Cache Design with Domain Wall Memory

Rangharajan Venkatesan, *Student Member, IEEE*, Vivek J. Kozhikkottu, *Student Member, IEEE*, Mrigank Sharad, *Student Member, IEEE*, Charles Augustine, *Member, IEEE*, Arijit Raychowdhury, *Senior Member, IEEE*, Kaushik Roy, *Fellow, IEEE*, and Anand Raghunathan, *Fellow, IEEE*

Abstract—Domain wall memory (DWM) is a recently developed spin-based memory technology in which several bits of data are densely packed into the domains of a ferromagnetic wire. DWM has shown great promise in enabling non-volatile memory with very high density and energy efficiency, and has been explored for secondary storage and off-chip memory. In this work, we explore the use of DWM within the on-chip cache hierarchy of general purpose computing platforms. Our work is motivated by the fact that DWMs enable much higher density compared to SRAM, DRAM, and other spin-based memory technologies such as STT-MRAM. However, DWMs also pose the unique challenge of serial access to the bits stored in a cell, leading to large and variable access latencies. In addition, DWMs share the inherent write inefficiency of other spin-based memories. We propose TapeCache, a DWM-based cache design that employs device, circuit, and architectural techniques to address these challenges. At the device level, we perform write optimization by employing a new write mechanism based on domain wall shifts to achieve fast, energy-efficient writes in DWM. At the circuit level, we propose different DWM bit-cell designs that are tailored to the distinct architectural requirements of different levels in the cache hierarchy. At the architecture level, we propose a new cache organization and suitable management policies that mitigate the performance penalty arising from serial access to bits in a DWM cell. We show that the holistic device-circuit-architecture co-design enables all the levels in the cache hierarchy to be realized using DWM and benefit from its improved density. Over a wide range of SPEC CPU 2006 benchmarks, TapeCache achieves an average energy improvement of 7.5×, with virtually identical performance and 7.8× improvement in area, compared to an iso-capacity SRAM cache. Compared to an iso-capacity STT-MRAM cache, TapeCache obtains 3.1× improvement in area and 2× average energy savings along with 1.1 percent performance improvement.

Index Terms—Domain wall memory, spintronics, non-volatile memory, cache

1 Introduction

In modern processors, major fractions of the chip transistor count, area, and power are consumed by cache memories. The growing processor-memory gap, driven by increases in on-chip parallelism, along with increasingly complex applications and data sets, fuel an ever-increasing demand for larger caches as illustrated in Fig. 1. Traditionally, SRAM has been the workhorse for cache design. However, with technology scaling, challenges such as increased leakage and process variations increasingly impact SRAM design. Consequently, there is great interest in the on-chip use of emerging memory technologies that have very high density and low leakage power.

Several new memory technologies—Ferroelectric RAM (FeRAM), Phase Change Memory (PCRAM), Spin-Transfer Torque Magnetic RAM (STT-MRAM), and Domain Wall Memory (DWM) have been proposed as potential replacements for various levels of the memory hierarchy. Fig. 2 compares various key metrics for different memory technologies. DWM, a spintronic non-volatile memory technology, can achieve much higher densities with similar access time and idle power compared to other emerging memory technologies. For these reasons, it is considered to be highly promising and several research efforts have focused on understanding its device-level characteristics [1], [2], [3]. In recent years, functioning prototypes of DWM have been demonstrated [4], [5], [6]. Considering the potential of DWM, it is critical to understand the design implications and tradeoffs associated with the use of DWMs in future computing platforms. DWM was initially envisioned as a replacement for secondary storage due to its excellent density. However, recent efforts have suggested the use of DWMs as on-chip memories in the context of domain specific computing platforms [7], [8].

In this work, we propose the use of DWM to realize on-chip caches in general purpose processors. Despite possessing a number of favorable characteristics such as very high density, non-volatility, and low leakage, DWM poses unique challenges compared to all other memory technologies. From an architectural perspective, a DWM device looks like a tape that can store several (upto hundreds of) bits, with a read/write port to access them. Another key characteristic is that the bits stored in a DWM device/tape can be shifted in either direction. This enables the sharing of read/write ports across the bits stored in a tape, resulting in very high density. However, the time taken to access a bit stored in the tape depends on its location relative to the

- R. Venkatesan, V. J. Kozhikkottu, M. Sharad, K. Roy, and A. Raghunathan are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906. E-mail: {rvenkate, vkozhikk, msharad, kaushik, raghunathan}@purdue.edu.
- C. Augustine is with the Circuit Research Labs, Intel Corporation, Hillsboro, OR 97124. E-mail: charles.augustine@intel.com.
- A. Raychowdhury is with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332. E-mail: arijit.raychowdhury@ece.gatech.edu.

Manuscript received 4 Feb. 2014; revised 15 May 2014; accepted 16 July 2014. Date of publication 7 Dec. 2015; date of current version 17 Mar. 2016.
Recommended for acceptance by H. Assadi, P. Ienne, and H. Sarbazi-Azad.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2015.2506581

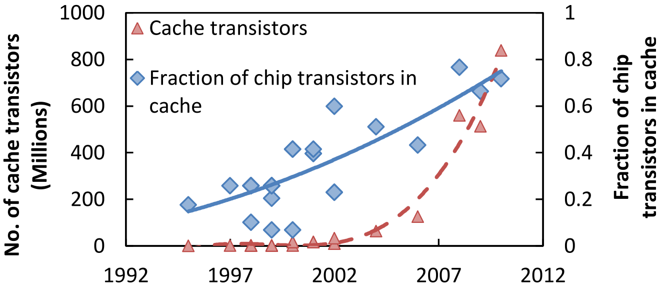


Fig. 1. Growth in on-chip memory (Intel processors).

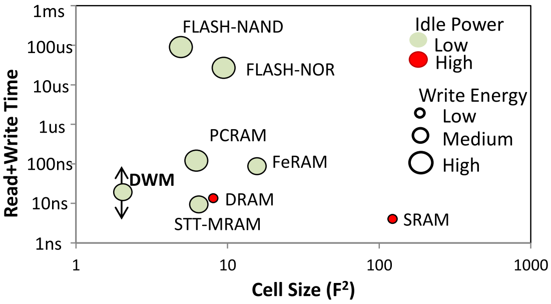


Fig. 2. Comparison of different memory technologies (data from [4]).

read/write port, leading to variable access latencies. For the bit stored closest to the read/write port, the access latency (best case) is lower compared to SRAM/STT-MRAM due to smaller bitline delays resulting from higher density. However, the overall performance of a DWM cache is determined by the average number of shift operations required per access. Hence, realizing a DWM-based cache requires the development of suitable circuit/architecture design techniques that exploit its strengths while reducing the performance penalties associated with shift operations.

On the other hand, DWM, much like other spintronic memories such as STT-MRAM, suffers from inefficient writes. This is because, the read/write port in DWM is typically designed using a magnetic tunneling junction (MTJ) that requires high switching energy and latency. While previous research efforts [9], [10], [11], [12] have proposed write optimizations for MTJ-based writes, the write inefficiency still remains a major bottleneck for realizing on-chip memories. Fortunately, DWM also presents a unique opportunity to optimize writes by utilizing the shift mechanism as an alternative write scheme. In summary, DWM poses the challenge of shift operations that is unique among emerging memory technologies, and it offers a unique solution to the common challenge of inefficient write operations. We pursue both these directions in this work.

The contributions of this work are as follows:

- We explore the design of on-chip caches using DWM. We propose TapeCache, a novel cache design in which all the levels in the cache hierarchy are realized using DWM. We explore two bit-cells that are tailored to the differing needs of different memory arrays within the cache hierarchy. For the latency-sensitive L1 cache, we design both the data and tag arrays using 1bitDWM that is optimized for latency. For the L2 cache, we propose a hybrid organization in which the data array is designed using a combination of 1bitDWM and multibitDWM. In order to determine whether a cache access is a hit or miss with low latency, the tag array is designed using 1bitDWM.
- We design bit-cells that perform write operations using domain wall shift based writes, which are fundamentally different from, and have been experimentally demonstrated to be faster and more energy-efficient than, MTJ-based writes [13].
- We propose circuit and architectural techniques to address the performance penalty arising from shift operations in multibitDWM. Considering the read-write asymmetry and spatial locality of memory accesses found in most applications, we propose (i) a multi-port read-optimized multibitDWM design that is more efficient for reads, (ii) an efficient array organization and suitable cache management policies to maximally harness the performance potential of DWM-based caches.
- We evaluate TapeCache through architectural simulations of benchmarks with diverse read/write characteristics, miss rates and working sets. Our results show that an iso-capacity replacement of SRAM-based cache with TapeCache can result in large benefits in area and energy at iso-performance. We also demonstrate that DWM can significantly outperform STT-MRAM, which is considered to be one of the most promising emerging memory technologies, in the context of on-chip cache design.

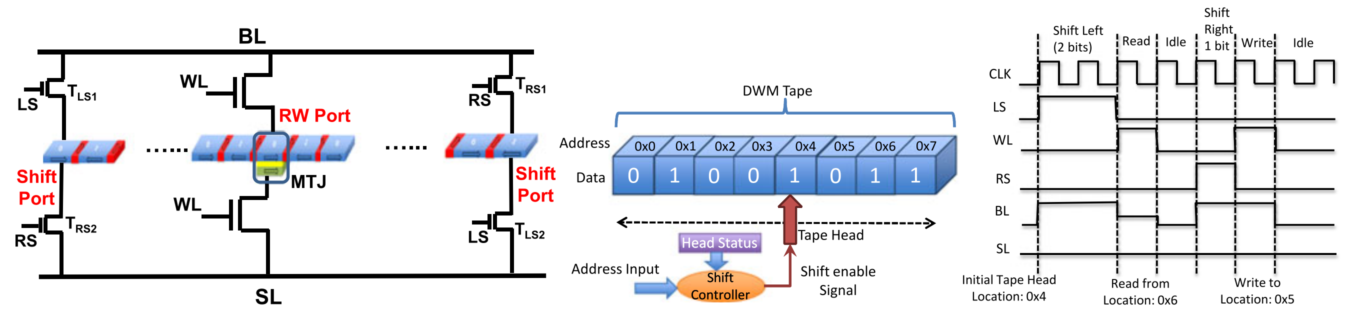
The rest of this paper is organized as follows. Section 2 outlines the fundamental concepts of DWM. Section 3 describes the proposed 1bitDWM and multibitDWM designs and the shift-based write scheme. Section 4 presents the proposed multi-port read-optimized multibitDWM design. Section 5 describes the proposed TapeCache architecture. Section 6 presents our methodology for modeling and evaluating TapeCache. Section 7 presents experimental results. In Section 8, we present a brief survey of related work and Section 9 concludes the paper.

2 Background

Fig. 3a shows the schematic of a multibitDWM cell consisting of a ferromagnetic wire, a MTJ and access transistors. The ferromagnetic wire can have multiple domains that are separated by domain walls. Each domain can be separately programmed to a certain magnetization direction, and can therefore store a single bit. Hence, a multibitDWM bit-cell is capable of storing multiple bits of data.

Logically, a DWM can be thought of as a tape that represents the ferromagnetic wire along with a tape head that represents a read/write port, as shown in Fig. 3b. Accessing a bit from a DWM tape involves shifting the tape head to the required location and then performing the required read/write operation. Since multiple bits are stored in a multibitDWM, address bits are needed to indicate which bit in the tape is being accessed. The movement of the tape head along the DWM tape is controlled by a shift controller, which determines the number of shift operations required by comparing the address bits with the head status, which represents the current location of the tape head.

An illustration of the working of a multibitDWM bit-cell is shown in Fig. 3c. Initially, the tape head is located at



(a) Schematic of a multibitDWM cell (b) Logical view of a multibitDWM cell (c) MultibitDWM operation

Fig. 3. MultibitDWM cell structure and operation.

address 0×4. In order to read the bit stored at address 0×6, we shift the tape head to the right (in reality, bits in the tape shift left. For convenience, we refer to the shifting of the bits in terms of equivalent shifting of the tape head) by two positions. This is done by connecting BL to V_{DD} and SL to GND and turning on access transistors T_{LS1} and T_{LS2} by driving the left-shift wordline LS high. Then we perform the read operation by connecting the wordline WL to V_{DD}, bitlines BL to $0.25V_{DD}$ and SL to the GND. Suppose we then write 0 to address 0×5, which requires shifting the tape head to the left (in reality bits shift right) by one position. This is done by driving the right-shift wordline RS high to shift the bits towards the right and then connecting wordline WL to V_{DD} and bitlines BL to V_{DD} and SL to GND. For writing 1, the voltages of the bitlines would be reversed.

In order to avoid losing data during shift operations, the number of domains in the ferromagnetic wire needs to be twice the number of bits stored in multibitDWM bit-cell. As we show in Section 3.2, this does not incur any area penalty as the multibitDWM bit-cell area is determined by the area occupied by the access transistors.

3 DWM with Shift-Based Writes

In conventional DWM, read/write operations are performed using an MTJ and domain wall shifts are used only to align the appropriate bit with the read/write port. This MTJ-based write mechanism requires high energy as well as latency and is therefore inefficient. Recently, it was shown experimentally that domain wall shifts can also be used to perform write operations [13]. This is accomplished by designing a DWM in which a free domain is sandwiched between two fixed domains having opposite spin orientations, as shown in Fig. 4. The fixed domains act as sources of 0/1, and the free domain can be written by performing a shift in the appropriate direction. Note that the magnetic orientation of the fixed domains remains constant and cannot be modified by the shift operation. A comparison of the switching current and latency requirements of the two write mechanisms in Table 1 underscores the efficiency of shift-based writes in DWM. We apply this concept to 1bitDWM and multibitDWM designs that are optimized for performance and area, respectively. We describe these designs in the following sections.

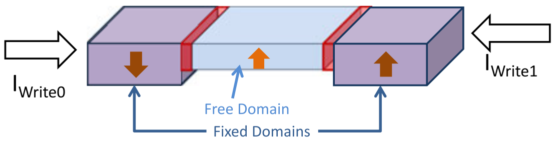


Fig. 4. Domain wall shift based write mechanism.

3.1 1bitDWM

Fig. 5 shows the schematic and layout of the 1bitDWM cell. The cell consists of a ferromagnetic wire, an MTJ and two access transistors. The data stored in the cell is determined by the magnetic orientation of the free domain. When the magnetic orientation of the free domain is parallel to that of the MTJ fixed layer, the MTJ offers low resistance, indicating the '0' state. When the magnetic orientations are anti-parallel, the MTJ offers high resistance, indicating the '1' state. The bit-cell also contains two separate access transistors—a read access transistor (T1) and a write access transistor (T2), which are used to control the direction of currents during the read/write operations.

3.1.1 Read/Write Operation

The voltage conditions for performing the read and write operations are presented in Table 2. In order to read the contents of the cell, the read access transistor (T1) is turned ON and the bitline BL is driven to appropriate read voltage (V_{read}). The current that flows from BL to GND varies depending on the resistance offered by the MTJ, and is used to determine the value stored in the cell. The write operation in the proposed design is performed by shifting the appropriate magnetization from the fixed domains to the free domain of the ferromagnetic wire. In order to write 0, write access transistor (T2) is turned ON, bitline BL is driven high and SL is connected to GND. This shifts the domains towards the left, thereby writing 0 into the bit-cell. For writing 1, the voltage conditions of the bitlines are reversed.

Note that in the layout of the 1bitDWM cell, the DWM device and the access transistors are stacked on top of each other in a 3D fashion. The area of the bit-cell is determined

TABLE 1
Characteristics of MTJ and Shift Based Writes [14][1]

Parameter	MTJ-based write	Shift-based write
Switching current (uA)	191	30
Switching Time (ns)	2	0.25

[1] We assume 32nm×32nm×2nm magnets with Perpendicular Magnetic Anisotropy (PMA) for spin memories.

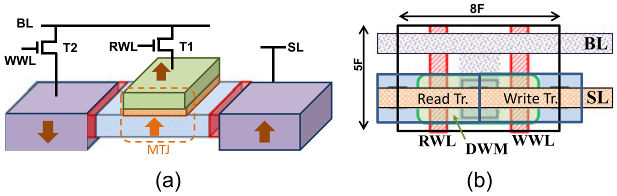


Fig. 5. Schematic and layout of a 1bitDWM cell.

by the two access transistors as the dimensions of the DWM are relatively insignificant. Further, the two access transistors can be minimum sized as the current requirements for both read and write operations in the DWM bitcell are typically low. At the 32 nm (F) technology node, the 1bitDWM cell area ($40F^2$) is $\sim$3.6× lower than that of an SRAM bit-cell ($146F^2$) and is comparable to STT-MRAM ($\sim 40F^2$). Finally, the 1bitDWM cell enjoys other advantages inherent to DWMs such as efficient write energy and negligible leakage power.

3.2 MultibitDWM

The schematic of multibitDWM cell that uses domain wall shifts to perform write operation is shown in Fig. 6. It consists of a ferromagnetic wire capable of storing multiple bits, a read-write port, and shift ports. The read/write port in a multibitDWM cell is made up of two fixed domains, one MTJ and four access transistors. In addition to read/write ports, the multibitDWM cell has shift ports consisting of an access transistor at each end of the ferromagnetic wire. Note that such a 2D DWM device structure has been proposed and prototyped in the context of domain-wall logic [15]. In this work, we use it to achieve high density and efficient write operations simultaneously.

3.2.1 Read/Write/Shift Operation

Three kinds of operations can be performed in a multibitDWM cell—read, write and shift. The voltage conditions for performing these operations are presented in Table 3. Reading/writing of data to the domain at the read/write port is performed in a manner similar to a 1bitDWM cell, as described above. Shifting of bits in a multibitDWM cell is accomplished by turning ON the shift access transistors and precharging the bitlines to appropriate voltages. For shifting the bits towards the right, BL is connected to V_{DD} and SL to GND. For shifting in the opposite direction, the voltage conditions of the bitlines are reversed.

The key benefit of the multibitDWM cell is that it achieves very high density by sharing the read/write ports across multiple bits that are stored in the ferromagnetic wire. Note that in the layout of the multibitDWM cell shown in Fig. 6b, the area is determined by the CMOS access transistors. For a multibitDWM cell with one read/write port and two shift ports, CMOS transistors require $144F^2$. In comparison, the ferromagnetic wire, which is stacked on top of the CMOS, requires only $18F^2$ for storing 16 bits. The multibitDWM cell storing 8 bits, therefore requires only $18F^2$/bit. With suitable layout optimizations [16], [17], even higher numbers of bits can be stored in the multibitDWM cell without any additional area penalty. While sharing of access ports across multiple bits leads to higher density, it also introduces the need to shift the bits to the read/write port before they can be accessed. Shift operations introduce additional latency for accessing the bits stored in multibitDWM cell. In order to address this problem of increased latency, we propose a multi-port read-optimized multibitDWM cell and suitable architectural techniques that we describe in the following sections. Another implication of the shift operations is the need for extra "overflow" bits beyond the shift ports to prevent loss of data. Note that these extra bits typically do not incur area penalty as the cell area is dominated by the access transistors.

TABLE 2
1bitDWM Cell Operation

	RWL	WWL	BL	SL
Read	V_{DD}	0	V_{read}	0
Write 0	0	V_{DD}	V_{write}	0
Write 1	0	V_{DD}	0	V_{write}
Idle	0	0	0	0

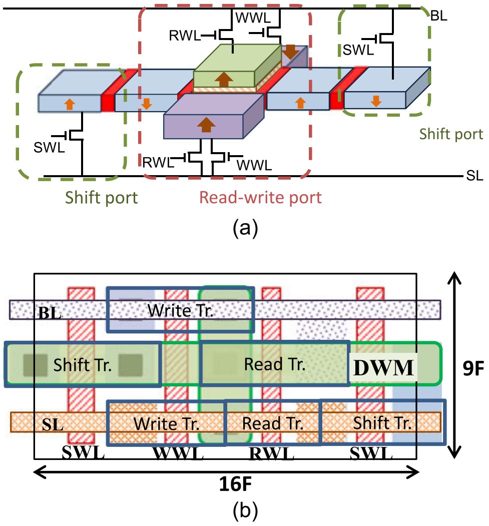


Fig. 6. Schematic and layout of a MultibitDWM cell.

4 Multi-Port Read-Optimized MultibitDWM

In this section, we present a multi-port read-optimized multibitDWM cell design that significantly reduces read access latencies, while retaining most of the density benefits associated with DWMs. This is achieved by introducing additional read-only ports, effectively reducing the number of shift operations that need to be performed. It also helps expand the design space by introducing additional design parameters that can be utilized to perform more fine-grained energy-performance tradeoffs.

TABLE 3
MultibitDWM Cell Operation

	RWL	WWL	SWL	BL	SL
Read	V_{DD}	0	0	V_{read}	0
Write 0	0	V_{DD}	0	V_{write}	0
Write 1	0	V_{DD}	0	0	V_{write}
Shift Left	0	0	V_{DD}	0	V_{shift}
Shift Right	0	0	V_{DD}	V_{shift}	0
Idle	0	0	0	0	0

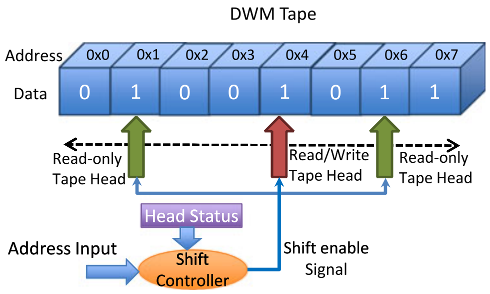


Fig. 7. Logical view of a multi-port read-optimized multibitDWM cell.

The logical view of a multi-port read-optimized multibitDWM cell is shown in Fig. 7. It has multiple read-only tape heads distributed across the DWM tape. In order to access a bit from the tape, the shift controller determines the appropriate tape head and computes the number of shift operations required. It makes these decisions by comparing the address bits with the current locations of the tape heads, referred to as the head status. It is important to note that there can be no relative movement between tape heads in a multi-port multibitDWM cell as it is the bits stored in the tape that physically shift and not the tape heads. The multi-port configuration helps reduce the average number of shift operations per read access, thereby reducing the average read access time. The motivation for having additional read-only ports is two-fold: (i) from an architectural perspective, reads are more performance critical than writes and reads also outnumber writes (across a wide range of SPEC benchmarks stores account for less than 25 percent of the total memory instructions). (ii) A read-only port can be realized using only one minimum sized transistor, preserving the density benefits of DWM. For instance, adding a read-only port to a single-port multibitDWM cell achieves 2× reduction in worst case read access latency with only 13 percent increase in area. On the other hand, halving the number of bits per tape achieves the same reduction in worst case read access latency with a 2× increase in area.

The proposed multi-port multibitDWM cell design expands the design space and offers the following design parameters:

1. *Number of bits per tape*: Varying the number of bits per tape offers a tradeoff between density versus worst case access latency for both reads and writes. Note that this tradeoff can be performed only at a coarse granularity as the number of bits per tape can be varied only in powers of 2 (to keep the addressing scheme simple).

2. *Number of read-only tape heads*: Varying the number of read-only tape heads alters only the read access latency (in contrast to bits per tape, which impacts read and write latencies), albeit in a more cost-effective manner. Furthermore, the number of read-only tape heads can be tuned in a more fine-grained manner as the number of read-only ports need not be a power of 2.

3. *Head selection*: The multi-port multibitDWM cell provides the flexibility of selecting the best tape head for a given read operation, depending on the relative location of the tape heads and the bit to be accessed. Architectural policies for head selection are proposed in Section 5.3.

In the next section, we discuss the architecture of a cache designed using the proposed multi-port multibitDWM cell.

5 TapeCache Architecture

The key architectural decisions involved in the design of TapeCache include:

- *Choice of DWM cells:* 1bitDWM and multibitDWM represent two different design points in the latency versus density tradeoff space. One of these two design options must be chosen for each memory array in the cache hierarchy.
- *Data organization:* MultibitDWM is capable of storing multiple bits in a single cell. Logically, a cache is divided into cache blocks, which must in turn be mapped to the DWM cells. One alternative mapping scheme is to store bits from the same cache block in each multibitDWM cell, while another option is to spread the bits of a cache block across multiple multibitDWM cells.
- *Addressing policy:* Sharing of read/write ports across multiple bits in a multibitDWM cell introduces the need for shifting bits before performing read/write operations. The addressing logic should not only select the multibitDWM cell to be accessed but also determine the number of shift operations required to access the required data.
- *Tape head management:* The additional latency caused by the varying number of shift operations for different bits stored in a multibitDWM cell necessitates the use of suitable cache management policies to reduce the performance penalty.

Due to the above considerations, cache design with DWM differs significantly from traditional caches. In this section, we provide an overview of the proposed cache architecture and describe its key features.

Fig. 8 depicts the overall organization of the proposed TapeCache architecture, which uses DWM to realize all levels in the cache hierarchy. Each level varies significantly in size and the required access latency. The L1 cache is responsible for providing fast access and its latency significantly impacts the overall performance of the system. Based on this consideration, we design both the data and tag arrays of the L1 cache (instruction cache and data cache) using latency optimized 1bitDWM cells, as shown in Fig. 8. When we consider the L2 cache, it is responsible for reducing the number of off-chip accesses. The L2 cache is usually of much larger size, and its leakage contributes significantly to the total energy consumption. Considering these requirements, we propose a hybrid organization consisting of both 1bitDWM and multibitDWM cells, as described in Section 5.1. In addition, the L2 cache contains a head status array along with shift control logic that are used to manage the shift operations required to access data in each multibitDWM cell. The multibitDWM cells in the data array are grouped into DWM Block Clusters (DBCs), which are capable of storing multiple cache blocks in a bit-interleaved fashion as explained in Section 5.2. In order to access a cache block, we need to select the correct DBC and determine the position of the block within that DBC. The index bits traditionally used for cache addressing are hence subdivided into decode bits and shift bits as shown in Fig. 8. The decode bits are used to select the correct DBC and the shift bits are used along with the head status

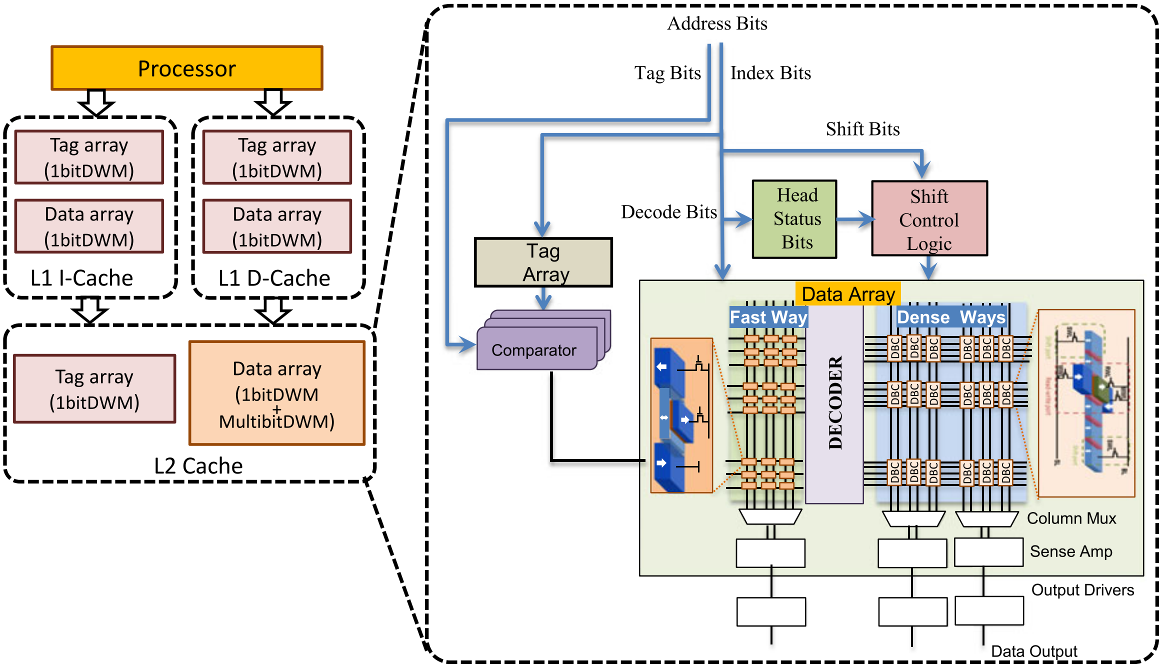


Fig. 8. TapeCache organization.

of the DBC to determine the number of shift operations required for accessing the cache block.

In the following sections, we present a detailed description of the L2 cache organization and management policies used in TapeCache.

5.1 Hybrid L2 Cache Design

A cache contains two major arrays—the tag array, which is used to determine whether an access is a cache hit or miss, and the data array, which stores the actual data. Let us consider a simple L2 cache organization in which both the data array and the tag array are designed using multibitDWM. This design would result in maximum benefits in terms of both area and power due to the high density and non-volatile nature of multibitDWM. However, the above configuration would require two variable latency operations per access, one for determining the block status from the tag array and the other for fetching the block from the data array.[2] This would considerably degrade the performance of the cache. Further, the area and energy benefits attainable from a multibitDWM-based tag array are relatively small, as the tag array represents a small fraction of the total area and power consumption of a cache. For instance, in a 1 MB cache, the tag array contributes only 4.8 percent to the total cache area. Hence, we propose to design the tag array of L2 cache using 1bitDWM cells.

In order to address the performance penalty from shift operations of multibitDWM cells in the data array, we propose a hybrid organization in which we partition the cache into *fast ways* and *dense ways*. The fast ways of the cache are designed with 1bitDWM, while the dense ways of the cache are designed with multibitDWM, as shown in Fig. 8. The motivation behind such an organization is to simultaneously exploit the latency benefits of 1bitDWM and the density benefits of multibitDWM. The fast ways are used to store frequently accessed cache blocks, thereby enabling lower latency access to performance-critical data. The remaining cache blocks are stored in dense ways, resulting in an overall improvement in cache density.

Fig. 9 demonstrates the working of a cache access under this hybrid organization. When a cache access results in a hit in the L2 cache, we check if the cache block is stored in a fast way or a dense way. If the cache block is present in a fast way, then no action is required. However, if the cache block is stored in a dense way, then it would require shift operations leading to higher access latencies. In this scenario, we need to determine if the cache block is frequently accessed and migrate it to a fast way if required. For this purpose, we check if the cache block has already been marked as most recently used (MRU) by the cache replacement policy. If so, this cache block is swapped with the least recently used (LRU) block in the fast ways. Therefore, in this cache migration policy, a cache block is determined to be frequently accessed only if a particular cache block is accessed consecutively twice - the

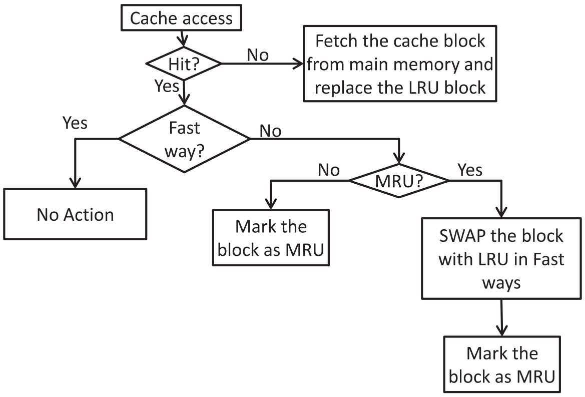

Cache access
Hit?
No
Fetch the cache block from main memory and replace the LRU block
Yes
Yes
Fast way?
No
No Action
No
MRU?
Yes
Mark the block as MRU
SWAP the block with LRU in Fast ways
Mark the block as MRU

Fig. 9. Hybrid L2 cache migration policy.

[2] In lower level caches, the tag and data array access are usually serialized so as to avoid energy overheads associated with reading all ways of a cache simultaneously.

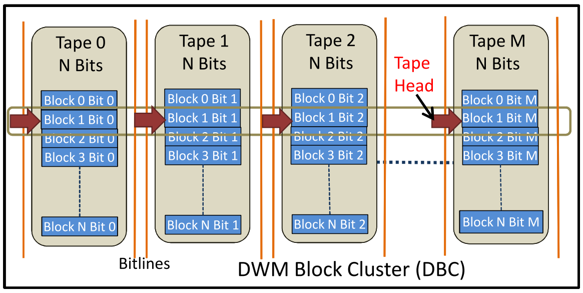


Fig. 10. Bit-interleaved data array organization.

first access would update the cache block as MRU, and the second access would initiate the block swap to a fast way. A key benefit of this cache migration policy is that it utilizes the state information already maintained by the cache replacement policy. Therefore, the hardware overhead incurred by this scheme is negligible.

The swap operation between the fast and dense ways is implemented as follows. First, we determine if the cache block being accessed will initiate a swap operation based on the tag bits using the migration policy described above. If a swap operation is required, the LRU cache block from the fast ways ($Block_{LRU}$) and the accessed block ($Block_{acc}$) from the dense ways are read simultaneously and stored in swap buffers. After reading $Block_{acc}$ from the dense ways, we keep the tape heads aligned to the current location and write $Block_{LRU}$ immediately to this location, thereby eliminating the need for any additional shift operations. Simultaneously, $Block_{acc}$ is also written to the fast ways to complete the swap operation. In the proposed design, the swap buffer has only two entries and is implemented using 1bitDWM, resulting in negligible hardware overhead. The performance penalty from the swap operations is also greatly reduced (only two cycles per swap) by (i) eliminating the need for additional shift operations, and (ii) overlapping the accesses to the fast ways with those of the dense ways. Also, the excellent energy efficiency of the proposed bit-cells greatly reduces the energy overhead associated with the swap operations. Therefore, the overheads from the proposed hybrid cache design are found to be very small, and are included in our evaluation in Section 7.

5.2 Bit-Interleaved DWM Block-Cluster Organization

Let us next consider how the multiple bits in a cache block can be mapped to the bits in a DWM tape (multibitDWM cell). One possible scheme involves mapping all the bits in a cache block to the same DWM tape (if a cache block is larger than a tape, then it can be stored in multiple tapes). In this scenario, a cache access would involve N serial read/write operations, where N is the number of bits stored in a DWM tape. This mapping would incur a very high access latency compared to a traditional SRAM-based cache.

In order to reduce the high latency overhead, we propose a bit-interleaved data array organization in which each cache block is spread across several DWM tapes. Fig. 10 illustrates how N 64-bit blocks are stored in 64 DWM tapes each containing N bits of data. The ith bit of each block is placed in tape number i. Within a tape, the jth bit location stores a bit that belongs to block j. In this scheme, all the tape heads move in a lock-step fashion such that at a given time instance, all bits of some block are aligned to their respective tapes' heads. Hence, in order to read a cache block, we perform the required number of shift operations and read all the bits belonging to the block in parallel. In this scenario, the number of shift operations required to access a block could vary from 0 to $N-1$, in contrast to the constant access latency of N observed in the previous scheme. The average access latency can be kept small in practice through the use of suitable head management policies, as discussed in the next section. One of the overheads introduced by this scheme is the need for additional decoding logic to determine the number of shifts required to access a given block. However, our experimental evaluations indicate that this overhead is negligible. Another overhead introduced by the bit-interleaved organization arises from performing shift operations in multiple tapes within a DBC for every cache access. Since the shift operation is highly energy efficient [13], performing shift operations in multiple tapes does not considerably increase in the total energy consumption of the cache. Further, the tape heads in a DBC move in lock-step fashion, thereby amortizing the control hardware overhead across multiple tapes in a DBC.

5.3 Head Management Policies

As described earlier, while accessing a cache block stored in a DBC, we need to perform an appropriate number of shift operations. The number of shift operations depends on the location of the block to be accessed relative to the position of the tape heads. Therefore, one needs an effective head management policy[3] in order to reduce the performance overhead due to shift operations. In a multi-port DWM cache design, head management policies involve (i) selection of the appropriate tape head for accessing the required data (tape head selection), and (ii) positioning of the tape head after the cache access (tape head update).

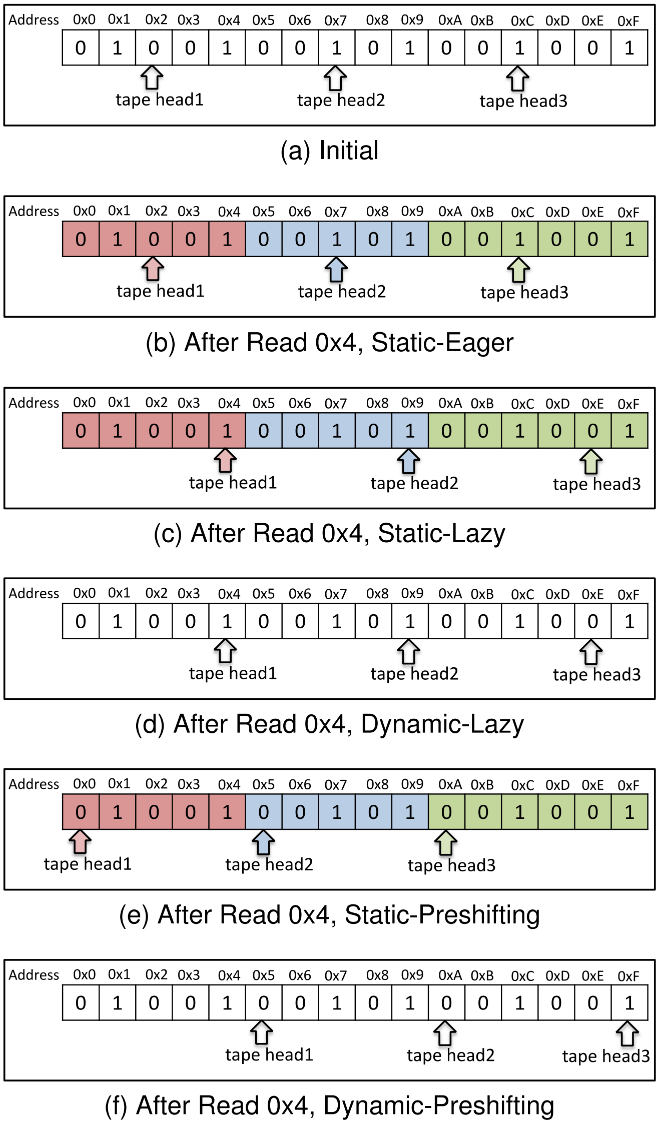
5.3.1 Tape Head Selection

The tape head selection policy applied to the multi-port multibitDWM design, and decides which of the tape heads on a cell is used to access a given bit. In this work, we consider two different tape head selection policies—*Static* and *Dynamic*. In the *Static* tape head selection policy, each cache block is assigned a tape head statically depending on its initial location within the DBC. This policy has the advantage that the tape head can be determined solely based on the address of the cache block. On the other hand, in the *Dynamic* policy, we select the tape head that is nearest to the required cache block at run-time depending on the current DBC head status. For this purpose, we use the head status array to store the locations of tape heads and activate the appropriate tape head to access the data.

5.3.2 Tape Head Update

The tape head update policy governs what we do with the tape head after an access. We explore three different tape

[3] The head management policies proposed in this section are applicable to both read-only tape heads and read/write tape heads. For read operations, we can use either read-only tape heads or read/write tape heads. For write operations, we need to use read/write tape heads.


Address 0x0 0x1 0x2 0x3 0x4 0x5 0x6 0x7 0x8 0x9 0xA 0xB 0xC 0xD 0xE 0xF
0 1 0 0 1 0 0 1 0 1 0 0 1 0 0 1
tape head1
tape head2
tape head3

(a) Initial

(b) After Read 0x4, Static-Eager

(c) After Read 0x4, Static-Lazy

(d) After Read 0x4, Dynamic-Lazy

(e) After Read 0x4, Static-Preshifting

(f) After Read 0x4, Dynamic-Preshifting

Fig. 11. Comparison of different cache management policies.

head update policies: *Eager*, *Lazy*, and *Preshifting*. In the *Eager* policy, the tape heads are restored to their original default locations after each access. This policy simplifies the tape head selection, since we can assign the optimal tape head for every block statically. Also, this results in simplified shift control logic as the number of shift operations required to access a given cache block can be determined from the block address alone, and the head status does not need to be stored. In the *Lazy* policy, we do not restore the tape head to its default initial position after performing an access. Instead, we have status bits for each tape head to keep track of its location. When we perform a read/write operation, we calculate the difference between the block location within the tape and the current location of the tape head and then perform the required number of shifts. The *Lazy* policy is motivated by the spatial locality of memory accesses, which implies that the current tape head location tends to be close to the next block to be accessed. In the *Preshifting* policy, we predict the next cache block that is likely to be accessed and align it with appropriate tape head. The concept of "preshifting" is analogous to prefetching but is unique to DWM and unlike prefetching, does not lead to any traffic to memory.

Enumerating the combinations of tape head selection and tape head update policies described above results in five distinct head management policies: *Static-Eager* (SE), *Dynamic-Lazy* (DL), *Static-Lazy* (SL), *Static-Preshifting* (SP), and *Dynamic-Preshifting* (DP). Note that *Dynamic-Eager* would be the same as *Static-Eager* as the best port can be assigned statically for each location.

Fig. 11 illustrates these five cache management policies using a DWM tape with three tape heads. For this illustration, we consider read operations, and therefore, the tape heads can be either read-only tape heads or read/write tape heads. Fig. 11a shows the initial status of the tape. Figs. 11b-11f show the status of the tape after performing a read operation at address 0×4. In Figs. 11b, 11c, 11e, the bits and tape head are shaded to indicate the static head assignment. Let us now consider the tape head selection and number of shift operations required for the next access to address 0×5. The SE policy would use head2 and would require two left shift operations. The SL policy would also use head2 but require four left shift operations. This shows that in the SL policy, a series of accesses to consecutive blocks would activate the worst case access latencies. The DL policy would choose head1 and require one right shift operation. Therefore, the DL policy exploits spatial locality, thereby reducing the average case shift latency overhead. However, this policy can skew the position of DWM tape head, thereby increasing the worst case latency for subsequent accesses. This is overcome by restoring the DWM tape configuration during cache idle periods when it gets skewed beyond a certain threshold. For instance, access to address 0×0 after 0×4 would incur a large access latency with the DL policy. However, performing a restore operation after access to address 0×4 would shift tape head2 to address 0×4. This would enable the DL policy to exploit spatial locality while eliminating higher access latencies due to skewed tape head positions. We employ a low overhead scheme to determine the idle period of the cache by checking if the cache access queue is empty. The skew threshold for restoring the tape heads is determined empirically (in our experiments, we found 4 to be a suitable threshold). When we consider the SP and DP policies, both these policies do not require any shifts as the preshifting is successful. However, the two policies would use different tape heads (SP uses tape head2 and DP uses tape head1) for accessing the cache block. Note that, the DP policy can also skew the configuration of the tape and can lead to large worst-case access latencies. However, preshifting offers a unique flexibility that is not present in other update policies. Preshifting involves two tape head selections—(i) first, when we perform the preshift operation, (ii) second, when the preshift is unsuccessful. Therefore, different strategies can be employed at these two steps, taking into consideration the performance criticality of different operations (read/write). This, as we explain below, helps us to reduce the performance impact from increased worst-case latencies.

5.3.3 Adaptive Preshifting (AP) Policy

Fig. 12 shows the different steps involved in the proposed adaptive preshifting policy. Initially, each cache block is

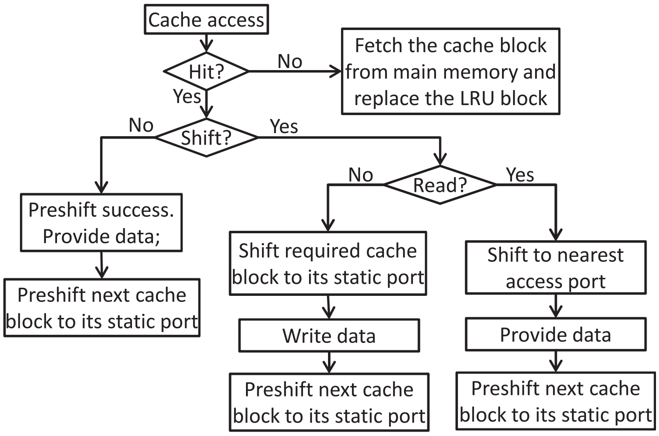

Cache access
Hit?
No
Fetch the cache block from main memory and replace the LRU block
Yes
No
Shift?
Yes
No
Read?
Yes
Preshift success. Provide data;
Preshift next cache block to its static port
Shift required cache block to its static port
Write data
Preshift next cache block to its static port
Shift to nearest access port
Provide data
Preshift next cache block to its static port

Fig. 12. Adaptive preshifting policy.

assigned a tape head statically based on its location. After a cache access, we predict the next cache block likely to be accessed and preshift the block to its *statically* assigned tape head. The use of statically assigned tape heads for preshifting reduces the skew in the tape configuration, resulting in reduced worst case latency (Fig. 11e versus Fig. 11f). If the prediction is successful, then no shift operation will be required, resulting in faster cache access. However, if the preshift fails and the cache access is a read operation, then we determine the nearest read port to the required cache block and use it to perform the required read access. This enables us to reduce the performance penalty due to misprediction for performance critical read operations. On the other hand, if the cache access is a write operation, we use the statically allocated write port. This is because, it ensures that the number of extra bits required to avoid loss of data during shifting is small. Note that, a DWM tape storing N_b bits with N_{rw} read/write ports would require N_b/N_{rw} extra bits when we use statically allocated ports for writes compared to N_b extra bits if we use the nearest port for performing writes.[4] Also, a wide range of prediction mechanisms [18] can be employed to predict the next cache block for preshifting. In this work, we use a sequential prediction scheme. After performing an access to the cache block at address i, the location of the cache block that is likely to be accessed next is predicted as address $i+1$ and the corresponding tape heads are aligned to this location.

Note that the head management policies are orthogonal to traditional cache management policies. For example, block replacement strategies such as LRU can be used unchanged in TapeCache.

6 Experimental Methodology

In this section, we present a brief description of the modeling framework and the experimental setup used to evaluate TapeCache.

6.1 Modeling Framework

TapeCache differs significantly from traditional memories in terms of both the device structure as well as cache architecture. In order to accurately evaluate the characteristics of the proposed cache design, we have developed a tool, DWM-CACTI, that is based on the CACTI framework [19]. First, we used a self-consistent device simulation framework [2] to accurately capture the characteristics of domain wall motion using spin-torque and obtain (i) the current required for shifting at different shift latencies, and (ii) the resistance offered by the MTJ for different magnetic orientations. The DWM parameters thus obtained were then used as technology parameters in DWM-CACTI to model the characteristics of TapeCache. The DWM-CACTI tool takes the number of bits per DWM tape, the number of read/write ports, and the number of read-only ports, along with the usual inputs to CACTI, to compute the area, energy and access latencies of TapeCache. DWM-CACTI takes into account the overheads due to the head-status array, additional wordlines, and shift control logic while modeling TapeCache.

TABLE 4
System Configuration

Processor Core	Alpha 21,264 pipeline, Issue Width - 4
Processor Frequency	2 GHz
Functional Units	Integer - Eight ALUs, Four Multipliers Floating Point - Two ALUs, Two Multipliers
L1 D/I-Cache	32 KB, direct mapped, 32 byte line size, Two cycle hit latency
L2 Unified Cache	1 MB, Four-way associative, 64 byte line size, hit latency depends on technology

6.2 Experimental Setup

In our experiments, we perform an iso-capacity replacement for L1 and L2 caches and compare the area, energy and performance of the proposed design with that of CMOS SRAM and STT-MRAM. All memory technologies considered are based on a 32 nm technology node. The processor configuration used in our analysis is provided in Table 4. We evaluate SRAM memories using CACTI [19], STT-MRAM using a modified CACTI tool [20], and TapeCache using our DWM-CACTI tool. We perform architectural simulations over a wide range of benchmarks from the SPEC 2006 suite using SimpleScalar [21] for 1 billion instructions after we warm up the cache by fast forwarding for 1 billion instructions.

7 Experimental Results

In this section, we present results comparing the benefits of using TapeCache with iso-capacity SRAM and STT-MRAM based caches. Note that DWM-based caches can also be used in an iso-area scenario wherein the density benefits can be translated into increased cache sizes for achieving higher performance. However, in this work, we focus on improving the energy and area efficiency of last level caches. We first present the results summarizing the benefits of the proposed design in terms of area, energy and performance. We then present results comparing the L1 and L2 cache characteristics of TapeCache with other memory technologies. Finally, we present architecture level results comparing the energy and performance of the proposed design across a wide range of benchmarks. In our experiments, we

[4] Assuming that the number of shifts required for writes will be higher than that for reads due to the presence of read-only ports.

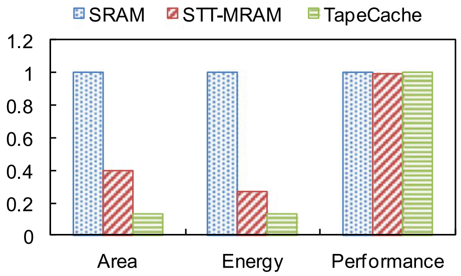


Fig. 13. Comparison of area, energy and performance across different memory technologies.

consider a multibitDWM cell with one read/write port and three read-only ports, which is capable of storing 32 bits unless mentioned otherwise.

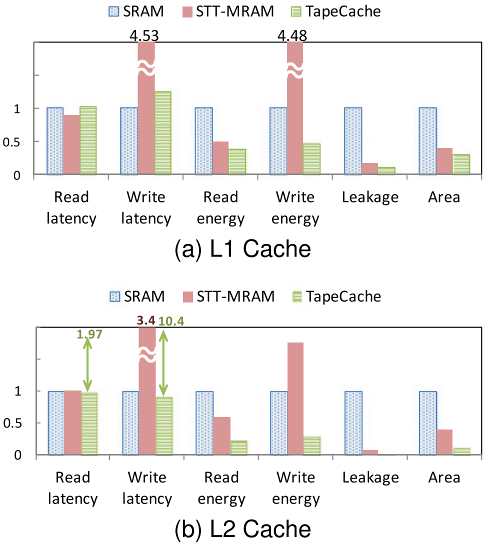
7.1 Results Summary

Fig. 13 summarizes the benefits of TapeCache compared to SRAM and STT-MRAM based caches. Compared to an SRAM-based cache, TapeCache achieves 7.5× improvement in energy and 7.8× improvement in area at virtually identical performance. When we compare the results with an STT-MRAM based cache, TapeCache achieves 3.1× improvement in area and 2× improvement in energy along with a marginal performance improvement of 1.1 percent. Next, we will examine the benefits of TapeCache in greater detail.

7.2 Cache Characteristics

In this section, we present the results comparing the characteristics of the proposed L1 and L2 cache designs with those of SRAM and STT-MRAM based caches.

Figs. 14a and 14b compare the L1 and L2 cache characteristics, respectively, across different memory technologies. As shown in the figure, the density of the L1 cache designed with 1bitDWM is similar to STT-MRAM and that of the hybrid L2 cache designed with both 1bitDWM and multibitDWM is significantly higher than both SRAM and STT-MRAM due to the higher density of multibitDWM. When we compare the leakage power, we can see that spin-based memory technologies can achieve significant reductions in leakage power consumption compared to SRAM due to their non-volatility. When we compare DWM with STT-MRAM, the small reduction in leakage power consumption is due to smaller peripheral circuitry. The wordline and bitline drivers in the STT-MRAM cache need to be sized larger due to the increased capacitive load from the large access transistors. This marginally increases the leakage power consumption of the STT-MRAM cache compared to TapeCache.



(a) L1 Cache

(b) L2 Cache

Fig. 14. Comparison of L1 and L2 cache characteristics.

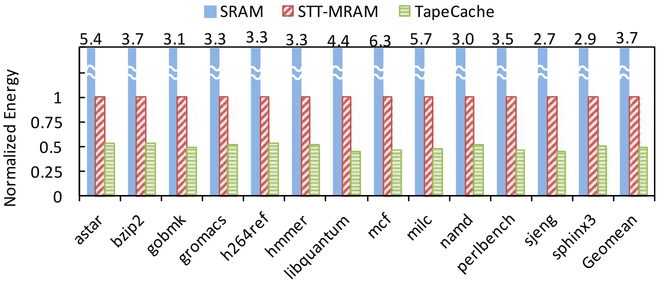


Fig. 15. Comparison of cache energy consumption across different memory technologies.

When we compare the access latencies of different L1 caches, we can see that both the read and write latencies of TapeCache are comparable to an SRAM cache. Due to the inefficiency of MTJ-based writes, the STT-MRAM based L1 has very high write latency. On the other hand, shift-based write enables us to improve the write latency significantly. When we consider the access latencies of different L2 caches, the access latency of TapeCache varies due to the variable access latency of multibitDWM, with the best case being comparable to SRAM. The effectiveness of preshifting results in average access latencies that are close to the best case.

Next, when we consider the read energies, all spin-based memories achieve significant benefits due to reduced bitline and wordline capacitances arising from improved density. Moreover, the energy efficiency of shift-based writes enables TapeCache to achieve significant reduction in write energy compared to SRAM and STT-MRAM based caches.

7.3 Architectural Evaluation

In this section, we present architecture level results comparing the energy and performance of TapeCache with SRAM and STT-MRAM caches across a wide range of benchmarks.

7.3.1 Energy Consumption of TapeCache

Fig. 15 compares the energy consumed by TapeCache with SRAM and STT-MRAM caches, normalized to the STT-MRAM cache. As we can see from the figure, TapeCache achieves significant reduction in the total cache energy consumption compared to both SRAM and STT-MRAM. An STT-MRAM based cache reduces the leakage and read energy while increasing the write energy. TapeCache achieves reduction in all the three energy components-leakage, read and write. In addition, it achieves even higher reduction in leakage and read energy compared to STT-MRAM caches. As a result, TapeCache enables us to achieve maximum benefits in the total cache energy consumption.

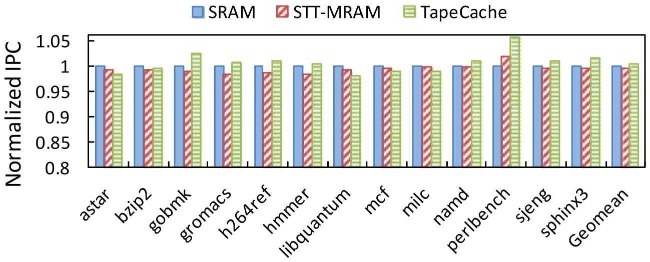


Fig. 16. Performance comparison between different memory technologies.

7.3.2 TapeCache Performance Evaluation

One of the main challenges in designing an L2 cache using multibitDWM is the variable access latency due to shift operations. The performance of a TapeCache-based system depends on (i) the access latency to different bits in a DWM tape, and (ii) the access pattern, which determines the number of shift operations required. In the case of TapeCache, the improvement in density results in reduced access latency to the bits stored at the tape head compared to SRAM and STT-MRAM. This improves its best-case access latency. The introduction of multiple ports reduces the number of shifts required to access the bits in a DWM tape, thereby reducing the worst-case access latency. Further, the proposed cache management policies ensure that most of the cache accesses require small numbers of shifts, which reduces the average access latency of the proposed L2 cache. Fig. 16 presents a comparison of the performance of different cache designs. Note that TapeCache results in performance improvement for benchmarks having high locality and predictable cache access patterns (gobmk, namd, perlbench, sjeng, sphinx3). On the other hand, for benchmarks (astar, libquantum, mcf, milc) that exhibit low degrees of locality, there is a reduction in performance due to increased shift penalty.

7.4 Design Space Exploration

7.4.1 Hybrid Cache Organizations

Using DWM, we can design two different hybrid caches: inter-layer and intra-layer. The inter-layer design uses different bit-cells to realize different levels in the cache hierarchy. The intra-layer design, on the other hand, uses different bit-cells even within a single level. In Section 5, we described an intra-layer hybrid cache organization. The inter-layer hybrid cache can be realized by designing the L1 cache with 1bitDWM and the L2 cache with multibitDWM. A comparison of these hybrid designs shows that the proposed intra-layer design achieves 23 percent improvement in performance at comparable energy and area, thereby validating our design choice.

7.4.2 Number of Bits Per Tape

TapeCache offers a unique parameter—number of tape heads (read-only ports and read/write ports)—to tradeoff energy with performance. In Fig. 17, we vary the number of read-only ports, read/write ports, and the tape head management policy and study their impact on cache energy consumption and performance. All the numbers in Fig. 17 are normalized to an SRAM cache. In the figure, a tape with x bits per tape, y read/write ports, and z read-only ports is denoted by x,y,z near different markers. As can be seen from Fig. 17, increasing the number of read-only ports and read/write ports reduces the average read latency while increasing the energy consumption of TapeCache. It is interesting to note the relationship between the average number of bits per read port and the tape head management policy. For configurations with smaller numbers of bits per tape head, the SE policy which reduces the worst-case access latency performs better than the DL policy. On the other hand, for configurations with higher numbers of bits per tape head (less no. of read ports), the DL policy outperform SE since it exploits the spatial locality in access patterns. The proposed AP policy combines the benefits of both DL and SE policies and is found to be optimal across all the tape configurations. Also, note that the AP policy achieves significant improvement in performance (>9.5 percent) over the SE policy for the (32,1,0) configuration, which has a large number of bits per tape head (large variation in access latencies). As the number of tape heads per tape increases, the variation in access latencies reduces, resulting in diminishing benefits from the management policies. The figure also shows the corresponding iso-capacity design point for the STT-MRAM cache, which is clearly inferior to TapeCache.

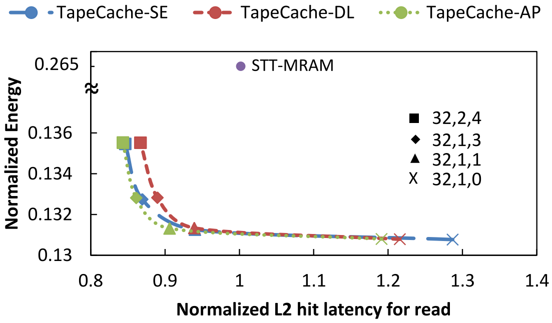


Fig. 17. Design space exploration for TapeCache.

8 Related Work

In recent years, several emerging memory technologies have been explored as potential replacements for CMOS memories [22]. Among the various memory technologies, STT-MRAM and PCRAM have been considered the most promising and various research efforts have addressed their shortcomings through architectural optimizations. In this section, we present a brief survey of representative research efforts that have explored these memories at different levels of design abstraction.

A key drawback common to both STT-MRAM and PCRAM is the energy required to perform writes. At the device level, researchers have optimized the write operation by designing different kinds of MTJ structures such as Dual-pillar MTJ, tilted MTJ, Dual barrier MTJ etc. [23], [24]. Many of the proposed device structures decouple the read and write paths, thereby relaxing the read versus write design conflicts that are commonly present in memory design. At the circuit level, proposals to use 2T-1R structures with dual source lines and early write termination are also aimed at reducing the write energy consumption [9], [25].

At the architecture level, the most common approach adopted to address this issue is to design a hybrid memory organization [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], comprising of both CMOS and STT-MRAM/PCRAM. The motivation behind such an approach is to selectively direct memory blocks that incur large number of writes to CMOS memory, while storing the rest in STT-MRAM/PCRAM. An alternative approach to address the write-inefficiency is to eliminate redundant writes to memory, either by comparing the data before performing the write operation [9], [36], [37] or by tracking dirty blocks at a finer granularity [38], [39]. A third approach to reduce write energy, called volatile STT-MRAM, was proposed in [40], [41]. In this case, the non-volatile nature of STT-MRAM is sacrificed for write-efficiency and suitable refresh schemes are employed for sustained data retention. In order to address the performance implications of inefficient writes, write buffers [39], [42] and scheduling mechanisms [43] that prioritize write requests to idle cache banks have been proposed. In addition to the write energy/latency, PCRAM suffers from limited endurance. As a solution, [12] proposes wear-leveling, in which the write operations are evenly spread across the entire memory array.

While the above efforts have addressed challenges related to STT-MRAM and PCRAM, we focus on utilizing DWMs, which offer unique challenges, in the design of the general purpose cache hierarchy.

Earlier research efforts on DWM have mainly been at the device and circuit levels [2], [14], [44], [45]. There were also multiple efforts to address the issues related to fabrication [1], [5], [46] and a prototype of a DWM array has been demonstrated recently [6]. Layout optimization to preserve the density benefits of DWM was proposed in [17]. Considering the unique characteristics of DWM, research efforts have primarily considered DWM in the context of domain-specific applications for implementing FIFOs [7] and shift registers [47], that directly match the device characteristics.

In a preliminary version of this work, we investigated, for the first time, the use of DWM in the cache hierarchy of general-purpose processors [14], [48]. Building upon those efforts, in this work, we perform a more comprehensive and detailed device-circuit-architecture co-optimization and design all the levels in the cache hierarchy using DWM. We also propose the use of a hybrid L2 cache architecture consisting of fast ways and dense ways along with suitable cache management policies to maximally harness the performance of DWM.

Like other emerging technologies like STT-MRAM, PCRAM, etc., many challenges associated with the fabrication and mass production of DWM remain to be addressed. While earlier research efforts [1], [5], [6], [46] have demonstrated the working of DWM through prototypes, the challenges associated with mass production need to be explored in the future.

9 Conclusion

Domain Wall Memory is an emerging spin-based memory technology that has a much higher density and good energy efficiency compared to current memory technologies (SRAM, DRAM), as well as other candidates for future memories such as STT-MRAM, PCRAM, etc. We explored the use of DWM for designing on-chip caches in general-purpose computing platforms. We proposed cache organization and management policies that reduce the performance penalty due to the shift latency of DWMs. We performed architectural simulations to evaluate the benefits of a DWM-based cache. Our results demonstrate that DWM based caches offer great potential in improving the energy-performance profile of a wide range of applications, while significantly reducing cache area.

Acknowledgments

This material is based upon work supported by Intel Corporation, STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA and Semiconductor Research Corporation under the GRC program.

References

[1] L. Thomas, R. Moriya, C. Rettner, and S. Parkin, "Dynamics of magnetic domain walls under their own inertia," *Science*, vol. 330, no. 6012, pp. 1810–1813, Dec. 2010.

[2] C. Augustine, A. Raychowdhury, B. Behin-Aein, S. Srinivasan, J. Tschanz, V. K. De, and K. Roy, "Numerical analysis of domain wall propagation for dense memory arrays," in *Proc. Int. Electron. Devices Meeting*, Dec. 2011, pp. 17.6.1–17.6.4.

[3] P. Agrawal, U. Bauer, and G. S. D. Beach, "Spontaneous domain nucleation under in-plane fields in ultrathin films with Dzyaloshinskii-Moriya interaction," *J. Appl. Phys.*, vol. 117, no. 17, pp. 17c744-1–17c744-4, Apr. 2015.

[4] S. Parkin, M. Hayashi, and L. Thomas, "Magnetic domain-wall racetrack memory," *Science*, vol. 320, no. 5873, pp. 190–194, Apr. 2008.

[5] E. R. Lewis, D. Petit, L. O'Brien, A. Fernandez-Pacheco, J. Sampaio, A.-V. Jausovec, H. T. Zeng, D. E. Read, and R. P. Cowburn, "Fast domain wall motion in magnetic comb structures," *Nature*, vol. 9, no. 12, pp. 980–983, Dec. 2010.

[6] A. J. Annunziata, M. C. Gaidis, L. Thomas, C. W. Chien, C. C. Hung, P. Chevalier, E. J. O'Sullivan, J. P. Hummel, E. A. Joseph, Y. Zhu, T. Topuria, E. Delenia, P. M. Rice, S. S. P. Parkin, and W. J. Gallagher, "Racetrack memory cell array with integrated magnetic tunnel junction readout," in *Proc. Int. Electron Devices Meeting*, Dec. 2011, pp. 24.3.1–24.3.4.

[7] R. Venkatesan, V. Chippa, C. Augustine, K. Roy, and A. Raghunathan, "Energy efficient many-core processor for recognition and mining using spin-based memory," in *Proc. Int. Symp. Nanoscale Archit.*, Jun. 2011, pp. 122–128.

[8] R. Venkatesan, V. K. Chippa, C. Augustine, K. Roy, and A. Raghunathan, "Domain-specific many-core computing using spin-based memory," *IEEE Trans. Nanotechnol.*, vol. 13, no. 5, pp. 881–894, Sep. 2014.

[9] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "Energy reduction for STT-RAM using early write termination," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2009, pp. 264–268.

[10] Y. Xie, "Modeling, architecture, and applications for emerging memory technologies," *IEEE Design Test Comput.*, vol. 28, no. 1, pp. 44–51, Jan./Feb. 2011.

[11] K. Lee and S. Kang, "Development of embedded STT-MRAM for mobile system-on-chips," *IEEE Trans. Magn.*, vol. 47, no. 1, pp. 131–136, Jan. 2011.

[12] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable high performance main memory system using phase-change memory technology," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2009, pp. 24–33.

[13] S. Fukami, T. Suzuki, K. Nagahara, N. Ohshima, Y. Ozaki, S. Saito, R. Nebashi, N. Sakimura, H. Honjo, K. Mori, C. Igarashi, S. Miura, N. Ishiwata, and T. Sugibayashi, "Low current perpendicular domain wall motion cell for scalable high-speed MRAM," in *Proc. IEEE Symp. VLSI Technol.*, Jun. 2009, pp. 230–231.

[14] R. Venkatesan, M. Sharad, K. Roy, and A. Raghunathan, "DWM-TAPESTRI—An energy efficient all-spin cache using domain wall shift based writes," in *Proc. Design, Autom. Test Eur.*, Apr. 2013, pp. 1825–1830.

[15] D. A. Allwood, G. Xiong, C. C. Faulkner, D. Atkinson, D. Petit, and R. P. Cowburn, "Magnetic domain-wall logic," *Science*, vol. 309, no. 5741, pp. 1688–1692, Sep. 2005.

[16] S. Motaman, A. Iyengar, and S. Ghosh, "Synergistic circuit and system design for energy-efficient and robust domain wall caches," in *Proc. Int. Symp. Low Power Electron. Design*, 2014, pp. 195–200.

[17] Z. Sun, W. Wu, and H. Li, "Cross-layer racetrack memory design for ultra high density and low power consumption," in *Proc. Design Autom. Conf.*, Jun. 2013, pp. 53:1–53:6.

[18] T.-F. Chen and J.-L. Baer, "Effective hardware-based data prefetching for high-performance processors," *IEEE Trans. Comput.*, vol. 44, no. 5, pp. 609–623, May 1995.

[19] N. Muralimanohar, R. Balasubramonian, N. P. Jouppi, "CACTI 6.0: A Tool to Model Large Caches," Tech. Report, HPL-2009-85, Apr. 2009.

[20] X. Dong, X. Wu, G. Sun, and Y. Xie, "Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement," in *Proc. Design Autom. Conf.*, Jun. 2008, pp. 554–559.

[21] T. Austin, E. Larson, and D. Ernst, "Simplescalar: An infrastructure for computer system modeling," *Computer*, vol. 35, pp. 59–67, Feb. 2002.

[22] K. Bernstein, R. Cavin, W. Porod, A. Seabaugh, and J. Welser, "Device and architecture outlook for beyond CMOS switches," *Proc. IEEE*, vol. 98, no. 12, pp. 2169–2184, Dec. 2010.

[23] N. Mojumder and K. Roy, "Proposal for switching current reduction using reference layer with tilted magnetic anisotropy in magnetic tunnel junctions for Spin-Transfer Torque (STT) MRAM," *IEEE Trans. Electron Devices*, vol. 59, no. 11, pp. 3054–3060, Nov. 2012.

[24] C. Augustine, A. Raychowdhury, D. Somasekhar, and J. Tschanz, "Numerical analysis of typical STT-MTJ stacks for 1T-1R memory arrays," in *Proc. Int. Electron Devices Meeting*, Dec. 2010, pp. 22.7.1–22.7.4.

[25] Y. Kim, S. K. Gupta, S. P. Park, G. Panagopoulos, and K. Roy, "Write-optimized reliable design of STT MRAM," in *Proc. Int. Symp. Low Power Electron. Design*, 2012, pp. 3–8.

[26] N. Goswami, B. Cao, and T. Li, "Power-performance co-optimization of throughput core architecture using resistive memory," in *Proc. Int. Symp. High Perform. Comput. Archit.*, Feb. 2013, pp. 342–353.

[27] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, "Hybrid cache architecture with disparate memory technologies," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2009, pp. 34–45.

[28] J. Hu, C. J. Xue, Q. Z., W. C. Tseng, and E. H. M. Sha, "Towards energy efficient hybrid on-chip scratch pad memory with non-volatile memory," in *Proc. Design Autom. Test Eur.*, Mar. 2011, pp. 1–6.

[29] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, "Design exploration of hybrid caches with disparate memory technologies," *ACM Trans. Archit. Code Optimiz.*, vol. 7, no. 3, pp. 15:1–15:34, Dec. 2010.

[30] J. Cong, K. Gururaj, H. Huang, C. Liu, G. Reinman, and Y. Zou, "An energy-efficient adaptive hybrid cache," in *Proc. Int. Symp. Low Power Electron. Design*, Aug. 2011, pp. 67–72.

[31] A. Jadidi, M. Arjomand, and H. S. Azad, "High-endurance and performance-efficient design of hybrid cache architectures through adaptive line replacement," in *Proc. Int. Symp. Low Power Electron. Design*, Aug. 2011, pp. 79–84.

[32] G. Dhiman, R. Ayoub, and T. Rosing, "PDRAM: A hybrid PRAM and DRAM main memory system," in *Proc. Design Autom. Conf.*, Jul. 2009, pp. 664–669.

[33] J. Zhao and Y. Xie, "Optimizing bandwidth and power of graphics memory with hybrid memory technologies and adaptive data migration," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2012, pp. 81–87.

[34] B. Wang, B. Wu, D. Li, X. Shen, W. Yu, Y. Jiao, and J. Vetter, "Exploring hybrid memory for GPU energy efficiency through software-hardware co-design," in *Proc. Int. Conf. Parallel Archit. Compilation Techn.*, Sep. 2013, pp. 93–102.

[35] M. Rasquinha, D. Choudhary, S. Chatterjee, S. Mukhopadhyay, and S. Yalamanchili, "An energy efficient cache design using Spin Torque Transfer (STT) RAM," in *Proc. Int. Symp. Low Power Electron. Design*, Aug. 2010, pp. 389–394.

[36] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "A durable and energy efficient main memory using phase change memory technology," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2009, pp. 14–23.

[37] S. Cho and H. Lee, "Flip-N-Write: A simple deterministic technique to improve PRAM write performance, energy and endurance," in *Proc. Int. Symp. Microarchit.*, Dec. 2009, pp. 347–357.

[38] S. P. Park, S. Gupta, N. Mojumder, A. Raghunathan, and K. Roy, "Future cache design using STT MRAMs for improved energy efficiency: Devices, circuits and architecture," in *Proc. Design Autom. Conf.*, Jun. 2012, pp. 492–497.

[39] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable DRAM alternative," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2009, pp. 2–13.

[40] C. W. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, and M. R. Stan, "Relaxing non-volatility for fast and energy-efficient STT-RAM caches," in *Proc. Int. Symp. High Perform. Comput. Archit.*, Feb. 2011, pp. 50–61.

[41] A. Jog, A. K. Mishra, C. Xu, Y. Xie, V. Narayanan, R. Iyer, and C. R. Das, "Cache revive: Architecting volatile STT-RAM caches for enhanced performance in CMPs," in *Proc. Design Autom. Conf.*, Jun. 2012, pp. 243–252.

[42] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen, "A novel architecture of the 3D stacked MRAM L2 cache for CMPs," in *Proc. Int. Symp. High Perform. Comput. Archit.*, Feb. 2009, pp. 239–249.

[43] A. Mishra, X. Dong, G. Sun, Y. Xie, N. Vijaykrishnan, and C. Das, "Architecting on-chip interconnects for stacked 3D STT-RAM caches in CMPs," in *Proc. Int. Symp. Comput. Archit.*, 2011, pp. 69–80.

[44] N. Vernier, D. Allwood, D. Atkinson, M. Cooke, and R. Cowburn, "Domain wall propagation in magnetic nanowires by spin polarized current injection," *Europhys. Lett.*, vol. 65, no. 4, pp. 526–532, Feb. 2004.

[45] M. Sharad, R. Venkatesan, A. Raghunathan, and K. Roy, "Multi-level magnetic RAM using domain wall shift for energy-efficient, high-density caches," in *Proc. Int. Symp. Low Power Electron. Design*, Sep. 2013, pp. 64–69.

[46] D. Chiba, G. Yamada, T. Koyama, K. Ueda, H. Tanigawa, S. Fukami, T. Suzuki, N. Ohshima, N. Ishiwata, Y. Nakatani, and T. Ono, "Control of multiple magnetic domain walls by current in a Co/Ni nano-wire," *Appl. Phys. Exp.*, vol. 3, no. 073004, pp. 1–3, Jul. 2010.

[47] W. Zhao, D. Ravelosona, J. Klein, and C. Chappert, "Domain wall shift register-based reconfigurable logic," *IEEE Trans. Magn.*, vol. 47, no. 10, pp. 2966–2969, Oct. 2011.

[48] R. Venkatesan, V. Kozhikkottu, C. Augustine, A. Raychowdhury, K. Roy, and A. Raghunathan, "TapeCache: A high density, energy efficient cache based on Domain Wall Memory," in *Proc. Int. Symp. Low Power Electron. Design*, Jul. 2012, pp. 185–190.

Rangharajan Venkatesan received the BTech degree in electronics and communication engineering from the Indian Institute of Technology, Roorkee, India, in 2009. He is currently working toward the PhD degree in the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. During the PhD degree, he was a research intern at Intel Corporation, Hillsboro from May 2012 to September 2012 and June 2013 to September 2013, where he worked on developing low power design methodologies for graphics processors and designing circuits for enabling fine-grained power gating. His research interests include circuit-architecture codesign for emerging technologies, neuromorphic hardware architectures, approximate computing, and variation-aware design methodologies. He was awarded the Ross Fellowship for the year 2009-10 and the Bilsland Dissertation Fellowship for the year 2013-14 by the Graduate School, Purdue University. He has received Best Paper Award in International Symposium on Low Power Electronics and Design, 2012. He is a student member of the IEEE.

Vivek J. Kozhikkottu received the PhD degree in computer engineering from Purdue University in March 2014. His thesis was on Variation Aware SoC Design. He received the master's and bachelor's degrees in electrical engineering from the Indian Institute of Technology, Madras, in 2009. He is currently with the Extreme Scale Technology Group at Intel. His research interests include high performance computing, low power and resilient architectures. He is a student member of the IEEE.

Mrigank Sharad received the BTech and MTech degrees in electronics and electrical communication engineering from the Indian Institute of Technology, Kharagpur, India, in 2010, where he specialized in microelectronics and VLSI design. He is currently working toward the PhD degree in electrical and computer engineering, Purdue University. His primary research interests include low-power digital/mixed-signal circuit design. His current research is focused on device-circuit codesign for low power logic and memory, with emphasis on exploration of post-CMOS technologies like, spin-devices. He has also worked on application of spin-torque devices in approximate computing hardware, interconnect and memory design. He was awarded Prime Minister of India Gold Medal for his academic performance by IIT Kharagpur. He received Andrews Fellowship from Purdue University in 2010. He has authored more than 40 papers in international journals and conferences. He is a student member of the IEEE.

Charles Augustine received the bachelor's degree in electronics from BITS, Pilani, India, in 2004, and the PhD degree in electrical and computer engineering from Purdue University in 2011. He is currently a senior research scientist in the Circuit Research Lab Intel Corporation, Hillsboro, OR. His primary research interests include ultra-low-power memory and logic circuits for GPUs, and circuit/ architecture design for machine learning algorithms such as image and audio recognition. He received the Best Paper Award in International Symposium on Low Power Electronics and Design in 2012, Best Paper in Session Award at SRC Techcon in 2009, AMD Design Excellence Award from Purdue in 2008, nominated for Best Paper Award at International Symposium on Quality Electronic Design in 2009, and won Bronze medal for academic excellence from BITS, Pilani, in 2004. He has held positions at Texas Instruments, ST Microelectronics, Philips Semiconductors, and Freescale Semiconductor, where he worked on CMOS digital integrated circuits and memories, including spin-torque based memories. Charles has published more than 45 papers in refereed journals and conferences and has filed seven patents (pending). He is a member of the IEEE.

Arijit Raychowdhury received the BE degree in electrical and telecommunication engineering from Jadavpur University, India, and the PhD degree in electrical and computer engineering from Purdue University. He is currently an associate professor in the School of Electrical and Computer Engineering, Georgia Institute of Technology where he joined in January, 2013. His industry experience includes five years as a staff scientist in the Circuits Research Lab, Intel Corporation and a year as an analog circuit designer with Texas Instruments Inc. His research interests include digital and mixed-signal circuit design, design of on-chip sensors, memory, and device-circuit interactions. He holds more than 25 US and international patents and has published more than 100 articles in journals and refereed conferences. He is the winner of the Intel Labs Technical Contribution Award, 2011; Dimitris N. Chorafas Award for outstanding doctoral research, 2007; the Best Thesis Award, College of Engineering, Purdue University, 2007; Best Paper Awards at the International Symposium on Low Power Electronic Design 2012, 2006; IEEE Nanotechnology Conference, 2003; SRC Technical Excellence Award, 2005; Intel Foundation Fellowship 2006, NASA INAC Fellowship 2004, and the Meissner Fellowship 2002. He is a senior member of the IEEE.

Kaushik Roy received the BTech degree in electronics and electrical communications engineering from the Indian Institute of Technology, Kharagpur, India, and the PhD degree from the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign in 1990. He was with the Semiconductor Process and Design Center of Texas Instruments, Dallas, where he worked on FPGA architecture development and low-power circuit design. He joined the electrical and computer engineering faculty at Purdue University, West Lafayette, IN, in 1993, where he is currently Edward G. Tiedemann Jr. Distinguished Professor. His research interests include spintronics, device-circuit co-design for nano-scale Silicon and non-Silicon technologies, low-power electronics for portable computing and wireless communications, and new computing models enabled by emerging technologies. He has published more than 600 papers in refereed journals and conferences, holds 15 patents, graduated 60 PhD students, and is coauthor of two books on Low Power CMOS VLSI Design (Wiley & McGraw Hill). He received the US National Science Foundation Career Development Award in 1995, IBM faculty partnership award, ATT/Lucent Foundation award, 2005 SRC Technical Excellence Award, SRC Inventors Award, Purdue College of Engineering Research Excellence Award, Humboldt Research Award in 2010, 2010 IEEE Circuits and Systems Society Technical Achievement Award, Distinguished Alumnus Award from Indian Institute of Technology, Kharagpur, Fulbright-Nehru Distinguished Chair, and Best Paper Awards at 1997 International Test Conference, IEEE 2000 International Symposium on Quality of IC Design, 2003 IEEE Latin American Test Workshop, 2003 IEEE Nano, 2004 IEEE International Conference on Computer Design, 2006 IEEE/ACM International Symposium on Low Power Electronics & Design, and 2005 IEEE Circuits and System Society Outstanding Young Author Award (Chris Kim), 2006 IEEE Transactions on VLSI Systems Best Paper Award, 2012 ACM/IEEE International Symposium on Low Power Electronics and Design Best Paper Award, 2013 IEEE Transactions on VLSI Best Paper Award. He was a Purdue University Faculty scholar (1998-2003). He was a Research Visionary board member of Motorola Labs (2002) and held the M.K. Gandhi Distinguished Visiting faculty at Indian Institute of Technology (Bombay). He has been in the editorial board of *IEEE Design and Test*, *IEEE Transactions on Circuits and Systems*, *IEEE Transactions on VLSI Systems*, and *IEEE Transactions on Electron Devices*. He was the guest editor for Special Issue on Low-Power VLSI in the *IEEE Design and Test* (1994) and *IEEE Transactions on VLSI Systems* (June 2000), *IEE Proceedings—Computers and Digital Techniques* (July 2002), and *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* (2011). He is a fellow of the IEEE.

Anand Raghunathan received the BTech degree in electrical and electronics engineering from the Indian Institute of Technology, Madras, India, and the MA and PhD degrees in electrical engineering from Princeton University, Princeton, NJ. He is currently a Professor in the School of Electrical and Computer Engineering, Purdue University, where he directs research in the Integrated Systems Laboratory in the areas of system-on-chip design, domain-specific computing, post-CMOS circuits and architecture, and heterogeneous parallel computing. Previously, he was a senior research staff member at NEC Laboratories America in Princeton, NJ, where he led research projects related to system-on-chip architectures, design methodologies, and design tools. He has coauthored a book (*High-Level Power Analysis and Optimization*) and eight book chapters, and has presented several full-day and embedded conference tutorials in the above areas. He holds 20 US patents and has authored more than 200 refereed conference and journal publications. He has received eight Best Paper Awards and four Best Paper Award nominations at leading IEEE and ACM conferences. He received a Patent of the Year Award (an award recognizing the invention that has achieved the highest impact), and two Technology Commercialization Award from NEC. He was chosen by MIT's Technology Review among the TR35 (top 35 innovators under 35 years, across various disciplines of science and technology) in 2006, for his work on "making mobile secure". He has been a member of the technical program and organizing committees of several leading conferences and workshops. He has served as a program cochair for the ACM/IEEE International Symposium on Low Power Electronics and Design, the ACM/IEEE International Conference on Compilers, Architecture and Synthesis for Embedded Systems, the IEEE VLSI Test Symposium, and the IEEE International Conference on VLSI Design. He has served as an associate editor of the *IEEE Transactions on CAD*, the *IEEE Transactions on VLSI Systems*, *ACM Transactions on Design Automation of Electronic Systems*, *IEEE Transactions on Mobile Computing*, *ACM Transactions on Embedded Computing Systems*, *IEEE Design & Test of Computers*, and the *Journal of Low Power Electronics*. He received the IEEE Meritorious Service Award (2001) and Outstanding Service Award (2004). He is a Fellow of the IEEE and was elected a Golden Core Member of the IEEE Computer Society in 2001, in recognition of his contributions.