Advancing STTRAM Caches for Runtime Adaptable Energy-Efficient Microarchitectures

Kyle Kuan and Tosiron Adegbija
Department of Electrical & Computer Engineering, University of Arizona, Tucson, AZ, USA
Email: {ckkuan, tosiron}@email.arizona.edu

***Abstract*—Spin-Transfer Torque RAM (STTRAM) is a promising alternative to SRAM in on-chip caches, due to advantages including non-volatility, low leakage, high integration density, and CMOS compatibility. However, STTRAM's wide adoption in resource-constrained systems is hindered by high write energy and latency. An increasingly popular method to address this challenge involves trading off the non-volatility for reduced write speed and write energy by relaxing the STTRAM's data retention time. However, the retention time, which defines how long the cache can retain a cache block without power, is one of the most important cache requirements that may vary for different applications. In this work, we promote the idea of runtime adaptable STTRAM caches by introducing *logically adaptable retention time STTRAM (LARS) cache* for level one (L1) cache, and *highly adaptable last level STTRAM (HALLS)* cache for last level cache (LLC).**

I. Introduction and Motivation

Spin-Transfer Torque RAM (STTRAM) is an emerging and increasingly popular alternative to SRAM for implementing on-chip caches due to several advantages, including non-volatility, higher storage density than SRAM, low leakage power, and compatibility with CMOS technology [1], [2]. However, dynamic operations in STTRAM caches accrue significant overheads, compared to SRAM caches, due to long write latency and high dynamic write energy [2]. To address these challenges, we focus on reduced retention times for STTRAM caches, since the intrinsic retention time offered by STTRAM is unnecessary for caches [3]. Typically, cached data only need to remain in the cache for no more than one second [1]. Therefore, to reduce the write latency and energy, the retention time only needs to be long enough to hold cached data.

We propose adaptable retention time STTRAM caches to solve two major issues. First, the retention time can dynamically adapt to runtime retention time requirements, in order to mitigate dynamic energy and latency overheads [4], [5]. Second, by using the right-provisioned retention time, benefits of refreshing data blocks to maintain their integrity in reduced retention STTRAM caches, as in prior work [2], become marginal [4], [5]. Since the refresh buffer used for refresh operations also consumes considerable leakage power [6], eliminating refreshes and associated buffers does not only reduce dynamic energy from refresh activity, but also the leakage energy of refresh circuits.

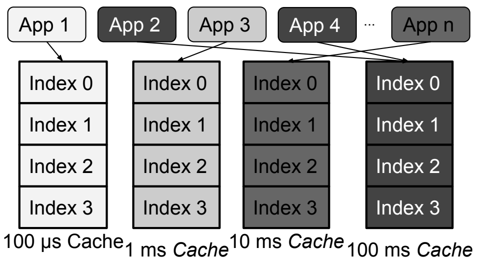

Fig. 1: Adapting retention times to different applications

II. Logically Adaptable Retention time STTRAM (LARS) L1 Caches

The basic idea of LARS is illustrated in Fig. 1. Given a set of potentially unknown applications running on a general-purpose system (e.g., smartphone), the cache is designed with multiple units such that it has a set of retention times that can satisfy different applications' runtime requirements. Based on the applications' cache block characteristics, and in effect, their retention time requirements, each application is executed on the retention time unit that best satisfies the retention time needs. This design can be achieved without incurring significant area overheads due to the density benefits of STTRAM.

To facilitate runtime adaptability, LARS also involves a hardware structure that samples the application's characteristics during its very first run and determines the best retention time. The retention time can then be associated with the application in a low-overhead table, and reused during future runs. The proposed LARS architecture comprises of four STTRAM units with four different retention times, which are empirically determined at design time to satisfy a range of application needs, depending on the target system and applications.

III. Highly Adaptable Last Level STTRAM (HALLS) Cache

We also studied the cache block characteristics of last level cache (LLC) accesses and found substantial variation in the retention needs of the data blocks. As such, we proposed HALLS to implement the idea of multiple retention times in the LLC. The HALLS cache architecture comprises of multiple banks (e.g., a 1MB LLC with 32 32KB banks), wherein different clusters of banks feature different retention times.

The retention times are determined through empirical analysis to select retention times that satisfy a variety of application requirements.

Furthermore, to enable additional adaptability and energy savings, HALLS allows the cache size to be configured by shutting down cache banks, e.g., the sample 1MB L2 cache can be configured into a 512KB cache by shutting down 16 banks or into a 128KB cache by shutting down 28 banks. The cache banks are organized in clusters, with each cluster designed with a different retention time. Each cluster features a $ClusterID$ to distinguish the different retention time clusters. HALLS also features a tuning algorithm that determines the best cache configuration and retention time based on an executing application's characteristics. Based on these configurations, HALLS opportunistically maps data accesses and references to the appropriate cache banks and clusters during runtime. Additional details of HALLS are provided in [5].

IV. Results and Comparison to Prior Work

We compared LARS and HALLS to SRAM and to an amalgam of prior work that used dynamic refresh schemes in reduced retention STTRAM caches (for brevity, we refer to prior work as DRS). We used GEM5 and NVSim to model the different techniques, and benchmarks from the SPEC CPU2006 benchmark suite to model both single and multi-programmed workloads.

A. LARS Results

To evaluate LARS, we modeled a 32KB L1 cache featuring four cache retention times per cache and experimented using twelve applications from the SPEC CPU2006 suite. On average across all the applications, LARS reduced the energy by 87.56% as compared to SRAM. This substantial energy reduction was achieved, in part, as a result of the significant reduction in leakage power of STTRAM compared to SRAM. The energy savings was achieved with only a marginal latency degradation of 0.7%. Compared to DRS, LARS achieved energy savings for all the benchmarks considered, with average savings of 25.31%. On average, LARS *increased* the latency by 2.3% compared to DRS.

B. HALLS results

To evaluate HALLS, we used a quad-core system featuring a two-level cache, running ten multi-programmed workloads in total. The LLC was modeled as a 1MB L2 cache with 32 physical banks organized as 8-bank clusters, with each cluster featuring a different retention time. On average across the workloads, HALLS reduced the average energy by 70.12% and 60.57%, as compared to SRAM and DRS, respectively. For majority of the workloads, HALLS's adaptability reduced the energy by more than 50%, illustrating the benefits of specializing the STTRAM cache configurations to the variety of execution requirements exhibited by applications in multicore systems. However, HALLS incurred some latency overheads of 5.16% and 1.47%, as compared to SRAM and DRS, respectively. These latency overheads occurred because HALLS increased the cache misses for several data blocks whose lifetimes exceeded the available retention time during the tuning process. The latency overheads were most pronounced for workloads that featured write-intensive applications [5]. Overall, this work illustrates the benefits of adaptable STTRAM caches for low-overhead energy savings, especially in resource-constrained systems.

V. Concluding Remarks and Future Directions

In this paper, we presented *logically adaptable retention time STTRAM (LARS) cache* and *highly adaptable last level STTRAM cache (HALLS)* to demonstrate the benefits of runtime adaptable STTRAM caches. LARS comprises of four STTRAM units with different retention times; only one unit is used at a time, depending on an application's needs. Experiments show that LARS can reduce the average energy by up to 25.31%, as compared to prior related work, without incurring significant latency or area overheads. We designed HALLS as a 1MB L2 cache with 32 physical banks organized in 8-bank clusters, and a different retention time for each cluster. On average, HALLS reduced the cache access energy by 60.57% compared to prior work, while introducing 1.47% of latency overhead.

The benefits of adaptable retention time demonstrated in LARS and HALLS motivates us to explore other optimization opportunities that leverage the diversity of cache block characteristics. Our analysis and findings reveal numerous research opportunities for future research. For example, in stride prefetchers, we plan to explore the interplay of cache block lifetimes and prefetch configurations in reduced retention STTRAMs. We may benefit from adjusting the prefetch degree during runtime based on resident blocks' retention time needs. We also plan to investigate similar ideas to LARS and HALLS in address translation hardware structures, such as the translation lookaside buffers (TLBs). Finally, we are interested in further exploring other retention-time-aware policies in the memory hierarchy, such as block replacement, cache coherence, and queuing policies to the unified cache.

References

[1] A. Jog, A. K. Mishra, C. Xu, Y. Xie, V. Narayanan, R. Iyer, and C. R. Das, "Cache revive: architecting volatile stt-ram caches for enhanced performance in cmps," in *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012, pp. 243–252.

[2] Z. Sun, X. Bi, H. H. Li, W.-F. Wong, Z.-L. Ong, X. Zhu, and W. Wu, "Multi retention level stt-ram cache designs with a dynamic refresh scheme," in *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2011, pp. 329–338.

[3] 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 *High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on*. IEEE, 2011, pp. 50–61.

[4] K. Kuan and T. Adegbija, "Energy-efficient runtime adaptable L1 STT-RAM cache design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2019.

[5] K. Kuan and T. Adegbija, "HALLS: An energy-efficient highly adaptable last level STT-RAM cache for multicore systems," *IEEE Transactions on Computers*, pp. 1–1, 2019.

[6] K. Kuan and T. Adegbija, "Mirrorcache: An energy-efficient relaxed retention L1 STTRAM cache," in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, ser. GLSVLSI '19. ACM, 2019, pp. 299–302. [Online]. Available: http://doi.acm.org/10.1145/3299874.3318022