2012 Sixth IEEE/ACM International Symposium on Networks-on-Chip

A Hybrid Buffer Design with STT-MRAM for On-Chip Interconnects

Hyunjun Jang, Baik Song An, Nikhil Kulkarni, Ki Hwan Yum and Eun Jung Kim
Department of Computer Science and Engineering
Texas A&M University
College Station, Texas, 77843-3112
Email: {hyunjun,baiksong,nikhilvk,yum,ejkim}@cse.tamu.edu

***Abstract*—As the chip multiprocessor (CMP) design moves toward many-core architectures, communication delay in Network-on-Chip (NoC) has been a major bottleneck in CMP systems. Using high-density memories in input buffers helps to reduce the bottleneck through increasing throughput. Spin-Torque Transfer Magnetic RAM (STT-MRAM) can be a suitable solution due to its nature of high density and near-zero leakage power. But its long latency and high power consumption in write operations still need to be addressed. We explore the design issues in using STT-MRAM for NoC input buffers. Motivated by short intra-router latency, we use the previously proposed write latency reduction technique sacrificing retention time. Then we propose a hybrid design of input buffers using both SRAM and STT-MRAM to hide the long write latency efficiently. Considering that simple data migration in the hybrid buffer consumes more dynamic power compared to SRAM, we provide a lazy migration scheme that reduces the dynamic power consumption of the hybrid buffer. Simulation results show that the proposed scheme enhances the throughput by 21% on average.**

***Keywords*-Network-on-Chip; STT-MRAM; router; input buffer;**

I. Introduction

With the continued advance of CMOS technology, the number of cores on a single chip keeps increasing at a rapid pace. And it is highly expected that many-core architectures with more than hundreds of processor cores will be commercialized in the near future. In a large-scale chip multiprocessor (CMP) system, network overheads are more dominant than computation power in determining overall system performance. While shared buses provide networking performance enough for a small number of CMP nodes, they cannot be good solutions for many-core systems due to the limitation on scalability. Accordingly, switch-based networks-on-chip (NoCs) are being adopted as an emerging design trend in many-core CMP environments. Since all components in a chip including processors, caches and interconnects must compete for limited area and power budgets, resources available for NoCs are tightly constrained compared to off-chip interconnects. Moreover, network performance becomes more significant with the increasing scale of CMP systems. Therefore, a new and innovative NoC design that can guarantee better performance with limited resources is necessary for many-core systems.

The advance of memory technology has ushered in new non-volatile memory (NVM) designs that overcome the drawbacks of existing memories such as SRAM or DRAM. Among them, Spin-Torque Transfer Magnetic RAM (STT-MRAM) is being regarded as a promising technology for a number of advantages over the conventional RAMs. STT-MRAM is a next-generation memory that uses magnetic materials as the main information carrier. It achieves lower leakage power and higher density compared to the existing SRAM. Also, STT-MRAM shows higher endurance compared to other NVM techniques such as Phase Change Memory (PCM) or Flash, which makes STT-MRAM more attractive for on-chip memories that must tolerate much more frequent write accesses compared to off-chip memories. However, one of the biggest weaknesses of STT-MRAM is long write latency compared to SRAM. Since the fast access time of memories on a chip must be guaranteed and cannot be negotiable, the slow write operations of STT-MRAM limit its popularity, even though it shows competitive read performance. Another serious drawback of STT-MRAM is high power consumption in write operations. This issue of high power consumption in STT-MRAM must be resolved in NoCs due to the limited power budgets.

Despite these weaknesses, using STT-MRAM in the NoC design has significant merits since an on-chip router can incorporate larger input buffers compared to SRAM with the same area budget because of the higher density of STT-MRAM. Larger input buffers contribute to improving the throughput of NoC, which results in the enhancement of overall system performance. However, the aforementioned challenges must be addressed first to exploit the benefit of STT-MRAM in NoC. Since the input buffer of an on-chip router must handle arriving flits on time, it is impossible in reality to use STT-MRAM without additional technique to hide the long write latency. Moreover, addressing the high write power issue of STT-MRAM is mandated in NoC environments.

In this paper, we explore the design issues of adopting STT-MRAM in on-chip interconnects. First, by relaxing the non-volatility of STT-MRAM, the latency as well as the power consumption in write operations can be reduced at the sacrifice of the retention time [1], [2]. Based on the observation of intra-router latency of flits, we find out

978-0-7695-4677-3/12 $26.00 © 2012 IEEE
DOI 10.1109/NOCS.2012.30

that the retention time needed for input buffers in NoC can be significantly shortened. We exploit the write latency reducing technique [1] in the input buffers of on-chip routers, and decrease the latency to less than 2ns that corresponds to 6 cycles in 3GHz clock frequency. Then we propose a hybrid design of input buffers combining both SRAM and STT-MRAM. By allowing each arriving flit to be stored in the SRAM buffer first and then migrated to STT-MRAM, the write latency of STT-MRAM is effectively hidden, thus increasing network throughput.

Simply migrating each flit from SRAM to STT-MRAM buffer causes significant power consumption due to the high write power of STT-MRAM, compared to existing SRAM-based input buffers. So we design a lazy migration scheme that allows the flit migration only when the network load exceeds a certain threshold, which helps to reduce the power consumption significantly. Simulation results show that the hybrid input buffers improve the network throughput by 21% in synthetic workloads and 14% in SPLASH-2 parallel benchmarks on average compared to pure SRAM-based buffers with the same area overheads. Also, the lazy migration scheme contributes to power reduction by 61% on average compared to the simple migration scheme that always migrates flits from SRAM to STT-MRAM.

The remainder of this paper is organized as follows. We discuss related work in Section II, followed by the performance and power model of STT-MRAM in Section III. In Section IV, we explain the hybrid buffer design using STT-MRAM in detail. Section V presents simulation results and analysis, and finally Section VI summarizes our work and makes conclusions.

II. Related Work

Since there has been no prior work using STT-MRAM in NoC design, we only summarize the relevant studies of STT-MRAM technologies as well as the application of NVM to diverse system domains such as processors and memories.

A. STT-MRAM

STT-MRAM is a next generation memory technology that takes advantage of magnetoresistance for storing data. It uses a Magnetic Tunnel Junction (MTJ), the fundamental building block, as a binary storage. An MTJ comprises a three-layered stack: two ferromagnetic layers and an MgO tunnel barrier in the middle. Among them, the fixed layer located at the bottom has a static magnetic spin, the spin of the electrons in the free layer at the top is influenced by applying adequate current through the fixed layer to polarize the current, and the current is passed to the free layer. Depending on the current, the spin polarity of the free layer changes either parallel or anti-parallel to that of the fixed layer. The parallel indicates a zero state, and the anti-parallel a one state. Figure 1 depicts the two parallel and anti-parallel states of an MTJ module. A single MTJ module is coupled with a transistor to form a basic memory cell of STT-MRAM called a 1T-1MTJ cell.

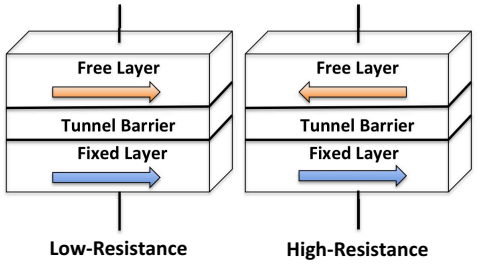


Figure 1: The Two States of An MTJ Module

B. Utilizing NVMs in Processors and Memories

Several schemes have been proposed to provide architectural support for applying NVMs to system components. Jog *et al.* [1] proposed to achieve better write performance and energy consumption of STT-MRAM-based L2 cache through adjusting data retention time of STT-MRAM. Similarly, Smullen *et al.* [2] reduced the write latencies as well as dynamic energy of STT-MRAM by lowering the retention time for designing on-chip caches. In [3], they integrated STT-MRAM into on-chip caches in a 3D CMP environment and proposed a mechanism of delaying cache accesses to busy STT-MRAM banks to hide long write latency. Prior to that, Sun *et al.* [4] stacked MRAM-based L2 caches on top of CMPs and reduced overheads through read-preemptive write buffer and hybrid cache design using both SRAM and MRAM. Guo *et al.* [5] resolved the design issues of microprocessors using STT-MRAM in detail for more power-efficient CMP systems.

PCM also has been constantly explored to replace existing SRAM or DRAM-based memory systems. Due to its lower endurance compared to SRAM or STT-MRAM, PCM is mainly adopted for off-chip memories rather than on-chip caches. Several designs of PCM-based main memory were discussed in [6], [7], [8]. In [9], adaptive write cancellation and write pausing policies were proposed to reduce energy and improve performance. Zhou *et al.* [10] suggested a new memory scheduling scheme that allows Quality-of-Service (QoS) tuning through request preemption and row buffer utilization.

III. Performance and Power Model of STT-MRAM

As an area model of STT-MRAM, we use ITRS 2009 projections [11] as well as the model used in [5], where a 1T-1MTJ cell size is $30F^2$ in the 32nm technology. When we assume that an SRAM cell size is approximately $146F^2$ with the same technology, one SRAM cell can be substituted by at least four STT-MRAM cells under the same area budget. Also, about 3.2ns of write latency can be achieved

with $30F^2$ STT-MRAM cell size [5]. It corresponds to 10 cycles in 3GHz clock frequency, which is quite long for on-chip routers compared to SRAM that completes both read and write accesses in a single cycle. Reducing retention time from 10 years to 10ms guarantees the same write latency with one third of original write current needed [1]. Using lower current is beneficial in terms of area overheads because it facilitates to implement STT-MRAM cells with smaller transistors, which reduces actual cell area.

In this study, we slightly increase write current to reduce this write latency of STT-MRAM further. The write latency reduces from 3.2ns to 1.8ns through increasing the write current from 50μA to 75μA under $125\,^{\circ}$C of a temperature. Note that even this increased current is far less than the original current needed for 10 years of retention time, while maintaining the same STT-MRAM cell size, $30F^2$. Also, the increased current does not hurt write energy consumption since the MTJ switching time decreases accordingly [5]. As a result, the write latency decreases from 10 to 6 cycles in 3GHz clock frequency. The increased write current may hurt the performance in terms of read latency. However, we verify that the reduction of write latency from 3 to 1.8ns affects the read latency to only a small extent [2]. Therefore, we can assume that the increased read latency can still be covered by a single cycle, considering the original read delay of 122ps [5], which is far shorter than 333ps, a cycle time in 3GHz clock frequency.

The relaxed retention time of 10ms may hurt the reliability of data stored in an STT-MRAM buffer, if the retention time is shorter than the intra-router delay of a flit, defined by the time difference between arrival time at the buffer and departure time in a router. Figure 2 depicts maximum intra-router latency for different injection rates ranging from 0.1 to 0.7 with various SRAM buffer sizes per VC, under uniform random synthetic workloads. We observe that the latency does not go up beyond 16 cycles, and it is almost negligible compared to 10ms, which corresponds to more than 30 million cycles in 3GHz clock frequency [1]. Hence, it is confirmed that even the reduced retention time is completely enough to hold a flit in STT-MRAM buffers safely. For the read and write energy model of STT-MRAM, we conservatively adopt the same parameters from [5], 0.01pJ and 0.31pJ per bit for read and write, respectively. Note that these are based on 3.2ns of write latency, so actual write energy becomes smaller after decreasing the latency to 1.8ns.

IV. An On-Chip Router Architecture with Hybrid Buffer Design

In this section, we describe a generic router architecture and a buffer structure in NoC and present our hybrid buffer design that maximizes the mutually complementary features of the two different memory technologies, SRAM and STT-MRAM, while minimizing the drawbacks of STT-MRAM, the long latency and high power consumption in write operations.

[1]Note that in deadlock situations, packets can stay in the network forever. In this study, we adopt deadlock-free routing algorithms, thus avoiding such situations.

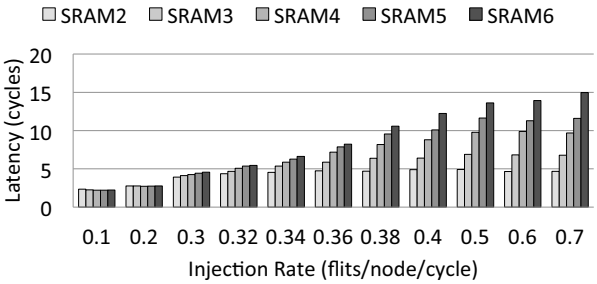


Figure 2: Maximum Intra-Router Latency of An On-Chip Router (*SRAM#*: SRAM Buffer Size per VC)

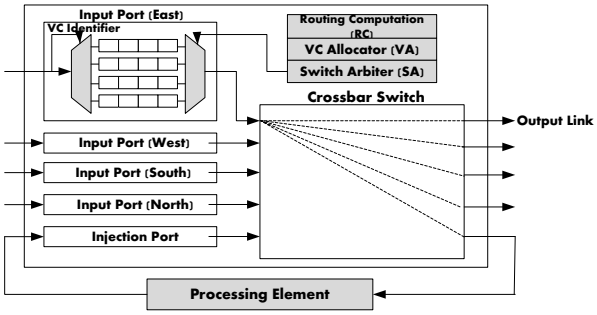


Figure 3: Generic Router Architecture

A. Generic Baseline Router Architecture

The generic NoC router architecture is depicted in Figure 3. It is based on the state-of-the-art speculative router architecture [12]. Each arriving flit goes through 2 pipeline stages in the router: routing computation (RC), VC allocation (VA) and switch arbitration (SA) at the first cycle, and switch traversal (ST) at the second cycle. A lookahead routing scheme [13] is adopted, which generates routing information of the downstream router for an incoming flit prior to the buffer write, thus removing the RC stage from the critical path. Each router has multiple VCs per input port and uses flit-based wormhole switching [14]. Credit-based VC flow control [15] is adopted to provide the back-pressure from downstream to upstream routers, thus controlling flit transmission rate to prevent packet loss due to buffer overflow.

Due to the limited area and power resources and ultra-low latency requirements, on-chip routers rely on very simple buffer structure. VC-based NoC routers consist of a number of FIFO buffers per input port where each FIFO corresponds to a VC as illustrated in Figure 4(a). Each input port has v

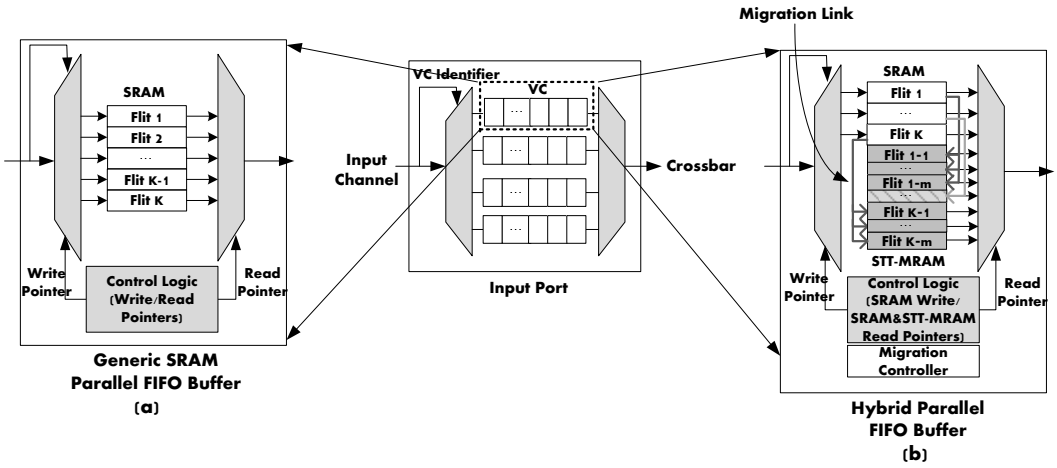


Figure 4: A Generic SRAM Input Buffer (a) and A Hybrid Input Buffer (b)

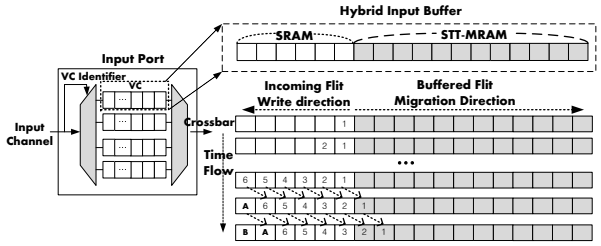


Figure 5: Simple Flit Migration Scheme in Hybrid Buffer Design

VCs, each of which has a k-flit FIFO buffer. Current on-chip routers have small buffers to minimize area overheads, thus v and k are much smaller than in macro networks. The necessity for ultra-low latency leads to a parallel FIFO buffer design as shown in Figure 4. Contrary to a serial FIFO implementation, the parallel structure eliminates unnecessary intermediate processes for a flit to traverse all buffer entries until it leaves the buffer [16]. This fine-grained control requires more complex logic, which manages read and write pointers to keep the FIFO order. The read and write pointers in the parallel FIFO registers control an input demultiplexer and an output multiplexer. The write pointer points to the tail of the queue, and the read pointer points to the head of the queue. For a read operation, the flit pointed by the head is selected and transmitted to a crossbar input port. Similarly, write operation leads the incoming flit to be written to the location pointed by the tail pointer. The pointers are promptly updated after each read or write operation. After a read operation, once the head is overlapped with the tail, the buffer becomes empty. After a write operation, likewise, if the tail moves to the same position pointed by the head, the buffer is full.

B. An On-Chip Router Architecture with Hybrid Buffer Design

In this section, we show an on-chip router architecture with hybrid buffer design that combines SRAM and STT-MRAM. The hybrid design aims to maximize advantages inherent in different memory technologies in a synergistic fashion for performance improvement while consuming power economically. The key idea is inspired by the nature of STT-MRAM that provides 4 times more buffer space than SRAM under the same area constraint due to its higher density characteristics [5], [17]. The increased buffer size contributes to making on-chip routers have spacious rooms for buffering, thus boosting the overall network throughput with no additional area overheads compared to a pure SRAM-based input buffer.

Figure 4(b) depicts the proposed hybrid input buffer of a VC. Compared to the pure SRAM buffer shown in Figure 4(a), the STT-MRAM is attached to each VC in parallel with the SRAM buffer. Each SRAM buffer entry is connected to m dedicated STT-MRAM buffer entries through separate migration links. The hybrid parallel FIFO buffer maintains read/write pointers. An incoming flit is first written to the SRAM buffer, thus the write pointer points to SRAM buffer entries only. But an outgoing flit may leave from either SRAM or STT-MRAM and the read pointer covers the entire buffer, both SRAM and STT-MRAM buffer entries.

A migration controller triggers the flit migration and determines if a certain flit is ready to be migrated to STT-MRAM. VC flow control is performed based on the availability of SRAM in downstream routers, meaning that the availability of STT-MRAM is not considered, because a write operation to STT-MRAM cannot finish in a single cycle.

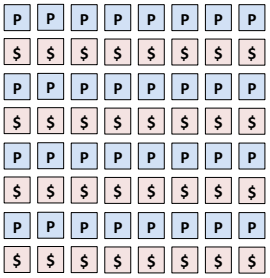


Figure 6: CMP Layout

1) Simple Flit Migration Scheme: The key design goal of the hybrid input buffer is to guarantee seamless read and write operations in every cycle to achieve higher throughput with an increased buffer size. To serve this purpose, we devise a flit migration scheme, which seamlessly migrates buffered flits from SRAM to STT-MRAM to secure more SRAM buffer space for incoming flits, while hiding the long write latency of STT-MRAM.

Figure 5 depicts an example of the migration scheme, where each VC consists of 6 SRAM and 12 STT-MRAM buffer entries. The STT-MRAM buffer write latency is assumed to be 6 cycles. When an incoming flit arrives, it is written to the SRAM buffer first, and the migration from SRAM to STT-MRAM begins immediately. Supposing that a new flit arrives every cycle, the SRAM buffer becomes full eventually in the 6th cycle. At the same time, the first flit is migrated to STT-MRAM successfully and one SRAM buffer entry becomes available. Then a subsequent incoming flit occupies the released SRAM buffer entry with no additional timing delay. Note that Figure 5 illustrates the concept in a logical way, and no physical shift occurs except the migration from SRAM to STT-MRAM. The placement of flits in STT-MRAM is logical and is not the physical placement described in Figure 4(b).

2) Power-Efficient Lazy Migration: In the simple migration scheme explained in the previous section, the migration begins immediately as soon as an incoming flit arrives at the SRAM buffer. The simple migration wastes lots of power in a low network load because most of the flits initially written to SRAM leave the buffer in the middle of migration to STT-MRAM.

Based on this observation, we propose a **lazy migration scheme**, which selectively triggers the migration of a flit based on the estimated network load per VC in the on-chip router. The network load is indirectly estimated by tracking the number of flits in the SRAM buffer. If the ratio of the number of flits in the SRAM buffer to the total SRAM buffer size exceeds a certain predefined threshold level, the flit migration is performed for every subsequent incoming flit as long as the the ratio exceeds the threshold. In this way, we can save total write power associated with the migration operation. To implement the lazy migration scheme, the migration controller is augmented to keep track of the flits in the SRAM buffer and triggers the migration adaptively. The write power is reduced by up to 79% in a low network load compared to the simple migration, which will be discussed in detail in Secton V.

Table I: CMP System Configuration

System Parameters	Details
Clock frequency	3GHz
# of processors	32
L1 I and D caches	direct-mapped 32KB (L1I) 4-way 32KB (L1D), 1 cycle
L2 cache	16-way 16MB, 20 cycles 32 banks, 512 KB/bank
Cache block size	64B
Coherence protocol	Directory-based MSI
Memory latency	300 cycles
Flit size	16B
Packet size	1 flit (Benchmark-control) 5 flits (Benchmark-data) 4 flits (Synthetic)

Table II: SRAM and STT-MRAM Parameters

Parameter	SRAM	STT-MRAM
Read Energy (pJ/flit)	5.25	3.826
Write Energy (pJ/flit)	5.25	40.0
Leakage Power (mW)	0.028	0.005

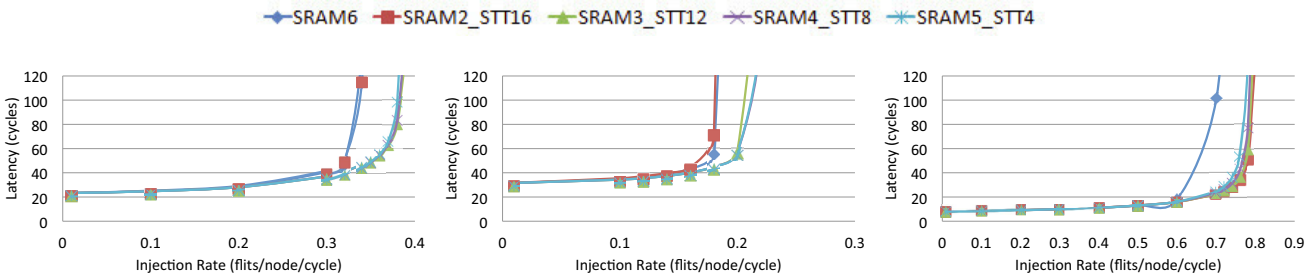
V. Performance Evaluation

In this section, we evaluate the proposed hybrid on-chip router to examine how much it improves the overall network performance while reducing the power consumption in NoC, using several benchmarks and synthetic workloads.

A. System Configuration

A cycle-accurate NoC simulator is used to conduct the detailed evaluation of the proposed scheme. It implements the pipelined router architecture with VCs, a VC arbiter, a switch arbiter and a crossbar. Under the 32nm process technology, all simulations are performed in an 8x8 network having 32 out-of-order processors and 32 L2 cache banks on a single chip as shown in Figure 6. The network is equipped with 2-stage speculative routers with lookahead routing [13]. The router has a set of v VCs per input port. Each VC contains a k-flit buffer with 16B flit size. In our evaluation, we assume that v is 4, and k may vary with different buffer configurations. A dimension order routing algorithm, XY, and O1TURN [18] are used with wormhole switching flow control.

A variety of synthetic workloads are used to measure the effectiveness of the hybrid on-chip router: uniform random **(UR)**, bit complement **(BC)** and nearest neighbor **(NN)**. To evaluate the proposed schemes under realistic environments, we also use SPLASH-2 [19] parallel benchmark traces. The traces are obtained using Simics [20], a full system



(a) UR (b) BC (c) NN

Figure 7: Performance Comparison with Synthetic Workloads

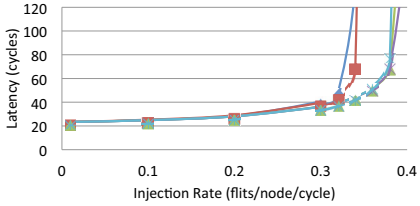
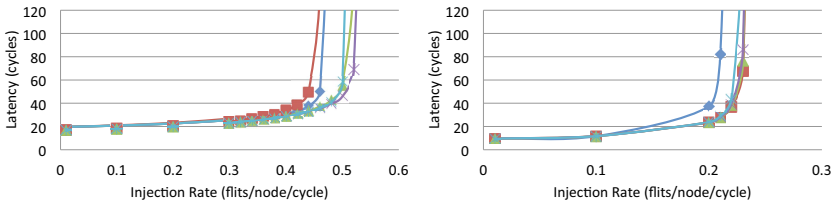
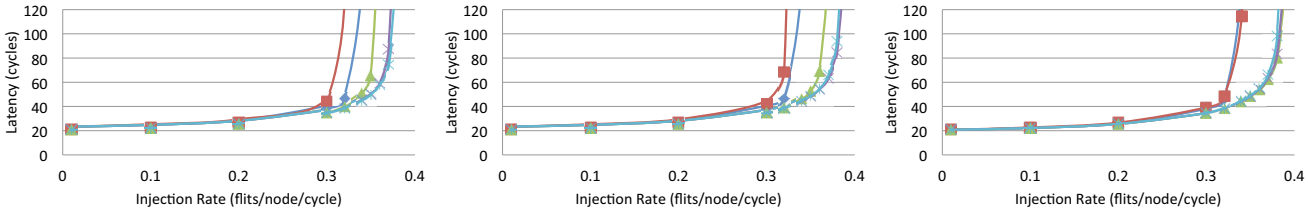


Figure 8: Performance Comparison with O1TURN Routing Algorithm



(a) 2D-Torus (b) Flattened Butterfly

Figure 9: Performance Comparison with Different Topologies



(a) 30 cycles (b) 10 cycles (c) 6 cycles

Figure 10: Performance Comparsion with Various STT-MRAM Write Latencies

simulation platform. Table I specifies the detailed CMP configuration we use to run benchmarks.

We use Orion 2.0 [21] to estimate router power consumption. In addition, parameters shown in Table II are cited from [11], [5], for both SRAM and STT-MRAM. The unit of parameter for the leakage power is *mW* per 1-flit buffer. Throughout this paper, the size of SRAM and STT-MRAM buffers are denoted by *SRAM#* and *STT#*, respectively. As stated in Section IV-B, STT-MRAM provides 4 times more buffer space compared to SRAM under the same area budget, thus *SRAM1* is equal to *STT4*. Unless otherwise stated, the write latency of STT-MRAM is 6 cycles based on the analysis in Section III.

B. Performance Analysis with Synthetic Workloads and Benchmarks

Figure 7 shows performance improvement for various hybrid input buffer configurations compared to the pure SRAM buffer, under UR, BC and NN traffic patterns. All results are measured under the same area budget, *SRAM6* per VC, for input buffers. In all cases, the hybrid design shows throughput improvement by 18% for UR, 28% for BC, and 17% for NN on average. These results indicate that although the STT-MRAM write latency is longer than that of SRAM, the performance loss is offset by the increased buffer size due to the high density of STT-MRAM, thus resulting in performance improvement.

We also evaluate the hybrid design using O1TURN [18] routing algorithm as well as various topologies: 2D-torus and flattened butterfly [22]. Figure 8 shows the performance with O1TURN in the 8x8 2D-mesh topology, where the overall throughput increases by 15% on average, while Figure 9 shows that the throughput is increased in 2D-torus and flattened butterfly by 13% and 15%, respectively.

To examine the impact of different write latencies of STT-MRAM on network performance, we conduct experiments under 2D-mesh and XY routing algorithm. Figure 10 shows the performance in terms of packet latency with 3 different write latencies of STT-MRAM: 30, 10, and 6 cycles. It clearly indicates that the overall network performance is

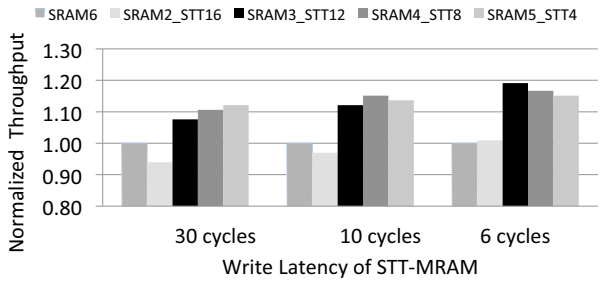


Figure 11: Throughput with Different STT-MRAM Write Latencies

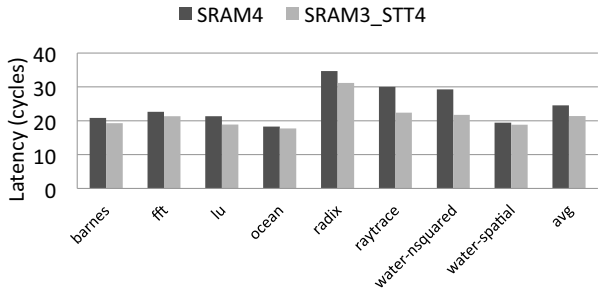
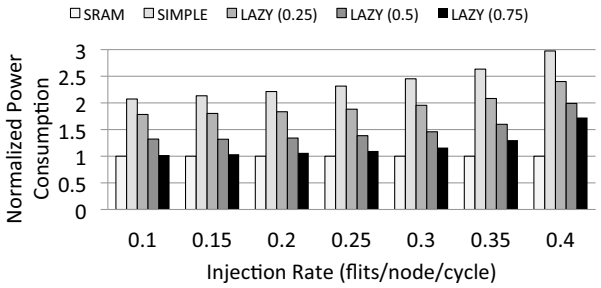
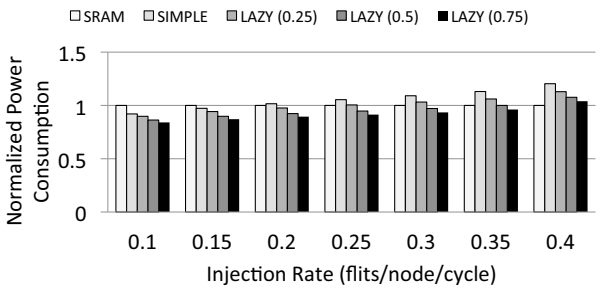


Figure 12: SPLASH-2 Benchmark Results



(a) Dynamic Power Consumption of Input Buffers



(b) Total Power Consumption of Routers

Figure 13: Comparison of Power Efficiency

affected by the duration of STT-MRAM write operation. Among the different hybrid configurations, *SRAM2_STT16* shows the worst performance. This is because the SRAM buffer space is too small to retain the incoming flits for sufficient period of time for migration, 6 cycles, which makes the simple flit migration scheme less efficient. Thus, the long write latency of STT-MRAM is not effectively hidden, resulting in the early saturation of the network. As shown in Figure 2, every flit stays in the buffer for at least 3 cycles. So the SRAM buffer size should be greater than or equal to 3 to run the migration scheme seamlessly.

If the write latency is long, 30 cycles, the performance is mostly determined by the SRAM size. This is because the long write latency lowers the possibility for flits to be migrated to the STT-MRAM buffer before network saturation. Therefore, *SRAM5_STT4* shows the best throughput improvement. On the contrary, if the write latency is sufficiently short, 6 cycles, the performance is greatly impacted by the total buffer size including both SRAM and STT-MRAM except the *SRAM2_STT16* case. Thus, *SRAM3_STT12* shows the highest throughput compared to other configurations.

To make a clear quantitative comparison of relative performance of the 3 different write latencies, we show network throughput normalized to the *SRAM6* in Figure 11, based on the results in Figure 10. Figure 11 confirms the aforementioned analysis. In case of a relatively long write latency, 30 cycles, the hybrid input buffer having the largest SRAM buffer outperforms the others by up to 11% compared to the pure *SRAM6* buffer. Likewise, in case of a low write latency, 6 cycles, except the *SRAM2_STT16* case, the one having the largest total buffer size, *SRAM3_STT12* beats the other configurations by up to 18% in terms of network throughput.

Figure 12 shows the average network latency with SPLASH-2 benchmark traces. We assume *SRAM4* per VC as an area budget, the same as a cache block size. In general, the hybrid input buffer outperforms the pure SRAM-based one, by approximately 14% on average. Specifically, *water-nsquared* shows the best improvement by 34.5% while *ocean* shows the least improvement by 3.2%. The amount of improvement varies depending on the traffic patterns. We observe that in the benchmarks showing higher improvement, hot spots exist in their communication, whereas in the benchmarks with slight performance improvement, communication is evenly spread across the whole network.

Finally, we make a sensitivity analysis of the number of buffer entries in NoC routers. Under two different area budgets, *SRAM4* and *SRAM6*, we compare the throughput of the pure SRAM-based buffer and the hybrid buffer that shows the best performance. As the budget decreases from *SRAM6* to *SRAM4*, the amount of improvement coming from the hybrid buffer increases by approximately 5.5%. This trend indicates that the hybrid buffer is more beneficial as the area budget in CMP environments becomes tighter.

C. Power Analysis

Since power is one of the main issues in the NoC router design, we evaluate power consumption of the hybrid input buffer and compare the effect of the two migration

schemes explained in Section IV. Figure 13(a) compares the dynamic buffer power consumption of 4 different migration schemes in *SRAM3_STT12*: simple and lazy with 3 different thresholds (0.25/0.5/0.75). All results are normalized to that of the pure SRAM-based buffer, *SRAM6*. The lazy migration scheme with the threshold 0.75 consumes significantly less amount of power, by 53% on average, compared to the simple migration scheme. In a low network load (0.1), the power consumption of the lazy migration scheme with the threshold 0.75 is almost equivalent to that of the baseline SRAM. In a high network load (0.4), however, the flit migration occurs more frequently in the hybrid buffer due to the highly congested network. Accordingly, the migration lowers the possibility of reducing the dynamic power, thus increasing the power consumption of the lazy migration by up to 1.7x more than the baseline SRAM.

Figure 13(b) compares the total router power consumption of the 4 migration schemes that includes both leakage and dynamic power consumption of all routers across the network. In a low network load (0.1), the total power consumption of routers with the hybrid buffer is less than that of routers with the pure SRAM buffer by 16%. This is due to much less leakage power consumption of STT-MRAM compared to SRAM as shown in Table II. As the network gets more congested, however, the hybrid buffer consumes more power compared to the baseline SRAM buffer. In a high network load (0.4), for instance, the lazy migration scheme with the threshold 0.75 consumes more power by up to 4% compared to the baseline SRAM buffer.

Note that as we increase the threshold value from 0.25 to 0.75 in the lazy migration scheme, the overall network throughput is slightly degraded but the amount of degradation is around 0.5% on average, which is negligible.

VI. Conclusions

In this paper, we have proposed a hybrid input buffer design using STT-MRAM with SRAM to achieve better network throughput with marginal power overheads in on-chip interconnection networks. The high density of STT-MRAM facilitates to accommodate larger buffer compared to the conventional SRAM under the same area budgets. Through the flit migration schemes, the long write latency of STT-MRAM is effectively hidden while minimizing the power overheads. Simulation results indicate performance improvement of around 21% and 14% on average under the synthetic workloads and benchmarks, respectively, compared to the conventional on-chip router with the SRAM input buffer.

For future work, we intend to devise an STT-MRAM-aware routing algorithm and provide an architectural support to reduce the overall power consumption and latency further.

References

[1] A. Jog, A. K. Mishra, C. Xu, Y. Xie, N. Vijaykrishnan, R. Iyer, and C. R. Das, "Cache Revive: Architecting Volatile STT-RAM Caches for Enhanced Performance in CMPs," The Pennsylvania State University CSE Dept., Tech. Rep. CSE-11-010, June 2011.

[2] 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 *Proceedings of HPCA*, 2011.

[3] A. K. Mishra, X. Dong, G. Sun, Y. Xie, N. Vijaykrishnan, and C. R. Das, "Architecting On-Chip Interconnects for Stacked 3D STT-RAM Caches in CMPs," in *Proceedings of ISCA*, 2011.

[4] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen, "A Novel Architecture of the 3D Stacked MRAM L2 Cache for CMPs," in *Proceedings of HPCA*, 2009.

[5] X. Guo, E. Ipek, and T. Soyata, "Resistive Computation: Avoiding the Power Wall with Low-Leakage, STT-MRAM Based Computing," in *Proceedings of ISCA*, 2010.

[6] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "A Durable and Energy Efficient Main Memory Using Phase Change Memory Technology," in *Proceedings of ISCA*, 2009.

[7] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable High Performance Main Memory System Using Phase-Change Memory Technology," in *Proceedings of ISCA*, 2009.

[8] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting Phase Change Memory as a Scalable DRAM Alternative," in *Proceedings of ISCA*, 2009.

[9] M. K. Qureshi, M. M. Franceschini, and L. A. Lastras-montao, "Improving Read Performance of Phase Change Memories via Write Cancellation and Write Pausing," in *Proceedings of HPCA*, 2010.

[10] P. Zhou, Y. Du, Y. Zhang, and J. Yang, "Fine-Grained QoS Scheduling for PCM-based Main Memory Systems," in *Proceedings of IPDPS*, 2010.

[11] ITRS, "International Technology Roadmap for Semiconductors: 2009 Executive Summary," http://www.itrs.net/Links/2009ITRS/Home2009.htm.

[12] L.-S. Peh and W. J. Dally, "A Delay Model and Speculative Architecture for Pipelined Routers," in *Proceedings of HPCA*, 2001.

[13] M. Galles, "Scalable Pipelined Interconnect for Distributed Endpoint Routing: The SGI SPIDER Chip," in *Proceedings of Hot Interconnect 4*, 2009.

[14] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. Comput.*, vol. 36, pp. 547–553, May 1987.

[15] W. J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. Parallel Distrib. Syst.*, vol. 3, pp. 194–205, March 1992.

[16] A. V. Yakovlev, A. M. Koelmans, and L. Lavagno, "High-Level Modeling and Design of Asynchronous Interface Logic," *IEEE Design and Test of Computers*, vol. 12, pp. 32–40, 1995.

[17] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, "Energy Reduction for STT-RAM Using Early Write Termination," in *Proceedings of ICCAD*, 2009.

[18] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi, "Near-Optimal Worst-Case Throughput Routing for Two-Dimensional Mesh Networks," in *Proceedings of ISCA*, 2005.

[19] S.C.Woo, M.Ohara, E.Torrie, J.P.Singh, and A.Gupta, "The SPLASH-2 Programs: Characterization and Methodological Considerations," in *Proceedings of ISCA*, 1995.

[20] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, B. Werner, and B. Werner, "Simics: A Full System Simulation Platform," *Computer*, vol. 35, no. 2, pp. 50–58, 2002.

[21] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration," in *Proceedings of DATE*, 2009.

[22] J. Kim, J. Balfour, and W. Dally, "Flattened Butterfly Topology for On-Chip Networks," in *Proceedings of MICRO*, 2007.