resistance ranges, but also incurs extremely long latency [4]. When combining MLC storage with the $4F^2$ cell size, PCM can offer much better storage density than DRAM.

2.4 STT-MRAM

Despite the many common characteristics with PCM, such as resistance-based storage, current-mode read/write operations, non-volatility, good scalability, soft error immunity, etc., Spin-Torque-Transfer Magnetic-RAM (STT-MRAM) is based on a unique storage mechanism using the Magnetic Tunnel Junction (MTJ) device as the storage element. Figure 18 illustrates an STT-MRAM cell showing the structure of a typical MTJ device. It includes two ferromagnetic layers separated by one oxide barrier layer. The reference layer has a fixed magnetization direction and the free layer has an alterable one. The magnetization direction of free layer is changed by passing a driving current spin-polarized by reference layer [19]: a current from reference layer to free layer rotates the direction of free layer to the opposite of reference layer, resulting an anti-parallel (AP) and high-resistance state of MTJ (logic '1'); a current in the other direction parallelizes directions of the two layers, resulting an parallel (P) and low-resistance state (logic '0'). The state of an MTJ device is preserved even after the cell is powered off, meaning that STT-MRAM is non-volatile. STT-MRAM also has good data retention time around ten years in general [81].

Besides the current direction requirement, in either AP- or P-writing, the write current must be larger than the threshold or switching current of the MTJ device, and it must be

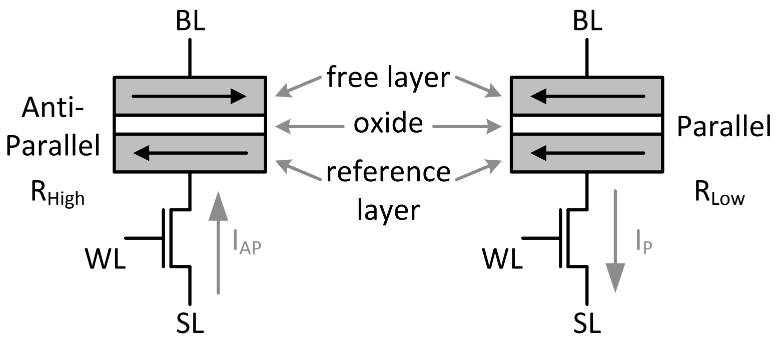


Figure 18: Conceptual view of an STT-MRAM cell showing MTJ structure.

applied for a certain amount of time (switching latency), to successfully alter the MTJ state [19]. Actually, during write operations, MTJ state and resistance change abruptly near the end of a write cycle [58, 6], different from the gradual, cumulative procedure in PCM writes. In other words, at the early stage of a write operation, an STT-MRAM cell still holds its valid old value. Also, there exist dependencies between switching current and switching latency: the larger the applied write current, the shorter the required time to flip the MTJ state [19, 12].

The underlying principle of MTJ switching is that the spin-polarized current disturbs the magnetic torque of free layer in one stable direction, turn the torque, and let it settle in the other stable direction. Similar to PCM write, such procedure also requires high current in the range of 50∼500μA [28, 19], as while as long period of 10∼100ns [19, 16, 36]. Therefore, STT-MRAM also suffers from high write power and energy per bit. Fortunately, unlike PCM, the MTJ device generally has unlimited write endurance (10^{15}) [81].

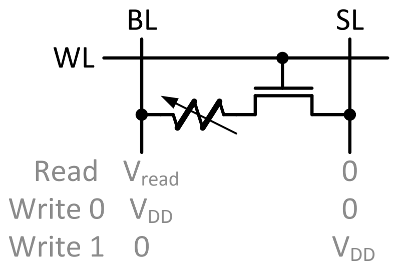


Figure 19: Schematic of an STT-MRAM cell. (MTJ = alterable resistor)

Because of such bipolar nature of MTJ switching, a MOS transistor is the undoubted choice of access device to provide bi-directional conductivity. Also, a pair of wires, one bitline

(BL) and one source-line (SL), is utilized to manipulate voltages on two ends of a cell, as shown in Figure 19. Due to the simple 1 MOS + 1 MTJ structure, the width and spacing of this wire pair usually determine the actuall cell size in an array [27, 13, 57].

Similar to PCM, reading data from an STT-MRAM cell is performed by sensing the resistance of the MTJ, as shown in Figure 20. When a regulated low voltage V_{read} is applied on BL and SL is grounded, the read current I_{read} that is inversely proportional to the cell resistance is compared by the sense amplifer (SA) with a reference current I_{ref}, which is $I_{read-AP} < I_{ref} < I_{read-P}$, to determine the logic value. Notice that the direction of read current can be determined at design time to be same as either AP- or P-writing. Such a choice is usually made by considering the combined effective resistance of MTJ and nMOS, for the purpose of maximizing sensing margin [36]. And read is usually faster than write with $<10ns$ latency [16, 12].

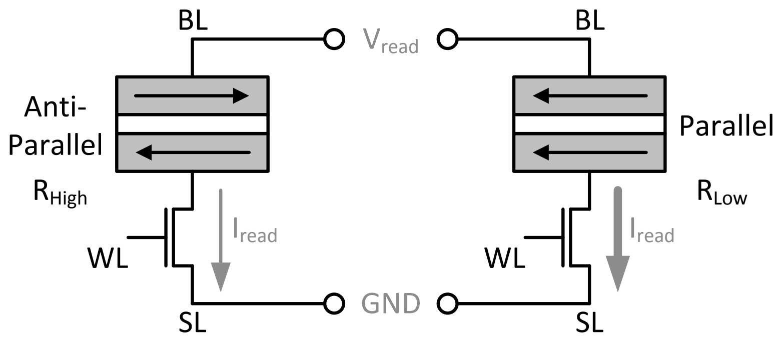


Figure 20: STT-MRAM cell read operation.

For read operation, one of the most important MTJ characteristics is the difference between high and low resistances, formally defined as Tunneling Magnetoresistance Ratio (TMR):

$$TMR = \frac{R_{AP} - R_P}{R_P}$$

Another crucial factor is the absolute MTJ resistance. Larger TMR and resistance are favorable to yield better sensing margin and delay [12, 1], because the resistance difference can be more distinguishable when parasitic resistances are present. Current STT-MRAM prototypes usually have TMR of 100%∼130% [19, 36, 12] and resistance in the range of

$2K\sim9.5K\Omega$ [19, 27]. These two parameters will be gradually improved over process generations via device level advancements [81, 12, 1].

STT-MRAM shares many benefits with PCM, e.g. efficient column multiplexing, zero cell leakage, non-volatility, soft error immunity, etc. As one advantage over PCM, STT-MRAM generally does not require high voltages to operate, making it more favorable as embedded memories for on-chip applications (also thanks to its unlimited write endurance). STT-MRAM also features very good scalability that the MTJ switching current can reduce with the shrinking of MTJ size [81, 12, 1]. This is a substantial improvement over previous generations of toggle MRAM technology which writes MTJ using magnetic field produced by the current on an adjacent wire [39]. With the toggle MRAM cell scaling down, such write current actually scales up to maintain the same energy density [1, 34]. Therefore, STT-MRAM is the absolute choice of future MRAM generations.

MTJ was also considered as a candidate in building cross-point memory arrays [34, 46]. Cross-point arrays achieve high cell density as there are no selection devices (typically much larger than the memory device itself) in cells. However, it is also difficult to select a cell without disturbing the adjacent ones, as a result of sneaky paths that lead to huge leakage current [34]. With MTJ, such sneaky path effect could be prohibitive as a result of its low TMR and low absolute resistance. For example, in a 64×64 array of a cross-point MRAM prototype [46], 97% of current is leaked away, and only 3% is effective in writing. This in turn limits the achievable sub-array size and area efficiency, and requires complex peripheral circuits to bias unselected cells, offsetting the benefit of high cell density. Therefore, MTJ is inferior to other bipolar memory devices with large resistance ratios, such as Memristor [10, 54], in building cross-point memory arrays.

3.0 BIT LEVEL ENERGY REDUCTION AND LIFETIME IMPROVEMENT FOR PCM

Despite the advantages like density, leakage, non-volatility, scalability and reliability, two major shortcomings of PCM as the main memory are high write energy and limited write endurance. Based on the observation of redundant bit-writes, in this chapter I develop circuits to implement the *Differential Write* (DW) technique that removes these unnecessary bit-write operations in PCM [75, 31]. Because only the cells that are actually changed are written, it is capable of both reducing write energy and extending cell lifetime. Evaluations show that DW offers 60% dynamic energy saving and 4.5× lifetime improvement on average. When combining DW with simple wear-leveling techniques, it is demonstrated that PCM-based main memory is practical in terms of lifetime [75, 31]. In addition, Differential Write helps reduce write power and opens new opportunities to throughput enhancement and power-budget based memory scheduling [78]. Thanks to its fundamentality and simplicity, DW can be seemlessly integrated into upper level techniques. Notably, it was used as the basis of some later studies [8, 18, 78], and also incorporated in a Samsung prototype [14].

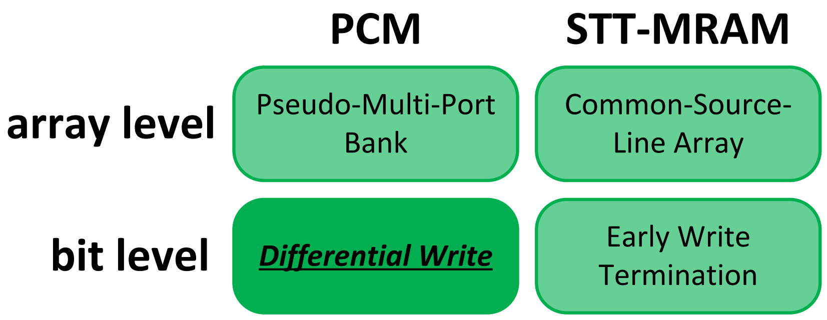


Figure 21: Bit level solution for PCM: Differential Write.

3.1 ENERGY AND LIFETIME PROBLEMS OF PCM

Both write energy and write endurance challenges are results of the phase change material characteristics. PCM's thermal-driven write processes incur high current injection in the range of 0.1$\sim$1mA [9, 29]. Such high currents are supplied from high voltage sources of 2$\sim$5V [7, 29]. Moreover, the currents keep flowing through the phase change material for 50$\sim$400ns [7, 29] to fully finish melting or crystallizing. Therefore, the per-bit write energy of PCM is quite high. For instance, assuming the conservative write current, voltage, and pulse width to be 100μA [9], 2V [7], and 50ns [29] respectively, this leads to $100\mu A \times 2V \times 50ns = 10pJ$ per bit, which is much higher than DRAM's $\sim$1.5pJ per bit [63]. What makes things worse is the fact that the high write voltages are generated by charge pump circuits from regular power supplies with *limited power efficiency*. So the actual energy/power consumption is even higher at chip level. Furthermore, the low leakage of PCM can lead to significant leakage saving over DRAM, given that leakage contributes to large portion of main memory energy due to the low activity. However such benefit may be overwhelmed by the excessive write energy.

On the other hand, due to the repeated heat stress in melting and crystallizing processes, a PCM cell can be written for a limited number of times, typically $10^8 \sim 10^9$ [81]. While this is better than the 10^5 write endurance of Flash, it is much worse than that of a DRAM cell (10^{16}) and is a big concern when PCM is used in main memory. To illustrate the problem, the "unprotected" lifetimes of a PCM main memory are tested using a variety of benchmarks including `SPEC2K`, `SPEC2006`, and `SPECWeb` [1]. Here I refer the lifetime of a PCM to the duration before the first cell starts to fail, and the number of rewrite cycles for a PCM cell is assumed to be 10^8. Figure 22 shows the projected lifetime of PCM memory without any enhancement technique. As shown in the figure, the results range from 25 days for `mcf` to 777 days for `specweb-banking`, and the average is only 171 days. Hence in order to make PCM main memory practical, lifetime improvement techniques are needed to extend PCM lifetime to an acceptable level.

[1]Architectural simulations in this chapter were performed by Ping Zhou at Department of Electrical & Computer Engineering, University of Pittsburgh.

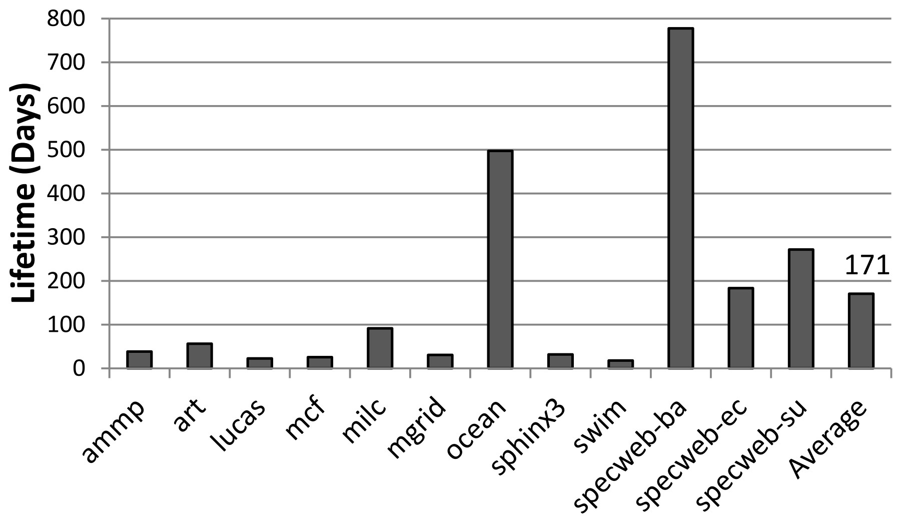


Figure 22: Lifetime of PCM memory without any improvement technique.

3.2 THE OPPORTUNITY: REDUNDANT BIT-WRITES

To improve the write energy and endurance of PCM memory, one intuitive step is to reduce the total number of bit-writes. Several existing studies have shown that there is a high probability that a write into a cache or memory location does not change its content, and therefore can be removed. Such an observation has been used at the word level for L1 cache [32], multiprocessors [33], and off-chip memories [30]. Intuitively, this phenomenon is more significant at bit level [75, 31].

In a conventional memory write, every bit in the request is written once. However, a great portion of these bit-writes are redundant. That is, in most cases, a write into a cell did not change its value. This is termed "redundant bit-writes" in this study. These bit-writes are hence unnecessary, and removing them can greatly reduce the write frequency of the corresponding cells. Figure 23 shows the percentages of redundant bit-writes for different benchmarks. They are calculated as the number of redundant bit-writes over the total number of bits in write accesses. The "SLC" series represents redundant bit-writes in single level PCM cells, i.e., each cell stores either '0' or '1'. The "MLC" series represents 2-bit multi-level PCM cells. That is, each cell stores 4 binary values.

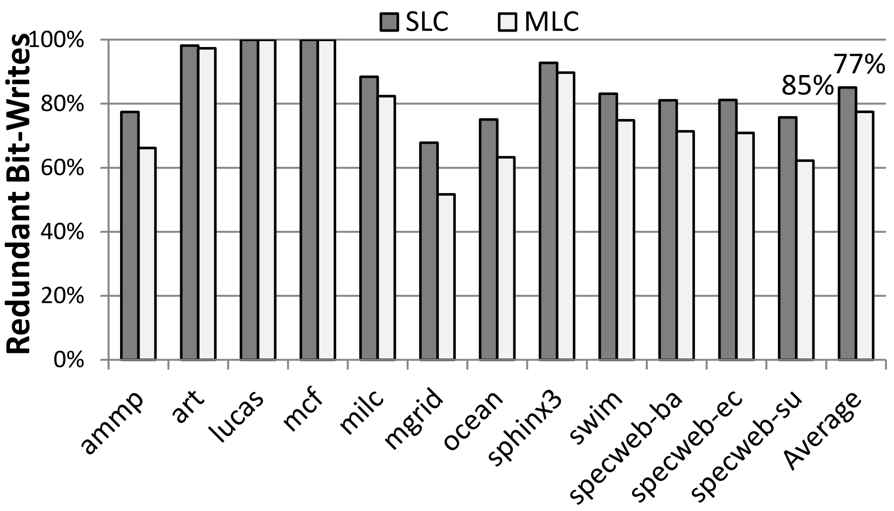


Figure 23: Percentage of redundant bit-writes for single-level and multi-level cells.

From the results, it it clear that all benchmarks exhibit high percentages of redundant bit-writes. Theoretically, for single-level cells, the statistical bit-write redundancy is 50% if writing a ‘0’ and ‘1’ is equally likely. For MLC cells, the redundancy probability is 25%. However, the measured redundancies for real workloads are much higher than the theoretic values, showing interesting *value locality.* The redundancy ranges for SLC and MLC cells are 68∼99% and 52∼99%, with averages of 85% and 77% respectively. This inspired the idea of removing redundant bit-writes, which leads to the Differential Write technique.

3.3 DIFFERENTIAL WRITE

The first step of Differential Write (DW) to remove redundant bit-writes is identifying which bit-writes are redundant. Before each write, DW performs a read first, and compares the stored data with to-be-written data, as illustrated in Figure 24. Next, based on the comparison, only the cells that are actually changed are written, all redundant bit-writes are suppressed.

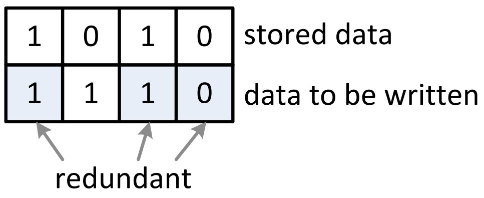


Figure 24: Identify redundant bit-writes based on comparison.

3.3.1 Circuit Design

The comparison logic can be simply implemented by adding an XNOR gate that takes read and write data as inputs, as illustrated in Figure 25. The XNOR output drives a pMOS transistor which can block the write current when the write data equals the currently stored data.

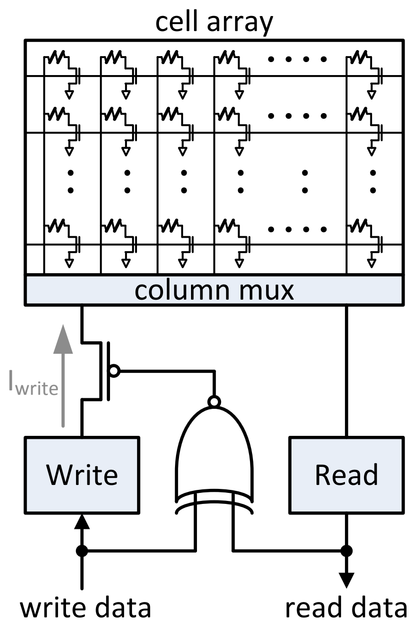


Figure 25: A micro-architectural view of Differential Write.

To keep the overhead low, a simple 6-transistor XNOR gate [21, 66] is utilized, as shown in Figure 26. Although the input node of its inverter suffers from V_{th} drop when both XNOR inputs are 0, this is not a problem given the relatively small V_{th} in the high-voltage environment of PCM, and the output is corrected to full-rail by the inverter [21, 66].

The implementation in Figure 25 is the simplest using only an XNOR gate plus a pMOS transistor [75, 31]. However the pMOS transistor represents certain extra parasitic resistance

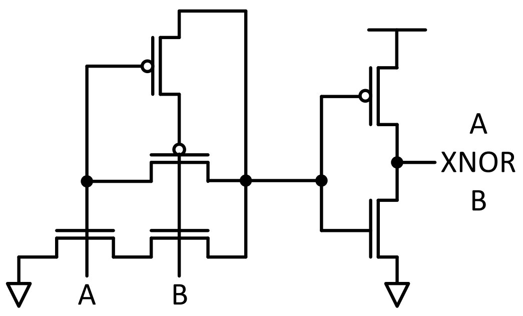
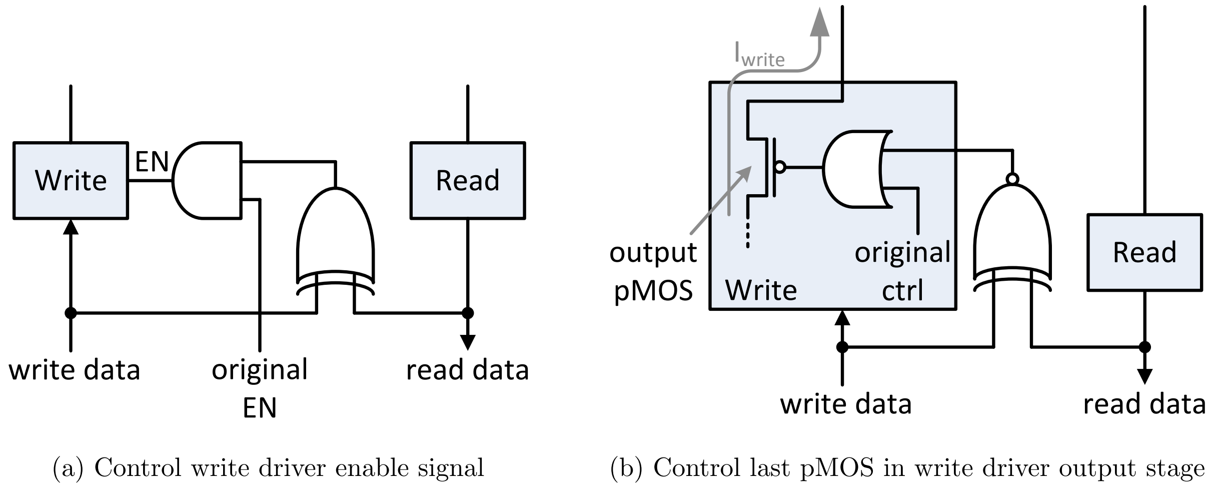


Figure 26: XNOR circuit implementation [21, 66].

on the write current path. This is unfavorable because a lot of efforts, e.g. boosted voltages and hierarchical bitline, are already spent on reducing the serial parasitic resistance of the write current path [26, 9]. Therefore, one design alternative is to combine the throttling signal out of data comparison with existing write circuit control signals. Figure 27 illustrates 2 design styles. In Figure 27(a), the comparison is done by an XOR gate whose output is then ANDed with the original write circuit enable signal [29] to generate the final enable. Here the XOR gate is simply a complementary design of the XNOR gate in Figure 26. In Figure 27(b), the XNOR output is ORed with the original control signal of the final output stage pMOS in the write circuit [26, 47]. These two designs guarantee better controlled write path resistance and only incur slightly higher overhead because of the AND/OR gate instead of a single pMOS in Figure 25.



(a) Control write driver enable signal

(b) Control last pMOS in write driver output stage

Figure 27: Two alternative design styles of Differential Write.

Notice that PCM write drivers usually drain write currents from a boosted voltage supplied by charge pumps [26, 29, 62, 47, 14, 9]. By reducing the number of bit-writes, DW can greatly decrease the current load on charge pumps and therefore improve their power efficiency [67]. No modification is required to the charge pump system because it can detect DW-caused load changes and automatically adapts its clock frequency.

3.3.2 Overhead

The delay and power overheads of the added simple circuits are $\sim 150ps$ and $\sim 20\mu W$ based on our SPICE simulations with $45nm$ technology, which are negligible in PCM. Instead, the main delay and energy overheads come from the pre-write read. In PCM operations, read is much faster than writes [2], so the delay increase here is much less than doubling the latency of a write. In addition, write operations are typically less critical than read operations, so increasing the effective write latency has less negative impact on the system performance. Also, a read operation consumes much less energy than a write because of the low read voltage/current and short read pulse. The extra read energy can be easily outweighted by the huge write energy saving of DW. Consequently, the overhead of extra circuits and pre-write read can be well justified by the benefits of DW at architecture level [75, 31].

3.4 EVALUATION RESULTS

3.4.1 Modeling

To provide a PCM model for architectural simulations, I modeled PCM in a hybrid manner that combines SPICE simulation with CACTI [84] estimation. PCM share similar peripheral circuits like decoders, interconnects, data buffers and I/O drivers with traditional memories. But it has differences in the implementation of cell arrays. Hence, the methodology I used is to simulate the essential circuits such as the cells, bitlines, wordlines, read/write circuits etc.

[2]Read latencies reported in published prototype chips are full round-trip latencies at chip I/O, including delays on all peripheral circuits. In contrast, the pre-write read in DW takes place purely inside the array on SAs, BLs and cells, and is much faster than writes.

at SPICE level, and then replace the CACTI results related to those essential circuits with SPICE results. In other words, only the skeleton of the CACTI DRAM model is used which is filled with the contents of SPICE PCM model. All SPICE simulations were performed with $45nm$ device models at 90°C. And CACTI was used to estimate a memory chip with 4Gb capacity. The numbers produced by this model, as listed in Table 2, were then applied in architectural simulations for evaluation of PCM memory. The modeled per-bit write energies are similar to that in [30].

Table 2: Latency and energy parameters used in architectural simulations.

	Latency (ns)	Energy (pJ)
Read	36.28 (row miss)	10.68 (row miss)
	6.47 (row hit)	3.77 (row hit)
Write	90.27 (0)	26.8 (0)
	120.27 (1)	13.7 (1)

With Differential Write, PCM's write latency is not fixed. If a write request is completely redundant (i.e. every bit of the line to be written is same as the old data in memory), then it can be terminated after the pre-write read and comparison operations, resulting in shorter latency. Also, the per access write energy is not fixed. It can be calculated as:

$$E_{DW-write} = E_{fixed} + E_{read} + E_{bitchange}$$

E_{fixed} is the "fixed" portion of energy charged for each PCM write on peripheral circuits including decodings, row selecting, interconnects, etc., plus the added DW circuits. This part is $4.1nJ$ per access as measured from SPICE simulation. E_{read} is the energy to read out the row for comparison and this part is approximately $1.075nJ$. The $E_{bitchange}$ part depends on the number of updated bits ($0 \to 1$ or $1 \to 0$): $E_{bitchange} = E_{1\to 0}N_{1\to 0} + E_{0\to 1}N_{0\to 1}$, where $E_{1\to 0}$ and $E_{0\to 1}$ are listed in Table 2. Therefore, per access write energy for PCM (nJ) can be expressed as:

$$E_{DW-write} = 4.1 + 1.075 + 0.0268 \times N_{1\to 0} + 0.0137 \times N_{0\to 1}$$

Therefore, given the significant opportunity of redundant bit-writes as shown in Figure 23, Differential Write can significantly reduce PCM's write energy by reducing $N_{1\to 0}$ and $N_{0\to 1}$, which can also benefit the lifetime.

3.4.2 Energy and Lifetime Results

Figure 28 compares the dynamic energy breakdown of the PCM main memory with and without DW, shown as the "DW" and "raw" bars respectively. The energy of pre-write reads are counted into DW write energy, so in all benchmark workloads the read energies remain the same. Although reads are usually more frequent than writes in main memory, write energy dominates in most workloads mainly due to the high per-bit write energy. Especially, in write-intensive workloads like `lucas` and `mcf`, almost the entire dynamic energy comes from writes. With DW, a great portion of write energy saving can be achieved, and the percentage of saving is proportional to the percentage of redundant bit-writes in Figure 23. Overall, on average DW offers 60% dynamic energy saving.

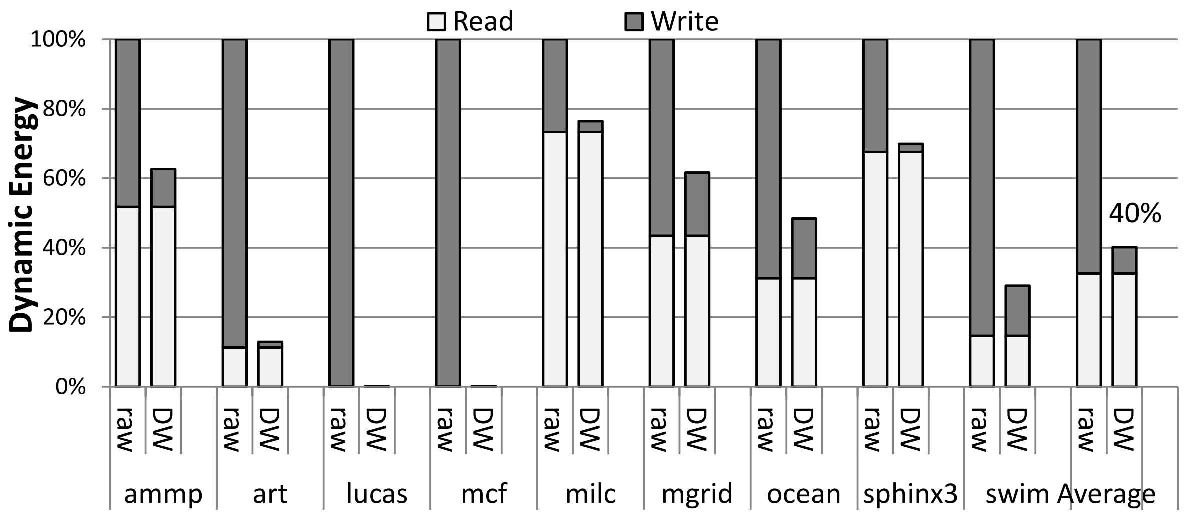


Figure 28: Dynamic energy breakdown showing energy savings of Differential Write (SLC).

After applying DW, the lifetime of PCM main memory on average is extended to 770/592 days, or 2.1/1.6 years, for SLC/MLC respectively, as shown in Figure 29. However, even though DW achieves 4.5/3.5× improvements, the 2.1/1.6 years of lifetime is still too short for main memory. Further improvement can be obtained with simple wear-leveling techniques like row shifting and segment swapping, so that PCM-based main memory becomes practical in terms of lifetime [75, 31]. Also, DW results in localized bit changes inside each row. This provides more room for wear-leveling.

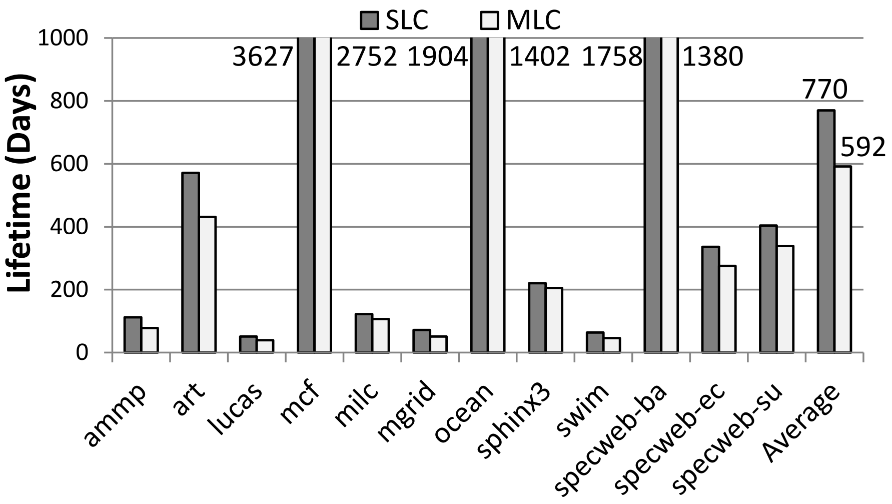


Figure 29: Lifetime (days) after applying Differential Write.

3.5 BEYOND ENERGY AND LIFETIME

Differential Write is not only an effective technique to reduce the write energy and improve the lifetime of PCM main memory, it also opens new opportunities for upper level architecture design. For example, in order to improve PCM memory throughput without breaking its power envelop, an intuitive approach is to improve the utilization of available power budget. This can be accomplished by reducing the number of bit-writes in each write request, so that more write requests can be served concurrently under same power limit. The Differential Write technique offers a great opportunity here: with DW, about 85% of bit-writes can be avoided in PCM write requests. Moreover, as studied in [78], DW provides important bit-change information that can be leveraged by power budgeting techniques for better estimation of power demands.

4.0 BIT LEVEL ENERGY REDUCTION FOR STT-MRAM

As discussed in Chapter 1 and 2, STT-MRAM shares the similar challenge of high write energy to PCM. Although STT-MRAM writes consume less energy than PCM writes, it is still in the same order as DRAM and much higher than SRAM and eDRAM, which is prohibitive in an on-chip cache environment. Fortunately, the opportunity of redundant bit-writes is not exclusive to main memory. Therefore PCM's solution of Differential Write can be borrowed by STT-MRAM. In this chapter, following the idea of Differential Write in PCM, I develop *Early Write Termination* (EWT), a novel technique to significantly reduce write energy with no performance penalty [76]. Because EWT circuits detect redunct bit-write and cut off write current at the early stage of a write, no pre-write read is triggered. Evaluations show that EWT can reduce 52% of dynamic energy on average. It can be combined with volatile-write [52] for further energy savings.

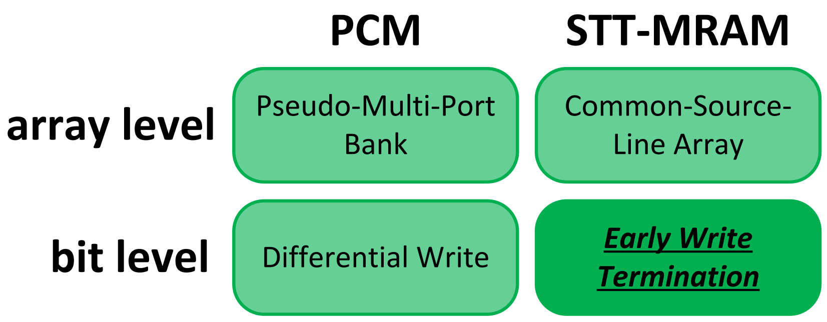


Figure 30: Bit level solution for STT-MRAM: Early Write Termination.

4.1 ENERGY PROBLEM OF STT-MRAM

Similar to PCM, the high write energy of STT-MRAM is also determined by the storage device. STT-MRAM write uses spin-polarized current flowing through the MTJ to disturb its magnetic torque in one stable direction, turn the torque, and let it settle in the other stable direction with different resistance. Such procedure also requires high current in the range of 50$\sim$500μA [28, 19], as while as long period of 10$\sim$100ns [19, 16, 36]. Therefore, STT-MRAM suffers from high per-bit write energy of conservatively $50\mu A \times 1.2V \times 10ns = 0.6pJ$ per bit, much higher than that of SRAM and eDRAM. It has been shown that STT-MRAM cache consumes 6$\sim$14 times more energy per write access than SRAM [55]. Several recent studies proposed to relax the non-volatility requirement from the typical ten year storage-class retention time, to reduce the write pulse width and thus write energy [52]. However, even with such volatile writes, the write energy still dominates total energy of an STT-MRAM based cache, offsetting the energy savings from low leakage [5]. Therefore, reducing write energy of STT-MRAM is important to improving its energy efficiency.

4.2 THE OPPORTUNITY: REDUNDANT BIT-WRITES

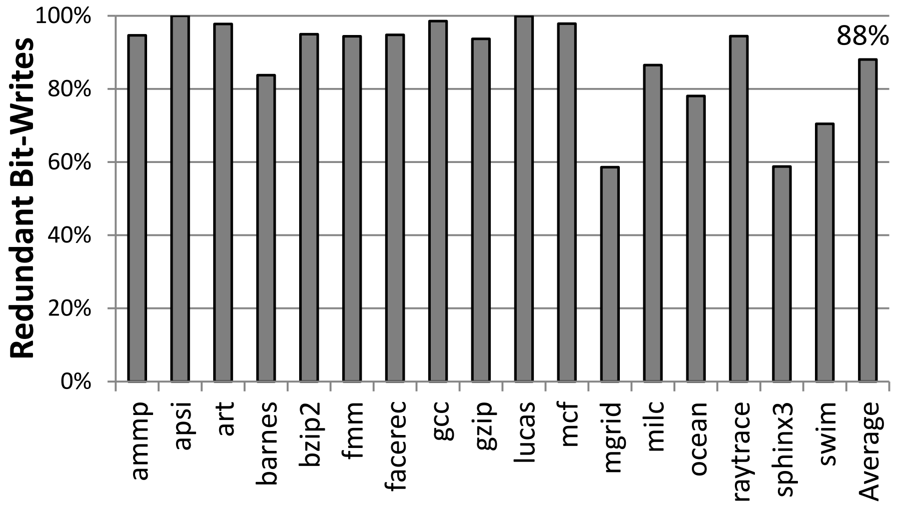


Figure 31: Redundant bit-writes in 16MB STT-MRAM L2 cache.

Similar to main memory, redundant bit-writes also widely exist in caches. Figure 31 shows the results of an evaluation with a 16MB L2 cache [1]. On average, about 88% of bit-writes are redundant, which implies a significant amount of removable bit-writes and a great potential of energy saving in an STT-MRAM cache.

4.3 EARLY WRITE TERMINATION

4.3.1 Rationale

To exploit the opportunity of redundant bit-writes, the approach of Differential Write could be employed: read out the cache content, compare it with the new value, and write back only the different bits. However, this method entails that every write is preceded with a read operation, as discussed in Chapter 3. Although reads consume much less energy and are much faster than writes in STT-MRAM, the increased latency could be much more expensive in the delay-sensitive L2 cache than in main memory. Fortunately, we could do better than Differential Write based on the following unique features of STT-MRAM:

- When writing an STT-MRAM cell, the change of MTJ resistance is not a gradual and cumulative procedure like in PCM. Instead, resistance changes abruptly near the end of a write cycle [58, 6]. This means that at the early stage of a write operation, an STT-MRAM cell still holds its valid old value.
- A read operation is performed by applying a voltage between bitline and source-line, followed by sensing the resulting current to determine the MTJ resistance. A write operation follows exactly the same scenario of flowing current through the cell, expect its higher voltage and longer pulse.
- Because write voltage and current are larger than read, the resulting current difference between high and low resistance states is also larger. This means the *sensing margin* is better in distinguishing high and low resistances using write current, and thus the sensing delay can be shorter.

[1]Architectural simulations in this chapter were performed by Ping Zhou at Department of Electrical & Computer Engineering, University of Pittsburgh.

- A write operation in STT-MRAM is much longer than a read. A typical write pulse of STT-MRAM is at least $10ns$, below which the switching current increases rapidly [19, 12]. However, reading from a cell can complete in around $1ns$ [55]. This not only gives us adequate room to sense the old value during the early stage of a write, but also implies great opportunity in saving energy by terminating the write current as soon as redundancy is detected.

Based on these observations, we propose a novel write scheme with the capability of early termination in case of a redundant write. The main goal is to greatly reduce the write energy without impact on performance. The basic idea is to sample the resistance of the MTJ (old value) at early stage of a write operation, and throttle the write current if old value is same as the new value. To achieve this goal, we need an additional comparator circuit to accompany each write driver to sense the resistance of the cell during a write operation, and some other additional circuits to generate the control signals. The procedure of such Early Write Termination (EWT) technique is as follows:

1. When a write operation begins, write voltage is applied between BL and SL to generate the write current.
2. When the signals are stabilized, a comparator is enabled to sense the resistance of the cell (i.e. the old value).
3. If the old value is same as the new value, a control signal shuts off the write current path and terminate the operation on this cell. Otherwise, write operation on this cell continues normally.

As we can see, the above process does not require an extra read to precede a write because sensing the old value is performed together with the write operation. Compared to the Differential Write scheme used in PCM that mandates a read before a write [75, 31], EWT does not introduce any overhead in performance. In fact, experiments show that EWT sometimes even improves performance a little because some write requests (an entire L1 cache line) can be completely throttled and terminated in their early stage.

4.3.2 Circuit Design

The overview of proposed EWT implementation is illustrated in Figure 32. It does not require any change to existing read or write circuit. Instead, it is designed to work as an "add-on" to a functional STT-MRAM. The main circuit components added for EWT include a voltage comparator, multiplexers, and an AND gate.

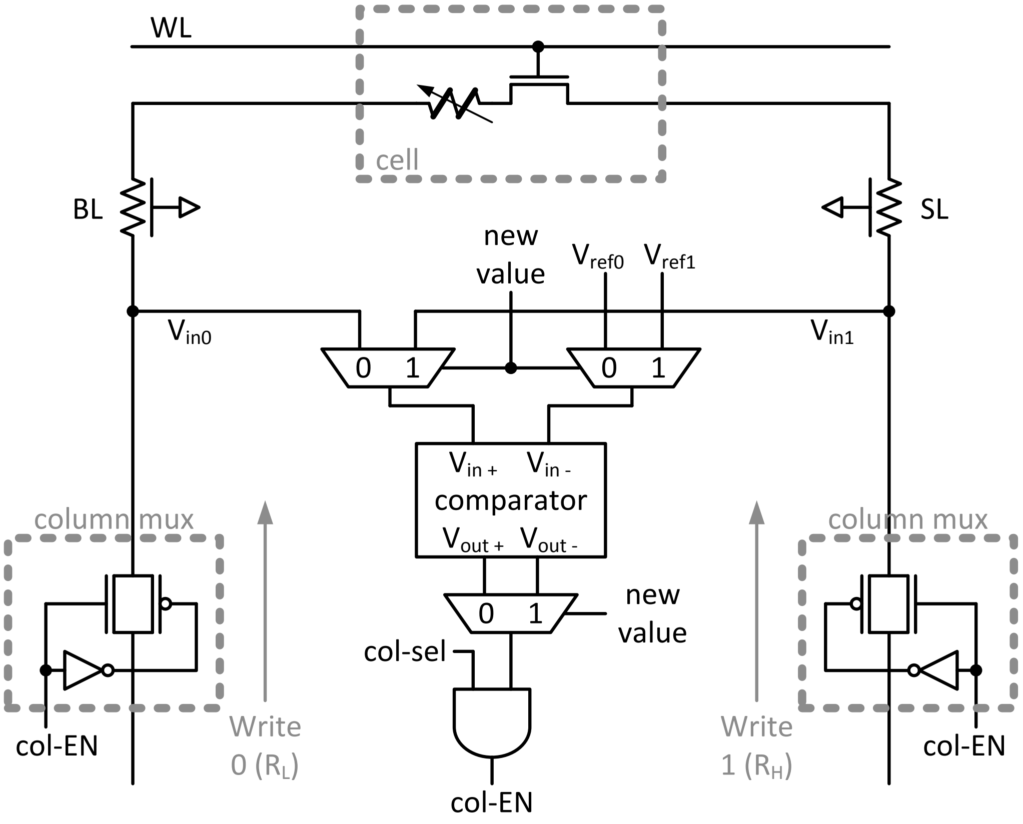


Figure 32: Overview of EWT circuit design.

The write operation starts with applying a positive voltage between BL and SL for writing a '0' (low resistance), or negative voltage between BL and SL for writing a '1' (high resistance). The existing column multiplexer circuit is used with revised control, shown as the pass-gates on BL and SL in Figure 32. They are now controlled by the column enable signal (col-EN) which is generated by ANDing the comparator output with their orignal column selection control (col-sel). These pass-gates serve two purposes: 1) when it is detected that the write is redundant, the pass-gates are turned off to cut the write current on BL and SL; 2) together with other resistances further away from the cell, they act as small loads on the write current path to convert the write current difference into voltage difference of V_{in0} or V_{in1}. This is used by the comparator to detect the stored value in the cell.

For example, as shown in Figure 33, when a '0' is written, a positive voltage is applied between BL and SL creating a current flow from BL to SL. Due to the voltage distribution along the write current path, there is a voltage drop from power supply to V_{in0}. The magnitude of this drop is determined mainly by the resistance of MTJ (other wire loads are constant). If it is storing a '1', meaning that the resistance is high, the voltage drop across the cell is larger, so the voltage drop from V_{DD} to V_{in0} is relatively small and V_{in0} is relatively high. On the other hand, if the MTJ is storing a '0', V_{in0} is relatively low. A reference voltage V_{ref0} between high V_{in0} and low V_{in0} is used for comparison and detection of the stored value in MTJ.

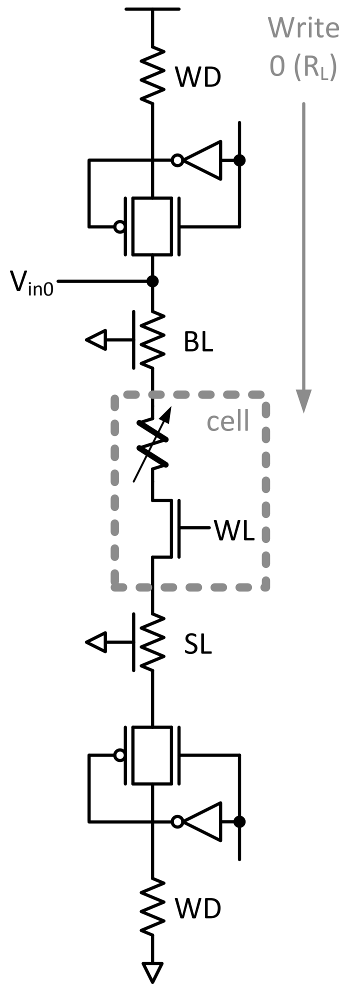


Figure 33: The voltage divider behavior of the write current path.

Because writing 1 and writing 0 are asymmetric, the magnitudes and variations of $\mathrm{V_{in1}}$ in writing 1 is different from that of $\mathrm{V_{in0}}$ in writing 0. Therefore another reference voltage $\mathrm{V_{ref1}}$ is needed in case of writing 1. The $\mathrm{V_{in0}}/\mathrm{V_{in1}}$, $\mathrm{V_{ref0}}/\mathrm{V_{ref1}}$, and comparator outputs $\mathrm{V_{out+}}/\mathrm{V_{out\text{-}}}$ are configured by the new value to be written through the multiplexers, so that the comparator circuit is reused. Table 3 summarizes signal conditions and induced actions in all write cases.

Table 3: Signal conditions and induced actions in different write cases.

Old value	New value	Comparator			col-EN	Action
		input condition	$\mathrm{V_{out+}}$	$\mathrm{V_{out\text{-}}}$		
0 ($\mathrm{R_L}$)	0	$\mathrm{V_{in0}} < \mathrm{V_{ref0}}$	0	1	0	cut off
1 ($\mathrm{R_H}$)	0	$\mathrm{V_{in0}} > \mathrm{V_{ref0}}$	1	0	1	continue
0 ($\mathrm{R_L}$)	1	$\mathrm{V_{in1}} < \mathrm{V_{ref1}}$	0	1	1	continue
1 ($\mathrm{R_H}$)	1	$\mathrm{V_{in1}} > \mathrm{V_{ref1}}$	1	0	0	cut off

Notice that Figure 32 is a simplified illustration not showing the multiplexing of the added EWT circuits. Because EWT functionality is added on a per-write-driver basis, it can also be shared by multiple columns through column multiplexer, similar to the write driver. In each column, the EWT multiplexer (not shown in Figure 32) simply consists of two pMOS transistors that respectively connect $\mathrm{V_{in0}}$ and $\mathrm{V_{in1}}$ to the input multiplexer (shown) of the comparator. They are co-located with the existing column multiplexer (shown) and controlled by $\overline{\text{col-EN}}$. Also, one AND gate is added to the comparator output for each column that shares the EWT circuit (only one shown in Figure 32). And it is controlled by the comparator output together with the col-sel signal of that column.

The comparator circuit, shown in Figure 34, comes from the generic comparator commonly used in many mixed-signal designs like A/D converters [64]. This topology consists of a differential amplifier part (Ni1, Ni2 and Nb) to sense the input voltage difference, a cross-coupled latch (Pr1, Pr2, Nr1 and Nr2) using positive feedback for full-rail regeneration/amplification, and precharge transistors (Pc1 $\sim$ Pc4) to reset voltages of output and internal nodes. Because input voltages $\mathrm{V_{in0}}/\mathrm{V_{in1}}$ and $\mathrm{V_{ref0}}/\mathrm{V_{ref1}}$ are relatively high (close to V_{DD}), high V_{th} transistors are used as Ni1 and Ni2 such that they can be biased in their high-gain region by the inputs. High V_{th} also means relatively small V_{th} variation, which greatly helps in the matching of Ni1 and Ni2 such that the comparator is more accurate and more immune to offset voltage and variations.

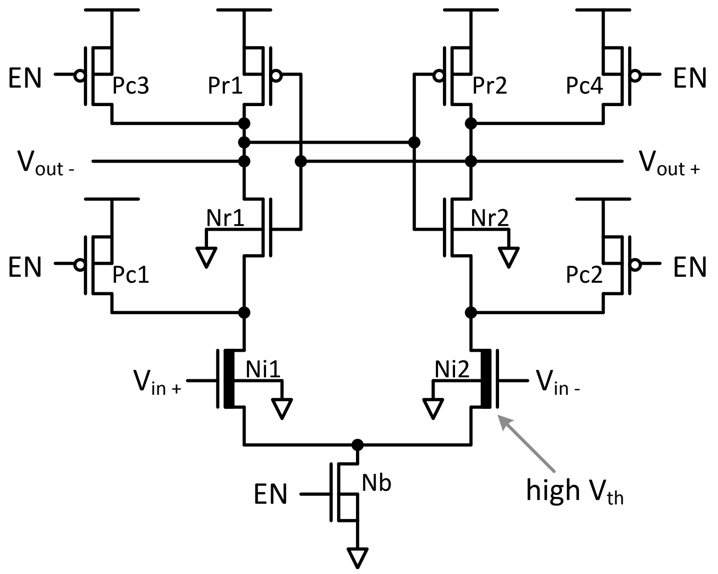


Figure 34: Comparator circuit.

In my SPICE-level simulation, an inactive col-EN can be generated in 500ps after the wordline is selected. Therefore, redundant bit-writes can be detected and throttled at a very early stage.

4.3.3 Overhead

I implemented the EWT circuits and simulated them at SPICE level using 45nm technology. I measured the additional energy introduced by EWT circuits, including the comparator, the multiplexers, and the AND gate. These components are added on a per-write-driver basis. The energy overhead is 74.4fJ per bit-write. This is negligible comparing to the pJ-level per-bit write energy. The estimated area introduced by EWT circuit is about 8.96μm^2 per write driver. The additional circuits to generate reference voltages incur very small area overhead because they are shared by many EWT circuits. Given that there are 16 banks and each bank has 512 write drivers (64B cache line size), the total area overhead is about 73400.32μm^2 which contributes to <1% area increase of a 16MB STT-MRAM cache (estimated by CACTI [84]).

Since EWT is carried within a write operation, there is no performance overhead to the write latency. On the contrary, some write requests can even finish earlier if all the bits are the same as what have been already stored in the cache. This leads to a slight performance gain, which will be seen in Section 4.4.

4.4 EVALUATION RESULTS

4.4.1 Modeling

To measure how much energy saving we can achieve through EWT design, I modeled an STT-MRAM L2 cache in both performance and energy, and then compared it to a baseline STT-MRAM without EWT. The same hybrid approach in Differential Write modeling was adopted: the core circuits of STT-MRAM such as the cells, bitlines, wordlines, read/write circuits etc. were simulated at SPICE level, and the SPICE results were then combined with CACTI [84] estimation of peripheral circuits like decoders, interconnects, data buffers and I/O drivers. All SPICE simulations were performed with $45nm$ device models at 90°C. The numbers produced by this model, as listed in Table 4 and 5, were then applied in architectural simulations.

Table 4: Per-access read/write energy.

	Read	Write (Base)	Write (EWT)
Peripheral	$0.192nJ$	$0.203nJ$	$0.203nJ$
Overhead	–	–	$0.0457nJ$
Cells	$0.013nJ$	$1.417nJ$	E_{change}=$2.767pJ$ $E_{unchange}$=$0.148pJ$
Total	$0.205nJ$	$1.620nJ$	Variable

The breakdown of dynamic energy for reads and writes are shown in Table 4. For dynamic cell energies, we referred to the results in [55], and scaled them to $45nm$ technology node. When EWT is enabled, the write energy is no longer a fixed value. Instead, it is the sum of three parts: peripheral energy $E_{peripheral}$, overhead energy $E_{overhead}$ and a varying cell energy E_{cells} due to value changes:

$$E_{EWT-write} = E_{peripheral} + E_{overhead} + E_{cells}$$

$E_{peripheral}$ is the energy consumed by the peripheral circuits. This is $0.203nJ$, same as in baseline. $E_{overhead}$ is the energy consumed by the EWT circuits. This part is $0.0457nJ$ per write access, calculated as per bit-write overhead multiplied by the number of bits in a cache line (512 in our case) since there is one set of EWT circuit per write driver. E_{cells} is the energy required by those cells that are updated. This variable part depends on how many cells are actually changed in a write request. It can be expressed as:

$$E_{cells} = N_{change} \times E_{change} + N_{unchange} \times E_{unchange}$$

Where E_{change} is the energy used to change one cell, which is $2.767pJ$ in our model. This was obtained by scaling the results in [55] to $45nm$ technology node. Write operations on unchanged cells are terminated at the end of $0.536ns$, which amount to $0.148pJ$ per cell for $E_{unchange}$. In summary, per-access write energy with EWT can be expressed as:

$$E_{EWT-write} = E_{peripheral} + E_{overhead} + N_{changed} \times 2.767pJ + N_{unchanged} \times 0.148pJ$$

Table 5: Per-access read/write latency.

	Read (ns)	Write / EWT (ns)
H-tree in	2.010	2.010
Word-line + Decoder	0.544	0.544
Bit-line	0.800	–
Sense-amp	1.006	–
H-tree out	1.872	–
Write Pulse	–	10 / 0.5
Total	6.232	12.554 / 3.054

The breakdown of read and write component latencies are listed in Table 5. The latencies are rounded to CPU cycles when used in architecture simulator. We referred to a previous work on STT-MRAM cache [55] for STT-MRAM cell latencies. For read operation, this is essentially the sense amplifier delay, which is assumed to be 20% slower than the sense amplifier of SRAM [55]. For write operations, a $10ns$ pulse width was used. However, if the entire write access (a cache line) is throttled, the write latency equals to the time required for redundancy detection which is $0.536ns$, as measured from SPICE simulation. Therefore, a write with EWT may take shorter time than in the baseline.

4.4.2 Energy and Performance Results

The write energy and read energy are combined together to evaluate saving in total dynamic energy. Figure 35 shows the measured results in each workload, normalized to the baseline. With EWT, up to 80% reduction of write energy is observed. Among all 17 workloads, 14 of them get more than 60% reduction of write energy. Even for workloads with lower bit-write redundancy such as `mgrid`, `sphinx3` and `swim`, EWT still achieves 40%∼60% savings. As write energy contributes to more than 70% of total dynamic energy in baseline, applying EWT leads to significant reduction in total dynamic energy (52% on average).

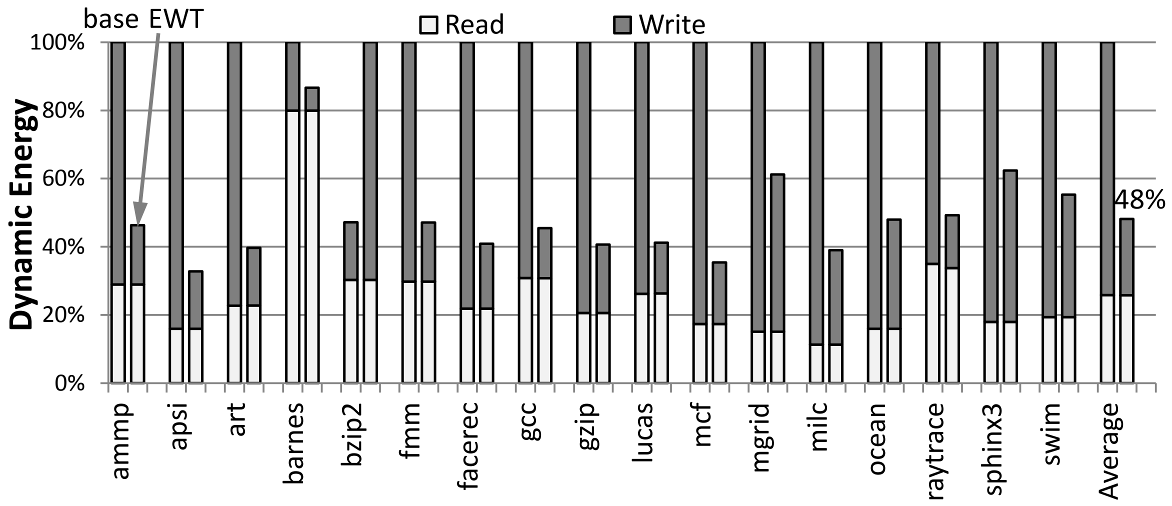


Figure 35: Dynamic energy breakdown showing energy savings of EWT.

As discussed previously, EWT does not introduce any performance penalty to cache accesses. Instead, write requests may finish early if no bit change is needed. Therefore, EWT can reduce average write latency and the contention on cache banks, resulting in slight improvement in performance. Figure 36 shows the results in Cycles Per Instruction (CPI), normalized to the baseline. 3%∼7% of CPI improvements were observed in memory intensive workloads such as `mcf`, `art`, `lucas`, and the average CPI improvement over all workloads is 1%.

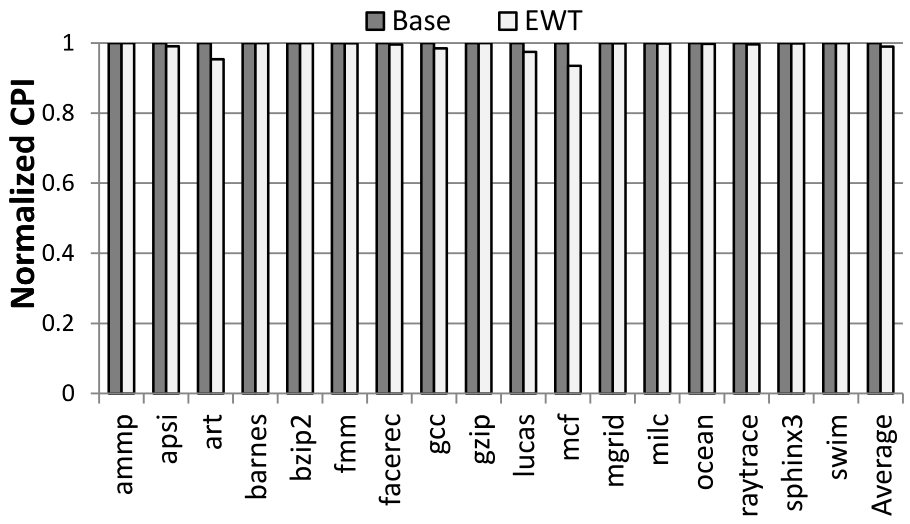


Figure 36: Performance improvements.

5.0 ARRAY LEVEL THROUGHPUT IMPROVEMENT FOR PCM

As discussed in Chapter 1 and Section 3.5, the proposed PCM bit-level solution, Differential Write, can effectively remove large portion of bit-writes and thus provides power headroom for throughput improvement. However, the throughput of PCM is also tightly bounded by another two factors: the long write latency and limited number of read/write circuits. Because multi-port is not an option in PCM cell engineering, array level becomes the lowest level for searching for a throughput enhancement solution.

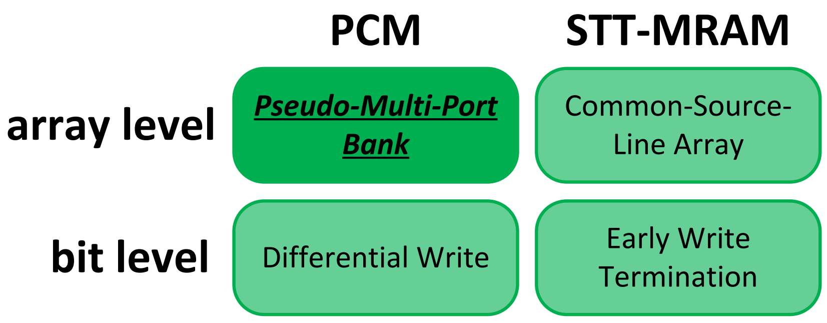


Figure 37: Array level solution for PCM: Pseudo-Multi-Port Bank.

In this chapter, I propose a *Pseudo-Multi-Port Bank* design to exploit the intra-bank or sub-array level parallelism [78]. It allows all of its arrays to operate relatively independently by leveraging the hierarchical wordline and bitline architectures, so that the bank can, instead of accommodating one access at a time, serve 2 writes and 2 reads simultaneously. The goal is to create more request parallelism and fully utilize the limited number of read and write circuits, without dividing a bank for the purpose of preserving cell density which is critical in PCM designs [9]. Next, I show that such bank design opens up potential for novel memory scheduling enhancements to fully take advantage of intra-bank parallelism and further improve throughput. Additionally, my design is cell-type independent and is applicable to SLC/MLC and MOS/BTJ/diode-selected PCM cells, or other high-density new memory technologies that suffer from low throughput in the similar manner. Experiments show that our novel bank design plus simple scheduling enhancement can improve throughput by 58% on average over a baseline PCM design.

5.1 THROUGHPUT PROBLEM OF PCM

Besides write energy and endurance, one major obstacle for PCM to replace DRAM is another write induced challenge – low write throughput. For example, a state-of-the-art PCM chip can achieve 40MB/s program throughput [9], while that of even an old DDR2-800 DRAM is 100MB/s per chip. Though many efforts have been spent on write energy and endurance problems, the throughput problem largely remains untouched. Because of its device and circuit level origins, throughput problem of PCM cannot be tackled by architectural techniques directly. A DRAM/PCM hybrid design [41] can only improve the overall throughput of the entire hybrid memory system, but not PCM throughput. The write-cancellation and write-pause techniques [43] can only help in improving read latency, but not memory throughput because writes and reads are still exclusive to each other and must be served in serial. On the other hand, at circuit level, a multi-port cell is not feasible or favorable because 1) it is very hard to guarantee the isolation between ports using a compact cell structure; 2) each port needs its dedicated set of peripheral circuits that will significantly hurt density.

Due to the unique characteristics of PCM, its throughput is mainly constrained by the following three factors:

(1) **High write power**. Given the large write power per bit, large number of concurrent bit-writes can raise concerns of voltage droop and power supply noise [14]. Hence the number of cells written in parallel has to be restricted [26], which is already constrained by the chip power budget.

Fortunately, as demonstrated in Chapter 3, our bit level solution Differential Write technique [75, 31] can effectively remove large portion of bit-writes and thus provide power headroom for throughput improvement [78]. Similarly, the Flit-n-Write technique [8] based on our Differential Write is another effective solution [14]. Moreover, good scalability of phase change materials implies that aggressive write current scaling can be expected [7, 9]. Furthermore, main memory interfaces like DDR generally feature much larger power capability than stand-alone Flash interface. Therefore, the write power/energy factor is less a concern here.

(2) **Long bank occupation time** as a result of long write latency. PCM's write operation is not only slow but much slower than its read operation, which is determined by the device characteristics. A typical set (crystallizing) procedure takes at least 120ns [7], and the write latency of multi-level PCM is much worse due to the multi-iteration program-and-verify procedure [4]. Moreover, because of the limited number of concurrent bit-writes (due to both factor (1) and (3)), writing a line (e.g. 512 bits) is usually completed in several iterations, with each iteration writing part of the line [26, 29], incuring $\sim$1000ns page write latency [85].

When a memory bank is serving a write for a long time, no other operations can be performed in this bank. In other words, such long writes increase memory bank occupation time and thus block the subsequent accesses. This is especially harmful for system performance if subsequent reads, which are on critical path of the CPU, are blocked and the effective read latencies are significantly increased.

(3) **Limited number of read/write circuits** due to the large sizes of SA/WD *vs.* small size of cell. PCM cell is quite small and scaled agressively during the last several year, thanks to the reduced write current from improved phase change material and the corresponding shrinking and evolution/migration of access devices. On contrast, read and write circuits (sense amplifier (SA) and write driver (WD)) of PCM are usually quite large as a result of high-current, high-voltage operations and complex control. Although write current kept scaling down, write voltage rised with increased parasitic resistances (due to narrower local bitlines to accommodate smaller cells, and longer global bitlines for better area efficiency [9]) and increased access device threshold (especially with diode-

selected cell [29]). Such a conflict between cell size and SA/WD size is represented in two dimensions:

First, in the row dimension, a limited number of WDs and SAs are placed to pitch-match the large number of cells in a row. For example, 16∼32 WDs and SAs can be accommodated within the width of 1024 cells [26, 29]. Therefore, PCM chips usually employ high degree of column multiplexing through two levels (global and local) of multiplexers.

Second, in the array or column dimension, WDs and SAs are usually globally placed and shared among many arrays in the same bank/chip [7, 26, 29, 4, 62, 47, 14, 9]. This is because equipping each array (or every two arrays) with its dedicated set of WDs and SAs, like in SRAM and DRAM, is simply too expensive for density and cost-effectiveness. Too many peripheral circuits lead to low *area efficiency* and thus low effective density, offsetting the advantage of small cell size [9].

Therefore, it is quite common that a high capacity PCM chip possesses very limited number of WDs and SAs, which ultimately constrains achievable throughput.

5.2 PSEUDO-MULTI-PORT BANK DESIGN

As discussed above, the bit level factor, (1) high write power, is less a concern for PCM throughput problem. Therefore, my Pseudo-Multi-Port Bank design targets the two array level factors: (2) long bank occupation time and (3) limited number of read/write circuits. For these two factors, the goals of my design are to provide intra-bank parallelism so writes and reads do not block each other, and to fully utilize these limited resources of read/write circuits to support such parallelism. It is based on the potential that when writing a bank, only the write circuits are occupied and the read circuits are idle. If circuit modifications are developed to allocate the read circuits for a parallel read operation, the design goals can be achieved.

5.2.1 Overview

Our PCM chip and physical bank organizations follow a prototype from Samsung [26] with minor modifications. The chips are organized into a memory system similar to a typical DRAM DIMM. Figure 38 shows our design of a 2GB PCM memory rank on a standard 64-bit channel. The rank consists of 8 256MB PCM chips, each having 8 32MB banks. In each channel transition, all PCM chips work together to deliver 64 bits of data, with each chip producing 8 bits which are generated from one bank within that chip. Hence, a read request of 64B data will require 8 transitions in 4 channel cycles (DDR interface). I also assume the same signal sequence as a DDR memory interface: row address arrives first, followed by column address and finally read or write command.

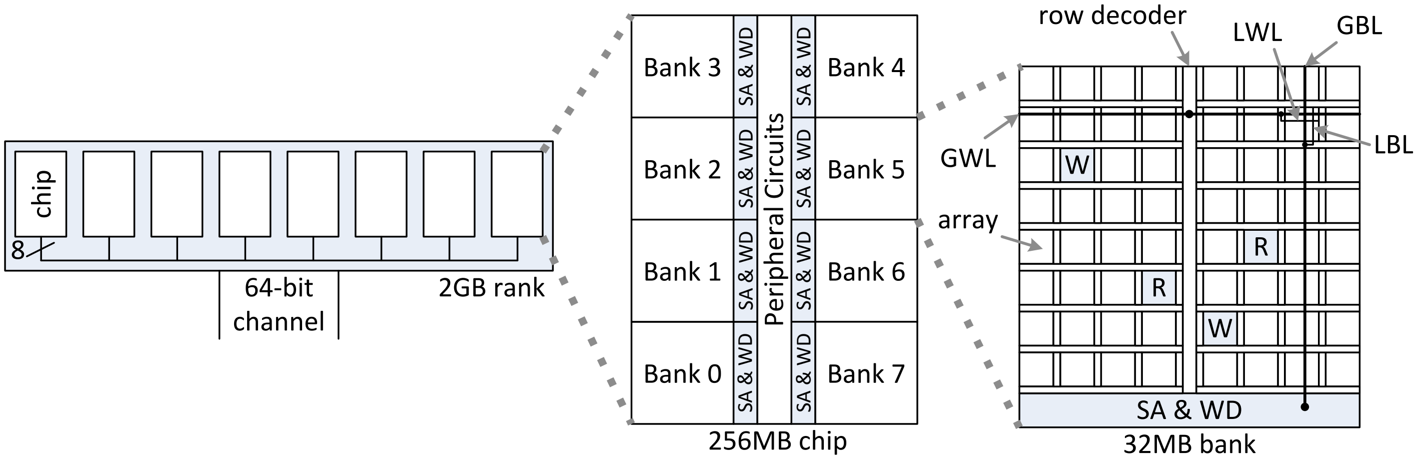


Figure 38: Overview of memory organization showing 4 concurrent accesses in a bank. SA – sense amplifier, WD – write driver, GWL – global wordline, LWL – local wordline, GBL – global bitline, LBL – local bitline. Chip and bank floorplans are from [26]

Inside each PCM bank, the 32MB capacity is divided into 64 4Mb (2048-row × 2048-column) cell arrays. These arrays are partitioned into left and right halves of the bank that can work concurrently. The row decoder, shared by both halves, is in the middle of the bank to avoid long wordline driving. The read and write circuits, sense amplifiers (SA) and write drivers (WD), are at the bottom of the bank and are shared by all the arrays. Because a bank handles 64 bits of data in serving each memory access, 64 sets of SA & WD are placed below each half bank. This many SAs and WDs can fit into the width of 4 arrays (8192 columns) [26, 29].

When a write is performed, the row decoder selects, drives and holds one wordline to open the cells in that row. Proper bitlines corresponding to the address are then selected to start writing. One write always activates only one array in the bank. To make a write non-blocking, another operation is enabled whose activity is in a different array of the bank. This is fundamentally feasible because when a write is in-progress, only the write circuit is occupied but the read circuit is idle. With proper circuit changes, the read circuit can be used to serve a different read in concurrent with the write.

However, there are two challenges to the exploiting of this opportunity in circuit implementation. The first challenge is the sharing of circuit resources. For example, the wordline is held by the row decoder for the entire duration of the write. So when a read request comes, there is no row decoder to use. The decoder needs to be freed in order to decode a second address and drive a second wordline. Even if a second wordline can be activated for read, a second challenge is the interference between write and read. The read wordline will interfere with write because it opens cells at its cross-points with the write's bitlines, and these cells would be mistakenly written. Likewise, the write's wordline will also interfere with the read's bitlines. This is depicted as the two crosses in Figure 39, which will destroy both the write and the read.

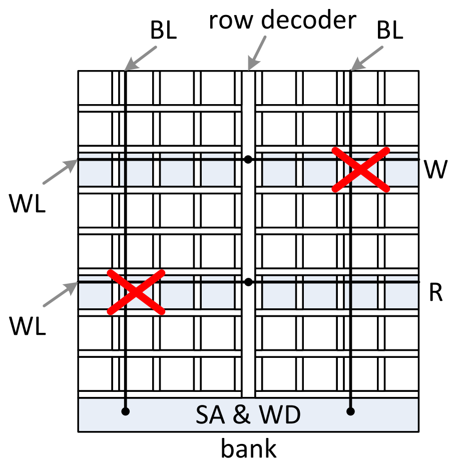


Figure 39: Interference among wordlines and bitlines for write and read.

The first challenge can be addressed by latching the output results of the shared circuits and time-multiplexing them between requests. The second challenge can be addressed by making arrays relatively independent and offloading memory accesses to individual arrays. My circuit implemention leverages and revises the existing hierarchical wordline and bitline architecture, i.e. one global wordline/bitline (GWL/GBL) with local wordlines/bitlines (LWL/LBL) in each array, as shown in Figure 38. For instance, once activated, the LWL signal can be latched locally in each array so it is possible to dismiss the GWL and row decoder for a subsequent access. However, the GBL cannot be released as it connects with the read/write circuit. Such connection must be maintained throughout an access. Hence, two operations that fall within the same column of arrays cannot be performed concurrently because they need to share the same GBLs which are dedicated to one operation at a time.

Therefore, arrays in my bank design are in one of three states, as illustrated in Figure 40. An *idle* array is open for any memory access. An *active* array is busy serving one request. Because the sharing of GBLs is uncompromisable, all other arrays in the same column of an active array must be *disabled* such that they will not support or be affected by another parallel access. Implementation details can be found in the following sections.

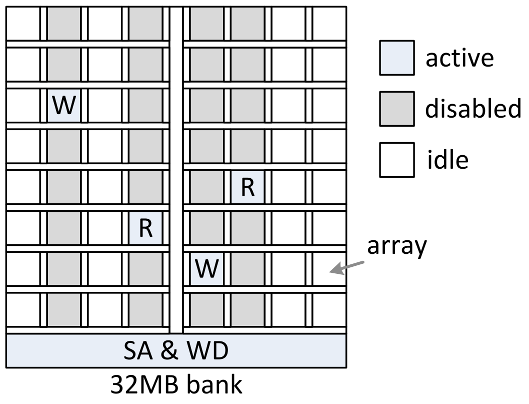


Figure 40: Three array states in a bank.

Notice that the latencies of these shared circuits (e.g. row/column decoders), and therefore the required time to recycle and reuse them, are negligible comparing to the write and read latencies on PCM cells. So the delay expense to enjoy this novel intra-bank parallelism is minimum. Consequently, my novel bank design creates the illusion of the multi-port access, where comes the name of Pseudo-Multi-Port Bank.

Also, I remark that my design is not simply breaking a bank into two (or throwing in more banks) for more parallelism. Those designs would dramatically reduce the PCM density because each bank must be equipped with a new set of peripheral circuits including decoders, drivers, I/O buffers, and address/data/control/power routings. My design is based on a Samsung prototype [26] except that I utilize its idle SAs or WDs for parallelize-able requests, and time-multiplex the shared decoders to enable parallelism. For example, when the left half is serving a write, the WDs below the right half and all SAs in the bank are idle, but the center row decoder is busy. I let free the decoder so that it can serve another read or write and utilize the idle SAs and WDs. As I will show in Section 5.2.4 that my design adds only 5% (conservative) of hardware overhead, which is much more lightweight than dividing a bank, or using more banks to achieve the same parallelism. This is the key advantage of my bank design.

To summarize, each half bank in my design can serve 1 read and 1 write at the same time. Hence a bank can serve up to 2 reads and 2 writes concurrently. There is also hardware limitation on this concurrency: two requests that access arrays on the same column cannot be active at the same time.

5.2.2 Circuit Design

My PCM bank design follows a prototype from Samsung [26] with revised hierarchical GWL/GBL + LWL/LBL control. In those designs, hierarchical organization of wires provides more flexibility to optimizing GBL for low resistance, which is critical to delivering current to cells. I use this organization for implementing parallel accesses within a bank. The GWL/GBL will first be selected, followed by opening LWL/LBL in a local array. Then, further circuit support is needed to hold the LWL for the duration of an access while the GWL can be released for a new access. Next, I elaborate my hierarchical GWL/GBL + LWL/LBL control based on a realistic memory organization.

To make each array independently accessible, I place all 64 bits of a data in the same array, instead of distributing them into 4 arrays [26]. Because of such data placement, 64 GBLs are needed per column of arrays, as depicted in Figure 41. Such number of GBLs is physically feasible [29] with no area occupation as GBLs traverse over the arrays. The 64 GBLs are connected to 2048 LBLs through local column multiplexer (LCM), and the global column multiplexer (GCM) switches 64 SAs and WDs between four groups of 64 GBLs. The GCM actually contains two separate multiplexers for SA and WD respectively [29].

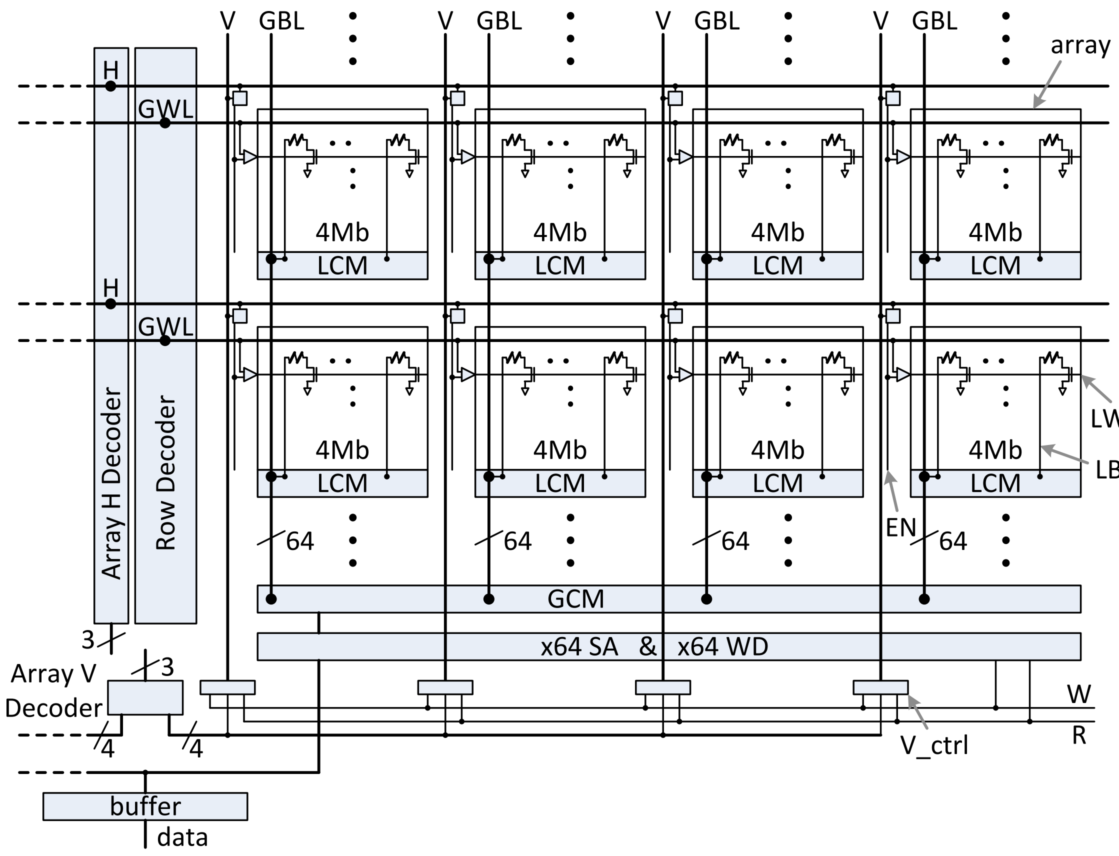


Figure 41: Architecture of Pseudo-Multi-Port Bank design showing the right-half bank.

Recall that the first challenge in my design is to release the shared circuits, e.g. a row decoder, after they generate their outputs. I first introduce a set of horizontal (H) and vertical (V) signals as shown in Figure 41. The intersection of an activated H and V opens an array and generates the corresponding enable signal (EN) for intra-array use. Since array and thus H and V signals are arranged in an 8×8 manner, two 3:8 decoders are sufficient for locating an array. When a row address arrives at the decoder, both GWL and H are driven high. One V wire is then selected by the array V decoder when the column address arrives. This V signal then closes the latch in the intersection of V and H (Figure 42), which latches H internally. Hence, the array H decoder can be freed. Meanwhile, the EN together with the selected GWL activates one LWL, and at the same time closes the pass-gate in every LWL driver in the active array (Figure 43), which latches GWL internally. Hence, the row decoder can be freed now.

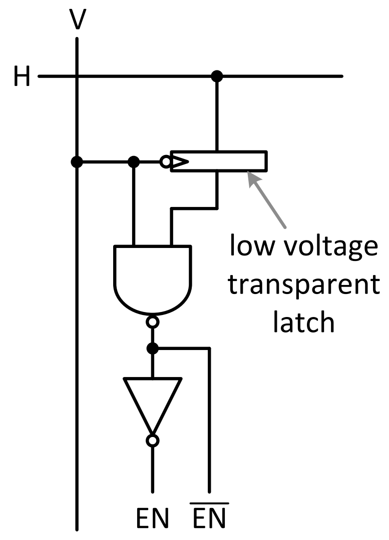


Figure 42: Array enable circuit.

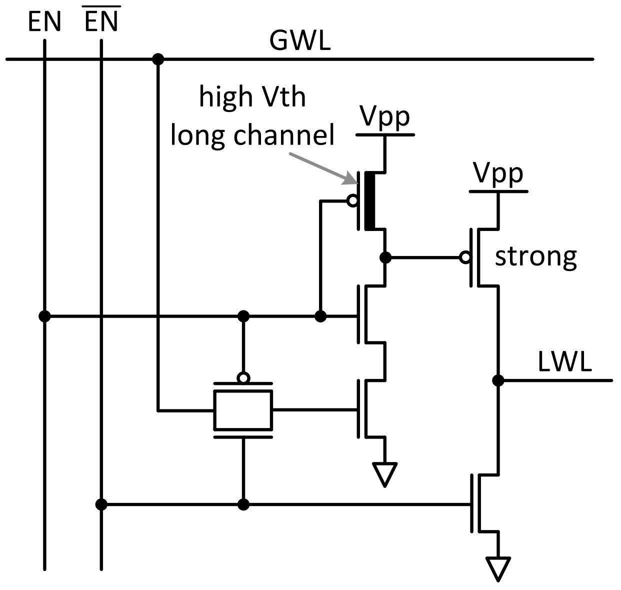


Figure 43: LWL driver.

The second challenge is to prevent the interference between the active write and read in their wordline and bitline cross-points. This challenge can be addressed by closing all arrays in the column so that they will not be affected by signals of a new request. Since an active V signal closes the latch in the array enable circuit as shown in Figure 42, any further changes on H will not be seen as long as the V is active. Hence, V signal in fact locks the selection states of all arrays in the same column. In the active array, the active EN signal held by

V closes the pass-gates, so the LWL drivers will not see changes on GWLs. In disabled arrays, the inactive EN signal, also held by V, makes LWL drivers not responsive to GWLs. Furthermore, after being generated by the array V decoder, the V signal holding the entire column is in turn held by the write/read enable signals, shown as W and R in Figure 41, through the V_{ctrl} circuit shown in Figure 44. Thus, the array V decoder can also be freed for another access.

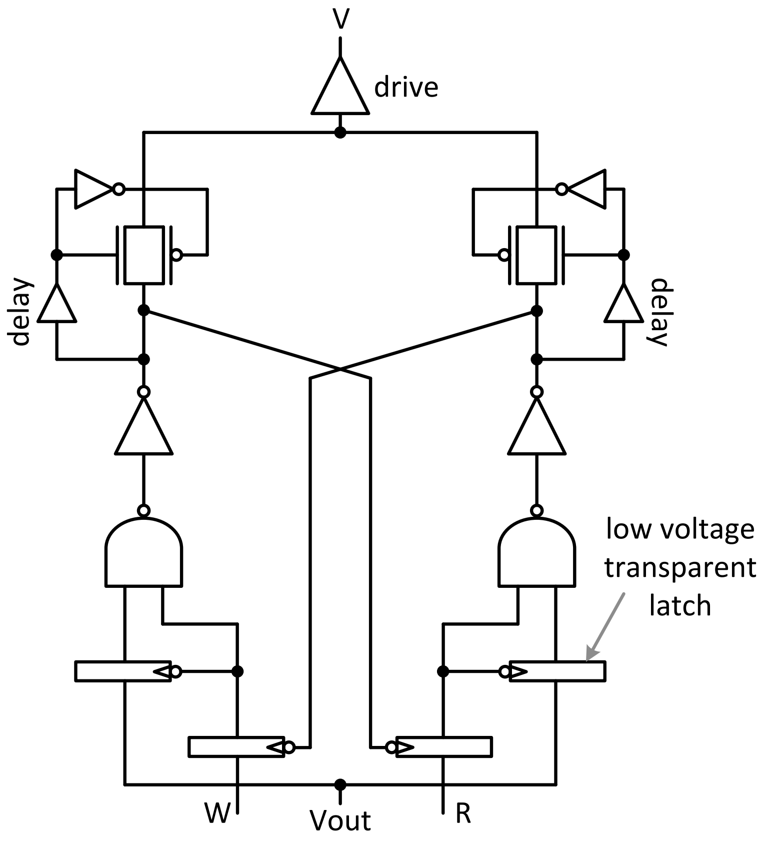


Figure 44: V_{ctrl} circuit to hold the V signal while releasing the array V decoder.

The V_{ctrl} circuit takes inputs from both the array V decoder and a R or W command. If the input from the array V decoder is high (the corresponding V signal should be driven), and either R or W command is high, it activates the V signal and locks itself until the R or W drops low, i.e. the operation finishes. The circuit has a symmetric and cross-locked structure. It ensures that when this V signal is selected, and if one command signal is high, its output will stay constant and not be affected by changes on the other command signal. Through this way, the V_{ctrl} circuit can hold the V signal using W or R command and dismiss the array V decoder for next request.

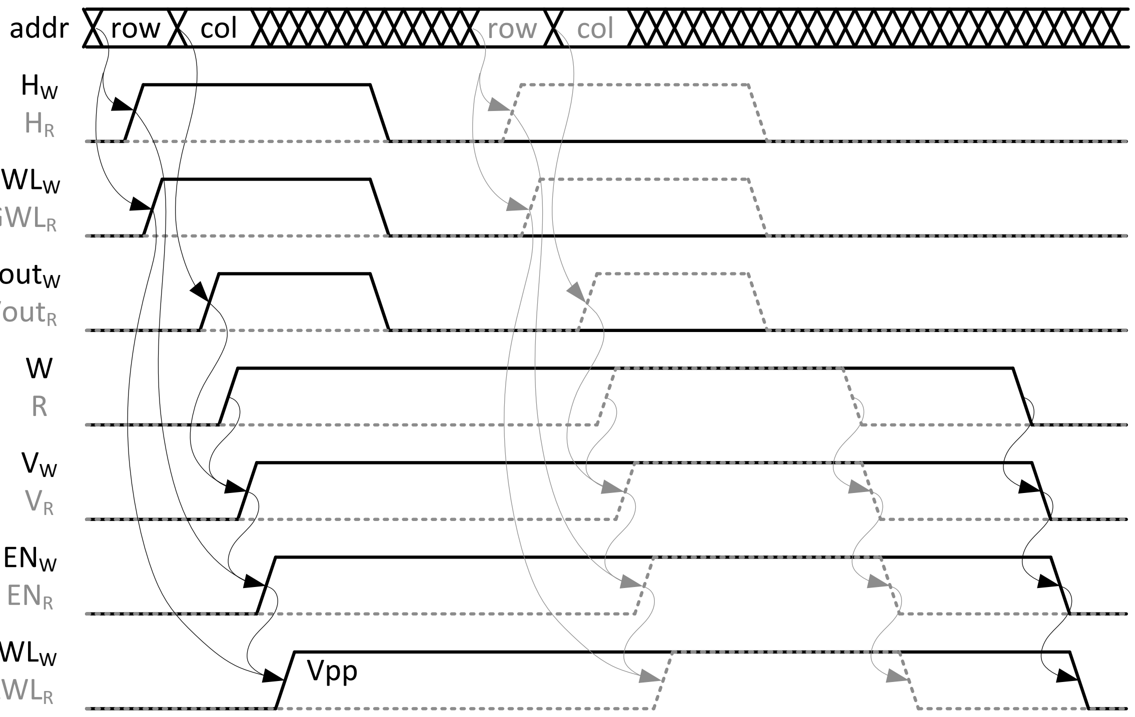
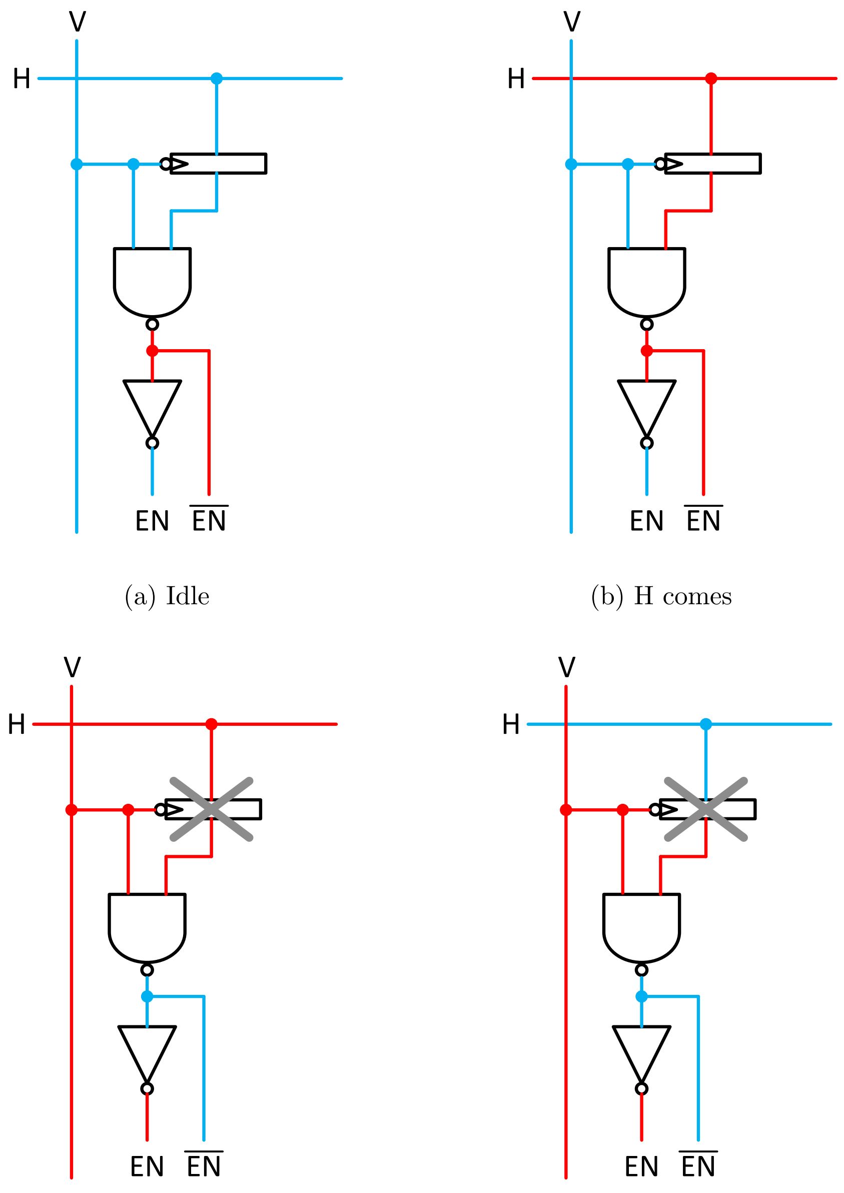


Figure 45: Timing graph of a read carried in parallel with a write (not to scale).

To summarize the procedure of issuing two accesses that overlap in time, I use a timing graph shown in Figure 45 (not to scale) to illustrate the sequence of control signals for a write parallelized with a read. When a write request is issued to PCM, its row address is first decoded by the array H decoder to select and pull up one H signal, H_W. At the same time, the row decoder activates one GWL, GWL_W. When the column address arrives, the array V decoder produces an output $Vout_W$ which makes one V_{ctrl} circuit ready for the upcoming W command. These signals will be discharged soon to take the next read operation. When the write command becomes high, the V_{ctrl} circuit activates a V signal, V_W, which will be held high as long as the W signal remains active. EN_W and LWL_W are then enabled by V_W. When the write is in progress, the dismissed row decoder, array H and V decoders are used by the following read access to generate H_R, GWL_R and $Vout_R$. The read command R then triggers a similar series of signals, all of which occur in a different column of arrays. Since read access is much faster than write access in PCM, R ends sooner leading to a sequential discharge from V_R to LWL_R which turns off the cells being read. If there is another read coming in at this time, the process will repeat again without any problem.

5.2.3 Circuit Component Details



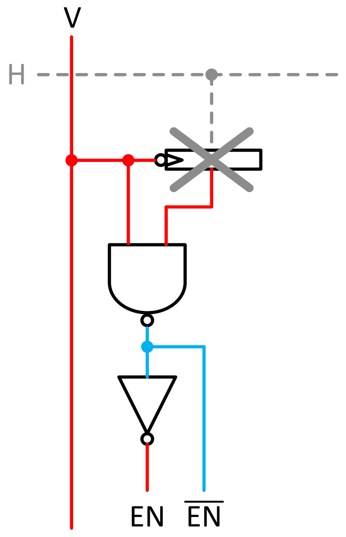
(a) Idle (b) H comes

(c) V comes (d) H goes

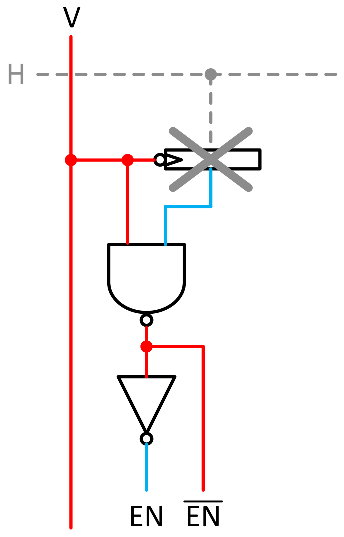
Figure 46: Steps to select/enable an array.

The array enable circuit shown in Figure 42 is to activate or disable an array according to H and V signals. It simply ANDs H and V to generate the intra-array enable signals EN and $\overline{\text{EN}}$. Figure 46 illustrates the steps to enable an array. When idle, the circuit is open for inputs to switch. When H signal comes, it passes through the latch to the NAND gate but the array state remains unchanged. Only when the V signal becomes active, the NAND gate is triggered to flip EN and $\overline{\text{EN}}$, at the same time the latch is turned opaque by V, latching the active H signal on the NAND input. Therefore, H signal and array H decoder can be dismissed.

Figure 47 shows the states of this circuit in different arrays. With an active V, it latches active and inactive H's in active and disabled arrays respectively, and therefore holding the enable signals. As a result, any further changes on H will not be seen as long as the V is active. Hence, V signal in fact locks the selection states of all arrays in the same column.



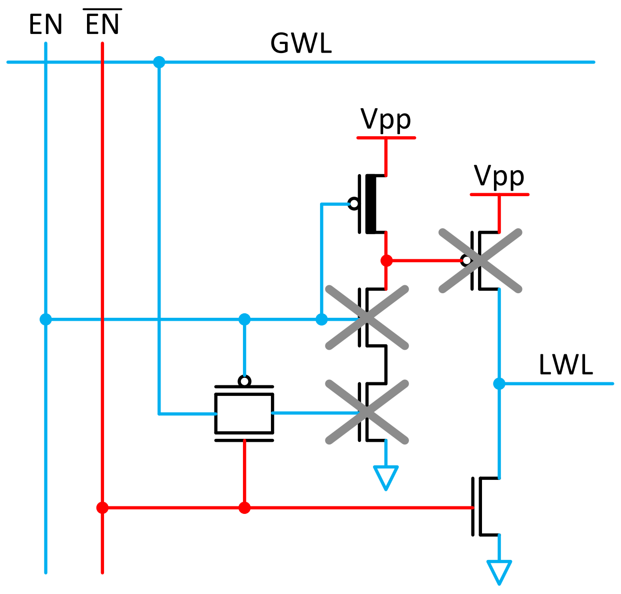
(a) In active array



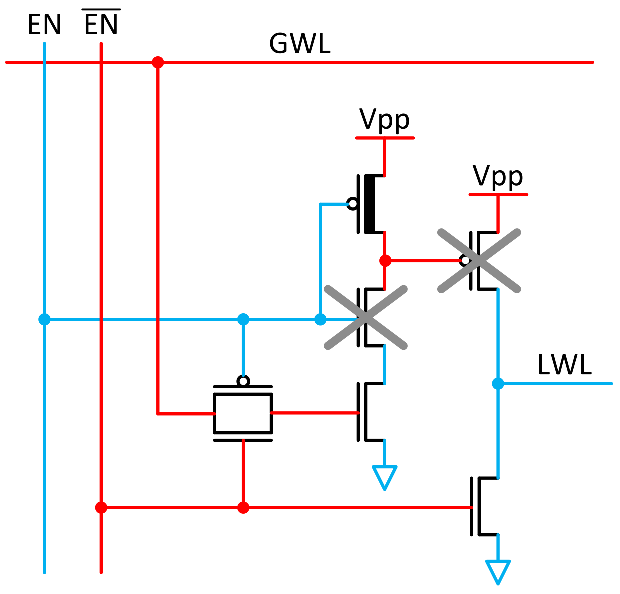
(b) In disabled array

Figure 47: States of array enable circuits in different arrays.

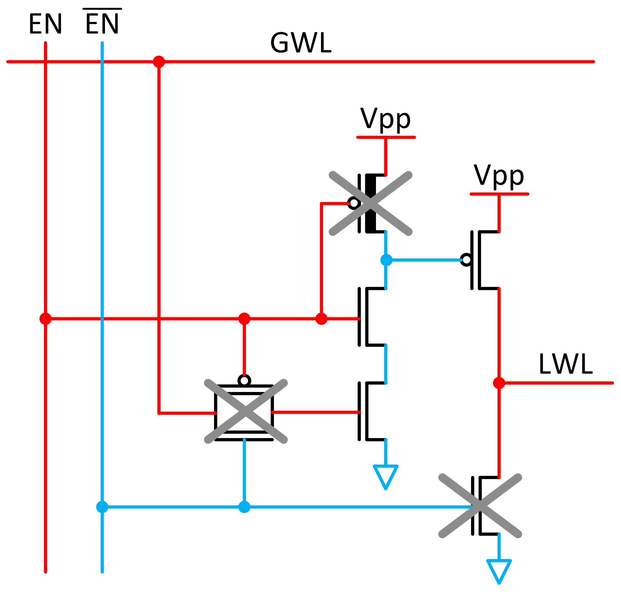
The LWL driver shown in Figure 43 is to activate LWL according to GWL and array enable signals. Figure 48 illustrates the steps to enable a LWL. When idle, the circuit is open for inputs to switch. When GWL comes, it passes through the pass-gate and the nMOS it controls is turned on, but the LWL remains grounded. Only when the array becomes active, the driver is triggered to charge LWL, at the same time the pass-gate is turned off by EN and $\overline{\text{EN}}$, latching the active GWL internally. Therefore, GWL and row decoder can be dismissed. Once the array is deactivated, i.e. EN and $\overline{\text{EN}}$ are turned off by array enable circuit when V signal is pulled down, it discharges LWL and becomes available to inputs again.



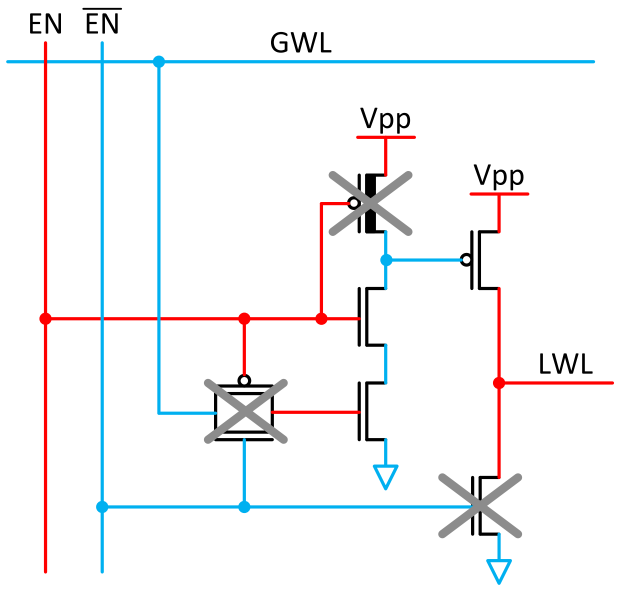
(a) Idle



(b) GWL comes



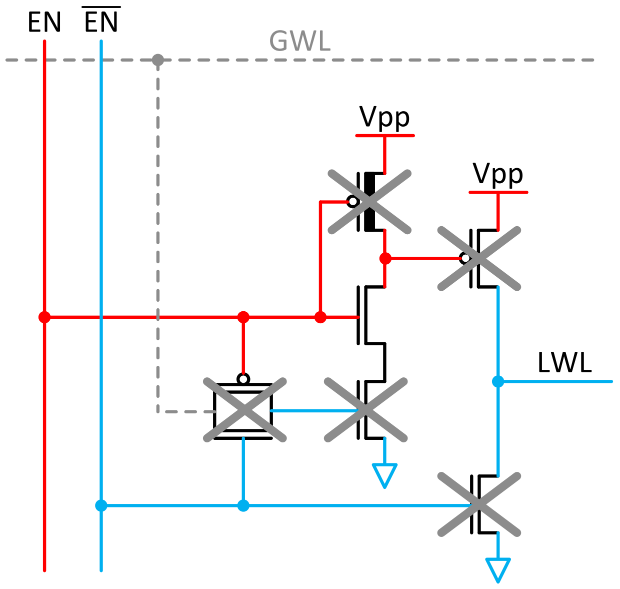
(c) Array enable



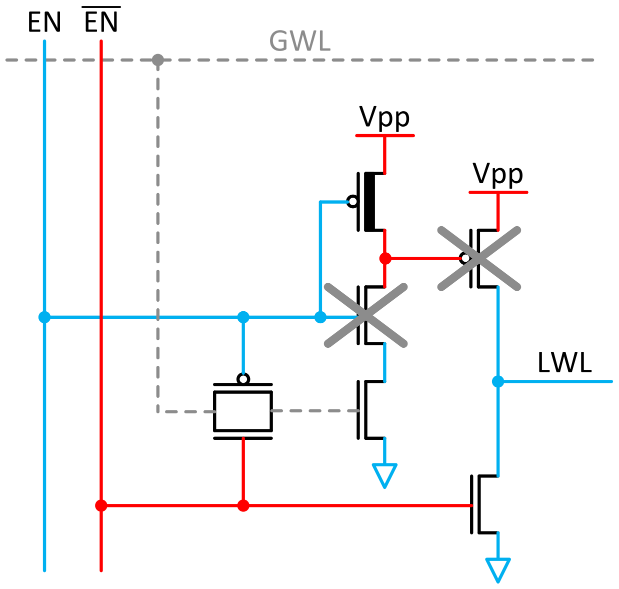
(d) GWL goes

Figure 48: Steps to activate a LWL.

Figure 49 shows the states of this circuit in different arrays. In the active array, the active enable signals held by V turn off the pass-gates, latching active and inactive GWLs for selected and unselected LWLs respectively, and therefore holding the LWLs. As a result, the LWL drivers will not see further changes on GWLs. In disabled arrays, the inactive enable signals, also held by V, make LWL drivers not responsive to GWLs.



(a) Unselected LWL in active array

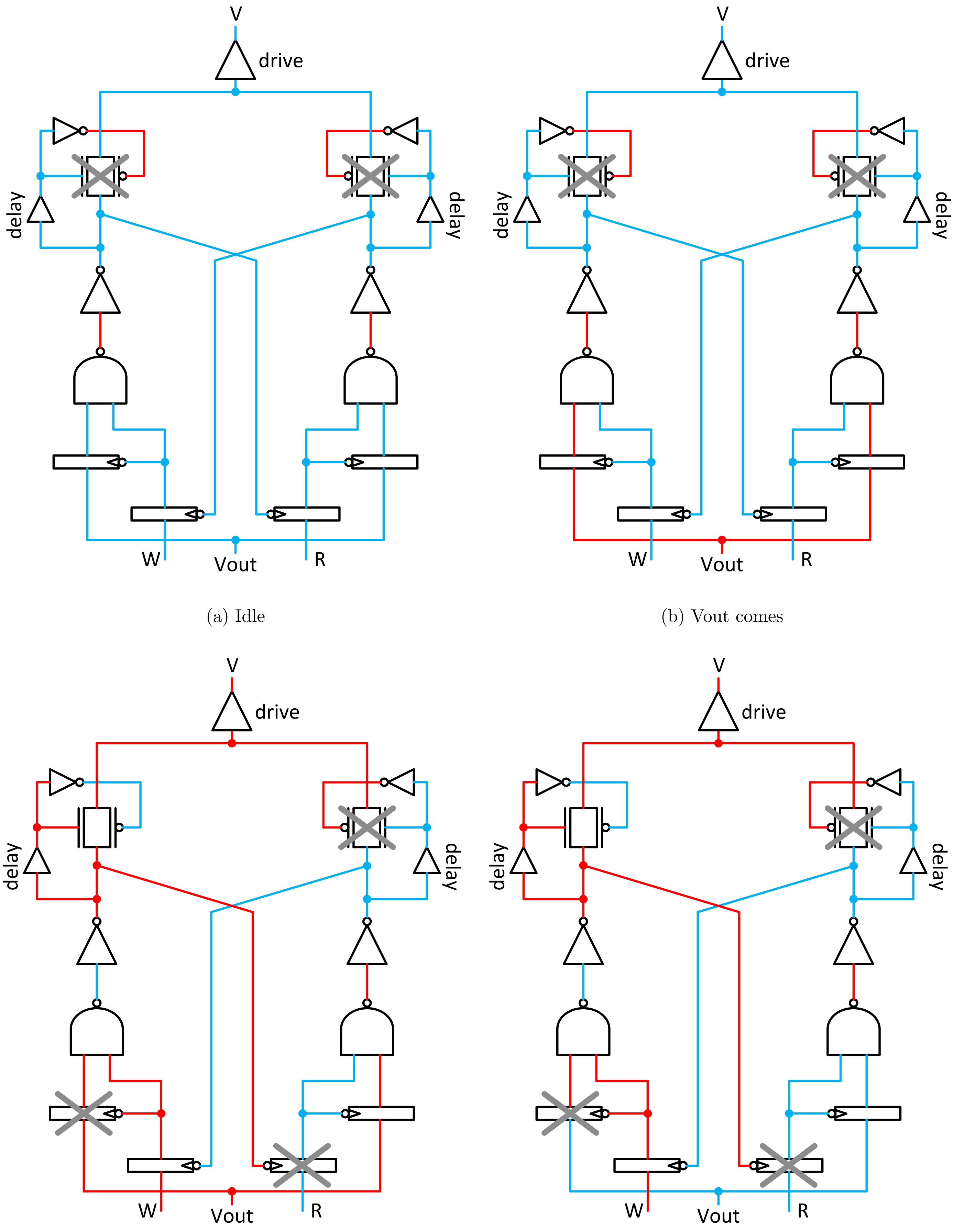


(b) In disabled array

Figure 49: States of LWL drivers in different arrays.

The V_{ctrl} circuit shown in Figure 44 is to activate V signal according to Vout and R or W command. It features a symmetric and cross-locked structure to resolve the control interference between R and W commands. Figure 50 illustrates the steps to enable a V signal for the first access (e.g. write) into an idle bank. When idle, the circuit is open for inputs to switch. When Vout comes, it passes through the latches on both sides of the circuit but neither side is fired. When the W command becomes active, the W side is triggered to drive the output V signal, at the same time the Vout latch is turned opaque by W, latching the active Vout within the W side. Therefore, Vout and array V decoder can be dismissed. Meanwhile, the inactive R command is latched within the R side by the internal signal of the W side. As a result, any further changes on R command and Vout (which is illegal) will not be seen as long as W command, and the write operation, is active.

Figure 51 illustrates the steps to enable another V signal in the same bank to parallelize a second access (e.g. read) with the on-going first write access. Because R and W commands are inputs of all V_{ctrl} circuits, the active W command of the first access already latched the inactive Vout within the W side of the V_{ctrl} circuit in all other columns. Therefore, when the second Vout comes, it only passes through to the R side, without mistakenly firing the



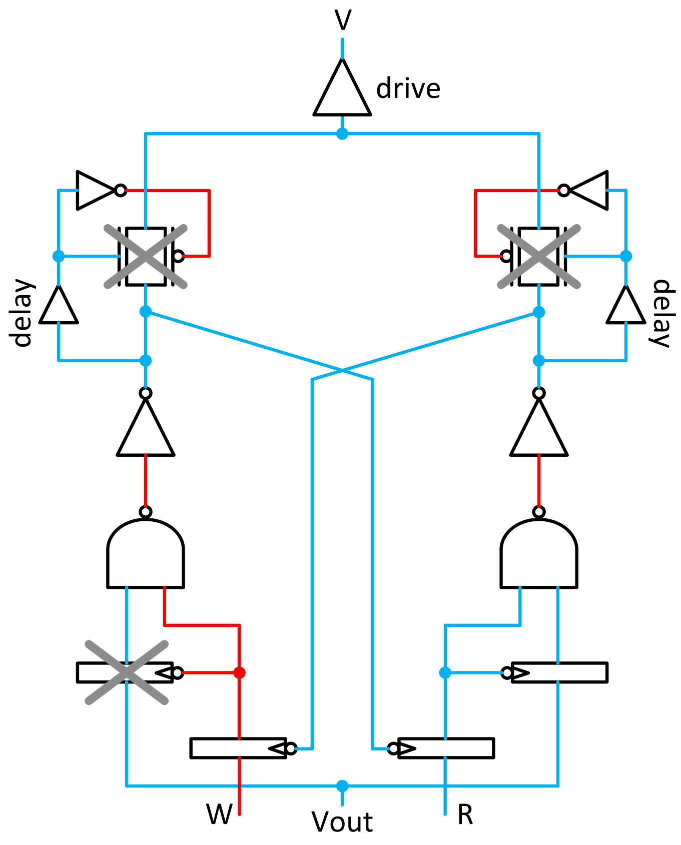
(a) Idle

(b) Vout comes

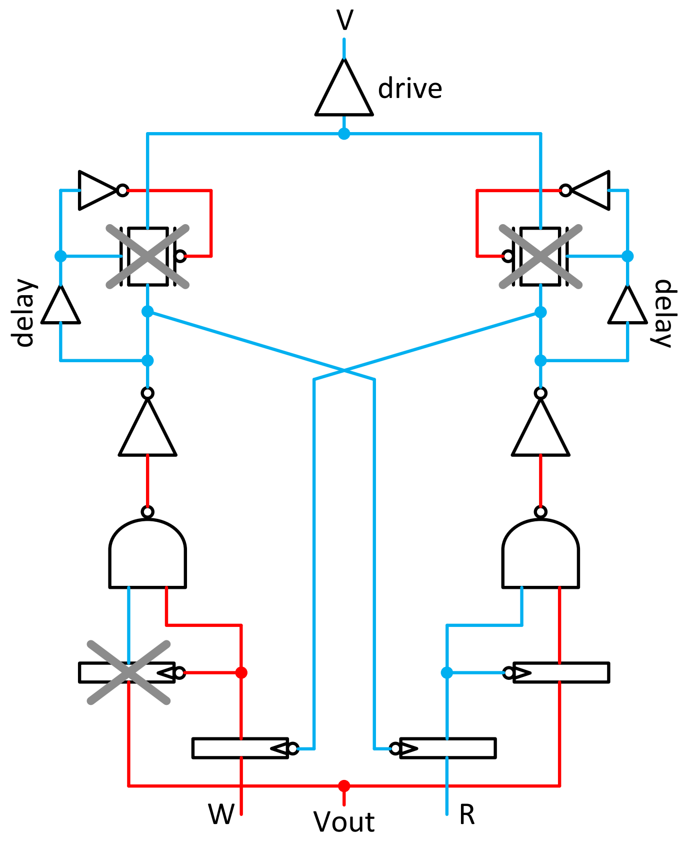
(c) W comes

(d) Vout goes

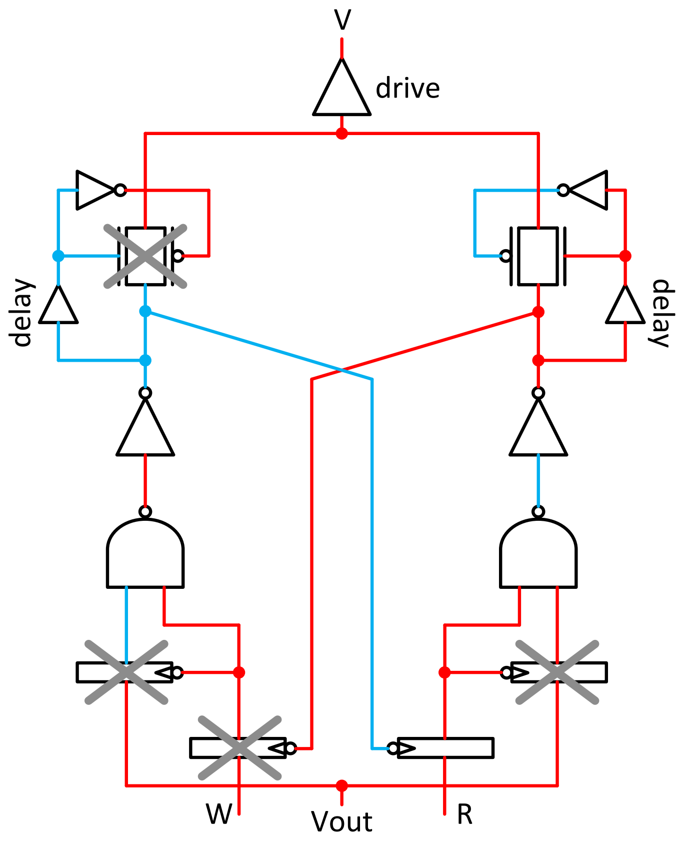
Figure 50: Selecting a V signal for the 1st access (e.g. write).



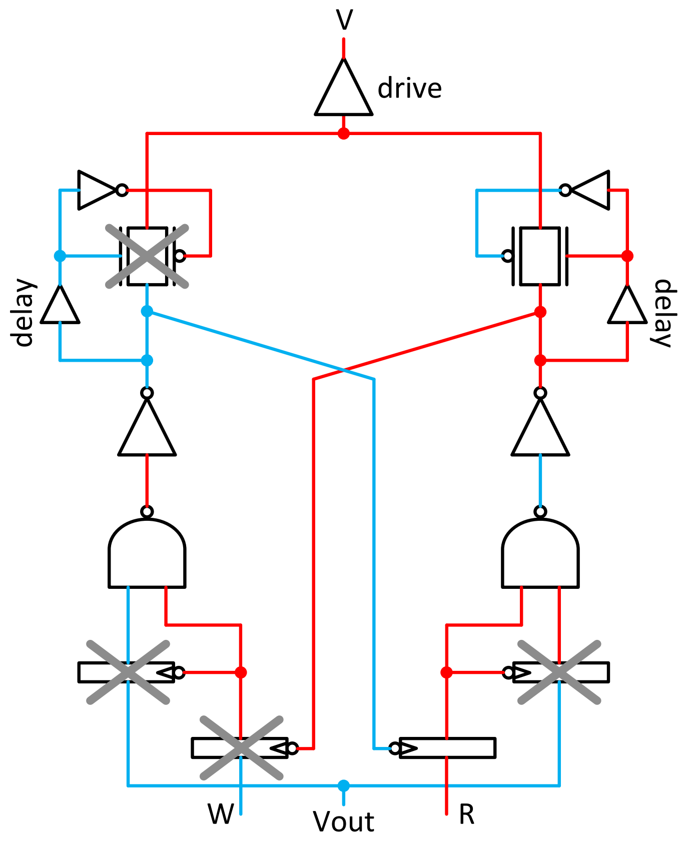
(a) Write is on-going in another column



(b) Vout comes



(c) R comes



(d) Vout goes, write finishes

Figure 51: Selecting a V signal for the 2nd access (e.g. read) in parallel with the 1st access.

W side due to the active W command. Next when the R command becomes active, the R side is triggered to drive the output V signal, at the same time the active Vout is latched within the R side so Vout and array V decoder can be dismissed. Meanwhile, the active W command, together with the already latched inactive Vout, is latched within the W side. As a result, the write access can finish without affecting the read, and any further changes on W command and Vout (which is illegal) will not be seen as long as the read operation is in progress. In summary, this V_{ctrl} circuit is only responsive to the *first* command that activated it.

5.2.4 Overhead

Table 6: Delay, energy and area overheads.

		# in one bank	Delay (ps)	Energy (fJ)	Area (μm^2)		
					Each	In one bank	% of bank
(a)	3:8 decoders	2	80	23.2	25.02	50.04	0.0005
(b)	H signal driver	8	240	499.8	10.62	84.96	0.0008
(c)	V_{ctrl}	8	850	446.8	32.4	259.2	0.0025
(d)	array EN	64	105	250.7	14.13	904.32	0.0087
(e)	LWL driver	64×2048	110	242.5	4	524288	5.03
	TOTAL	–	1145	1486.4	–	525586.52	5.0425

Most parts of my design such as chip organization, array partition, and hierarchical WL/BL are adopted from existing prototypes [26, 29], and thus do not incur overhead. The circuit components I added per PCM bank include: (a) array H and V decoders, which are two 3:8 decoders; (b) H signal driver (×8); (c) V_{ctrl} circuit (Figure 44, ×8); (d) array enable circuit (Figure 42, ×64); (e) LWL driver (Figure 43, 64×2048). Although LWL drivers exist in the prototype I referred to, I still conservatively study its overheads since the design is revised. I built and tested my added circuit components in SPICE with customized 45nm PTM device models [82], and measured their delay, energy and area overheads. For area overhead, I also convert the number into percentage of the whole bank. To obtain the area of a bank, I scaled the dimensions reported in [26] from 100nm to 45nm. I also assume that the H and V wires can traverse over cell arrays and thus do not occupy silicon area. All SPICE simulations were performed with a power supply of 1.2V at 90°C. The results are presented in Table 6.

5.2.5 More Discussions on Circuit Design

LWLs must use a high voltage, denoted as V_{PP} here, to increase the conductivity and thus reduce the effective resistance of the cell access transistor for easier current injection [26]. V_{PP} is boosted from and higher than the supply voltage V_{DD}. In contrast, peripheral circuits and signals that purely provide logic functions, such as row decoders and GWLs, do not need to be operated at higher voltages. Therefore, these two voltage domains interface in the LWL driver circuit which is driven by GWL and EN in V_{DD} domain and drives LWL in V_{PP} domain, as shown in Figure 52.

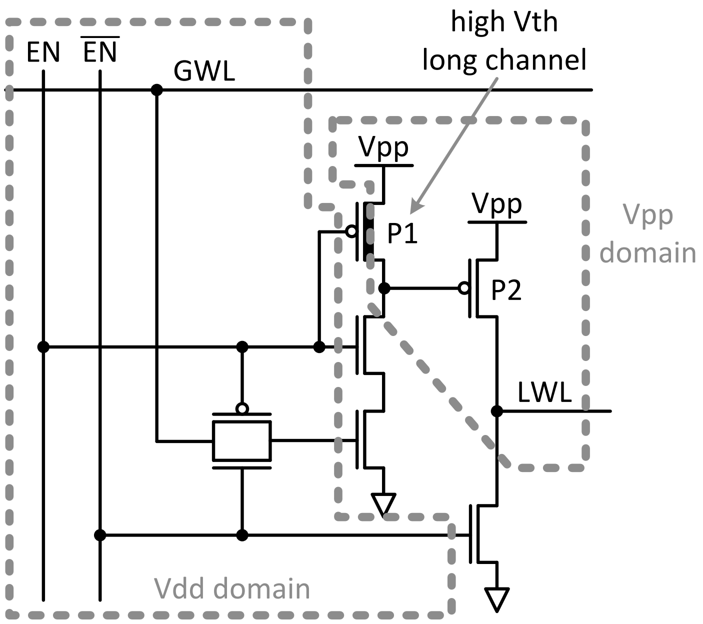


Figure 52: Two voltage domains in a LWL driver.

Unfortunately, this LWL driver suffers from *voltage domain interfacing* problem because signals in the lower V_{DD} domain cannot directly drive pMOS transistors in the higher V_{PP} domain. More specifically, this problem manifest itself when EN signal tries to turn off the pMOS transistor P1, which sits on the domain interface. When EN is at V_{DD}, the pMOS transistor P1 in the V_{PP} domain has V_G=V_{DD}, V_S=V_{PP}, and thus $|V_{GS}| = |V_{DD} - V_{PP}|$. If this exceeds $|V_{th}|$ of P1, the pMOS will turn on and burn contention current. Even if the difference is less than $|V_{th}|$, P1 will suffer substantially increased leakage.

This problem can be alleviated by using a high-V_{th} pMOS device as P1 if the voltage difference between domains is small enough [66]. In addition, increasing its channel length also helps to further suppress leakage, as leakage has exponential dependency on channel length. Nevertheless, high-V_{th} and long channel imply a weak driving strength and may incur extra delay overhead. However, because the load of P1 is only the pMOS P2, such delay penalty is quite limited despite the large size of the strong P2.

Moreover, it is worth noting that this partial-off problem only exists in the LWL driver that drives an active LWL. In such a case, both of the nMOS transistors stacked with P1 are on and the contention current can flow through them to ground. In all other LWL drivers, including the ones driving unselected LWLs in the active array and the ones in disabled and idle arrays, there is at least one nMOS turned off in the stack, thus quenches the leakage burning of P1.

On the other hand, if the voltage difference between V_{PP} and V_{DD} is relatively large, on each EN wire a simple voltage level converter circuit shown in Figure 53 can be used to adapt EN signal from V_{DD} domain to V_{PP} domain (no need to convert $\overline{\text{EN}}$). This completely eliminates the voltage domain interfacing problem, and adding one simple level converter circuit per array is negligible overhead.

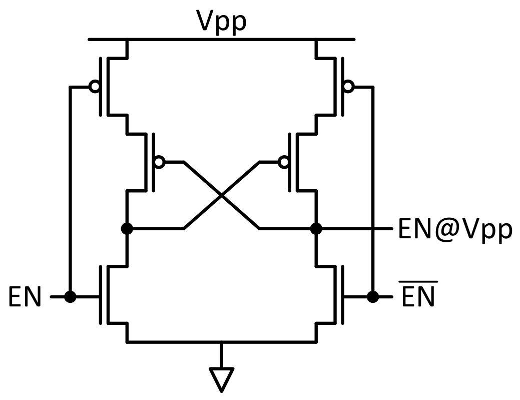


Figure 53: Level converter circuit to adapt EN signal to V_{PP} domain.

Figure 54 shows the LWL driver design for diode-selected PCM cell. In this design, EN signal must be adapted to V_{PP} domain using the level converter circuit shown in Figure 53 because it directly controls the pMOS that drives LWL.

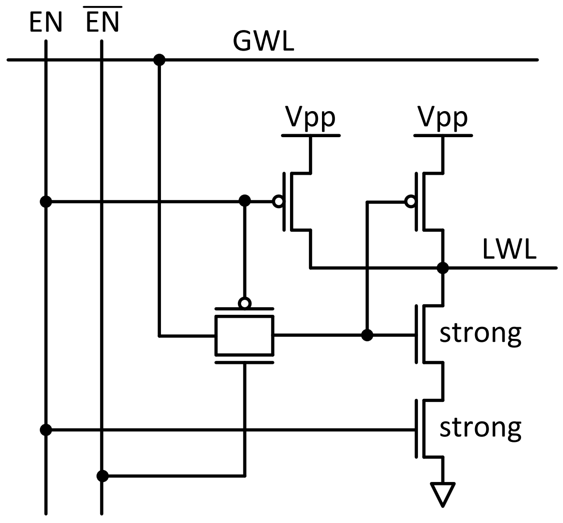
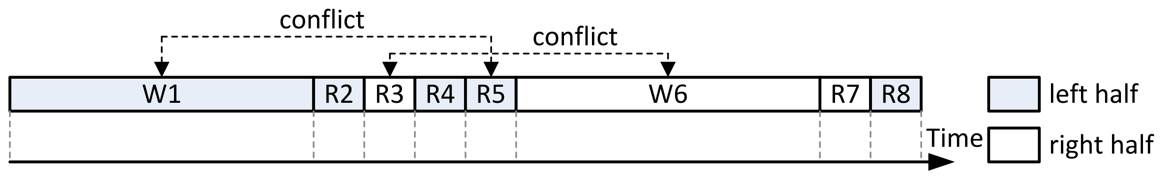


Figure 54: LWL driver design for diode-selected PCM cell.

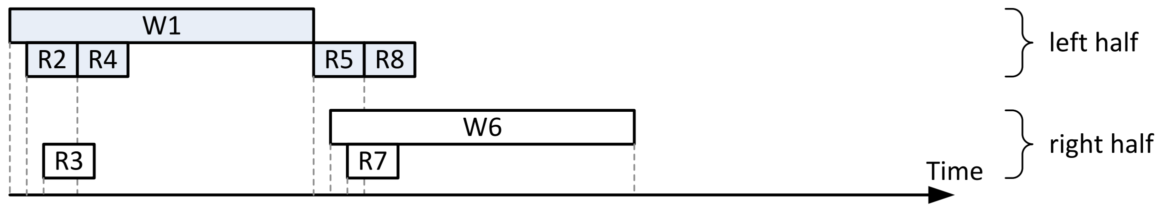
5.3 MEMORY SCHEDULING OPPORTUNITIES

Although my proposed circuit design can provide larger concurrency inside each PCM bank, the final throughput also depends on whether the memory requests issued to the bank can fully take advantage of such *intra-bank parallelism*. Existing memory scheduling schemes aim to exploit *inter-bank parallelism* and are not aware of this new opportunity. Once dispatched, requests in one bank queue are issued in order. This could lose significant opportunities to further improve throughput: if the head request of the bank queue "conflicts" with an on-going request because they fall into the same column of arrays, as previously discussed, it cannot be issued until the on-going request finishes. All subsequent requests are blocked even though some of them do not have such conflict. This is even worse with PCM's extreme latency asymmetry between read and write. I now use an example shown in Figure 55 to illustrate this problem.

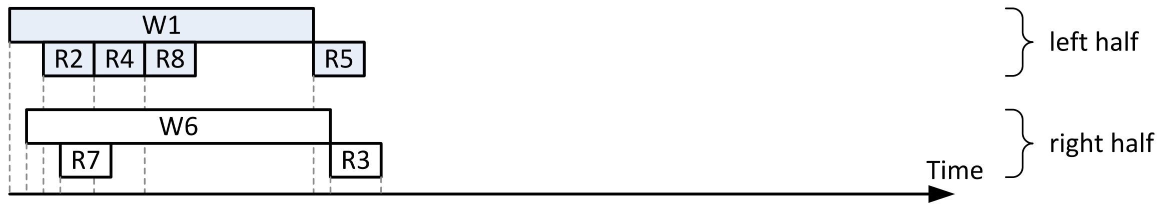
Consider a bank queue containing request sequence {W1, R2, R3, R4, R5, W6, R7, R8}. Among these requests, {W1, R2, R4, R5, R8} access the left half and {R3, W6, R7} access the right half of the bank, as shown in Figure 55(a). W1 conflicts with R5 and R3 conflicts with W6. We assume $1000ns$ and $50ns$ for write and read respectively [85].



(a) Blocking bank. Requests in a bank queue are issued in order.



(b) Non-blocking Pseudo-Multi-Port Bank. Requests are issued in order.



(c) Non-blocking Pseudo-Multi-Port Bank. Requests are issued out of order.

Figure 55: Impact of scheduling on requests finish time.

- **Figure 55(a)**. In baseline architecture, each bank can only serve one request at a time. Requests are issued in order. Total time to finish the sequence is ~2300ns.
- **Figure 55(b)**. We use non-blocking Pseudo-Multi-Port Bank without any scheduling enhancement. Requests in the bank queue are still issued in order. Since R5 conflicts with W1, it cannot be issued until W1 finishes. This immediately delays all subsequent requests. However, W6 falls into the other half of the bank, so W6 could have been issued in parallel with W1. Also, it does no harm to issue R8 sooner (than R5) as it does not conflict with W1. Nevertheless, the total completion time is approximately 1000ns (W1) + 1000ns (W6) = 2000ns, a 13% improvement over baseline.
- **Figure 55(c)**. We reorder the requests in the bank queue to exploit intra-bank parallelism, assuming dependencies among them have been resolved earlier. In this sequence, W1 and W6 are parallelized. All read requests except R3 and R5 are parallelized with writes. The total time spent is approximately 1000ns (W1 and W6) + 50ns (R3 and R5) = 1050ns. Comparing to baseline, the completion time is reduced by more than 54%.

A key point shown in this example is that in my non-blocking Pseudo-Multi-Port Bank design, *reordering requests* is critical to the overall throughput and the average read latency. Also, due to the significant gap between read and write latencies, it is utmost important to *overlap* writes as much as possible to shorten the total latency of the entire sequence. This often requires to move writes ahead of many reads. But such move will not hurt the reads too much because they can be parallelized with writes most of the time.

5.4 EVALUATION RESULTS

We study the memory throughput improvements of my Pseudo-Multi-Port Bank design with or without memory scheduling enhancement. Throughput is calculated as number of requests served over their total finish time. Figure 56 show results normalized to the baseline blocking design [1]. As we can see, using my Pseudo-Multi-Port Bank design alone (PMP) results in 35% throughput improvement on average because it provides much more parallelism. Further parallelism can be achieved through scheduling enhancement. A scheme that issues requists out-of-order (PMP+OoO) achieves another 23% improvement because of its aggressive reordering algorithm.

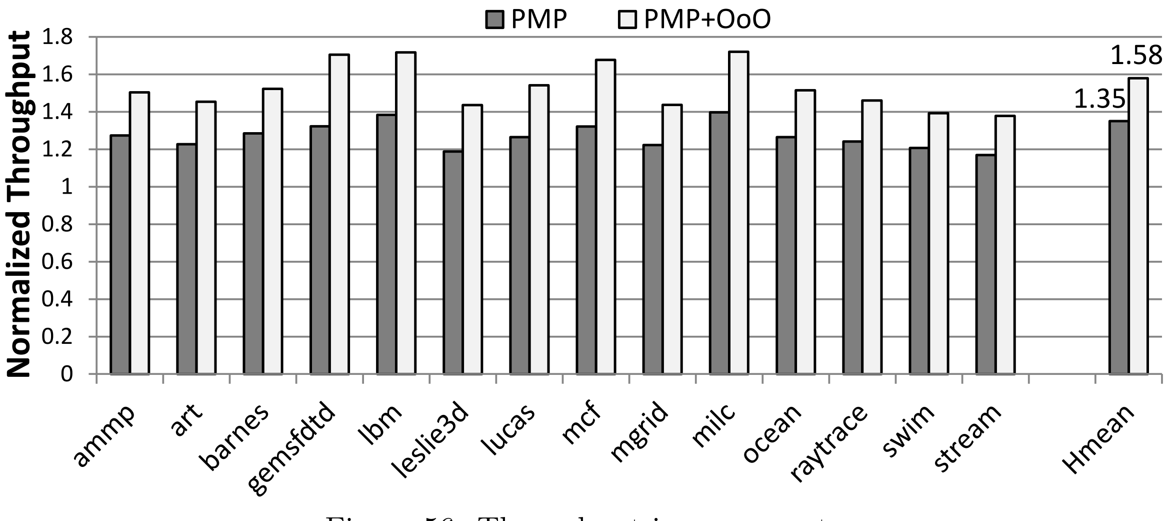


Figure 56: Throughput improvements.

[1]Architectural simulations in this chapter were performed by Ping Zhou at Department of Electrical & Computer Engineering, University of Pittsburgh.

6.0 ARRAY LEVEL DENSITY AND SCALABILITY IMPROVEMENTS FOR STT-MRAM

To provide bi-directional currents to an MTJ, a classic array structure utilizes a pair of bitlines to control voltages on two ends of a cell, similar to that of SRAM. However, with aggressive scaling of the MTJ device, the wire width and spacing of the bitline pair become the bottleneck to further shrinking memory area, diminishing the benefit of device scaling. In this chapter I propose a novel *Common-Source-Line Array* architecture, in which one wire in the bitline pair is moved to rows, leaving only one bitline per column of cells. Therefore within the proposed array, cell size is again determined by the access device, similar to that in DRAM and Phase Change Memory (PCM), leading density improvement back to the track of device scaling.

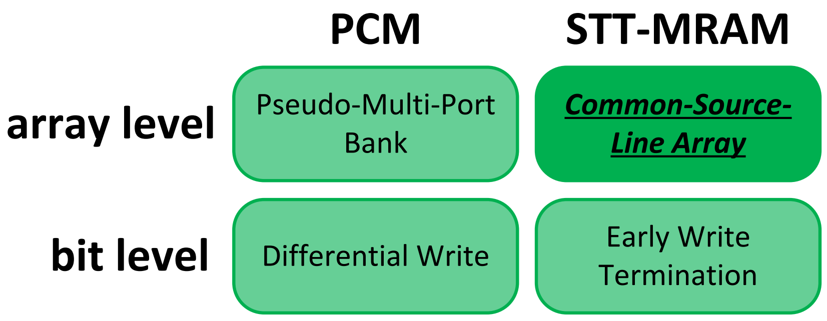


Figure 57: Array level solution for STT-MRAM: Common-Source-Line Array.

In this chapter, I describe in detail my design flow for a reliable Common-Source-Line Array architecture [72, 73], and demonstrate the viability of Common-Source-Line Array in STT-MRAM. Results show that with comparable latency and energy consumption, the Common-Source-Line Array can save 33% area, compared with corresponding dual-bitline array. I also thoroughly discuss possible design styles of a Common-Source-Line Array with respect to different applications and integration processes.

6.1 DENSITY AND SCALABILITY PROBLEMS OF DUAL-BITLINE ARRAY ARCHITECTURE

In building random access memories with MTJ, both cell and array designs must respect its bipolar nature. In cell design, an nMOS is used as the selection device to provide bi-directional conductivity. In array design, two bitlines (BL) are set on each side of the cell to provide reversible voltage drop from the write circuits. Figure 58 illustrates such *dual-bitline* structure for a 2×2 cell array. By convention, one wire in the bitline pair is called source line (SL). It is symmetric with the other bitline both logically and physically.

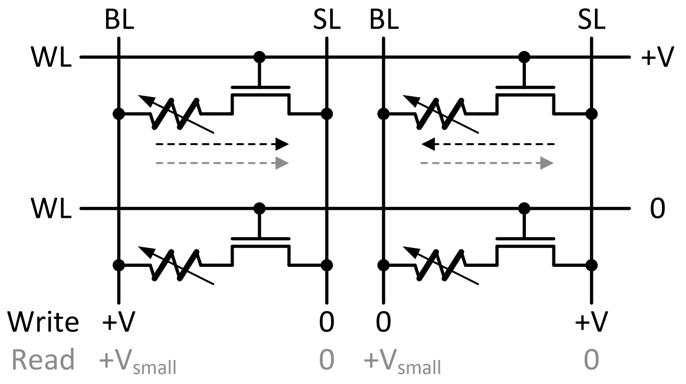


Figure 58: Illustration of a dual-BL array.

To read a cell, a small voltage difference is applied between BL and SL, resulting in a current proportional to the resistance of the MTJ device. The read current is then sensed by a sense amplifier to output the stored data. When writing a cell, a large positive or negative voltage difference is applied between BL and SL for writing one state or the other.

In memory technologies that require specially-processed memory device such as DRAM, PCM and MRAM, the memory device is stacked on top of its access transistor which is made as small (narrow) as possible to achieve high density. In such a case, the cell area of a dual-BL array is usually wire pitch dominant. In other words, the transistor (diffusion) width plus the diffusion spacing is smaller than two times the bitline (metal wire) pitch. This is illustrated in Figure 59 which shows the layout of a group of eight cells. The polysilicon is used as wordline routing, and the parallel BLs and SLs traverse in column direction on Metal 2 level. Two neighborring cells in the same column share a common diffusion and the via to SL. The MTJ is at Metal 1 level, within the via/contact stack from diffusion to bitlines [61, 13]. As can be seen in the figure, the cell width is determined by the pitch of BL and SL [27, 13, 57], not the transistor width. For ease of fabrication and cost control, in most STT-MRAM prototypes [19, 16, 39, 27, 57], the MTJs are implemented in the top metal layer, after the formation of all metal layers. The dominance of wire pitch is even more pronounced in such designs as the pitch of higher metal levels are usually several times that of bottom metal levels.

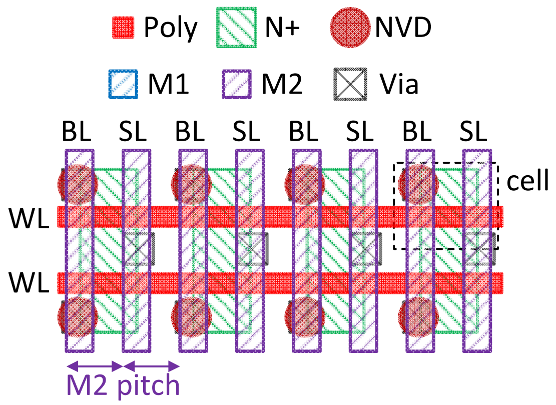


Figure 59: Layout of dual-BL array.

Obviously, with such wire pitch dominance, the reductions in MTJ switching current and thus access transistor width can no longer drive memory area shrinking. In other words, the device/cell level advantages cannot be translated into array level benefits. This problem will be even more evident in highly scaled technologies (e.g. 28nm and below) where wires can no longer keep up with transistor scaling [81].

6.2 COMMON-SOURCE-LINE ARRAY DESIGN

6.2.1 Common-SL Layout

To reduce the area and eliminate the wire pitch dominance of such an array, I propose to turn the SLs by 90° such that they span across all columns, as illustrated in Figure 60 (with the cross-section view along its bitline). That is, all cells in a row share a single SL, eliminating the areas taken by N SLs previously, where N is number of columns. Hence, cell width is narrowed down to the transistor (diffusion) width plus diffusion spacing, and the area of an array can be considerably reduced compared to a dual-BL array.

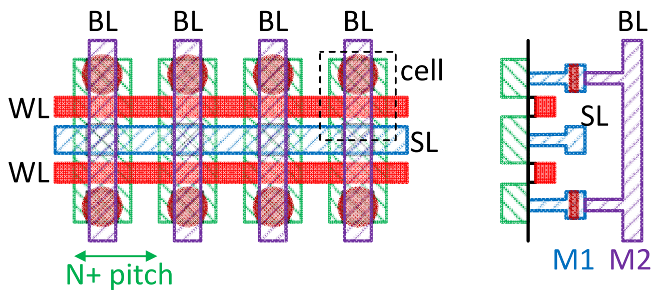
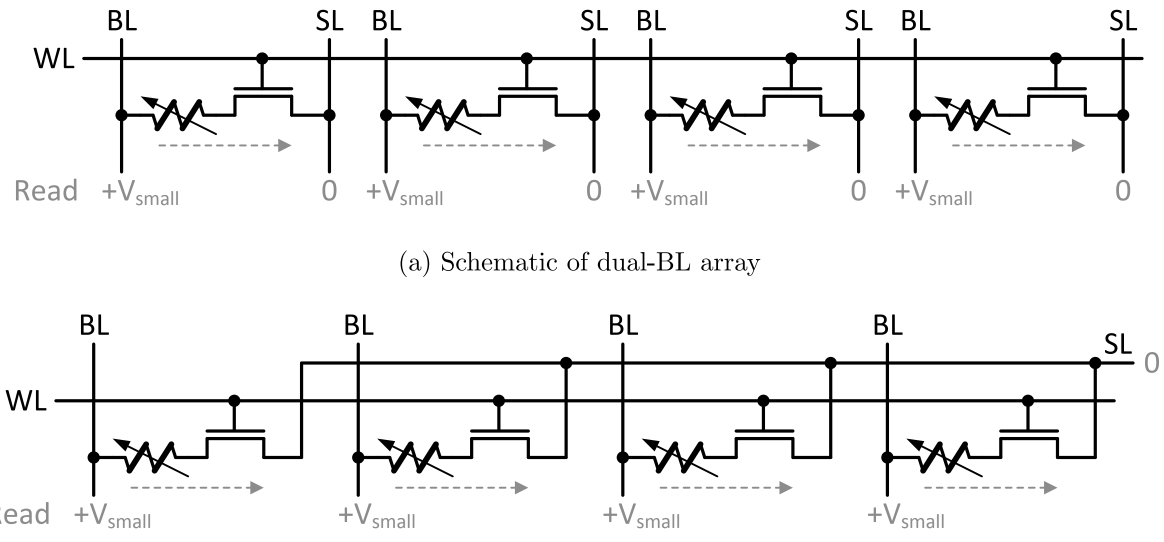


Figure 60: Layout of common-SL array with cross-section view along bitline (BL).

6.2.2 Read and Write

With the common-SL design, the memory accesses become different from before. Figure 61 shows the schematic comparison between dual-BL and common-SL arrays. The voltage configuration for read operation is also marked. These two array designs essentially share the same read scheme.

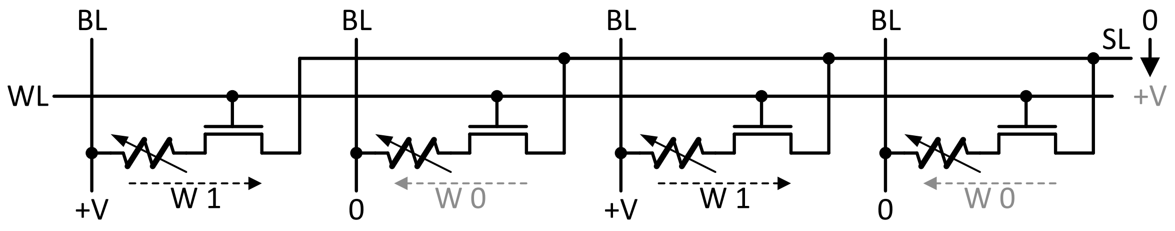


(a) Schematic of dual-BL array

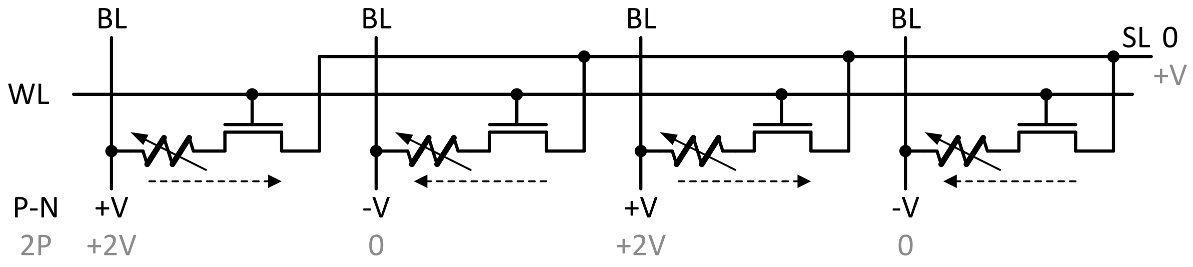
(b) Schematic of common-SL array

Figure 61: Schematic comparison between dual-BL and common-SL arrays.

For writes, the dual-BL array can use a positive or negative voltage applied to each pair of BL and SL, depending on the value to be written into the cell. However, in the common-SL array, writing different cells in a row are no longer independent due to the shared SL. Hence, writing bit '1' and '0' should be performed in two separate rounds. As shown in Figure 62(a), the bitlines voltage are first set according to the values to be written. Then the SL is set to 0 for writing bit '1', and in the next round to +V for writing bit '0'. Therefore, this write scheme doubles the write latency due to the common-SL. Obviously. such doubling is very expensive in on-chip cache designs, given that the long write latency is already the performance limiter in STT-MRAM based caches [55].



(a) Flip-V scheme



(b) P-N (or 2P) scheme

Figure 62: Two write schemes in common-SL array.

Instead, I propose to concurrently write all cells in a row, achieving a write latency comparable to traditional dual-BL array. This is achieved by applying both +V and -V to corresponding BLs and 0 to SL, producing current/voltage in two directions simultaneously. This is illustrated in Figure 62(b) as positive-negative voltage (P-N) scheme. Alternatively, one can also shift all voltages by V leading to an equivalent scheme with no negative voltages, shown as the 2× positive voltage (2P) scheme in the figure. In both of these schemes, WL swings between lowest and highest voltage levels. I will use this P-N scheme for better illustration in the remainder of this chapter.

The boosted voltages -V or +2V can be regulated by charge pumps. A voltage doubler charge pump for +2V(can use its complement implementation for -V) can be made compact and fast in an embedded environment [53]. Because only one level of boosted voltage is needed in both P-N and 2P write schemes, such voltage could also be supplied directly from off chip [16], eliminating charge pump circuits.

6.2.3 Concerns of Gate Oxide Breakdown

As I choose the P-N (or 2P) write schemes to not compromise latency, one reliability issue arises as the largest possible voltage drop on two sides of the transistor gate is now 2× the power supply voltage V_{DD}. This is especially a concern when it comes down to a failure mechanism known as *gate oxide breakdown* [20]. A *soft* gate oxide breakdown shares similar underlying mechanism as Negative Bias Temperature Instability (NBTI) in that electrons can be trapped into gate oxide. Gradually, the accumulated traps may stack together and form a path through the oxide, making it more conductive. A *hard* gate oxide breakdown, which is the concern here, happens when the voltage across gate oxide exceeds its maximum sustainable value. As a result, the oxide is punched through, melted, and no longer insulating.

The occurrence of hard gate oxide breakdown can be effectively eliminated by having appropriate oxide thickness (T_{ox}) in design. Traditionally, T_{ox} was scaled proportionally to V_{DD} and V_{th} for better performance and guaranteed reliability. At $45nm$ technology node with a V_{DD} of $0.9\sim1.1V$, T_{ox} in bulk CMOS is within the range of $1.5\sim2.4nm$ [81, 80, 82]. For my common-SL array design I pick $V_{DD} = 1V$ and thus the maximum voltage drop is $2V$. Previous studies have shown that a $4nm$ T_{ox} is safe for $2V$ [20, 79], and $4nm$ is about 2× the T_{ox} in nominal $45nm$ process.

The downside of thicker gate oxide is the reduced current capability, or larger effective resistance, of the access transistor. This in turn translates into degradation in performance or write stability. Equation 6.1 (in first order) demonstrates how I_{DS} is reduced due to increased T_{ox}. To offset this effect, one could boost the wordline voltage by a factor of 2, doubling the V_{GS}, in equation 6.1. This will result in extra energy costs on wordlines. However I found in my study that wordline energy is still negligible compared to read/write energy of the cell. Lowering V_{th} and using high-k_{ox} dielectric can also help regain drive strength.

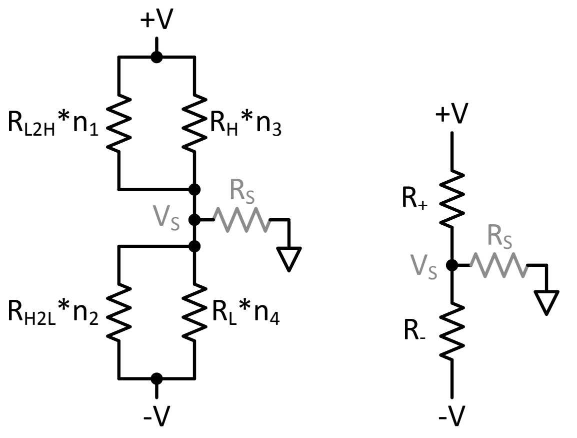
$$I_{DS} = \begin{cases} \frac{\mu k_{ox}\epsilon_0 W}{2T_{ox}L}[2(V_{GS}-V_{th})V_{DS}-V_{DS}^2](1+\lambda V_{DS}) \\ \frac{\mu k_{ox}\epsilon_0 W}{2T_{ox}L}(V_{GS}-V_{th})^2(1+\lambda V_{DS}) \end{cases} \tag{6.1}$$

In high voltage transistors that possess thick gate oxide, e.g. I/O transistors, the minimum gate length usually increases proportionally. This is to accommodate the possibly large V_{DS} across the channel. However in the proposed common-SL array, a $V_{DS} \leq V_{DD}$=1V can be guaranteed on all cell transistors in either read or write operation (Figure 61 and 62). Thus it is feasible to maintain the nominal minimum gate length. This is exactly the same case as DRAMs, which utilize >3V wordline voltage swings while still featuring $\leq 8F^2$ cell sizes [69, 81]. I use PTM model [82] to generate my 45nm custom nMOS model with 4nm T_{ox}, which is used in my circuit simulations. Simulation showed that at 2V, the thicker T_{ox} nMOS is slightly weaker than a nominal nMOS at 1V.

6.3 DESIGN FOR RELIABILITY

In the proposed Common-Source-Line Array architecture, SLs are shared among cells in the same row, which are read and written simultaneously on each access. As discussed earlier, the read/write operations on individual cells are no longer independent, while such isolation is guaranteed in a dual-BL array. This is not a problem in reading a row because all the cells are exposed to the same voltage configuration, which is essentially identical to reading a dual-BL array, as shown in Figure 61. However, the write operation is more complicated. I now formulate the problem using a static model.

In a write operation, a cell's resistance experiences one of the four state changes: from high to low (H2L), from low to high (L2H), staying high (H), and staying low (L). Here I avoid the logic abstraction of '0' and '1' and just use high and low resistances of the MTJ device for consistency. For these four cases, I extract the effective resistance of a cell, including both the MTJ and the access transistor. This resistance represents the state of the cell at the *beginning* of a write, when the MTJ state has not yet changed. This is because the state change of MTJ is an abrupt process. Therefore, writing a group of cells sharing a common SL can be generalized into an equivalent circuit as shown in Figure 63(a), assuming a positive BL writes high resistance and a negative BL writes low resistance. The node in the middle represents the common SL and its resistance R_s. Here $n_1 \sim n_4$ are the number of cells in the four state changes respectively, and $N = n_1 + n_2 + n_3 + n_4$ is the number of written cells sharing a common SL. It can be further simplified into the circuit shown in Figure 63(b).



(a) (b)

Figure 63: Equivalent circuits of write operation.

Figure 63(b) is essentially a voltage divider circuit. The voltage on the common SL, shown as V_s, is supposed to stay grounded to provide identical voltage drop on all cells. However, the imperfection of common SL and the global sources that drive it, represented by R_s, breaks such balance and introduces *voltage drift* on the SL node. Such drift places negative impact on write operations, especially on those cells with smaller voltage drops. For STT-MRAM, reduced voltage may directly causes write failures as the induced current may not be larger than the switching current of MTJ. I now apply KCL to express V_s analytically:

$$V_s = \frac{V \cdot (R_- - R_+)}{R_+ + R_- + \dfrac{R_+ R_-}{R_s}} \tag{6.2}$$

From equation 6.2 and Figure 63, it can be derived that V_s is determined by the following parameters:

(1) Old data stored in each cell and new data to be written

(2) Number of written cells, N, sharing a common SL

(3) Driving capability of SL node, R_s

Parameter (1) decides the distribution of $n_1 \sim n_4$. Hence, it determines R_+ and R_-, together with parameter (2). However, (1) is governed by data patterns generated by applications, and thus is hard to control at design time. On the other hand, we do have control over both N and R_s. Therefore, to mitigate V_s drift, I will first find the worst-case data pattern $n_1 \sim n_4$, i.e. the pattern that leads to largest V_s drift with given N and R_s, and then find an array design with proper N and R_s that work reliably/robustly under such worst cases. The parameters of the MTJ device used in my analytical models and circuit simulations are summarized in Table 7.

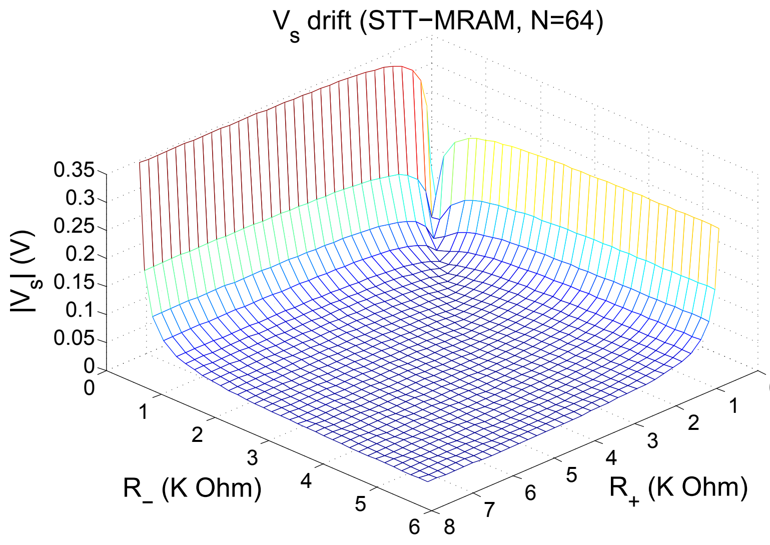
Table 7: STT MTJ parameters [19, 28].

R_L (P)	R_H (AP)	Switching Current	
		P2AP	AP2P
2KΩ	4KΩ	55μA	30μA

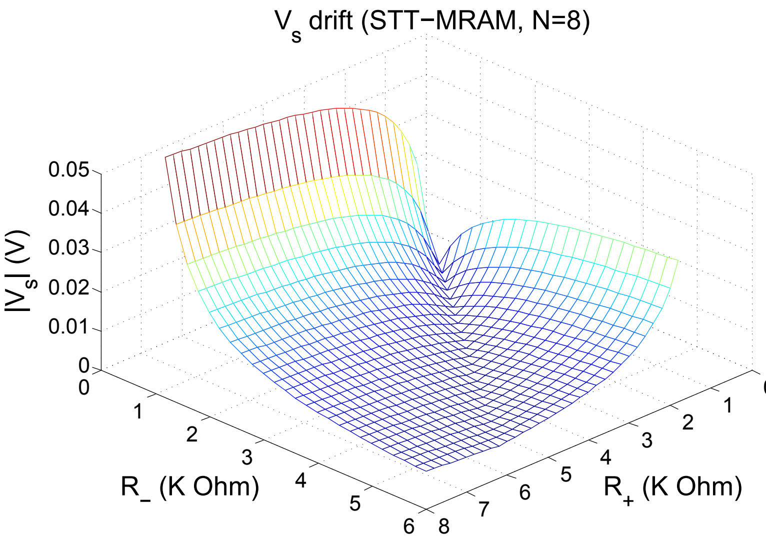
6.3.1 Mitigating V_s Drift

To find the worst-case data pattern, I plot V_s drift in the entire range of R_+ and R_-, for N={8, 16, 32, 64} with a constant R_s=30Ω. Figure 64 shows plots for N=64 and 8, and all other cases lie in between. All the plots agree on the same tendency: the absolute value of V_s reaches its extremes when R_- reaches its minimum. The worst case happens when R_+ is at its maximum. For example, when $R_+ = 7K\Omega$ and $R_- < 0.5K\Omega$, V_s drift surges to 340mV when N=64 and 45mV when N=8. The drift is relatively moderate with all other R_+s and R_-s.

The worst case of resistance distribution happens when there are N R_Ls, i.e., n_1=n_2=n_3 =0, n_4=N. In this case, $R_+ = \infty$ and $R_- = R_L/N$, representing an extreme imbalance. However, it does not alter any cell and thus not helpful in evaluating the harmfulness of V_s drift. I therefore use a next-to-worst case data pattern that involves one cell resistance change from high to low, i.e. n_1=n_3=0, n_2=1, n_4=$N-1$, as shown in Figure 65. In this case,



(a) N=64



(b) N=8

Figure 64: V_s *vs.* R_+ and R_- for N=64 and 8, R_s=30Ω.

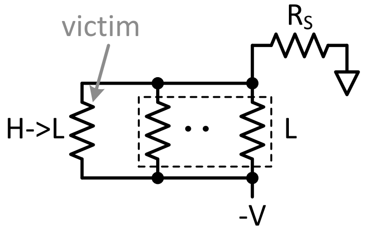


Figure 65: Equivalent circuit of the worst-case data pattern.

only one cell is written and this cell is a victim of the parallel R_Ls that lower the effective resistance.

I then use this worst-case data pattern and study V_s drift as a function of N and R_s, as plotted in Figure 66.

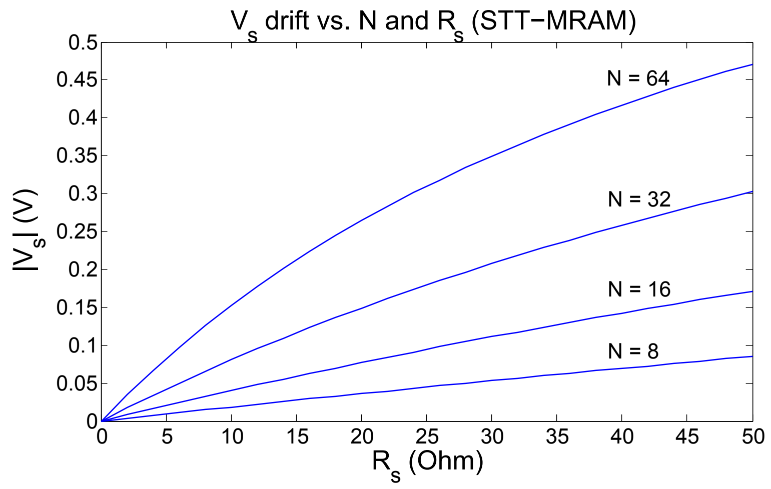


Figure 66: V_s drift as a function of N and R_s.

As can be seen, $|V_s| \propto N$, $|V_s| \propto R_s$. Furthermore, with large N and R_s, V_s drift can reach ∼450mV, which is prohibitive as the voltage on a victim cell is nearly halved from what it should be. I will show next on how to design the array for lowest R_s and best N to achieve high reliability.

6.3.2 Array Design

R_s is the resistance between SL node and the "ideal" ground/power. On a fabricated chip, ground/power is supplied from off-chip through pads which drive the on chip ground/ power rings (wide metal wires) surrounding the core area where all the circuits lies. The ground/power are then delivered to the entire core area through hierarchical mesh networks in high level metal layers, as illustrated in Figure 67. The number of pads are usually large and the rings are usually considered as ideal boundary conditions in ground/power network design and analysis [50]. Therefore, for integrated circuit designers, the ground/power rings can be considered "ideal" and all the RLC and IR-drop of supply network seen by circuits can be assumed as the result of the mesh network. Hence, the center region of the network will incur the largest effective resistance due to the "imperfection" of supply network. I denote this resistance as R_{mesh}. Based on the chip size of [61] and the analysis in [50], I estimate the worst-case R_{mesh} to be $\sim 15\Omega$. [1]

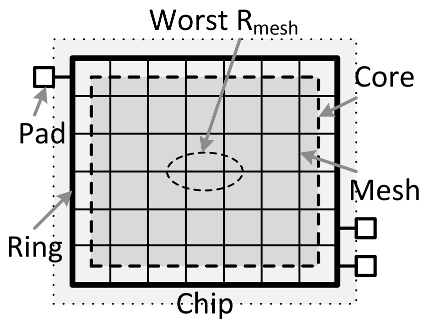


Figure 67: Simple illustration of ground/power supply system.

Other parts of R_s come from within an array. Figure 68 depicts an example of a common-SL array design. It has 1024 rows and 256 columns of cells, same as the prototype in [61]. The *rib* wires in wordline direction are the common SLs of width N. The *spine* wires in bitline direction connect the ribs to ground (power) meshes outside the array. Here it shows the configuration of N=128, 64 on each side of a spine. Hence, in a 256-column wide array, $\frac{256}{N}$=2 spines are needed.

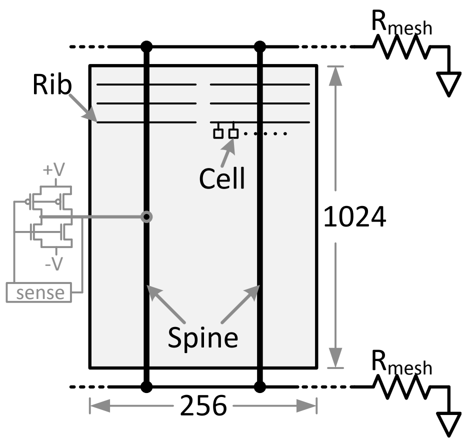


Figure 68: An array example with N=128.

I also conservatively assume the resistance of a rib (common SL) is seen by all cells sharing it, regardless of their relative positions. Hence, we have

[1]Notice that the wire bonding packaging as assumed here generally yields higher supply network resistance than more advanced flip chip packaging (a.k.a. Controlled Collapse Chip Connection (C4)). So my R_{mesh} estimation is conservative in the following analysis.

$$R_s = R_{mesh} + R_{spine} + R_{rib}$$

Early analysis from Figure 66 shows that keeping both R_s and N small helps reducing V_s drift. While R_{mesh} has been determined by the global ground network, the tradeoff between R_{spine} and R_{rib} is dependent on N, which is studied in Figure 69. Spine straps are constrained by total width of N cells and are thus wider at larger N. So R_{spine} decreases with increasing N. R_{rib} however increases linearly with N and thus becomes dominant at larger N. As spines are additional area overheads, layout techniques can be apply to make the area overhead per spine less than a Metal 2 pitch, while still maintaining low resistance, as shown in Figure 70. The idea is to use narrowest wires to reach ribs on Metal 1, then back them up using wide Metal 3 straps that traverse over the cells.

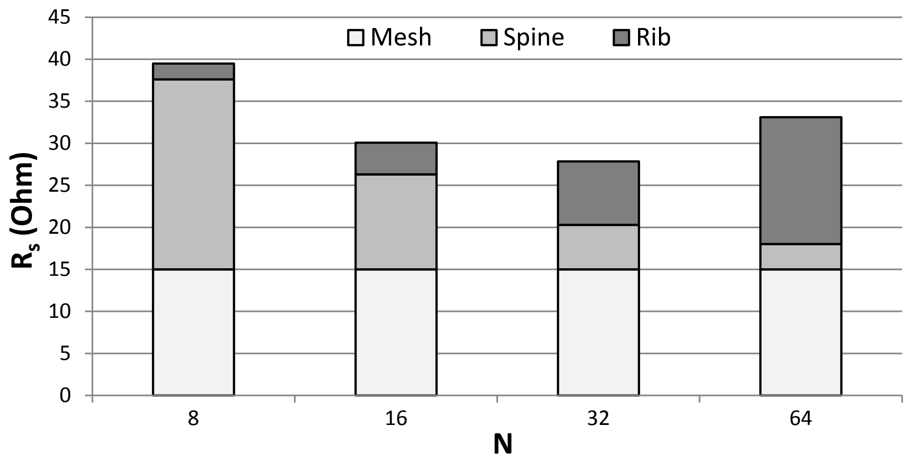
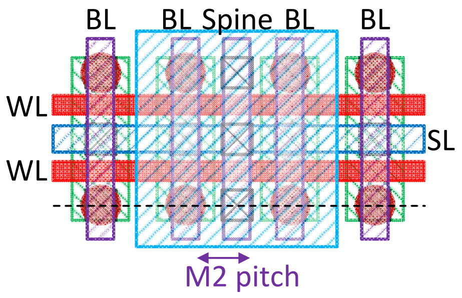
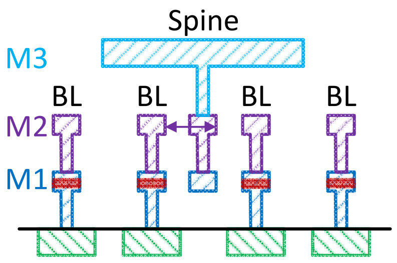


Figure 69: R_s tradeoffs.



(a) Layout



(b) Cross-section along dashed line

Figure 70: Physical design of spines.

6.3.3 Compensation Circuit

To further mitigate V_s drift, I develop a compensation circuit that can detect large V_s drift, and compensate it at run time. Its simple illustration is shown in Figure 68 in gray. A set of sensing circuit and compensation transistors are attached through wide wires to the middle of a spine. The sensor can be built out of compact voltage comparators like sense amplifiers in SRAM and DRAM. When it detects a V_s drift larger than a reference, it opens the compensation transistor that helps in balancing voltage distribution by reducing R_+ or R_-, whichever is larger. Figure 71 demonstrates the equivalent circuit of write operation with compensation.

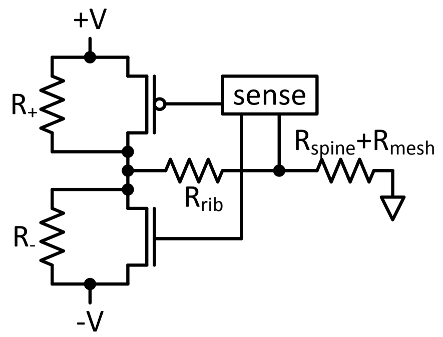


Figure 71: Equivalent circuit of write operation with compensation.

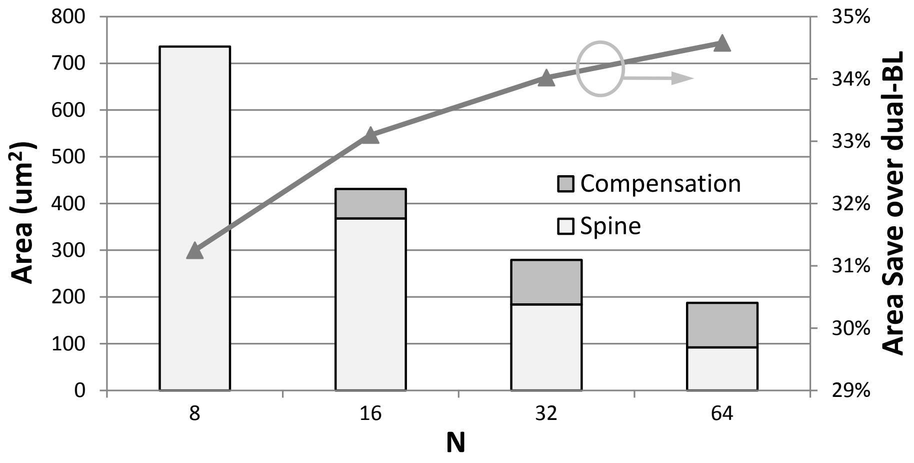


Figure 72: Area estimations (per array).

When N is small, V_s drift is tiny even in worst cases, so the compensation may not be necessary. When N is large, V_s drift also becomes large, so compensation needs to be carried at different levels of strength. Hence, several sensors are equipped on each spine, controlling varying number of parallel transistors to open or close for different data patterns. As we can see, the area of compensation circuit increases with N. However, larger N also implies fewer spines, and less area overhead for the array (as indicated in Figure 70). This tradeoff is studied in Figure 72, which is estimated based on the rules in Table 8. The results show that the area overhead of spines drops linearly with increasing N, while the area of the compensation circuit increases only mildly. It also shows the percent of area reduction (the curve in the figure) considering both overheads, when comparing the common-SL with the dual-BL array design. Even with the largest area overhead (N=8), the common-SL array still holds more than 30% area reduction.

Table 8: Layout rules used in area estimation.

Rule	Value (μm)
Polysilicon min width	0.045
Diffusion min width/spacing	0.09
Metal 1 min width/spacing	0.06
Metal 2 min width/spacing	0.07
Metal 3∼6 min spacing	0.35 (width > 1)
	0.45 (width > 1.5)
	0.75 (width > 2.5)
	1.25 (width > 3.5)
Metal X max density within $100\times100\mu m^2$ area	60%

With compensation, it is possible to trade spine area for much smaller compensation circuit area, and guarantee a controllable V_s drift. Combining the results in Figure 66, 72 and 69, it is easy to see that when N=16 a common-SL array achieves a low R_s, good area reduction, and easy-to-control V_s drift.

6.4 EVALUATION RESULTS

With the 45nm PTM model and device parameters given in Table 7, I built subsets of common-SL and dual-BL arrays for STT-MRAM, and simulated them in SPICE. My circuits include all supply network models and compensation circuits previously described. The RC parasitics of wordlines, bitlines and source-lines (for dual-BL array) are all properly modeled as well.

The metrics I measured are V_s drift, write latency and energy. Since they depend on different data patterns, I selected representative data patterns for evaluation, as listed in Table 9. The patterns include the most difficult ones that cause worst V_s drifts, and all other possibilities of cell state transition.

Table 9: Data patterns used in evaluations.

No.	H→L	L→H	stay H	stay L	Emphasize
1	1	0	0	N-1	L
2	1	0	1	N-2	
3	1	1	0	N-2	
4	1	1	(N-2)/2	(N-2)/2	H, L
5	1	1	N-2	0	H
6	0	1	N-1	0	
7	$N/4$	$3N/4$	0	0	L→H
8	1	N-1	0	0	
9	0	N-1	0	1	
10	0	N	0	0	
11	$N/4$	$N/4$	$N/4$	$N/4$	all
12	$N/2$	$N/2$	0	0	H→L, L→H
13	$3N/4$	$N/4$	0	0	H→L
14	N-1	1	0	0	
15	N-1	0	1	0	

For STT-MRAM common-SL arrays, the V_s drift will likely create write failure since the effective voltage on cells may be less than the threshold. However, if we can keep the drift within half of the natural IR-drop of a dual-BL array (denoted as $\frac{1}{2}$IR-drop), then we can gain the same reliability on writes. In a dual-BL array, the existence of resistive BL and SL introduces *IR-drops* into the write/read current path. Such IR-drops decrease the effective voltage applied on a cell, especially when it is physically far from its write driver. In contrast, the source-line resistance is minimized by my spine+rib design in the proposed common-SL array, leaving only BL resistance in write/read current path, which effectively halves these IR-drops. Hence, if we can control the V_s drift such that it does not exceed $\frac{1}{2}$IR-drop, or, the total effective voltage drop does not exceed that in a dual-BL array, then a common-SL array can guarantee the same write reliability as a dual-BL array:

$$|V_s| + IR_{cSL} \leq IR_{dBL} \qquad (IR_{cSL} \approx \frac{1}{2} IR_{dBL})$$

Figure 73 shows the absolute values of voltage drifts over all data patterns, with or without compensation. Patterns 5$\sim$10 represent cases of large V_s drifts that should be compensated. All the rest patterns generate acceptable V_s drifts and thus did not trigger compensation circuits. The worst IR-drop in dual-BL array is $\sim$80mV, obtained from simulation with MTJ parameters in Table 7. It can be seen that V_s drift together with IR-drop falls below the 80mV line, demonstrating the effectiveness of my common-SL design.

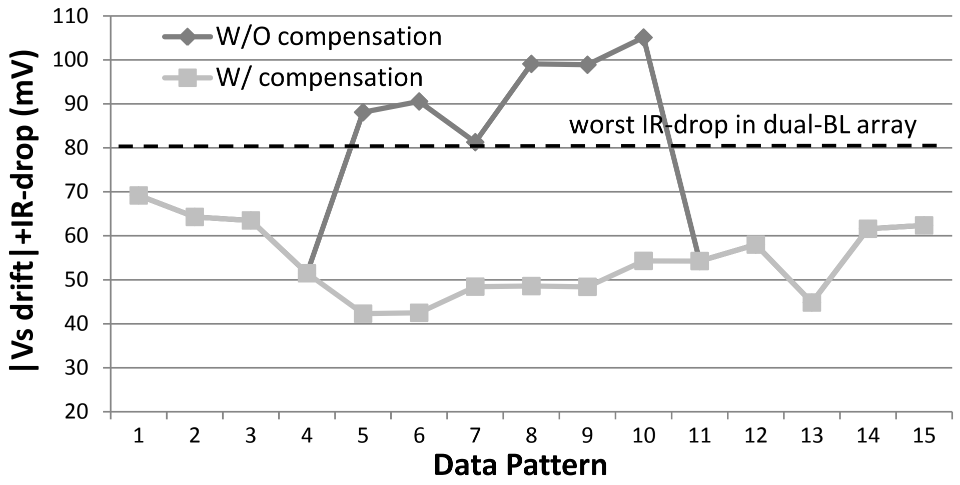


Figure 73: V_s drift + IR-drop *vs.* data pattern in STT-MRAM (N=16).

As long as the reliability is guaranteed, a common-SL array of STT-MRAM enjoys similar latency and energy to its dual-BL counterpart. Notice that although the voltage drop on write paths in common-SL array is 2× of a dual-BL array, their energy consumptions are comparable. This is because write current that flows through R_+ (writing L cells) is reused by R_- (writing H cells) in a common-SL array, and their mismatched part is supplied/drained by SL. While in a dual-BL array, write currents are drained by their own sinks. After accounting for all the overheads, the proposed Common-Source-Line Array architecture achieves a 33% area reduction over dual-BL array.

6.5 DESIGN STYLES WITH COMMON-SOURCE-LINE ARCHITECTURE

In recent years, STT-MRAM was extensively studied as universal memory due to its flexibilities. As each application possesses unique requirements, the proposed Common-Source-Line Array architecture can be utilized in different design styles.

For example, the design choice of P-N and 2P write schemes in above studies is based on an on-chip cache application, which demands low latency and high throughput especially in the multi/many-core era with ever increasing size and memory intensity of working sets. Therefore, the doubled write latency of the proposed flip-V scheme in Section 6.2.2 and Figure 62(a) is too expensive here. On the other hand, it is worth noting that column multiplexing is another solution to V_s drift in P-N and 2P write schemes. In other words, writing only a fraction of the cells on one common-SL and biasing the remaining unselected BLs at the SL voltage could mitigate V_s drift, because the unwritten cells have no contribution to the voltage divider behavior. This has the same effect of reducing the number of cells per common-SL (N in previous discussion) under full-width concurrent write. Hence, in applications with relaxed bandwidth specification, V_s drift could become even easier to control.

Moreover, for applications with very loose latency and bandwidth requirements like stand-alone flash replacement, the proposed flip-V scheme in Section 6.2.2 and Figure 62(a) is adequate. Thanks to the separation of writing-1 and writing-0, the flip-V scheme is free of boosted voltage and V_s drift at the expense of doubling the already-long write latency. Because here the common SL is no longer integrated to power/ground network directly, extra care should be carried out in the SL driver design.

Therefore, V_s drift is associated with the choice of design styles which is in turn determined by the application. This demonstrates the tradeoffs between performance and the overhead of reliability control, and represents the flexibility of my common-SL architecuture. In this study, the choice of P-N and 2P write schemes is not only due to the corresponding design style of high-performance cache, it can be viewed as an extreme-case study as well. Without loss of generality, it provides important insights to the reliability challenge associated with my novel array architecture, and demonstrates that this can be solved with lightweight techniques even at such extreme case, proving the feasibility of my design.

Besides application, another consideration is process integration. More specifically, choices of CMOS-magnetic process integration can yield different area savings of common-SL array over dual-BL array. This study assumes the most aggressive and costly process that the MTJ is tightly integrated into CMOS metal layers at local level (M1/M2) [61, 13]. This integration leads to most conservative area saving because cell size is dominated by transistor width/spacing after removing SLs from columns. But it also liberates the potential of scaling with improved MTJ device. On the other hand, if cost is at higher priority and thus the MTJ is stacked in the last step on top of all metal layers [19, 16, 39, 27, 57], a ∼50% area saving is easily achievable, because the higher level metal width/spacing is larger than that of cell transistor and thus still dominates cell size after removing one out of two wires per column. Especially, if MTJ is used in cache and stacked on logic process with ∼10 metal layers, the large pitch of top metal layers could easily exhaust the density advantage over SRAM/eDRAM even with the 50% discount from common-SL array. Therefore, future STT-MRAM designs should pursue lower-level integration for both scalability and absolute density.

To date, the smallest cell size reported in a dual-BL structure is $14F^2$ with M1/M2 level integration [13]. This implies that a $\sim 9F^2$ cell size is easily achievable with the proposed Common-Source-Line Array architecture. Because memory cell arrays can usually be fabricated using specially optimized processes (much more aggressive than the design rules in Table 8), an even smaller cell size could be expected. Furthermore, when switching current is large, the 2T1R cell structure has advantage on layout area over 1T1R cell [57]. However for the same equivalent transistor width, 2T1R cell suffers severer wire pitch dominance due to its folded transistor. My common-SL architecture is also applicable to 2T1R cell [57, 61, 13] with slight modification to also remove wire pitch dominance.

7.0 RELATED WORK

7.1 PCM

7.1.1 Prototype Chips & Circuit Designs

Various PCM prototype chips have been fabricated in the past years, showed the industrial evolution of PCM technology. In [15], Gill *et al.* presented a 4Mb BTJ-selected test memory in 0.18μm technology, and demonstrated the fundamental characteristics of the GST material. Another 0.18μm 4Mb MOS-selected experimental chip with μtrench structure storage node was then introduced by Bedeschi *et al.* [3] showing 45ns read access and 5MB/s write throughput. Also in 0.18μm technology, Cho *et al.* [7] achieved 64Mb capacity featuring 16F^2 MOS-selected cell, 512Kb sub-array, and separated SET/RESET control. Signal IR drop along the resistive BL is also discussed. Targeting the signal IR drop problem, Oh *et al.* [40] proposed cell position aware current regulator scheme and multiple step-down pulse generator scheme (inherited by [29]) to improve RESET and SET distributions, respectively.

Kang *et al.* presented a 0.1μm 256Mb chip in [26]. This design implemented many key features for high density PCM chip architectures. The chip is configured using hierarchical bitline and wordline architecture to make it area-efficient, in which sense amplifiers (SA) and write drivers (WD) are globally placed and shared by many sub-arrays via the global bitlines (GBL). The 256Mb capacity is organized into multiple banks, each has private row and column circuits to work independently. A sophisticated charge pump system is adopted to cope with the reducing supply voltage but non-reducing operation voltage. And it also helps in increasing write currents by reducing the parasitic resistance along the path from SAs/WDs to cells, which is prolonged by the hierarchical BL structure. In addition, FinFET technology is also adopted to increase the drive current of the tiny cell-access nMOS. To provide an extended write throughput, the chip uses an additional external power supply to support wider parallel write, and this scheme was inherited by [29, 9].

A 0.13μm 512KB (or 4Mb) embedded PCM was presented in [17]. In this design, Hanzawa *et al.* implemented a set of techniques tailored for the embedded PCM module with current-saving architecture. It features a sense amplifier prefetch serial write scheme to support serial write mode, a two-step pulse set method to reduce SET time, and a charge-transfer direct-sense scheme to achieve high-speed and low-power read operation. A cell write current of 100μA and a 20ns read access time was demonstrated.

Beyond previous MOS- and BTJ-selected designs, a 512Mb diode-selected PCM was developed by Lee *et al.* in 90nm process [29]. The vertical diode-switch stacked with the GST device achieves a 5.8F^2 cell size. However, a key challenge with diode design is that it requires higher operation voltages than the MOS-selected cell by the built-in potential (or threshold voltage) of a diode to turn it on during write and read. Therefore, read/write circuit techniques and a charge pump system for the diode-switch PCM were proposed. A write-verify scheme was used to enhance distribution and reliability of cell data, with an arbitrary slow-quench (ASQ) shaper scheme to improve the write time of the SET data.

In [4], Bedeschi *et al.* presented a 90nm 128M-cell PCM demonstrating 256Mb effective capacity with multi-level cell (MLC) storage. A programming algorithm suitable for 2-bit/cell storage achieving tightly placed inner states (in terms of cell current or resistance) was proposed. It is based on a Program-and-Verify (P&V) technique to ensure adequate control of the cell resistance. The cell is first programmed to its low-resistance state by a long SET sweep pulse to avoid any spread due to the previous programmed state. This is followed by a single RESET pulse with a fast quench to initialize the cell. Subsequent Stair-Case Up (SCU) algorithm applies a sequence of box-shaped program pulses (each followed by a verify step) that have the same width but increasing height. The MLC capability not only leads to complex and slow write, read latency is also increased (120ns) because the data has to be sensed out of more states.

A 45nm 1Gb chip in NOR Flash interface was presented by Villa *et al.* in [62]. The PCM array is built out of a basic structure made of 4 cells. To program a pattern of data, a sequence of SET pulse, RESET pulse, SET verify and RESET verify is given after a pre-read phase. In case the verify fails, the sequence is repeated using larger pulses. A whole set of SAs is dedicated for verify in addition to the read SAs.

In [47], Sandre *et al.* demonstrated an embedded PCM chip in standard 90nm CMOS technology with 6 metal layers. The storage element has been integrated using only 3 additional masks with respect to process baseline, making it very attractive in terms of process cost and simplicity. Moreover, the cell selector is implemented by a standard low voltage nMOS device with 2.1nm gate oxide for the 1.2V supply. A dual-voltage row decoder and a double-path column decoder are introduced, enabling a completely low voltage read operation. This confirms PCM technology as a viable solution in embedded environments.

A 58nm 1.8V 1Gb PCM prototype was presented by Chung *et al.* in [14]. The diode-selected cell is just $4F^2$ which is the lower extreme of achievable size. The chip is compatible with LPDDR2-NVM interface, and features multiple throughput enhancement techniques to make PCM viable for main memory. To cope with the huge bandwidth difference between PCM and DRAM, an SRAM-based 1KB program buffer with 800Mb/s write throughput is used and programming operation is internally controlled using FSM. On the PCM core side, the number of simultaneous program bits needs to be increased, leading to according increase in program current injected to core, which can cause a significant voltage bouncing and can be a burden for the pumping capacity. So the WDs are organized into two groups, and the skewed group is activated later than the main group by a timing delay of tSKEW. As as result, superpositioning of the program peak current is prevented without significant performance degradation. Furthermore, notably, this prototype integrates Differential Write [75, 31] and Flip-N-Write [8] techniques, with the name of "data comparison write with inversion flag (DCWI)". It also considers SET and RESET to have different weights for obtaining the number of effective bit changes. A mid-array pre-charge scheme is also proposed to greatly reduce RC delay on GBLs.

In highly scaled 20nm process, an 8Gb density was achieved by Choi *et al.* [9] with 40MB/s write bandwidth in LPDDR2-NVM interface. Cost-effectiveness is one of the top priorities for such high density memory chips. Without doubt, diode selector is the choice for smallest $4F^2$ cell size, and the buried wordline is built with $N+$ doped base and strapped by metal wires. A large 8Mb (4096 WL $\times$ 2048 BL) sub-array is built to diminish the area overhead of WL and BL selection switches per sub-array, which are required by the high voltage and current in PCM write operations. As a result, the chip size is only 70% of a DRAM chip at the same design rule, at the expense of increased parasitic resistances and capacitances. To cope with these overheads, dual-LY and multi-WL schemes are integrated to reduce the effective resistances of local BLs and WLs. A pre-emphasis write scheme and a cascode type current source were also developed to reduce the cell current rise time (to fight increased parasitic RC) and increase WD output resistance (to mitigate cell location dependency), respectively. Thanks to the significantly scaled GST device and write current ($I_{reset} = 100\mu A$), an unprecedented 40MB/s write throughput was accomplished with much wider parallel bit-writes.

Prior art and my work. Previous prototype chip designs have provided me with valuable insights of PCM characteristics and operation mechanisms, key parameters and guidelines used in my studies, and motivated my research work. While the Differential Write technique was incorporated in a Samsung prototype [14], my Pseudo-Multi-Port Bank design can also be easily integrated as it is heavily based on an existing prototype [26]. As the write current keeps scaling down and positively affects throughput [9], the Pseudo-Multi-Port Bank design, aiming at other throughput limiters, offers a perfect additional boost on further throughput improvements.

7.1.2 Architectural Innovations

In [30], Lee *et al.* proposed a design of a cache-organized row buffer in PCM memory: use a narrow row buffer entry to mitigate per-access write energy, and use multiple row buffer entries to improve locality and write coalescing. In addition, a partial write scheme was proposed to improve lifetime. It marks dirty words of each memory write access and only writes these dirty words. Since a dirty word may still contain many redundant bit changes, partial write cannot fully exploit the opportunity of value locality.

In [41], Qureshi *et al.* proposed a hybrid memory system using DRAM as a buffer for the PCM main memory. A page loaded from hard disk is only written to the DRAM buffer, and

PCM is only written when a page is evicted from DRAM buffer. Also, similar to the partial write scheme in [30], only the dirty lines within a page are written back. Some OS support is added to avoid writting PCM in case of streaming applications. Similar to [75, 31], a line rotating wear-leveling scheme was studied.

Based on Differential Write, Cho *et al.* proposed Flip-N-Write technique [8] that writes either the original value or its inversion, whichever results in fewer bit flips. Therefore, Flip-N-Write guarantees that no more than half of the bits in each write are changed. This in turn means that under the same instantaneous write power constraint, it can support a double-width write. As a result, Flip-N-Write can reduce write power/energy, and improve PCM throughput and lifetime.

In [42], Qureshi *et al.* proposed Start-Gap wear-leveling for hybrid memory system. All lines plus an extra empty GapLine can be regarded as forming a circular buffer, and GapLine is moved by 1 location periodically. Start-Gap has the advantage of low storage and computation overhead, however it has the shortcoming of slow line movements. To overcome this shortcoming, the memory is partitioned into smaller regions, each running Start-Gap independently.

To mitigate PCM's long write latency, Qureshi *et al.* proposed write-cancelation and write-pausing schemes in [43]. In these schemes, an on-going write can be canceled or paused, giving way to a subsequent read request to improve read latencies. The write request is restarted (in case of write-cancelation) or resumed (in case of write-pausing) afterwards. While write-cancelation supports both SLC and MLC, write-pausing only supports MLC with iterative write-and-verify process. In both techniques, only one request is served at a time in each bank, and writes and reads are still in serial but not in parallel. Hence, they do not help in improving the throughput of PCM memory.

In [44], a memory management scheme was proposed by Qureshi *et al.* for MLC PCM, based on the observation that systems are typically over-provisioned in terms of memory capacity although memory requirements vary between workloads. During a phase of low memory usage, it allows some MLC cells to only store a single bit with lower latency. When the workload requires high capacity, these cells can be restored to MLC. Experiments showed that 95% of all memory requests were served in low latency mode.

Because PCM is less susceptible to soft errors, hard errors becomes more important. Schechter *et al.* proposed a PCM-optimized new approach to error correction named Error Correct Pointer (ECP) [48]. ECP exploits the nature of hard errors (permanent and immediately detectable at write time). It corrects errors by permanently encoding the locations of failed cells into a table and assigning cells to replace them. ECP was reported to provide longer lifetimes than previously solutions with equivalent overhead.

In [49], a multi-bit stuck-at fault error recovery scheme called SAFER was proposed by Seong *et al.*. It exploits the key attribute that a failed cell with a stuck-at value is still readable, making it possible to continue using the failed cell to store data and reduce hardware overhead for error recovery. SAFER dynamically partitions a data block and ensures that there is at most one fail bit per partition. It then uses single error correction techniques per partition for error recovery. Comparing to ECP, SAFER can increase the number of recoverable fails and achieves better lifetime with smaller hardware overhead.

Yoon *et al.* proposed another improved scheme called FREE-p to handle both hard and soft errors [68]. Based on a key observation that a deemed dead block still has many functional bits that can store useful information, FREE-p embeds a fine-grain remapping pointer in it. Hence the mapped-out block (which is otherwise useless) is used as free storage for remapping information. FREE-p was also reported to achieve better lifetime improvement over ECP.

In [18], a power budgeting technique called "power token" was proposed by Hay *et al.*. The technique ensures that the number of concurrent bit-writes does not exceed a power budget, which is defined by the memory interface. To estimate power demands with minimum memory traffic overhead, it approximately counts the number of bit changes in each write with the help of last-level cache in a coarse granularity of 3 bits. Essentially, the power token scheme is a power gating technique enhanced with conservative bit change estimations.

Focused on the excessively long write latency problem of the iterative write scheme in MLC PCM, Jiang *et al.* proposed two architectural innovations in [22]. The write truncation (WT) design reduces the number of write iterations with the assistance of an extra error correction code (ECC). The form switch (FS) design reduces the storage overhead of the

ECC for WT. By storing highly compressible lines in SLC form, FS improves read latency as well.

Also for MLC PCM, in [24] Jiang *et al.* proposed Fine-grained write Power Budgeting (FPB) to improve write throughput. First, iteration power management (FPB-IPM) observes a global power budget and regulates power across write iterations according to the step-down power demand of each iteration. Second, FPB-GCP integrates a global charge pump on a DIMM to boost power for hot PCM chips while staying within the global power budget. These schemes also interact positively with PCM effective read latency reduction techniques, e.g. write-cancellation/pausing [43] and write truncation [22].

Prior art and my work. My implemented Differential Write technique was one of the early attempts of improving PCM as main memory [75, 31]. It is a bit-level circuit scheme tightly integrated into the PCM core. In contrast, majority of other energy reduction and lifetime improvement techniques are at much higher levels with much coarser granularities, aiming at reducing number of write accesses into PCM, and/or moving large data blocks around. Also, thanks to its fundamentality and simplicity, Differential Write can be seemlessly integrated into upper level techniques, and it is actually the basis of some later studies, including Flip-N-Write [8], power token [18], Bit-level Power Budgeting (BPB) [78], etc. Although Differential Write does not ensure an upper bound on number of actual bit-writes as Flip-N-Write does, it can provide accurate fine-grain information about the power demand of each write request, leading to larger power control opportunities especially with limited power budget [78, 18].

While many efforts have been spent on write energy and endurance problems, another write induced challenge – low write throughput, largely remains untouched. Because of its device and circuit level origins, write throughput problem of PCM cannot be tackled by architectural techniques directly. A DRAM/PCM hybrid design [41] can only improve the overall throughput of the entire hybrid memory system, but not PCM throughput. The write-cancellation and write-pause techniques [43] can only help in improving read latency, but not memory throughput because writes and reads are still exclusive to each other and must be served in serial. In contrast, as a circuit and micro-architecture level technique, my Pseudo-Multi-Port Bank design targets the essential shortcomings of PCM and can achieve substantial throughput improvement by exploiting intra-bank or sub-array level parallelism. Moreover, it is orthogonal to and easily combinable with device level evolutions (e.g. write current scaling [9]) and circuit level solutions (e.g. Differential Write, Flip-N-Write [8]), and also provides prominent potential for architectural level enhancements [78].

7.2 STT-MRAM

7.2.1 Prototype Chips & Circuit Designs

The distinguished switching mechanism of "spin torque transfer magnetization switching" was introduced by Hosomi *et al.* in [19] for the first time. While read system remains the same, the only main difference of this new STT-MRAM to a conventional toggle MRAM is in write operation mechanism that the previously necessary external magnetic field is eliminated. This has been accomplished owing to their tailored Magnetic Tunnel Junction (MTJ) with an oval shape of $100 \times 150nm$. Successful memory operations and promising characteristics were demonstrated confidently with the fabricated 4Kb prototype on a 4 level metal, $0.18\mu m$ CMOS process with 1 transistor + 1 MTJ cell structure. The features of this new programming mode and MTJ are thoroughly investigated, including MTJ shape/size effect, dependence of write pulse width on switching current, endurance, and scalability.

In [27], Kawahara *et al.* presented a $1.8V$ $0.2\mu m$ 2Mb STT-MRAM. This chip features 40% cell efficiency with 256Kb sub-arrays each consists of two 128Kb parts separated by the sense amplifiers (SA) and write drivers (WD) in the middle. During write, the WDs handle one side of the two 128Kb parts and the SAs are used as latches. During read, an open-bit architecture is configured and SAs use both 128Kb parts with dummy cells. The choice of read current direction is discussed in detail. Parallel-direction reading was chosen based on the concurrent considerations of minimizing read disturbance and maximizing sensing margin. The tradeoff between TMR ratio, sensing margin, and read voltage was also studied.

A $45nm$ 32Mb embedded STT-MRAM was presented by Lin *et al.* in [36]. This design was implemented in a standard CMOS logic platform that employs low-power (LP) transistors and Cu/low-k BEOL. The common cell structure, in which the free layer of MTJ is connected to BL and the reference layer is attached to the access transistor, suffers from a severe *source-degeneration* effect for the P→AP switching, causing $I_{MTJ(P\rightarrow AP)}$ significantly smaller than $I_{MTJ(AP\rightarrow P)}$. Unfortunately, on the switching current side, $I_{C(P\rightarrow AP)}$ is usually 20∼50% larger than $I_{C(AP\rightarrow P)}$. Therefore, the P→AP switching is difficult to achieve for such conventional cell. To mitigate this problem, a novel "reverse-connection" cell was developed that the connections between free/reference layers and BL/transistor are flipped. As a result, the source-degeneration effect influences only the less demanding AP→P switching. Device attributes and design windows have been examined by considering PVT variations of nine physical and operating parameters to secure operating margins.

Using modified high-density DRAM processes, the smallest bit cell to date of $14F^2$ was achieved by Chung *et al.* at the $54nm$ technology node [13]. The $14F^2$ (WL×BL=$3.5F\times 4F$) cell features 2 transistor + 1 MTJ structure with dual-WL. The 64Mb chip consists of 64Kb sub-arrays that share the globally placed SAs and WDs. To achieve sufficient operation margin, FinFET cell transistor and low resistive W-based SL and BL are utilized to increase current drivability and reduce parasitic resistances in the limited area. Based on the scaling trend and the statistical analysis of cell switching behavior, the authors estimated that the unit cell dimension below $30nm$ can be smaller than $8F^2$.

Taking advantage of lower switching currents of the perpendicular MTJ device, a $1.2V$ $65nm$ 64Mb STT-MRAM was presented by Tsuchida *et al.* in [61]. The read operation relies on a clamped-reference system including the SA, clamping voltage generators, and the reference cell located in an adjacent idle sub-array. All the reference cells are initially set as parallelized state, and their bias condition in read operation is also a weak parallelizing writing condition. Therefore, even though the access duty of reference cells is highest, read disturbance is always absent in them. In addition, an adequate-reference scheme was proposed to improve sensing margin, or relax the requirement on TMR ratio, by limiting the coupling noises on the long signal routing from the reference generator to SA. Regardless of column address, same reference cell is always selected and it can be changed by the ROM fuse programming in case of large variation. Also, the column switch transistors for BL and $\overline{\text{BL}}$ are placed in the opposite sides of the sub-array, such that the total BL resistance along the write or read current path can be constant regardless of the memory cell position. Such design was then re-implemented in [70].

In [16], Halupka *et al.* proposed novel negative-resistance read and write schemes for STT-MRAM in 0.13μm CMOS. The negative-resistance read scheme (NRRS) is to avoid the tradeoff between sense voltage and read margin, and to guarantee a non-destructive read. It shunt the MTJ with a negative resistance (-R) that dynamically allocates current to the MTJ depending on its state. The -R was chosen such that -R||R$_{AP}$ is negative while -R||R$_{P}$ is positive. A negative net resistance in parallel with the source-line capacitance makes an unstable system, while a positive net resistance makes a stable system. A small initial voltage causes V_{MTJ} to exponentially grow to V_{DD} in the unstable system, and decay to ground in the stable system, thus sensing the MTJ state and reading the stored bit. The negative-resistance write scheme (NRWS) saves power during write by moderating the current though the MTJ as its resistance drops from high R$_{AP}$ to low R$_{P}$. If a cell has R$_{AP}$, the driver exponentially increases I_{MTJ} as in an unstable system, until he MTJ switches its state. Then, I_{MTJ} exponentially decays as in a stable system, saving power. If a cell has R$_{P}$, I_{MTJ} decays right away.

In [57, 58], a 1.8V 0.15μm 32Mb STT-MRAM was demonstrated by Takemura *et al.*. Multiple circuit techniques for high-density STT-MRAM was proposed. First, the 2T1R (2 transistors + 1 MTJ) cell structure is shown to occupy smaller area than 1T1R cell with the same access transistor width. Second, to reduce effects of bitline/source-line parasitic resistance, localized bi-directional write drivers are used in combination with hierarchical BL/SL configuration. Hierarchical WL is also adopted. Third, to achieve fast reference level generation and make read operation stable against the temperature dependency of resistance, a '1'/'0' dual-array equalized reference scheme is proposed. Fourth, a disruptive reading and restoration scheme is developed. The basic idea is to improve the sensing margin and speed up the reading by utilizing large read current, and then perform restore when closing the row to overcome possible read disturbance caused by the large read current.

Targeting the reliability limitation of MTJ, Yu *et al.* proposed a cycling endurance optimization scheme [70] for a 1.1V 40nm 1Mb STT-MRAM. The required MTJ endurance may not be achieved if the MTJ is overstressed by the write voltage such that MgO thin-film breakdown may occur. To deal with this, the write path is redesigned with wire-resistance balance scheme, in which the current source and current sink are not placed on the same side but on opposite sides of an array. As a result, voltage stress on the cells near the write buffer is minimized, and voltage across MTJ becomes more uniform for cells from top to bottom of an array. Unfortunately, this technique is not novel as claimed by the authors, because it was already proposed in [61].

Prior art and my work. Previous prototype chip designs have provided me with valuable insights of STT-MRAM characteristics and operation mechanisms, key parameters and guidelines used in my studies, and motivated my research work. Besides our Early Write Termination design, the negative-resistance write scheme (NRWS) [16] is the only other circuit technique that actively saves write energy. However, it only handles parallelizing write and still use conventional scheme for anti-parallelizing write. On the other hand, as the wire dominance was reported in multiple prototype chips [27, 13, 57], my Common-Source-Line Array design offers a practical solution for area saving. Furthermore, in addition to the 1T1R cell assumed as our baseline, Common-Source-Line Array design is also applicable to 2T1R cell [57, 61, 13] which shows more efficient cell layout but severer wire pitch dominance due to its folded transistor.

7.2.2 Architectural Innovations

In [55], Sun *et al.* observed that directly stacking STT-MRAM atop CMPs as L2 cache might harm chip performance, due to its long write latency and high write energy. To solve this problem, an SRAM-MRAM hybrid L2 cache was proposed in which each cache set consists of a majority of STT-MRAM ways and a minority of SRAM ones. By keeping write intensive data in the SRAM part as much as possible and thus reducing the number of STT-MRAM writes, it takes advantage of both SRAM's low latency and STT-MRAM's high density. A read-preemption scheme was also proposed to allow read operations to terminate on-going write operations for the purpose of performance improvement and power reduction.

Also targeting the high write energy problem, in [45] Rasquinha *et al.* proposed policies that prevent premature eviction of lines from higher level caches to lower level STT-MRAM caches. The idea is to increase the residency of dirty lines in the L1/L2 to accommodate (ideally) all the stores to that line. This would prevent the line from being prematurely evicted to the STT-MRAM L2/L3 and being subsequently moved back to the L1/L2 on a near term store miss, at the penalty of increasing the read miss rate.

Though people usually refer STT-MRAM as a non-volatile technology, it is also possible to trade its non-volatility (i.e. its retention time) for better write performance and lower write energy, as studied by Smullen *et al.* in [52]. The planar area of the MTJ device is reduced to achieve better writability at the expense of lower retention time. Since ultra-low retention STT-MRAM may lose data, a simple DRAM-style refresh scheme was also proposed. This study also showed that a hybrid cache hierarchy of SRAM-based L1 with reduced-retention STT-MRAM L2 and L3 eliminates performance loss while still reducing the energy-delay product by over 70%.

Following the same idea, Sun *et al.* [56] extented the use of STT-MRAM to L1 cache as well as lower level ones. Their designs use STT-MRAM cells with various data retention times and write performances, also made possible by tuned MTJ designs. A counter controlled dynamic refresh scheme was proposed to save refresh energy over the simple scheme in [52]. For lower level caches with relatively large capacity, a data migration scheme was proposed to move data between portions of the cache with different retention characteristics so as to maximize the performance and power benefits.

In [38], Mishra *et al.* proposed solutions at the on-chip network level to circumvent the write overhead problem of the STT-MRAM cache in a 3D multi-core environment. The scheme is based on the observation that instead of staggering requests to a write-busy STT-MRAM bank, the network should schedule requests to other idle banks for effectively hiding the latency. This is made possible by 1) accurately estimating the busy time of each cache bank through logical partitioning of the cache layer and 2) prioritizing packets in a router requesting accesses to idle banks by delaying accesses to the STT-MRAM banks that are currently serving long latency write requests.

Prior art and my work. My implemented Early Write Termination technique was one of the early attempts of reducing STT-MRAM write energy [76]. It is a bit-level circuit scheme tightly integrated into the STT-MRAM core. In contrast, majority of other energy reduction techniques are at much higher levels with much coarser granularities, aiming at reducing number of write accesses into STT-MRAM. Also, thanks to its fundamentality and simplicity, Early Write Termination can be easily combined with other techniques (e.g. volatile write [52, 56]) to achieve even higher energy reduction.

On the other hand, because of its device and circuit level origins, density and scalability of STT-MRAM cannot be tackled by architectural techniques. Nevertheless, architecture designs can definitely take advantage of the benefits (e.g. higher density), and exploit the new potentials (e.g. new write schemes), offered by my Common-Source-Line Array design.

8.0 CONCLUSION & FUTURE WORK

8.1 FUTURE WORK

Although the proposed techniques in this dissertation have made significant improvements to PCM and STT-MRAM, there still remain some challenges or opportunities that are worth exploration, especially in future generations at scaled technology nodes.

8.1.1 PCM

PCM operations rely on high voltages to generate high operation currents and minimize parasitic resistances in serial current paths [26, 9]. With technology scaling, such operation voltages are not likely to scale due to the migration of access device to diode [29] and the increased parasitic resistances in high density array structures [9]. However, supply voltage keeps scaling over generations [81], conflicting with the operation voltage trend. This increasing gap between supply voltage and operation voltage is bridged by charge pump circuits.

In majority of PCM prototypes, charge pumps are used to maintain multiple boosted power supplies on-chip. Especially, designing charge pumps for the write driver is challenging because they have to supply high current and sustain high voltage at the same time [26]. There are mainly 3 problems associated with the utilization of charge pumps.

First, charge pumps work at *limited power efficiencies.* For example, in [29] the charge pump that supplies the write current has only 20% efficiency. In other words, 80% of power/energy is wasted during write, placing much heavier burden on the chip supply. Notice that power efficiency of a given charge pump is inversely related to its current load [67].

Second, to reduce standby currents and ensure device reliability by minimizing the duration of high-voltage exposure, the boosted supplies are usually discharged to V_{DD} (or another boosted "base" voltage [29]) after serving a request, and charged back up upon next request. These operations are very expensive in delay and energy. Requests need to wait for the kick-ups of these supplies to proceed [9], leading to additional latency of about 200∼300ns [26, 29]. Also, discharging these boosted voltages after each request is a considerable waste of the energy stored in the huge capacitances of these supply networks.

Third, because of the huge current loads and the coexistence of multiple supplies, charge pumps occupy significant chip area in high density PCM chips [14, 9]. And their area is dominated by the large internal capacitors built out of expensive MIM or MOS capacitors.

With Differential Write and Flip-N-Write [8] techniques that reduce number of bit-writes, the existing charge pumps become over-designed. Although they can regulate themselves with respect to the variable current load, their silicon area occupation can not change. Therefore, it is more meaningful to either shrink the charge pump design (e.g. reduce number of pump-units [1]) to save area, or support wider parallel writes with existing design.

It is also beneficial to make smart use of charge pumps by exploiting *temporal locality* of requests. If requests come back-to-back, or the predicted interval between requests is small, charge pumps can be left active between requests, so the kick-up delay is removed for the later request, and the saved discharging/charging energy can outweight the standby energy spent to keep charge pumps active. If the arrival time of next request after a long interval is predicted, charge pumps can be activated beforehand, saving the waiting time for kick-up. Moreover, intentionally enhancing the temporal locality further extends the potential. For example, by adding a write buffer and accumulating write requests in it, write requests can be issued into PCM in a burst when the buffer becomes full. Therefore the kick-up only happens once to serve many write requests, eliminating considerable latency and energy that are otherwise spent to drag the boosted supplies up and down. By incorporating my Pseudo-Multi-Port Bank design, the burst writes will not block incoming reads and performance impact is minimized. On the other hand, because Pseudo-Multi-Port Bank

[1]A charge pump system is built with multiple sub-pumps or pump-units connected in parallel, each contributes to a portion of the load current.

design can increase parallelism and thus reduce the total service time, standby intervals become longer giving charge pumps more opportunities to be powered off.

Another challenge associated with charge pump control and high voltage operations is the *cell location dependency*. Due to the IR drop on the highly resistive write/read current paths, a cell that is physically far away from the WD/SA will receive a smaller signal than a closer cell [7, 40]. This is especially bad for write, because the amount of write current directly determines the programmed cell resistance [9], and the closer cells may suffer from worse lifetime caused by over-programming [40]. Existing location compensation technique [40] divides the BL into 4 segments, and applys different voltages between segments but same voltage within a segment. However, this is at a coarse granularity and location dependency still exists within a segment. With longer current paths in high density array structures and worse parasitic resistances in highly scaled technology nodes, finer granularity charge pump tuning and other control techniques are necessary.

8.1.2 STT-MRAM

The excellent scalability of STT-MRAM lies in its aggressive reduction of switching current with technology scaling and device improvements, as shown in Figure 74 [81, 12, 1]. As STT-MRAM features current-driven operations instead of voltage-mode/charge-based operations in SRAM and DRAM, such current scaling in turn determines the voltage scaling, because the minimum voltage to operate STT-MRAM heavily depends on its current profiles.

Therefore, one significant advantage of scaled STT-MRAM is its ability to sustain lower voltages, or V_{DD-min}. Although lower voltage leads to longer latency, it also implies much lower power. This is especially helpful when memory load/activity is low and latency is less important. On the other hand, when performance is critical and power is less a concern, higher voltage could be applied. In STT-MRAM, performance benefits not only come from circuit latency's intrinsic dependency on V_{DD}, more importantly, the switching time of MTJ is a strong function of write current resulting from the V_{DD} applied [19, 58, 12]. With higher voltage, the access transistor can supply larger current, so a much shorter write pulse could be adequate to switch the MTJ, further reducing latency considerably.

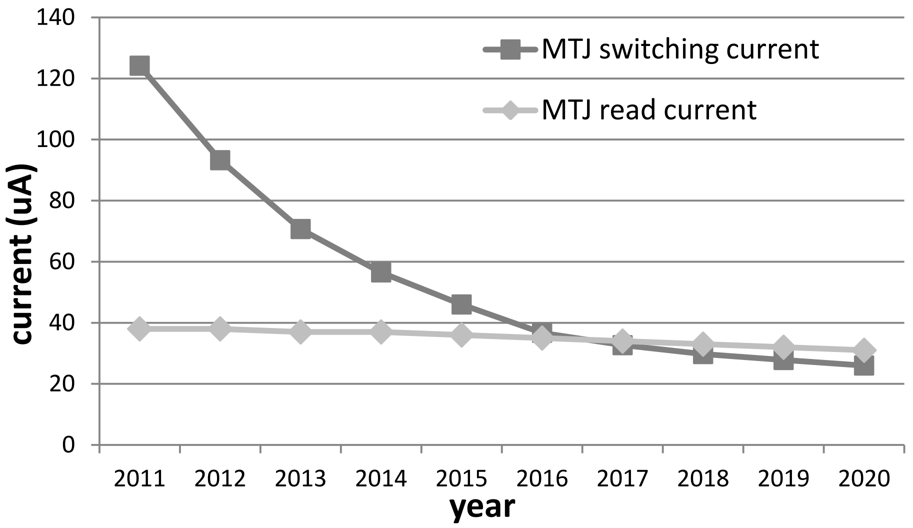


Figure 74: STT-MRAM current scaling trends.

As a result, STT-MRAM enjoys the potential to perform wide range of DVFS for the tradeoff between power and performance, which is not possible in SRAM and eDRAM based large caches. This is because of the high V_{DD-min} of the high density cells used in SRAM large caches, and DRAM's intrinsic dependency of retention time and sensing margin on supply voltage. Therefore, they are usually attached to a separate power supply that is not dynamically scaled. Using STT-MRAM, various DVFS management schemes could be designed to maximize power saving, maximize performance improvement, or to find the balance between power and performance for each individual application.

On the other hand, unlike the rapidly reducing switching current, STT-MRAM read current should generally remain constant to maintain reasonable sensing margin and delay [58]. Although improvment of Tunneling Magnetoresistance Ratio (TMR) can yield better sensing margin and delay [12, 1], read current is in turn bounded by the increasing process variations. Therefore, read current is projected to scale mildly over process generations [58], shown in Figure 74.

Recall that both read and write operations of STT-MRAM rely on flowing current through the MTJ device, and the only difference between them is the amount of current. When the amount of read current approaches switching current, the probability that the state of MTJ could be switched by the read current increases rapidly [58, 12], even though read pulse is generally shorter than write pulse and there is a dependency of switching current on pulse width [19]. Historically, switching current was quite large so a decent gap between switching current and read current was easily guaranteed. However, with the aggressive scaling of switching current and the mild scaling of read current, read current will soon reach the same magnitude of switching current and will finally exceed it [58], as shown in Figure 74. The *read disturbance* problem will be a crucial system-level concern when the disturbance probability is too high to be handled by any error correcting codes with reasonable overheads. And this is very likely to happen when read current exceeds switching current beyond 2017 node in Figure 74.

An intuitive and effective solution to fight read disturbance is to perform restore after each read operation, which basically writes the read data back to the cells [58]. Fortunately, because MTJ is a bipolar device that requires bi-directional currents to switch between states, and the read current direction is fixed at design time, only the cells in one of the two states could be possibly disturbed. For example, if read current is in the same direction of parallizing write, only the anti-parallelized cells could possibly be disturbed to parallelized state. As a result, restores are only necessary on these anti-parallelized cells. Therefore, based on the probabilities of logic '0' and '1' in stored data, an optimal encoding between logic '0'/'1' and parallelized/anti-parallelized states could be found, with respect to the read configuration, so that the number of restore bits and restore energy can be minimized. However, the latency overhead of restore cannot be hid. So intelligent management schemes can be developed to mitigate the overheads of restore, possibly by taking advantages of some error correction techniques to weaken the necessity of restore.

8.2 CONCLUSION

With the growing demand for high capacity, low power, scalability, and reliability, SRAM based large caches and DRAM based main memories are facing serious crises with the shrinking of process feature size, and alternatives are therefore needed. Although considered promising, both Phase Change Memory (PCM) and Spin-Torque-Transfer Magnetic-RAM (STT-MRAM) were born with disadvantages. Without proper improvements, the successful application of these emerging memories, in replacements of traditional SRAM and DRAM, could not be achievable.

Improvements can be carried out at different design levels: device, circuit, architecture. In this dissertation, I propose multiple circuit level solutions at bit and array granularities, targeting some unique but correlated drawbacks of PCM and STT-MRAM. For PCM, the bit level technique Differential Write greatly reduces write energy and extends lifetime of PCM by skipping redundant bit-writes. Following the same principle but different mechanism, Early Write Termination is a corresponding bit level technique that tackles the write energy problem of STT-MRAM, without the expense of performance overhead. In contrast to the similar bit level problems, at array level, PCM enjoys high density but suffers low throughput, while STT-MRAM possesses satisfying throughput but limited density and scalability. The Pseudo-Multi-Port Bank design for high density PCM exploits intra-bank parallelism to boost throughput with minimum overhead. The Common-Source-Line Array design liberates the scaling potential of STT-MRAM by removing the wiring dominance. Therefore, the main contribution of this dissertation is the proposal and evaluation of this comprehensive set of circuit techniques that offer substantial and efficient enhancements to both PCM and STT-MRAM.

On the other hand, it is also clear that besides the significant achievements made by the proposed designs, rooms still remain for further improvements. This is because the effective integration and utilization of a new memory technology actually rely on the systematic incorporation of efforts from different levels. Fortunately, all of the proposed techniques are orthogonal to and easily combinable with device and architectural level innovations to be part of a powerful, comprehensive solution. Moreover, with the evolution of PCM and

STT-MRAM technologies, new challenges and opportunities that are worth exploration may emerge, especially in future generations at scaled technology nodes.

Furthermore, we should keep in mind that solving problems, as my dissertation and many other research work did, is only part of the story. Being optimistic/excited with, and making good use of the unique advantages and potentials provided by these new memories, is equally important. After all my comprehensive studies and research efforts, I do believe in the bright future of PCM and STT-MRAM as next-generation memories.

BIBLIOGRAPHY

[1] D. Apalkov, *et al.*, "Spin-Transfer Torque Magnetic Random Access Memory (STT-MRAM)", *ACM Journal on Emerging Technologies in Computing Systems*, Vol. 9, No. 2, 2013.

[2] J. Barth, *et al.*, "A 45nm SOI Embedded DRAM Macro for POWER7 32MB On-Chip L3 Cache", *International Solid-State Circuits Conference*, pp. 342-343, 2010.

[3] F. Bedeschi, *et al.*, "4-Mb MOSFET-Selected μTrench Phase-Change Memory Experimental Chip", *IEEE Journal of Solid-State Circuits*, Vol. 40, No. 7, pp. 1557-1565, 2005.

[4] F. Bedeschi, *et al.*, "A Bipolar-Selected Phase Change Memory Featuring Multi-Level Cell Storage", *IEEE Journal of Solid-State Circuits*, Vol. 44, No. 1, pp. 217-227, 2009.

[5] M.-T. Chang, P. Rosenfeld, S.-L. Lu, B. Jacob, "Technology Comparison for Large Last-Level Caches (L^3Cs): Low-Leakage SRAM, Low Write-Energy STT-RAM, and Refresh-Optimized eDRAM", *International Symposium on High Performance Computer Architecture*, pp. 143-154, 2013.

[6] Y. Chen, X. Wang, H. Li, H. Liu, D. V. Dimitrov, "Design Margin Exploration of Spin-Torque Transfer RAM (SPRAM)", *International Symposium on Quality Electronic Design,* pp. 684-690, 2008.

[7] W. Y. Cho, *et al.*, "A 0.18-μm 3.0-V 64-Mb Nonvolatile Phase-Transition Random Access Memory (PRAM)", *IEEE Journal of Solid-State Circuits*, Vol. 40, No. 1, pp. 293-300, 2005.

[8] S. Cho, H. Lee, "Flip-N-Write: A Simple Deterministic Technique to Improve PRAM Write Performance, Energy and Endurance", *International Symposium on Microarchitecture*, pp. 347-357, 2009.

[9] Y. Choi, *et al.*, "A 20nm 1.8V 8Gb PRAM with 40MB/s Program Bandwidth", *International Solid-State Circuits Conference*, pp. 46-47, 2012.

[10] L. Chua, "Memristor - The Missing Circuit Element", *IEEE Transactions on Circuit Theory*, Vol. 18, No. 5, pp. 507-519, 1971.

[11] K. C. Chun, P. Jain, T.-H. Kim, C. H. Kim, "A 667 MHz Logic-Compatible Embedded DRAM Featuring an Asymmetric 2T Gain Cell for High Speed On-Die Caches", *IEEE Journal of Solid-State Circuits*, Vol. 47, No. 2, pp. 547-559, 2012.

[12] K. C. Chun, *et al.*, "A Scaling Roadmap and Performance Evaluation of In-Plane and Perpendicular MTJ Based STT-MRAMs for High-Density Cache Memory", *IEEE Journal of Solid-State Circuits*, Vol. 48, No. 2, pp. 598-610, 2013.

[13] S. Chung, *et al.*, "Fully Integrated 54nm STT-RAM with the Smallest Bit Cell Dimension for High Density Memory Application", *International Electron Devices Meeting*, pp. 304-307, 2010.

[14] H. Chung, *et al.*, "A 58nm 1.8V 1Gb PRAM with 6.4MB/s Program BW", *International Solid-State Circuits Conference*, pp. 500-501, 2011.

[15] M. Gill, T. Lowrey, J. Park, "Ovonic Unified Memory - A High-Performance Nonvolatile Memory Technology for Stand-Alone Memory and Embedded Applications", *International Solid-State Circuits Conference*, pp. 202-203, 2002.

[16] D. Halupka, *et al.*, "Negative-Resistance Read and Write Schemes for STT-MRAM in 0.13μm CMOS", *International Solid-State Circuits Conference*, pp. 256-257, 2010.

[17] S. Hanzawa, *et al.*, "A 512kB Embedded Phase Change Memory with 416kB/s Write Throughput at 100μA Cell Write Current", *International Solid-State Circuits Conference*, pp. 474-475, 2007.

[18] A. Hay, K. Strauss, T. Sherwood, G. H. Loh, D. Burger, "Preventing PCM Banks from Seizing Too Much Power", *International Symposium on Microarchitecture*, pp. 186-195, 2011.

[19] M. Hosomi, *et al.*, "A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM", *International Electron Devices Meeting*, pp. 459-462, 2005.

[20] C. Hu, "Gate Oxide Scaling Limits and Projection", *International Electron Devices Meeting*, pp. 319-322, 1996.

[21] D. Huang, W. Li, "On CMOS Exclusive OR Design", *Midwest Symposium on Circuits and Systems*, pp. 829-832, 1989.

[22] L. Jiang, B. Zhao, Y. Zhang, J. Yang, B. R. Childers, "Improving Write Operations in MLC Phase Change Memory", *International Symposium on High-Performance Computer Architecture*, pp. 1-10, 2012.

[23] L. Jiang, B. Zhao, Y. Zhang, J. Yang, "Constructing Large and Fast Multi-level Cell STT-MRAM based Cache for Embedded Processors", *Design Automation Conference*, pp. 907-912, 2012.

[24] L. Jiang, Y. Zhang, B. R. Childers, J. Yang, "FPB: Fine-grained Power Budgeting to Improve Write Throughput of Multi-level Cell Phase Change Memory", *International Symposium on Microarchitecture*, pp. 1-12, 2012.

[25] L. Jiang, Y. Du, B. Zhao, Y. Zhang, B. R. Childers, J. Yang, "Hardware Assisted Cooperative Integration of Wear-Leveling and Salvaging for Phase Change Memory", *ACM Transactions on Architecture and Code Optimization*, Vol. 10, No. 2, 2013.

[26] S. Kang, *et al.*, "A 0.1-μm 1.8-V 256-Mb Phase-Change Random Access Memory (PRAM) With 66-MHz Synchronous Burst-Read Operation", *IEEE Journal of Solid-State Circuits*, Vol. 42, No. 1, pp. 210-218, 2007.

[27] T. Kawahara, *et al.*, "2Mb SPRAM (SPin-Transfer Torque RAM) with Bit-by-Bit Bidirectional Current Write and Paralilelizing-Direction Current Read", *IEEE Journal of Solid-State Circuits*, Vol. 43, No. 1, pp. 109-120, 2008.

[28] T. Kishi, *et al.*, "Lower-current and Fast switching of A Perpendicular TMR for High Speed and High density Spin-Transfer-Torque MRAM", *International Electron Devices Meeting*, pp. 1-4, 2008.

[29] K.-J. Lee, *et al.*, "A 90 nm 1.8 V 512 Mb Diode-Switch PRAM With 266 MB/s Read Throughput", *IEEE Journal of Solid-State Circuits*, Vol. 43, No. 1, pp. 150-162, 2008.

[30] B. C. Lee, E. Ipek, D. Burger, "Architecting Phase Change Memory as a Scalable DRAM Alternative", *International Symposium on Computer Architecture*, pp. 2-13, 2009.

[31] B. C. Lee, P. Zhou, J. Yang, Y. Zhang, B. Zhao, E. Ipek, O. Mutlu, D. Burger, "Phase Change Technology and the Future of Main Memory", *IEEE MICRO, Special Issue: Micro's Top Picks from 2009 Computer Architecture Conferences*, Vol. 30, No. 1, pp. 131-141, 2010.

[32] K. M. Lepak, M. H. Lipasti, "On the Value Locality of Store Instructions", *International Symposium on Computer Architecture*, pp. 182-191, 2000.

[33] K. M. Lepak, M. H. Lipasti, "Silent Stores for Free", *International Symposium on Microarchitecture*, pp. 22-31, 2000.

[34] H. Li, Y. Chen, "An Overview of Non-Volatile Memory Technology and the Implication for Tools and Architectures", *Conference on Design, Automation and Test in Europe*, pp. 731-736, 2009.

[35] H. Li, Y. Chen, H. Liu, X. Wang, "Static Source Plane in ST-RAM", *US Patent 7,859,891.*

[36] C. J. Lin, *et al.*, "45nm Low Power CMOS Logic Compatible Embedded STT MRAM Utilizing a Reverse-Connection 1T/1MTJ Cell", *International Electron Devices Meeting*, pp. 279-282, 2009.

[37] N. Lu, *et al.*, "A Substrate-Plate Trench-Capacitor (SPT) Memory Cell for Dynamic RAM's", *IEEE Journal of Solid-State Circuits*, Vol. 21, No. 5, pp. 627-634, 1986.

[38] A. K. Mishra, X. Dong, G. Sun, Y. Xie, N. Vijaykrishnan, C. R. Das, "Architecting On-Chip Interconnects for Stacked 3D STT-RAM Caches in CMPs", *International Symposium on Computer Architecture*, pp. 69-80, 2011.

[39] R. Nebashi, *et al.*, "A 90nm 12ns 32Mb 2T1MTJ MRAM", *International Solid-State Circuits Conference*, pp. 462-463, 2009.

[40] H. Oh, *et al.*, "Enhanced Write Performance of a 64-Mb Phase-Change Random Access Memory", *IEEE Journal of Solid-State Circuits*, Vol. 41, No. 1, pp. 122-126, 2006.

[41] M. K. Qureshi, V. Srinivasan, J. A. Rivers, "Scalable High Performance Main Memory System Using Phase-Change Memory Technology", *International Symposium on Computer Architecture*, pp. 24-33, 2009.

[42] M. K. Qureshi, J. Karidis, M. Franceschini, V. Srinivasan, L. Lastras, B. Abali, "Enhancing Lifetime and Security of Phase Change Memories via Start-Gap Wear Leveling", *International Symposium on Microarchitecture*, pp. 14-23, 2009.

[43] M. K. Qureshi, M. M. Franceschini, L. A. Lastras-Montano, "Improving Read Performance of Phase Change Memories via Write Cancellation and Write Pausing", *International Symposium on High Performance Computer Architecture*, pp. 1-11, 2010.

[44] M. K. Qureshi, M. M. Franceschini, L. A. Lastras-Montano, J. P. Karidis, "Morphable Memory System: A Robust Architecture for Exploiting Multi-Level Phase Change Memories", *International Symposium on Computer Architecture*, pp. 153-162, 2010.

[45] M. Rasquinha, D. Choudhary, S. Chatterjee, S. Mukhopadhyay, S. Yalamanchili, "An Energy Efficient Cache Design Using Spin Torque Transfer (STT) RAM", *International Symposium on Low Power Electronics and Design*, pp. 389-394, 2010.

[46] N. Sakimura, *et al.*, "A 512Kb Cross-Point Cell MRAM", *International Solid-State Circuits Conference*, pp. 278-279, 2003.

[47] G. D. Sandre, *et al.*, "A 4 Mb LV MOS-Selected Embedded Phase Change Memory in 90 nm Standard CMOS Technology", *IEEE Journal of Solid-State Circuits*, Vol. 46, No. 1, pp. 52-63, 2011.

[48] S. Schechter, G. H. Loh, K. Straus, D. Burger, "Use ECP, not ECC, for Hard Failures in Resistive Memories", *International Symposium on Computer Architecture*, pp. 141-152, 2010.

[49] N. H. Seong, D. H. Woo, H.-H. Lee, "SAFER: Stuck-At-Fault Error Recovery for Memories", *International Symposium on Microarchitecture*, pp. 115-124, 2010.

[50] K. Shakeri, J. Meindl, "Compact Physical IR-Drop Models for Chip/Package Co-Design of Gigascale Integration (GSI)", *IEEE Transactions on Electron Devices*, Vol. 52, No. 6, pp. 1087-1096, 2005.

[51] J. Slaughter, "Recent Advancements in Spin-Torque Switching for High-Density MRAM", *International Symposium on Advanced Gate Stack Technology*, 2010.

[52] C. W. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, M. R. Stan, "Relaxing Non-Volatility for Fast and Energy-Efficient STT-RAM Caches", *International Symposium on High Performance Computer Architecture*, pp. 50-61, 2011.

[53] D. Somasekhar, *et al.*, "Multi-Phase 1 GHz Voltage Doubler Charge Pump in 32 nm Logic Process", *IEEE Journal of Solid-State Circuits*, Vol. 45, No. 4, pp. 751-758, 2010.

[54] D. Strukov, G. Snider, D. Stewart, S. Williams, "The Missing Memristor Found", *Nature*, Vol. 453, pp. 80-83, 2008.

[55] G. Sun, X. Dong, Y. Xie, J. Li, Y. Chen, "A Novel Architecture of the 3D Stacked MRAM L2 Cache for CMPs", *International Symposium on High Performance Computer Architecture*, pp. 239-249, 2009.

[56] Z. Sun, X. Bi, H. Li, W.-F. Wong, Z.-L. Ong, X. Zhu, W. Wu, "Multi Retention Level STT-RAM Cache Designs with a Dynamic Refresh Scheme", *International Symposium on Microarchitecture*, pp. 329-338, 2011.

[57] R. Takemura, *et al.*, "A 32-Mb SPRAM With 2T1R Memory Cell, Localized Bi-Directional Write Driver and '1'/'0' Dual-Array Equalized Reference Scheme", *IEEE Journal of Solid-State Circuits*, Vol. 45, No. 4, pp. 869-879, 2010.

[58] R. Takemura, *et al.*, "Highly-Scalable Disruptive Reading and Restoring Scheme for Gb-Scale SPRAM and Beyond", *Solid-State Electronics*, Vol. 58, No. 1, pp. 28-33, 2011.

[59] S. Taylor, *et al.*, "POWER7+: IBM's Next Generation POWER Microprocessor", *Hot Chips*, 2012.

[60] M. Togo, S. Iwao, H. Nobusawa, M. Hamada, K. Yoshida, N. Yasuzato, T. Tanigawa, "A Salicide-Bridged Trench Capacitor with a Rouble-Sacrificial-Si_3N_4-Sidewall (DSS) for High-Performance Logic-Embedded DRAMs", *International Electron Devices Meeting*, pp. 37-40, 1997.

[61] K. Tsuchida, *et al.*, "A 64Mb MRAM with Clamped-Reference and Adequate-Reference Schemes", *International Solid-State Circuits Conference*, pp. 258-259, 2010.

[62] C. Villa, *et al.*, "A 45nm 1Gb 1.8V Phase-Change Memory", *International Solid-State Circuits Conference*, pp. 270-271, 2010.

[63] T. Vogelsang, "Understanding the Energy Consumption of Dynamic Random Access Memories", *International Symposium on Microarchitecture*, pp. 363-374, 2010.

[64] Y.-T. Wang, B. Razavi, "An 8-Bit 150-MHz CMOS A/D Converter", *IEEE Journal of Solid-State Circuits*, Vol. 35, No. 3, pp. 308-317, 2000.

[65] J. Warnock, *et al.*, "5.5GHz System z Microprocessor and Multichip Module", *International Solid-State Circuits Conference*, pp. 46-47, 2013.

[66] N. Weste, D. Harris, "CMOS VLSI Design: A Circuits and Systems Perspective (4th Edition)", *Addison-Wesley Publishing Company*, 2010.

[67] O. Y. Wong, H. Wong, W. S. Tam, C. W. Kok, "A Comparative Study of Charge Pumping Circuits for Flash Memory Applications", *Microelectronics Reliability*, Vol. 52, No. 4, pp. 670-687, 2012.

[68] D. H. Yoon, N. Muralimanohar, J. Chang, P. Ranganathan, N. P. Jouppi, M. Erez, "FREE-p: Protecting Non-Volatile Memory against both Hard and Soft Errors", *International Symposium on High Performance Computer Architecture*, pp. 466-477, 2011.

[69] M. Yoshida, *et al.*, "An Embedded 0.405 μm^2 Stacked DRAM Technology Integrated with High-performance 0.2 μm CMOS Logic and 6-Level Metalization", *International Electron Devices Meeting*, pp. 41-44, 1999.

[70] H.-C. Yu, *et al.*, "Cycling Endurance Optimization Scheme for 1Mb STT-MRAM in 40nm Technology", *International Solid-State Circuits Conference*, pp. 224-225, 2013.

[71] B. Zhao, Y. Du, Y. Zhang, J. Yang, "Variation-Tolerant Non-Uniform 3D Cache Management in Die Stacked Multicore Processor", *International Symposium on Microarchitecture*, pp. 222-231, 2009

[72] B. Zhao, J. Yang, Y. Zhang, Y. Chen, H. Li, "Architecting a Common-Source-Line Array for Bipolar Non-Volatile Memory Devices", *Design, Automation and Test in Europe*, pp. 1451-1454, 2012.

[73] B. Zhao, J. Yang, Y. Zhang, Y. Chen, H. Li, "Common-Source-Line Array: An Area Efficient Memory Architecture for Bipolar Non-Volatile Devices", *ACM Transactions on Design Automation of Electronic Systems*, Vol. 18, No. 4, 2013.

[74] B. Zhao, Y. Du, J. Yang, Y. Zhang, "Process Variation Aware Non-Uniform Cache Management in 3D Die Stacked Multicore Processor", *IEEE Transactions on Computers*, Vol. 62, No. 11, pp. 2252-2265, 2013.

[75] P. Zhou, B. Zhao, J. Yang, Y. Zhang, "A Durable and Energy Efficient Main Memory Using Phase Change Memory Technology", *International Symposium on Computer Architecture*, pp. 14-23, 2009.

[76] P. Zhou, B. Zhao, J. Yang, Y. Zhang, "Energy Reduction for STT-RAM Using Early Write Termination", *International Conference on Computer-Aided Design*, pp. 264-268, 2009.

[77] P. Zhou, B. Zhao, Y. Zhang, J. Yang, Y. Chen, "MRAC: A Memristor-based Reconfigurable Framework for Adaptive Cache Replacement", *International Conference on Parallel Architectures and Compilation Techniques*, pp. 207-208, 2011.

[78] P. Zhou, B. Zhao, J. Yang, Y. Zhang, "Throughput Enhancement for Phase Change Memories", *IEEE Transactions on Computers*, 2013.

[79] Taiwan Semiconductor Manufacturing Company, $0.18\mu m$ and $0.13\mu m$ process data sheets and device models.

[80] Cadence Design Systems, $45nm$ Generic PDK data sheet and device models.

[81] International Technology Roadmap for Semiconductors, "ITRS Report, 2012 Edition", http://www.itrs.net

[82] Nanoscale Integration and Modeling (NIMO) Group at Arizona State University, "Predictive Technology Model (PTM)", http://ptm.asu.edu

[83] JEDEC, "DDR4 SDRAM Standard (JESD79-4)", September 2012, http://www.jedec.org/standards-documents/docs/jesd79-4

[84] HP Labs, "CACTI", http://www.hpl.hp.com/research/cacti/

[85] Numonyx, "Phase Change Memory and Its Impacts on Memory Hierarchy", Presentation at Carnegie Mellon University, September, 2009, http://www.pdl.cmu.edu/SDI/2009/slides/Numonyx.pdf