On-chip MRAM as a High-Bandwidth, Low-Latency Replacement for DRAM Physical Memories

Rajagopalan Desikan[⋆] Charles R. Lefurgy[†] Stephen W. Keckler Doug Burger
[⋆]Department of Electrical and Computer Engineering [†]IBM Austin Research Laboratory
Computer Architecture and Technology Laboratory
Department of Computer Sciences
The University of Texas at Austin
cart@cs.utexas.edu - www.cs.utexas.edu/users/cart

Department of Computer Sciences
Tech Report TR-02-47
The University of Texas at Austin

September 27, 2002

Abstract

Impediments to main memory performance have traditionally been due to the divergence in processor versus memory speed and the pin bandwidth limitations of modern packaging technologies. In this paper we evaluate a magneto-resistive memory (MRAM)-based hierarchy to address these future constraints. MRAM devices are non-volatile, and have the potential to be faster than DRAM, denser than embedded DRAM, and can be integrated into the processor die in layers above those of conventional wiring. We describe basic MRAM device operation, develop detailed models for MRAM banks and layers, and evaluate an MRAM-based memory hierarchy in which all off-chip physical DRAM is replaced by on-chip MRAM. We show that this hierarchy offers extremely high bandwidth, resulting in a 15% improvement in end-program performance over conventional DRAM-based main memory systems. Finally, we compare the MRAM hierarchy to one using a chipstacked DRAM technology and show that the extra bandwidth of MRAM enables it to outperform this nearer-term technology. We expect that the advantage of MRAM-like technologies will increase with the proliferation of chip multiprocessors due to increased memory bandwidth demands.

1 Introduction

Main memory latencies are already hundreds of cycles; often processors spend more than half of their time stalling for L2 misses [11]. Memory latencies will continue to grow, but more slowly over the next decade than in the last, since processor pipelines are nearing their optimal depths [34, 13]. However, off-chip bandwidth will continue to grow as a performance-limiting factor, since the number of transistors on chip is increasing at a faster rate than chip signaling pins [33]. Left unaddressed, this disparity will limit the scalability of future chip multiprocessors [14]. Larger caches can reduce off-chip bandwidth constraints, but consume area that could instead be used for processing, limiting the number of useful processors that can be implemented on a single die.

In this paper, we evaluate the potential of on-chip magneto-resistive random access memory (MRAM) to solve this set of problems. MRAM is an emerging memory technology that stores information using the magnetic polarity of a thin ferromagnetic layer. This information is read by measuring the current across an MRAM cell, determined by the rate of electron quantum tunneling, which is in turn affected by the magnetic polarity of the cell.

MRAM cells have many potential advantages. They are non-volatile, and they can be both faster, and potentially as dense, as DRAM cells. They can be implemented in wiring layers above an active silicon substrate as part of a single chip. Multiple MRAM layers can thus be placed on top of a single die, permitting highly integrated capacities. Most important, the enormous interconnection density of 100,000 vertical wires per square millimeter, assuming vertical wires have pitch similar to global vias (currently 24 λ thickness and 10 λ width), will enable as many as 10,000 wires per addressable bank within the MRAM layer. In this technology, the number of interconnects and total bandwidth are limited by the pitch of the vertical vias rather than that of the pads required by conventional packaging technologies.

Unsurprisingly, MRAM devices have several potential drawbacks. They require high power to write, and layers of MRAM devices may interfere with heat dissipation. Furthermore, while MRAM devices have been prototyped, the latency and density of production MRAM cells in contemporary conventional technologies remains unknown. To justify the investment needed to make MRAMs commercially competitive will require evidence of significant advantages over conventional technologies. One goal of our work is to determine whether MRAM hierarchies show enough potential performance advantages to be worth further exploration.

In this paper, we develop and describe access latency and area models for MRAM banks and layers. Using these models, we simulate a hierarchy that replaces off-chip DRAM physical memories with an on-chip MRAM memory hierarchy. Our MRAM hierarchy breaks a single MRAM layer into a collection of banks, in which the MRAM devices sit between two metal wiring layers, but in which the decoders, word line drivers, and sense amplifiers reside on the transistor layer, thus consuming chip area. The MRAM banks hold physical pages, and under each MRAM bank resides a small level-2 (L2) cache which caches lines mapped to that MRAM bank. The mapping of physical pages thus determines to which L2 cache bank a line will be mapped.

Since some MRAM banks are more expensive to access than others, due to the physical distances across the chip, page placement into MRAM banks can affect performance. An ideal placement policy would: (1) minimize routing latency by placing frequently accessed pages into MRAM banks close to the processor, (2) minimize network congestion by placing pages into banks that have the fewest highly accessed pages, (3) minimize L2 bank miss rates by distributing hot pages evenly across the MRAM banks. According to our results, the best page placement policy with MRAM outperforms a conventional DRAM-based hierarchy by 15% across 16 memory-intensive benchmarks. We evaluate several page placement policies, and find that in near-term technology, minimizing L2 miss rates with uniform page distribution outweighs minimization of bank contention or routing delay. That balance will shift as cross-chip routing delays grow in future technologies, and as both wider-issue and CMP processors place a heavier load on the memory subsystem.

Finally, we compare our MRAM hierarchy against another emerging memory technology called chip-stacked SDRAM. With this technology, a conventional DRAM chip is die-attached to the surface of a logic chip. This procedure enables the two chips to communicate through many more wires than can be found on conventional chip packages or even multi-chip modules (MCMs). Although the higher bandwidth is exploited in a manner similar to that of MRAM, the total I/O counts is still substantially lower. Our preliminary evaluation of these two hierarchies shows that the MRAM hierarchy performs best, with the caveat that the parameters used for both are somewhat speculative.

Section 2 describes MRAM device operation and presents a model for access delay of MRAM banks and layers. Section 3 proposes an MRAM-based memory hierarchy, in which a single MRAM layer is broken into banks, cached by per-bank L2 banks, and connected via a 2-D switched network. Section 4 compares a traditional memory hierarchy, with an off-chip DRAM physical memory, to the MRAM memory hierarchy. Section 5 presents a performance comparison of the MRAM hierarchy to a chipstack SDRAM memory hierarchy. Section 6 describes related work, and Section 7 summarizes our conclusions and describes issues for future study.

2 MRAM Memory Cells and Banks

Magnetoresitive random access memory (MRAM) is a memory technology that uses the magnetic tunnel junction (MTJ) to store information [31, 4]. The potential for MRAM has improved steadily due to advances in magnetic materials. MRAM uses the magnetization orientation of a thin ferromagnetic material to store information, and a bit can be detected by sampling the difference in electrical resistance between the two polarized states of the MTJ. Current MRAM designs using MTJ material to store data, are non-volatile and have unlimited read and write endurance.

Along with its advantages of small dimensions and non-volatility, MRAM has the potential to be fabricated on top of a conventional microprocessor, thus providing very high bandwidth. The access time and cell size of MRAM memory has been shown to be comparable to DRAM memory [36, 32, 28, 31]. Thus, MRAM memory has attributes which make it competitive with semiconductor memory.

2.1 MRAM Cell

Figure 1 shows the different components of an MRAM cell. The cell is composed of a diode and an MTJ stack, which actually stores the data. The diode acts as a current rectifier and is required for reliable readout operation. The MTJ stack material consists of two ferromagnetic layers separated by a thin dielectric barrier. The polarization of one of the magnetic layers is pinned in a fixed direction, while the direction of the other layer can be changed using the direction of current in the bitline. The resistance of the MTJ depends on the relative direction of polarization of the fixed and the free layer, and is minimum or maximum depending on whether the direction is parallel or anti-parallel to each other. When the polarization is anti-parallel, the electrons experience an increased resistance to tunneling through the MTJ stack. Thus, the information stored in a selected memory cell can be read by comparing its resistance with the resistance of a reference memory cell located along the same wordline. The resistance of the reference memory cell always remains at the minimum level.

As the data stored in an MRAM cell are non-volatile, MRAMs do not consume any static power. Also, MRAM cells do not have to be refreshed periodically like DRAM cells. However, the read and write power for MRAM cells are considerably different as the current required for changing the polarity of the cell is almost 8 times than that required for reading. A more complete comparison of the physical characteristics of competing memory technologies can be found in [9].

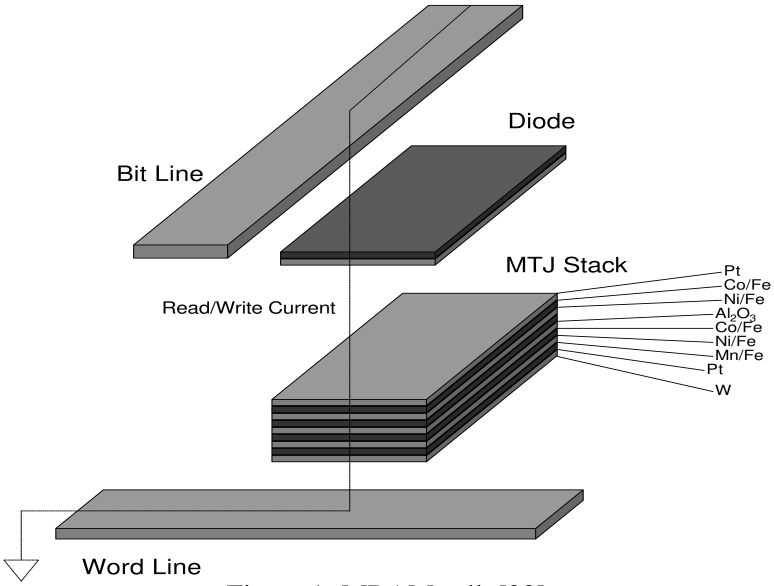

Figure 1: MRAM cell [38]

The MRAM cell consists of a diode, which currently can be fabricated using excimer laser processing on a metal underlayer, and an MTJ stack which can be fabricated using more conventional lithographic processes. The diode in this architecture must have a large on-to-off conductance ratio to provide isolation of the sense path from the sneak paths. This isolation is achievable using thin film diodes. Schottky barrier diodes have also been shown to be promising candidates for current rectification in MRAM cells. Thus, MRAM memory has the potential to be fabricated on top of a conventional microprocessor in wiring layers. However, the devices required to operate the MRAM cells namely, the decoders, the drivers, and the sense amplifiers, cannot be fabricated in this layer and hence must reside in the active transistor layers below. Thus, a chip with MRAM memory will have area overhead associated with these devices. The data cells and the diode themselves do not result in any silicon overhead since they are fabricated in metal layers.

One of the main challenges for MRAM scalability is the cell stability at small feature sizes, as thermal agitation can cause a cell to lose data. However, researchers are already addressing this issue and techniques have been proposed for improving cell stability down to 100 nm feature size. Also, IBM and Motorola are already exploring 0.18 um MRAM designs, and researchers at MIT have demonstrated 100 nm x 150 nm prototypes. While there will be challenges for design and manufacture, existing projections indicate that MRAM technology can be scaled, and with enough investment and research, will be competitive with other conventional and emerging memory technologies.

2.2 MRAM Bank Design

Figure 2 shows an MRAM bank composed of a number of MRAM cells located at the intersection of every bit and word line [39]. During a read operation, current sources are connected to the bit lines and the selected wordline is pulled low by the wordline driver. Current flows through the cells in the selected wordline and the magnitude of the current through each cell depends on its relative magnetic polarity. If the ferromagnetic layers have the same polarity, the cell will have lower resistance and hence more current will flow through the cell thus reducing the current flowing through the sense amplifiers. The current through the sense amplifiers is shown graphically in Figure 2, when the middle wordline is selected for reading. The bitline associated with the topmost cell experiences a smaller drop in current as the cell has higher resistance compared to the other two cells connected to the selected wordline. This change in current is detected using sense amplifiers,

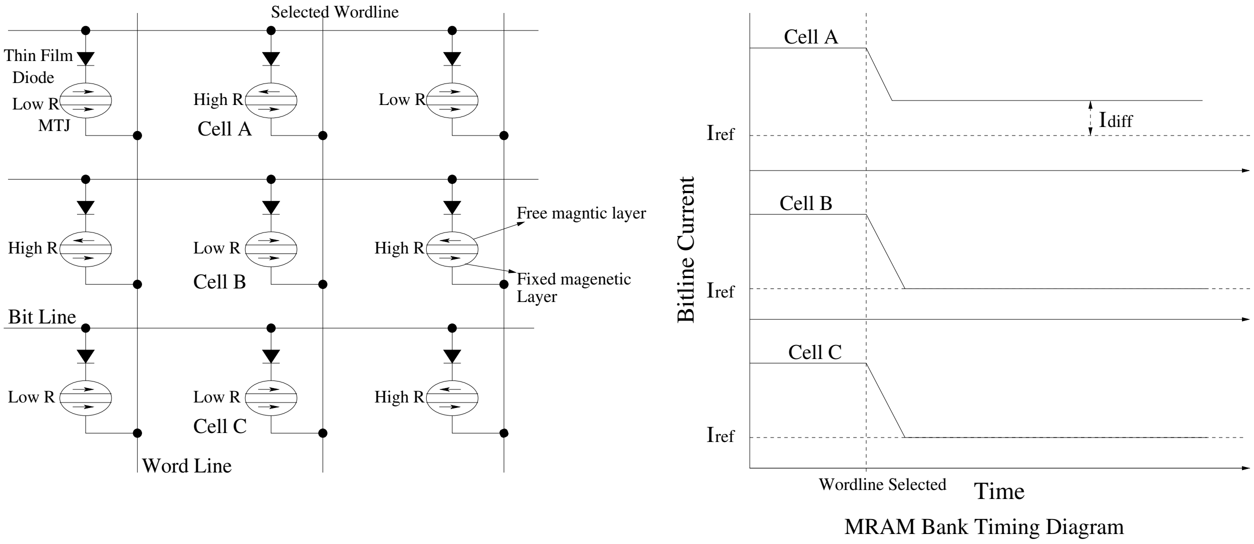

Figure 2: MRAM Bank Organization and Timing

and the stored data is read out. As the wordline is responsible for sinking the current through a number of cells, the wordline driver should be strong to ensure reliable sensing. Alternative sensing schemes have been proposed for MRAM which have increased sensing reliability but also increase the cell area [42].

2.3 MRAM Bank Modeling

To estimate the access time of an MRAM bank and the area overhead in the transistor layer due to the MRAM banks, we developed an area and timing tool by extending CACTI-3.0 and adding MRAM specific features to the model [41]. In our model, MRAM memory is divided into a number of banks which are independently accessible. The banks in turn are broken up into sub-banks to reduce access time. The sub-banks comprising a bank, however, are not independently accessible. Some of the important features we added to model MRAMs include:

1. The area consumed in the transistor layer by the devices required to operate the bank including decoders, wordline drivers, bitline drivers, and sense amplifiers.

2. The delay due to vertical wires carrying signals and data between the transistor layer and the MRAM layer [2].

3. MRAM capacity for a given die size and MRAM cell size.

4. Multiple layers of MRAM with independent and shared wordlines and bitlines.

We used the 2001 SIA roadmap for the technology parameters at 90 nm technology [33]. Given an MRAM bank size and the number of sub-banks in each bank, our tool computes the time to access the MRAM bank by computing the access time of the sub-bank and accounting for the wire delay to reach the farthest sub-bank. To compute the optimal sub-bank size, we looked at designs of modern DRAM chips and made MRAM sub-bank sizes similar to current DRAM sub-bank sizes [20, 21]. We computed the access latency for various sub-bank configurations using our area and timing model. This latency is shown in Table 1. From this table it is clear that the latency increases substantially once we increase the sub-bank size beyond 8 Mb. We fixed 4 Mb as the size for the sub-banks in our system. Our area and timing tool was

Size (Mb)	1	2	4	8	16	32	64
Latency (ns)	4.5	5.6	7.9	10.7	18.7	28.3	59.3

Table 1: MRAM sub-bank latency as a function of capacity.

then used to compute the delay for banks composed of different number of sub-banks. We added a fixed 5ns latency to the bank latency to account for the MRAM cell access latency which is half the access time demonstrated in current prototypes [32]. The 4Mb bank latency was then used in our architectural model which is described in the next section.

We used a single vertical MRAM layer in our evaluation. Future implementations with multiple MRAM layers will result in a much larger memory capacity, but will also increase the number of vertical buses and the active area overhead, if the layers have to be independently accessed. It might be possible to reduce the number of vertical buses and the active area overhead by either sharing the wordlines or the bitlines among the different layers. Sharing bitline among layers might interfere with reliable sensing of the MRAM cells. Evaluation of multiple layer MRAM architectures is a topic for future research.

3 A Chip-Level MRAM Memory Hierarchy

MRAM memory technology promises large memory capacity close to the processor. With global wire delays becoming significant, we need a different approach to managing this memory efficiently, to ensure low latency access in the common case [1]. In this section, we develop a distributed memory architecture for managing MRAM memory, and use dynamic page allocation policies to distribute the data efficiently across the chip.

3.1 Basic MRAM System Architecture

Our basic MRAM architecture consists of a number of independently accessed MRAM banks distributed across the chip. As described in Section 2, the data stored in the MRAM banks are present in a separate vertical layer above the processor substrate, while the bank controller and other associated logic required to operate the MRAM bank reside on the transistor layer. The banks are connected through a network that carries request and response packets between the level-1 (L1) cache and each bank controller. Figure 3 shows our proposed architecture, with the processor assumed to be in the center of the network.

To cache the data stored in each MRAM bank, we break the SRAM L2 cache into a number of smaller caches, and associate each one of these smaller caches with an MRAM bank. The SRAM cache associated with each MRAM bank has low latency due to its small size, and has a high bandwidth vertical channel to its MRAM bank. Thus, even for large cache lines, the cache can be filled with a single access to the MRAM bank on a miss. Each SRAM cache is fabricated in the active layer below the MRAM bank with which it is associated. The SRAM cache is smaller than the MRAM bank and can thus easily fit under the bank. The decoders, sense amplifiers, and other active devices required for operating an MRAM bank are also present below each MRAM bank. We assume MRAM banks occupy 75% of the chip area in the metal layer, and the SRAM caches and associated MRAM circuitry occupy 60% of the chip area in the active layer. Each node in the network has a MRAM bank controller that receives requests from the L1 cache and checks its local L2 cache first to see if the data are present in it. On a cache hit, the data are retrieved from the cache and returned via the network. On a cache miss, the request is sent to the MRAM bank which returns the data to the controller and also fills its associated L2 cache. We model channel and buffer contention in the network, and also model contention for the ports associated with each SRAM cache and MRAM bank.

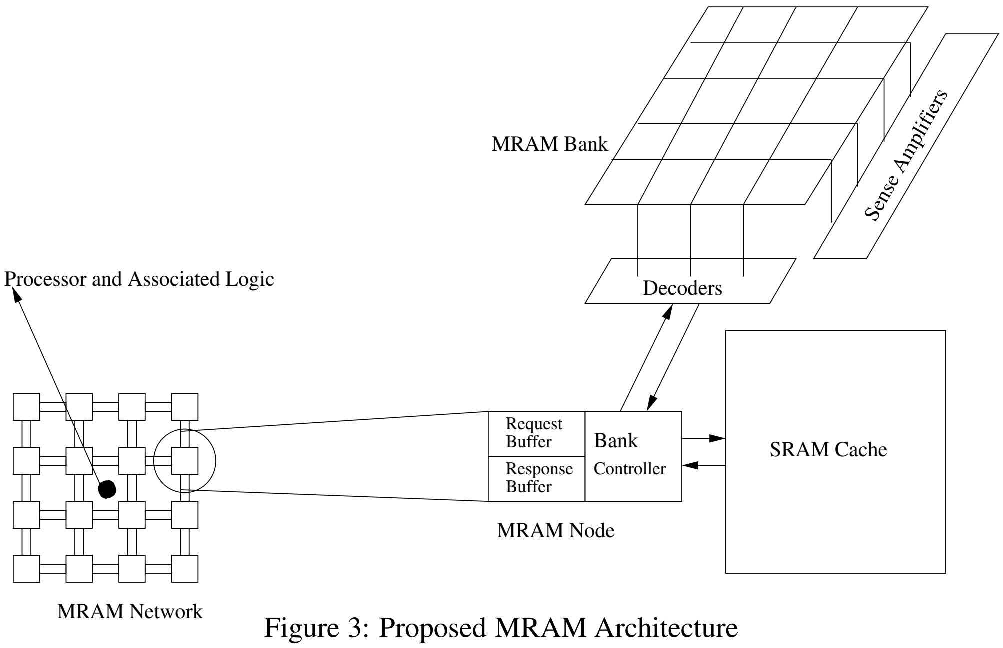

Figure 3: Proposed MRAM Architecture

3.2 Factors influencing MRAM design space

The cost to access data in an MRAM system depends on a number of factors. Since the MRAM banks are distributed and connected by a network, the latency to access a bank depends on the bank's physical location in the network. The access cost also depends on the congestion in the network to reach the bank, and the contention in the L2 cache associated with the bank. Understanding the trade-offs between these factors is important to achieve high performance in an MRAM system.

Number of banks: Having a large number of banks in the system increases the concurrency in the system, and ensures fast hits to the closest banks. However, the network traversal cost to reach the farthest bank also increases due to the increased number of hops. The amount of L2 cache associated with each bank depends on the number of banks in the system. For a fixed total L2 size, having more number of banks results in a smaller size for the L2 cache associated with each bank. However, the latency of each L2 cache is lower now because of its smaller size. Thus increasing the number of banks in the system results in reduced cache and MRAM bank latency (because of smaller bank size for a fixed total MRAM capacity), while increasing the potential miss rates in each individual L2 cache and increased latency to traverse the network due to greater number of hops.

Cache Line Size: Because of the potential for MRAM to provide a high-bandwidth interface to its associated L2 cache, we can have large line sizes in the L2 cache which can potentially be filled with a single access to MRAM on an L2 miss. Large line sizes can result in increased spatial locality but they also result in an increase in the access time of the cache. Thus, there is a trade-off between increased locality and increased hit latency which determines the optimal line size when bandwidth is not a constraining factor. In addition, the line size has an effect on the number of bytes written back into an MRAM array, which is important due to the substantial amount of power required to perform an MRAM write compared to a read.

Page Placement Policy: The MRAM banks are accessed using physical addresses, and the memory in the banks is allocated on a page granularity. Thus, when a page is loaded, the operating system can assign a

Processor Core	Alpha 21264 pipeline Fetch Width - 8 Issue Width - 8 Integer, 4 Floating Point Commit Width - 22
Processor Frequency	3.8 Ghz (Clock period of 10 FO4 inverters per stage)
Size of Structures	Fetch Queue - 8 Issue Queue - 40 Integer, 30 Floating Point Reorder Buffer - 160 Load/Store queue - 64 Load, 64 Store Physical Registers - 82 Integer, 82 Floating Point
Functional Units	Integer - 8 ALUs, 4 multipliers Floating Point - 2 ALUs, 2 multipliers
Branch Predictor	Alpha 21264 Tournament Predictor
L1 D-Cache	16 KB, 2-way set associative, 64 byte line size, 2 cycles hit latency
L1 I-cache	16 KB, 2-way set-predict, 64 byte line size, 1 cycle hit latency

Table 2: Simulated Processor Configuration

frequently accessed page to a closer bank to ensure low latency access. However, assigning a large number of pages to a particular bank can result in greater network congestion and pollution of the cache associated with the bank. Thus, a good page assignment minimizes the distance traveled between the processor and the location of the page, while mitigating potential contention and congestion for any individual cache bank.

4 Results

In this section we evaluate the performance of a uniprocessor MRAM system, and compare it against our base case SDRAM implementation. To determine the optimal number of banks and the optimal cache line size for each bank, we measure the performance of an MRAM system with different number of banks, and look at different cache line sizes for each bank organization. After determining the optimal number of banks and optimal cache line size, we compare the performance of three different page placement policies. We also perform a limit study of an MRAM system by comparing the performance of an ideal MRAM system, an MRAM system with perfect caches, and an MRAM system with the minimum network latency to all nodes with no network contention, i.e. a perfect network. Finally, we do a sensitivity analysis of MRAM bank latency for our best bank configuration.

4.1 Methodology

Simulated Processor: We use the results of our area and timing model to drive our architecture simulator which models a distributed MRAM network, and evaluate the performance of a set of memory intensive benchmarks. We use the sim-alpha simulator for modeling the processor, and extend the memory model to incorporate our MRAM model [8]. The clustered functional unit implementation found in the Alpha 21264 is disabled to support a wider issue processor. We use die photos of current high performance processors to estimate the percentage of die area devoted to L2 caches [10, 12]. We assume the same percentage of die area will be devoted to L2 caches in future processors at 90 nm technology, and use CACTI-3.0 to estimate the size of the cache which will fit in this area [41]. Table 2 lists the various parameters for our simulated processor.

Base SDRAM System: For our base case, we assume a conventional SRAM cache backed up by external off-chip DRAM. Our area estimates using CACTI yield a cache size of 4 MB if 60% of the die is devoted to L2 cache. To reduce the hit latency of a large monolithic L2 cache, Kim et al. suggest a distributed cache organization consisting of small lower latency cache banks [19]. We assume a similar cache organization in our base case and organize the L2 cache in an 8x8 network of 64 KB banks. The addresses are statically mapped to each bank. Each node in the network has buffers and logic to store and forward a request to the appropriate bank. There is a central bank controller which puts packets on the cache network, and on a miss, sends the request to external SDRAM. A miss must come back to the controller from a cache bank before it can be sent to external memory, as we assume a single memory channel in our organization. All cache banks have a single read/write port. Each individual cache bank is 2-way set associative and we vary the line size to evaluate the best possible organization. We assume a one cycle channel latency between banks. The minimum and maximum network latency to a bank in the system, assuming no channel or buffer contention, is 3 cycles and 15 cycles respectively. The width of the bus to main memory in our system is 16 bytes. We model contention on the bus and assume that it runs at 800 MHz. The SDRAM model, provided by Cuppu et al., models page mode SDRAM [7]. The SDRAM latency was determined by looking at the SDRAM of current high performance processors and assuming a 7% decrease in latency per year. The maximum latency to access the SDRAM without bus contention is 115 cycles.

Baseline MRAM System: Our baseline MRAM system has a cache organization similar to the SDRAM system in that we model a distributed cache connected by a routing network. However, each cache bank in the MRAM system is connected to a single MRAM bank and caches data only for that bank. The MRAM network in the simulator is fully configurable and uses Manhattan routing to route a packet to its destination. There are separate request and response buffers at each node to store the packets traversing the network. We assume that we can fabricate MRAM only over 75% of the chip and estimated the capacity of single layer MRAM as 200 MB using our area tool. Memory in MRAM banks is allocated on a page granularity, with each page being 8 KB in our simulated system. We estimated the overhead of the decoders, drivers, and sense amplifiers for operating the MRAM banks as 20% of the die area using our area tool. We subtract this area overhead from the amount available for the L2 cache. Thus, the total amount of L2 cache available in our MRAM system is 33% less than the SDRAM system and we divide this cache evenly among the MRAM banks in our system.

Benchmarks: To evaluate the performance of our memory system, we use six SPEC2000 floating-point benchmarks [35], five SPEC2000 integer programs, 4 scientific applications – 3 from the NAS suite [3] and smg2000 [5], and a speech recognition program, sphinx [26]. We chose these benchmarks because they have been previously shown to have high L1 miss rates, and simulate those instructions of the application which capture the core repetitive phase of the program [19]. Table 3 lists these benchmarks along with the number of instructions skipped to reach the desired phase, and the number of instructions simulated to capture the behavior of the desired phase. The table also lists each application's memory footprint. For all applications except *bt*, the footprint is less than the capacity of MRAM in our system. Application *bt*'s larger working set would result in page replacement for a single layer MRAM or would require larger capacity memory using multiple MRAM layers to avoid page replacements. We do not simulate page replacements in our simulator and assume that the working set can fit in a single layer MRAM.

4.2 MRAM System Cache Line Size and Number of Banks

Our evaluation of an MRAM system is a two step process. As described in Section 3 we first use our area and timing model to get the access latency for different size sub-banks. After choosing an optimal sub-bank

SPECINT2000	Phase FFWD	RUN	Memory Footprint	SPECFP2000	Phase FFWD	RUN	Memory Footprint
gcc	2.4B	300M	23.8 MB	mgrid	550M	1.1B	55.5 MB
mcf	5.0B	200M	39.2 MB	mesa	570M	200M	9.4 MB
parser	3.7B	200M	19.8 MB	applu	267M	650M	181 MB
bzip2	744M	1.0B	182 MB	art	2.2B	200M	3.6 MB
twolf	511M	200M	1.7 MB	galgel	4.0B	200M	37.9 MB
				equake	4.4B	200M	48.7 MB
Speech				Scientific			
sphinx	6.0B	200M	0.3 MB	NAS-cg	600M	200M	55.3 MB
				NAS-bt	800M	650M	293 MB
				NAS-sp	2.5B	200M	78.9 MB
				smg2000	1.2B	800M	122.2 MB

Table 3: Benchmarks used for performance experiments

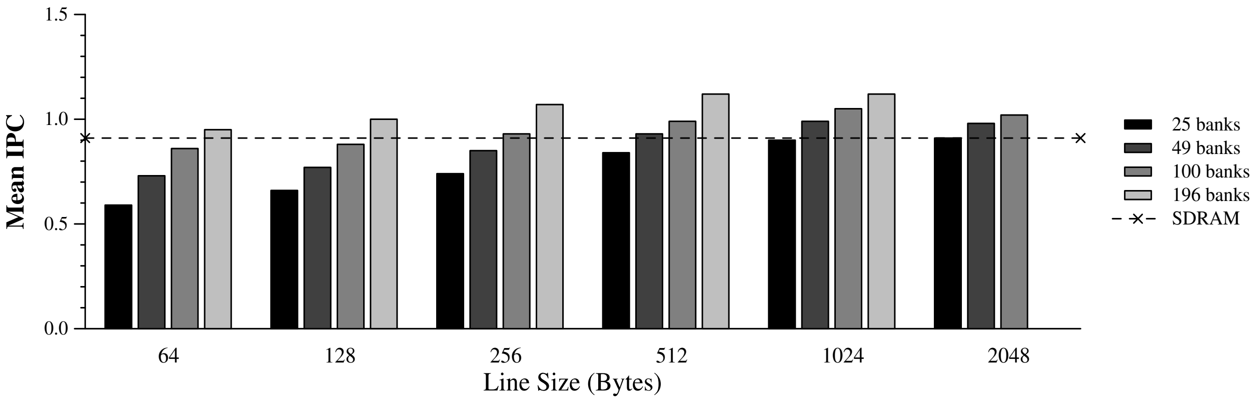

Figure 4: Mean IPC with different number of banks for different line sizes

size, we compute the worst case latency for banks with different numbers of sub-banks, which is then used in our architectural simulation model to compute the performance in Instructions Per Cycle (IPC) for the various organizations. By varying the number of MRAM banks, we consider four different MRAM systems with 25, 49, 100, and 196 banks respectively. An MRAM bank and the associated L2 cache capacity in the four cases are (8 MB, 128 KB), (4 MB, 64 KB), (2 MB, 32 KB), and (1 MB, 16 KB) respectively. The caches are all single ported and 2-way set associative. The page allocation policy is round-robin (described in the next section). A chief advantage of an MRAM system is that it can provide high bandwidth to the L2 cache sitting below it. Thus, we assume that the MRAM bank can fill the cache line in a single access for line sizes up to 2048 bytes. We obtained the latency of the SRAM cache for different line sizes using CACTI.

Figure 4 shows the mean IPC across our set of 16 benchmarks for different number of banks and different L2 cache line sizes. From Figure 4 we can see that performance improves for all bank configurations with larger line size, up to 1 KB, due to increased spatial locality. But beyond 1KB line size, IPC decreases due to the increase in the cache hit latency. Thus, the optimal line size is 1 KB. We do not show 2 KB line size for the system with 196 banks because the cache size of 16 KB is too small for CACTI to support the line size. We can also see from Figure 4 that performance increases as we increase the number of banks in the system. However, we found that for most benchmarks, the performance with 100 banks is higher than the performance with 196 banks. This trend can be seen from Figure 5. *art* is the only benchmark which shows a significant improvement in performance with 196 banks. *art* has a small memory footprint and increasing the number of banks from 100 to 196 effectively removes all conflict misses in the L2 cache banks. Hence,

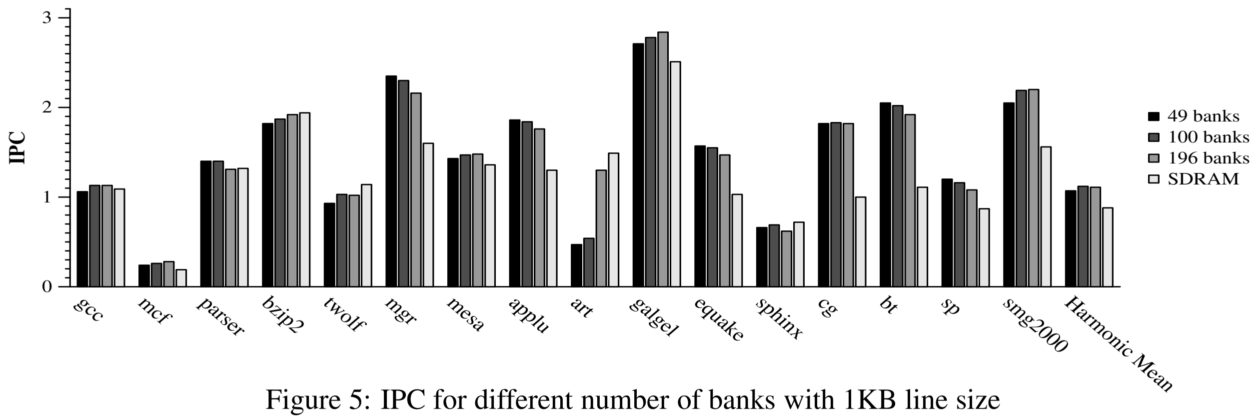

Figure 5: IPC for different number of banks with 1KB line size

we choose the 100 bank configuration with a 1KB line size as the default configuration for the rest of our experiments. We do not include *art* in the mean in Figure 5 because it biases the harmonic mean towards the 196 bank configuration, even though the majority of the benchmarks have higher performance with 100 banks.

4.3 Page Allocation Policy

In our MRAM system, pages are assigned statically to banks and there is no migration of pages in the system from one bank to another. Hence it is very important for the operating system to adopt a good page placement policy. We try three different policies in our system namely *random*, *round-robin*, and *least-loaded*. A random policy assigns pages randomly to banks. In the round-robin policy, we start assigning to the closest bank and progressively move out until we have assigned a page to all the banks, and then start the process over. In the least-loaded policy, we associate a cost with each bank. This is initially the distance of the bank from the controller. This cost is updated whenever a request to an MRAM bank comes back to the central network controller. The new cost is computed as :

$$\begin{aligned} cost &= (L2HitRate * L2HitLatency) \\ &\quad + (L2MissRate * MRAMBankLatency) \\ &\quad + currentLatencyToTheBank \end{aligned}$$

The L2 hit and miss rates are for the cache associated with the MRAM bank. Note that the dynamic latency on the network to access the bank takes into account both the physical distance of the bank and the amount of contention the bank is currently experiencing in the network, while the load on the L2 cache associated with the bank is measured using its hit and miss rates. When a page is first touched, it is assigned to the bank that has the lowest cost. The performance for these policies is shown in Figure 6.

Surprisingly, round-robin policy seems to perform the best among the different policies. Our least-loaded policy seems to perform well on some benchmarks but its average performance is worse than both the round-robin and the random policy. We examined the page allocation pattern of our benchmarks and found that for many of them, most of the pages are allocated at program start up. Thus, the dynamic mechanism of the least loaded scheme is not very effective in distributing the pages evenly among banks, and tends to increase cache pollution by allocating more pages to a small subset of banks.

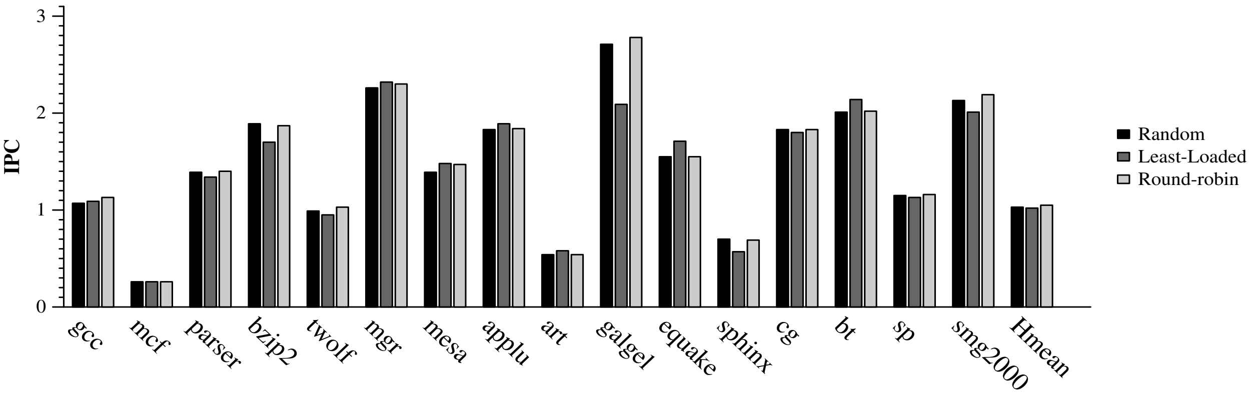

Figure 6: IPC for 100 banks with different page placement policies

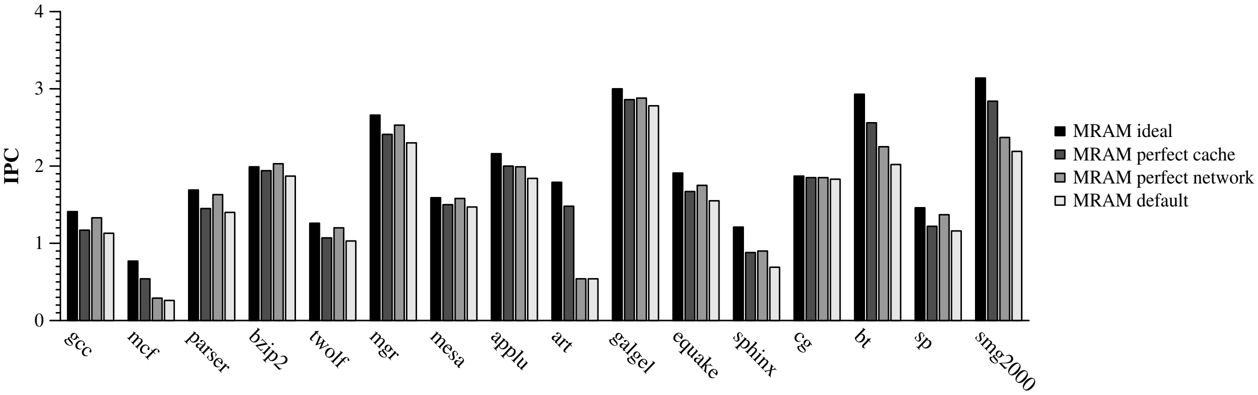

Figure 7: Performance breakdown for the latency components in an MRAM system

4.4 Cache and Network Influence

To determine the influence of the caches and the network on MRAM latency, we ran simulations to measure the performance of an ideal MRAM system, a system with perfect caches, and a system with perfect network. The perfect cache system models only network contention and assumes that the request always hits in the local L2 cache. The perfect network assumes real caches, but models no network contention and the latency to reach any bank is the minimum latency in the system. The ideal case combines perfect cache and perfect network i.e. the request always hits in the closest bank. These results are shown in Figure 7. We also show the performance of the default MRAM configuration derived in Section 4.2.

As can be seem from Figure 7, for some benchmarks like *bzip2* and *cg* the performance is relatively unaffected by the network latency and cache misses while for other benchmarks like *art*, *bt*, and *smg2000* there is considerable room for improvement. We can also see from this figure that the effect of the network and the cache on performance is quite benchmark dependent.

4.5 MRAM Latency Sensitivity

To study the sensitivity of our MRAM architecture to MRAM bank latency, we examine the performance of our MRAM system with increasing bank latencies. The mean performance of the system across our set of benchmarks for different bank access latencies is shown in Figure 8. The horizontal line represents the mean IPC for the conventional SDRAM system. As can be seen from this graph, the performance of our architecture is relatively insensitive to MRAM latency and breaks even with the SDRAM system only at MRAM latencies larger than the off-chip SDRAM latency. This phenomenon occurs because the higher

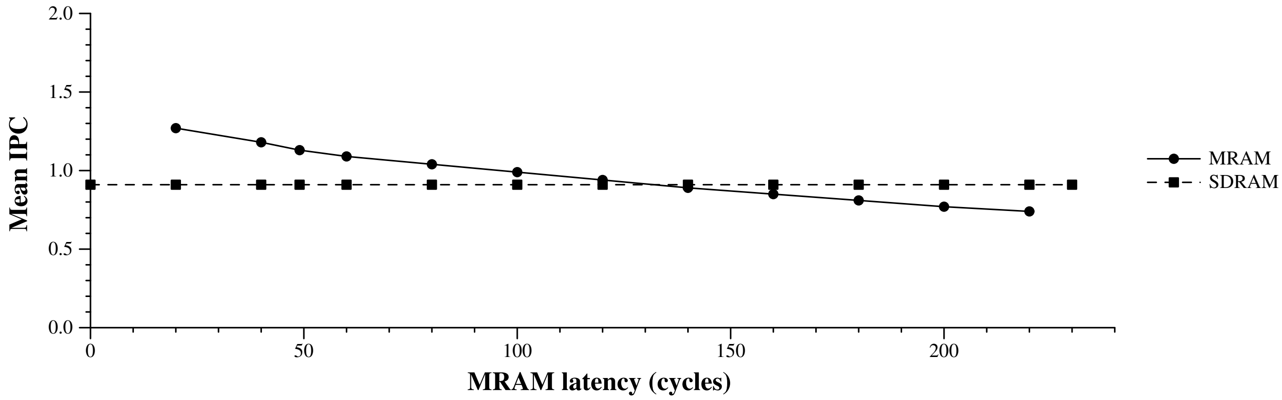

Figure 8: MRAM Latency Sensitivity

Line size	mcf	parser	twolf	mesa	galgel	equake	sphinx	sp
64	586.8	26.8	9.9	7.8	9.7	23.7	26.0	184.2
128	1187.2	28.8	21.8	8.2	15.8	24.5	50.4	184.9
256	1714.6	30.1	34.6	8.9	25.8	25.4	100.1	185.9
512	2738.2	31.9	61.5	11.4	51.8	26.0	190.8	190.5
1024	4721.9	34.1	112.2	26.9	98.1	26.9	344.1	227.1
2048	8633.1	38.7	211.3	83.5	169.3	28.3	595.7	429.3

Table 4: Megabytes written back into MRAM from L2 cache

bandwidth in the MRAM system is able to more easily tolerate larger latencies.

4.6 Cost of Writes

Writes to MRAM memory consume more power than reads because of the larger current needed to change the polarity of the MRAM cell. Hence, for a low power design, it might be better to consider an architecture which minimizes the amount of data written back into the MRAM banks from the L2 cache. In Table 4 we show the total number of bytes written back into MRAM memory for a 100 bank configuration with different line sizes. We show only a subset of the benchmarks as all the benchmarks show the same trend. From Table 4 we can see that the total volume of data written back increases with increasing line size. We found that even though the number of writebacks decreases with larger line size, the amount of data written back increases. This is because the decrease in the number of writebacks is offset by the increased line size. Thus, there is a power performance trade-off in an MRAM system as larger line sizes consume more power but yield better performance. We are currently exploring other mechanisms such as sub-blocking to reduce the volume of data written back when long cache lines are employed.

5 The Future of Solid-State Memory Technologies

While the feasibility of MRAM technology has not yet been proven, nearer term dense memory technologies are emerging that may help ameliorate memory latency and bandwidth. Embedded DRAMs (eDRAM) [40, 17, 16] are delivering higher memory density on the same die as logic devices. While bandwidth to this on-chip DRAM is high, for many applications, the total capacity may not be sufficient to employ eDRAM as a substitute for main memory and may be better suited for implementation as a cache.

Technology	Capacity	Number of vertical wires/mm^2	Access latency (cycles)	Total L2 cache
Stacked SDRAM	256 MB	25	115	4 MB
MRAM	200 MB	2048	49	2.7 MB

Table 5: Stacked SDRAM Vs MRAM Comparison

Chipstacking is a packaging solution that promises high bandwidth access to conventional DRAM memory stacked on top of a processing core [30, 27]. In chipstacking, a high bandwidth low cost connection is provided between the processor and commodity DRAM by using a C4 process, which provides a real connection contact between the chip and its carrier. A 400 mm^2 chip can support as many as 10,000 connections thus providing high bandwidth access to memory above the chip. However, since we are connecting two separate chips, the bandwidth is much less than that provided by the more tightly vertically integrated MRAM technology. Another advantage of MRAM is that capacity can be more easily increased by fabricating additional MRAM layers on top of one another, while the third dimension of interconnect for multiple stacked DRAM chips may not be possible.

To determine the relative effectiveness of MRAM to chipstacked DRAM, we estimated their differences in bandwidth and latency and simulated the chipstacked DRAM in a page-based organization comparable in structure to that of our MRAM architectures. We assumed that a total of 8192 connections could be implemented between the processor chip and the stacked DRAM chip. As shown in Table 5, the reduced memory bandwidth decreases the number of useful memory channels to 16, each with its own 256KB L2 SRAM cache. The SDRAM itself is similar to our base case, except that the data bus is much wider and runs at a faster rate of half the processor frequency. We used the round-robin page placement policy as that yielded the best performance in the MRAM system. As in the MRAM case, we selected the line size that yielded the best performance.

Figure 9 shows the performance of our best conventional SDRAM system, best stacked DRAM system, and best MRAM system. The MRAM system performs better by 15% and 30% over conventional DRAM and stacked DRAM. Not surprisingly, the stacked DRAM system performs better than conventional off-chip DRAM on benchmarks which have large bandwidth requirements. However, for a number of benchmarks, the off-chip DRAM actually performs better. This phenomenon occurs for benchmarks that have a high L1 miss rate but relatively low L2 miss rate. The size of each L2 bank is 4 times as large for stacked-DRAM than for conventional DRAM, as each L2 bank is connected to one stacked DRAM bank and memory channel; thus each L2 access is somewhat slower due to the larger capacity. We believe that the stacked-DRAM can be architected to always perform at least as well as conventional DRAM by reducing the bank size and having multiple L2 banks share a single stacked DRAM bank and memory channel. We will investigate this issue further in future work.

6 Related Work

Performance of on-chip DRAM memory was evaluated by Yamauchi et al., in the context of a uniprocessor and a chip multiprocessor (CMP) [43]. They found that DRAM-based CMP performed 52% better than an SRAM based architecture on floating point applications with high bandwidth requirements. The DRAM-based uniprocessor only performed 4% better. The authors concluded that high bandwidth memory is more useful for improving performance of chip multiprocessors which have significantly higher bandwidth requirements than uniprocessors. We expect that future studies on using MRAM for chip multiprocessors will show a similar benefit.

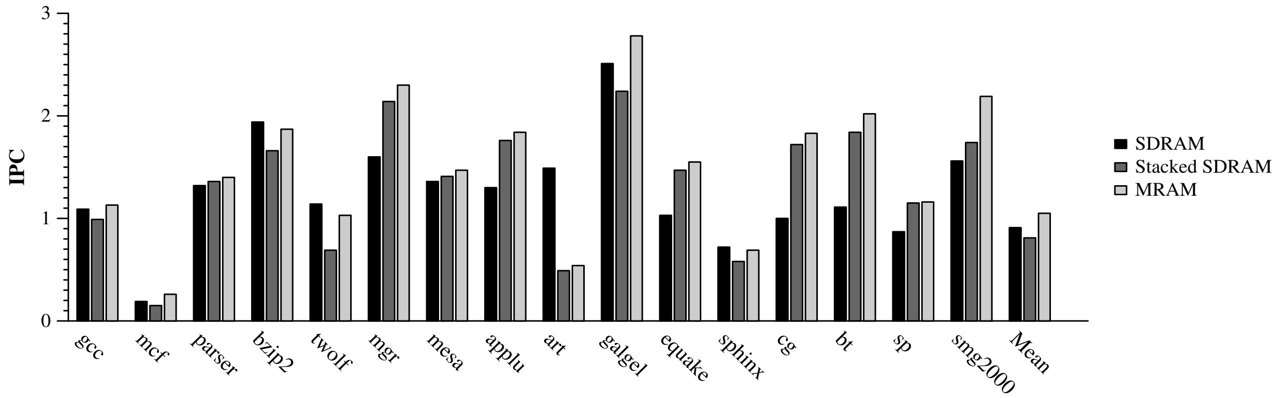

Figure 9: Performance comparison of Conventional SDRAM, Stacked SDRAM, and MRAM

More recently, Keltcher et al., did an equal area comparison of embedded DRAM and SRAM memory architectures for a chip multiprocessor [18]. The authors fixed the percentage of die area devoted to on-chip memory, and studied the impact of varying the die area devoted to L1 and L2 caches. The L1 is assumed to be SRAM cache, and the authors evaluate the L2 cache as SRAM and eDRAM both as a cache and as paged memory. The authors conclude that considering the cache as paged memory does not offer any significant improvement in performance. Our conclusions differ as we assume that MRAM capacity is large enough to capture the working set of the applications without paging during program execution. Although dynamic paging is likely to have an effect on total performance, we expect the likely benefits of MRAM over DRAM systems to persist.

A number of researchers have looked at architecting computing systems that integrate processors and DRAM on the same die [22, 29, 15, 23]. These studies each propose novel fine-grained processor architectures that are tightly coupled to DRAM banks and show good performance improvements on classes of easily parallelizable applications. In our study, we focus on more conventional processor architectures and aim to improve single-thread performance. From a technology perspective, mixed Logic-DRAM fabrication technologies still have a significant drawback: the logic circuits, the memory circuits, or both will suffer from not being implemented in a fabrication technology tailored to the circuit family. In principle, neither MRAM nor stacked-DRAM technologies have this drawback as the memory and logic devices can be fabricated separately.

Page migration has been extensively studied in multiprocessor systems [24, 6, 37]. This technique reduces memory access time by moving data pages into the memory of the processor that is accessing them. Our MRAM memory architecture will allow a similar optimization to be made for uniprocessor systems. Lebeck et al. have suggested that page allocation and migration policies can be tuned to reduce energy used in the memory hierarchy [25]. This is a natural fit with a partitioned L2 and main memory architecture such as ours. For example, the L2 cache could be turned off when the memory bank it serves is not used for long periods of time. We plan to evaluate such policies in future work.

7 Conclusions

In this paper, we have introduced and examined an emerging memory technology, MRAM, which promises to enable large, high bandwidth memories. MRAM can be integrated into the microprocessor die and avoid the conventional pin bandwidth limitations found in off-chip memory systems. We have developed a model for simulating MRAM banks and use it to examine the trade-offs between line size and bank number to derive the MRAM organization with the best performance. We break down the components of latency in the memory system, and examine the potential of page placement to improve performance. Finally, we have compared MRAM with conventional SDRAM memory systems and another emerging technology, chipstacked SDRAM, to evaluate its potential as a replacement for main memory.

Our results show that MRAM systems perform 15% better than conventional SDRAM systems and 30% better than stacked SDRAM systems. An important feature of our memory architecture is that the L2 cache and MRAM banks are partitioned. This architecture reduces miss conflicts in the L2 cache and provides high bandwidth when multiple L2s are accessed simultaneously. We studied MRAM systems with perfect L2 caches and perfect networks to understand where performance was being lost. We found that the penalty of cache conflicts in the L2 cache and the network latency had widely varying effects among the benchmarks. However, these results did show that page allocation policies in the operating system have a great potential to improve MRAM performance.

Our work suggests several opportunities for future MRAM research. First, our partitioned MRAM memory system allows page placement policies for a uniprocessor to consider a new variable – proximity to the processor. Allowing pages to dynamically migrate between MRAM partitions may provide additional performance benefit. Second, the energy use of MRAM must be characterized and compared to alternative memory technologies. Applications may have quite different energy use given that the energy required to write the MRAM cell is greater than that to read it. In addition, the L2 cache line size has a strong effect on the amount of data written to the MRAM and may be an important factor in tuning systems to use less energy. Third, since MRAM memory is non-volatile, its impact on system reliability over conventional memory should be measured. Finally, our uniprocessor simulation does not take full advantage of the large bandwidth inherent in the partitioned MRAM. We expect that chip multiprocessors will have additional performance gains beyond the uniprocessor model studied in this paper.

References

[1] V. Agarwal, M. S. Hrishikesh, S. W. Keckler, and D. Burger. Clock rate versus IPC: The end of the road for conventional microarchitectures. In *Proceedings of the 27th Annual International Symposium on Computer Architecture*, pages 248–259, June 2000.

[2] V. Agarwal, S. W. Keckler, and D. Burger. The effect of technology scaling on microarchitectural structures. Technical Report TR2000-02, Department of Computer Sciences, University of Texas at Austin, Austin, TX, Aug. 2000.

[3] D. Bailey, J. Barton, T. Lasinski, and H. Simon. The NAS parallel benchmarks. Technical Report RNR-91-002 Revision 2, NASA Ames Research Laboratory, Ames, CA, Aug. 1991.

[4] H. Boeve, C. Bruynseraede, J. Das, K. Dessein, G. Borghs, and J. D. Boeck. Technology assessment for the implementation of magnetoresistive elements with semiconductor components in magnetic random access memory MRAM architectures. *IEEE Transactions on Magnetics*, 35:2820–2825, Sep 1999.

[5] P. N. Brown, R. D. Falgout, and J. E. Jones. Semicoarsening multigrid on distributed memory machines. Technical Report UCRL-JC-130720, Lawrence Livermore National Laboratory, 2000.

[6] R. Chandra, S. Devine, B. Verghese, A. Gupta, and M. Rosenblum. Scheduling and page migration for multiprocessor compute servers. In Proceedings of the Sixth International Conference on Architectural Support for Programming Languages and Operating Systems, pages 12–24, San Jose, California, 1994.

[7] V. Cuppu, B. Jacob, B. Davis, and T. Mudge. A performance comparison of contemporary DRAM architectures. In *Proceedings of the 26th Annual International Symposium on Computer Architecture*, pages 222–233, May 1999.

[8] R. Desikan, D. Burger, S. W. Keckler, and T. M. Austin. Sim-alpha: a validated execution driven alpha 21264 simulator. Technical Report TR-01-23, Department of Computer Sciences, University of Texas at Austin, 2001.

[9] R. Desikan, S. W. Keckler, and D. Burger. Assessment of MRAM technology characteristics and architectures. Technical Report TR-01-36, Department of Computer Sciences, University of Texas at Austin, 2001.

[10] K. Diefendorff. Power4 focuses on memory bandwidth, October 1999.

[11] W. fen Lin, S. K. Reinhardt, and D. Burger. Reducing DRAM latencies with a highly integrated memory hierarchy design. In *7th Symposium on High-Performance Computer Architecture (HPCA)*, pages 301–312, Jan 2001.

[12] J. M. Hill and J. Lachman. A 900MHz 2.25MB cache with On-Chip cpu - now in Cu SOI. In *2001 IEEE International Solid-State Circuits Conference*, pages 176–177, 444, Feb 2001.

[13] M. Hrishikesh, K. Farkas, N. P. Jouppi, D. Burger, S. W. Keckler, and P. Sivakumar. The optimal logic depth per pipeline stage is 6 to 8 FO4 inverter delays. In *Proceedings of the 29th International Symposium on Computer Architecture*, pages 14–24, May 2002.

[14] J. Huh, D. Burger, and S. W. Keckler. Exploring the design space of future CMPs. In *Proceedings of the 2001 International Conference on Parallel Architectures and Compilation Techniques (PACT 2001)*, pages 199–210, Sep 2001.

[15] Y. Kang, W. Huang, S.-M. Yoo, D. Keen, Z. Ge, V. Lam, P. Pattnaik, and J. Torrellas. FlexRAM: Toward an advanced intelligent memory system. In *Proceedings of the International Conference on Computer Design, ICCD 1999*, pages 192–201, Oct 1999.

[16] D. Keitel-Schulz and N. Wehn. Issues in embedded DRAM development and applications. In *Proceedings of the 11th International Symposium on System Synthesis*, pages 23–28, Dec 1998.

[17] D. Keitel-Schulz and N. Wehn. Embedded DRAM development: Technology, physical design, and application issues. *IEEE Design and Test of Computers*, 18:7–15, May 2001.

[18] P. Keltcher, S. Richardson, and S. Siu. An equal area comparison of embedded DRAM and SRAM memory architectures for a chip multiprocessor. Technical Report HPL-2000-53, Computer Systems Technology HP Laboratories Palo Alto, Apr 2000.

[19] C. Kim, D. Burger, and S. W. Keckler. An adaptive, non-uniform cache structure for wire-dominated on-chip caches. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2002.

[20] T. Kirihata et al. A 390-mm^2, 16-bank, 1-Gb DDR SDRAM with hybrid bitline architecture. *International Journal of Solid-State Circuits*, 34:1580–1588, Nov 1999.

[21] T. Kirihata et al. A 113mm^2 600Mb/sec/pin 512Mb DDR2 SDRAM with vertically folded bitline architecture. In *2001 IEEE International Solid-State Circuits Conference*, pages 382–383, 468, Feb 2001.

[22] P. M. Kogge, S. C. Bass, J. B. Brockman, D. Z. Chen, and E. H. Sha. Pursuing a petaflop: Point designs for 100TF computers using PIM technologies. In *6th Symp. on Frontiers of Massively Parallel Computation*, pages 25–31, Oct 1996.

[23] C. E. Kozyrakis, S. Perissakis, D. Patterson, T. Anderson, K. Asanović, N. Cardwell, R. Fromm, J. Golbus, B. Gribstad, K. Keeton, R. Thomas, N. Treuhaft, and K. Yelick. Scalable processors in the billion-transistor era: IRAM. *Computer*, 30(9):75–78, 1997.

[24] R. P. LaRowe, Jr., C. S. Ellis, and M. A. Holliday. Evaluation of NUMA memory management through modeling and measurements. *IEEE Transactions on Parallel and Distributed Systems*, 3(6):686–701, 1992.

[25] A. R. Lebeck, X. Fan, H. Zeng, and C. S. Ellis. Power aware page allocation. In *Architectural Support for Programming Languages and Operating Systems*, pages 105–116, 2000.

[26] K.-F. Lee, H.-W. Hon, and R. Reddy. An overview of the SPHINX speech recognition system. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38:35–44, 1990.

[27] R. Montoye, E. Sprogis, and P. Hofstee. Chipstack and DCA DRAM: Maximize bandwidth and size at reasonable cost. May 2000.

[28] P. K. Naji, M. Durlam, S. Tehrani, J. Calder, and M. F. DeHerrera. A 256kb 3.0v 1T1MTJ nonvolatile magnetoresistive RAM. In *2001 IEEE International Solid-State Circuits Conference*, pages 122–123, 438, Feb 2001.

[29] M. Oskin, F. T. Chong, and T. Sherwood. Active pages: A computation model for intelligent memory. In *Proceedings of the 25th Annual International Symposium on Computer Architecture, ISCA 1998*, pages 192–203, 1998.

[30] M. Reber and R. Tielert. Benefits of vertically stacked integrated circuits for sequential logic. In *International Symposium on Circuits and Systems*, pages 121–124, May 1996.

[31] R. E. Scheuerlein. Magneto-resistive IC memory limitations and architecture implications. In *Proceedings of the International NonVolatile Memory Technology Conference*, pages 47–50, May 1998.

[32] R. E. Scheuerlein, W. Gallagher, S. Parkin, A. Lee, S. Ray, R. Robertazzi, and W. Reohr. A 10ns read and write non-volatile memory array using a magnetic tunnel junction and FET switch in each cell. In *2000 IEEE International Solid-State Circuits Conference*, pages 128–129, Feb 2000.

[33] The national technology roadmap for semiconductors. Semiconductor Industry Association, 2001.

[34] E. Sprangle and D. Carmean. Increasing processor performance by implementing deeper pipelines. In *Proceedings of the 29th International Symposium on Computer Architecture*, pages 25–34, May 2002.

[35] Standard Performance Evaluation Corporation. *SPEC Newsletter*, Fairfax, VA, Sept. 2000.

[36] S. Tehrani, B. Engel, J. M. Slaughter, E. Chen, M. DeHerrera, M. Durlam, P. Naji, R. Whig, J. Jenesky, and J. Calder. Recent developments in magnetic tunnel junction MRAM. In *IEEE Transactions on Magnetics*, volume 36, pages 2752–2757, Sep 2000.
[37] B. Verghese, S. Devine, A. Gupta, and M. Rosenblum. Operating system support for improving data locality on CC-NUMA compute servers. In Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems, 1996.
[38] S. Vernon. LDRD review presentation, livermore national labs, June 2001.
[39] F. Wang. A modified architecture for high-density MRAM. *Computer Architecture News*, pages 16–22, March 2001.
[40] N. Wehn and S. Hein. Embedded DRAM architectural trade-offs. In *Proceedings of Design, Automation and Test in Europe*, pages 704–708, Feb 1998.
[41] S. Wilton and N. Jouppi. CACTI: An enhanced cache access and cycle time model. In *IEEE Journal of Solid-State Circuits*, May 1996.
[42] K. Yamada, N. Sakai, Y. Ishizuka, and K. Mameno. A novel sensing scheme for a MRAM with a 5% MR ratio. In *2001 Symposium on VLSI Circuits Digest of Technical Papers*, pages 123–124, Mar 2001.
[43] T. Yamauchi, L. Hammond, and K. Olukotun. A single chip multiprocessor integrated with DRAM. Technical Report CSL-TR-97-731, Computer Systems Laboratory Stanford University, Aug 1997.