HyGain: High Performance, Energy-Efficient Hybrid Gain Cell based Cache Hierarchy

SARABJEET SINGH[*†], University of Utah, USA
NEELAM SURANA[*‡], NVIDIA Graphics, India
KAILASH PRASAD, Department of Electrical Engineering, Indian Institute of Technology, Gandhinagar, India
PRANJALI JAIN[‡], University of California, Santa Barbara, USA
JOYCEE MEKIE, Department of Electrical Engineering, Indian Institute of Technology, Gandhinagar, India
MANU AWASTHI, Ashoka University, India

In this paper, we propose a "full-stack" solution to designing high capacity and low latency on-chip cache hierarchies by starting at the circuit level of the hardware design stack. We propose a novel half V_{DD} precharge 2T Gain Cell (GC) design for the cache hierarchy. The GC has several desirable characteristics, including ~50% higher storage density and ~50% lower dynamic energy as compared to the traditional 6T SRAM, even after accounting for peripheral circuit overheads. We also demonstrate data retention time of 350 us (~17.5× of eDRAM) at 28 nm technology with V_{DD} = 0.9V and temperature = $27°C$ which, combined with optimizations like staggered refresh, makes it an ideal candidate to architect all levels of on-chip caches. We show that compared to 6T SRAM, for a given area budget, GC based caches, on average, provide 30% and 36% increase in IPC for single- and multi-programmed workloads, respectively on contemporary workloads including SPEC CPU 2017. We also observe dynamic energy savings of 42% and 34% for single- and multi-programmed workloads, respectively. Finally, in a quest to utilize the best of all worlds, we combine GC with STT-RAM to create hybrid hierarchies. We show that a hybrid hierarchy with GC caches at L1 and L2, and an LLC split between GC and STT-RAM is able to provide a 46% benefit in energy-delay product (EDP) as compared to an all-SRAM design, and 13% as compared to an all-GC cache hierarchy, averaged across multi-programmed workloads.

CCS Concepts: • **Computer systems organization → Heterogeneous (hybrid) systems**; • **Hardware → Memory and dense storage**.

Additional Key Words and Phrases: Cache Memory, Emerging Memories, Gain Cell

[*]Both authors contributed equally to this research.
[†]This work was carried out while author was at Ashoka University.
[‡]This work was carried out while authors were at Indian Institute of Technology, Gandhinagar.

Authors' addresses: Sarabjeet Singh, sarab@cs.utah.edu, University of Utah, USA; Neelam Surana, neelam.surana@alumni.iitgn.ac.in, NVIDIA Graphics, India; Kailash Prasad, kailash.prasad@iitgn.ac.in, Department of Electrical Engineering, Indian Institute of Technology, Gandhinagar, India; Pranjali Jain, pranjali_jain@ucsb.edu, University of California, Santa Barbara, USA; Joycee Mekie, joycee@iitgn.ac.in, Department of Electrical Engineering, Indian Institute of Technology, Gandhinagar, India; Manu Awasthi, manu.awasthi@ashoka.edu.in, Ashoka University, India.

1544-3566/2022/11-ART $15.00
https://doi.org/10.1145/3572839

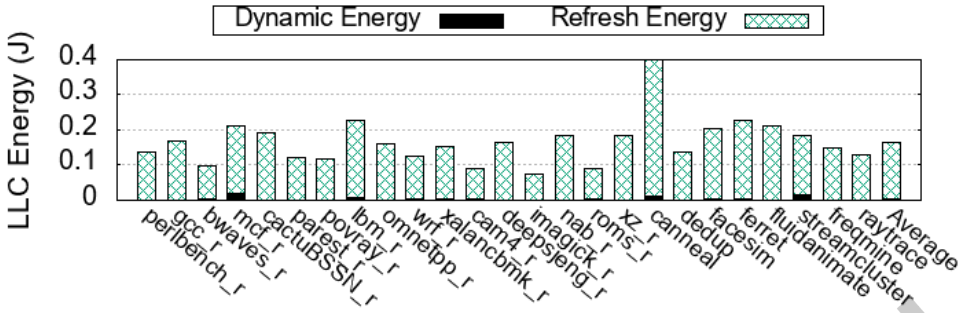


Fig. 1. eDRAM LLC energy breakdown.

1 INTRODUCTION

With the increasing number of cores on-chip [25], additional memory is needed to feed these cores. Emerging workloads have become significantly memory intensive and have large working set sizes [45, 62]. These factors have necessitated the need for high capacity, low latency, on-chip caches.

Several research efforts have been made to increase capacity and contain latency of on-chip caches, which has led to ever-increasing cache capacities and deeper cache hierarchies [45]. However, the memory technology, which makes up the bulk of on-chip caches, has remained unchanged. On-chip caches, at almost all levels of the memory hierarchy, have been devised using 6T SRAM. Even though SRAM suffers from low areal density, high leakage power and high dynamic energy requirements compared to other memory technologies [13, 57, 75], the latency superiority of SRAM, and its compatibility with logic fabrication technology has made it indispensable for creating low-latency caches.

Recently, a number of alternative memory technologies have started to emerge, and have been evaluated for use in caches. Contenders for SRAM replacement include non-volatile memory technologies like Spin-Transfer Torque RAM (STT-RAM) [64, 66, 80] and Phase Change Memory (PCM) [46], as well as volatile ones like embedded DRAM (eDRAM) [13].

In addition to providing data non-volatility, both STT-RAM and PCM provide 3-4× density benefits over 6T SRAM [46, 66], making them attractive candidates for high capacity caches. However, there are drawbacks inherent to both technologies, including higher write energy (up to ∼5× that of SRAM) and access latencies (∼1.5× read, 5× write latency for STT-RAM, PCM is worse) [46, 66]. In most cases, the drawbacks outweigh the benefits, rendering these technologies suitable for use only in last-level caches, where both capacities and access latencies are expected to be higher.

eDRAM has also been evaluated as a candidate for architecting caches [13, 76]. It has found adoption in multiple recent products, including IBM's Power series [74], Intel's Haswell [24] and Microsoft's Xbox 360 [9], again as a technology for LLCs. Since eDRAM is a DRAM variant, it provides higher density compared to SRAM and has favorable access latency profiles [13] as compared to NVMs. However, the traditional 1T1C eDRAM cells suffer from low Data Retention Times (DRTs) of 20 - 50 μs [13], requiring frequent refreshes in many eDRAM based design. These refreshes cause significant energy consumption as the data from an eDRAM row has to be read and written back [13, 76], making addressing refresh operations as the primary challenge in designing eDRAM caches. The energy consumption breakdown of dynamic and refresh energies for single programmed SPEC CPU2017 and PARSEC workloads in an eDRAM LLC (simulation parameters are listed in Section 6) is shown in Figure 1. As can be observed, refresh energy is many times higher than dynamic energy, which makes optimizing refresh operations an essential consideration for designing eDRAM based caches.

Table 1. Comparison of various memory technologies for on-die caches. Limitations of each technology in bold.

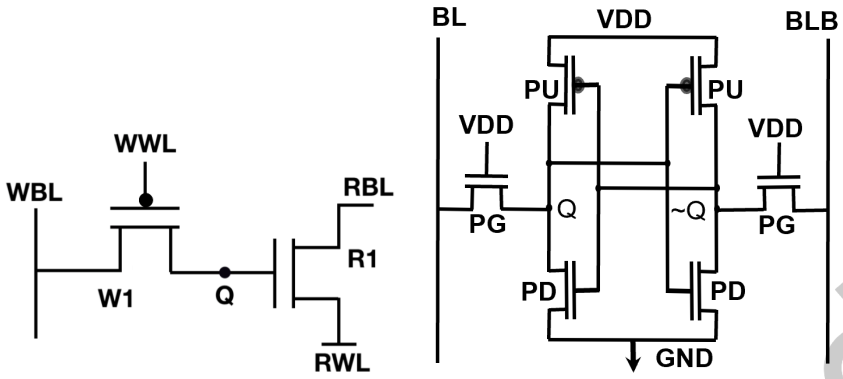
	SRAM	STT-RAM [30]	PCM[46]	1T1C eDRAM[3]	2T GC[65]	3T GC [22]	4T GC [20]	**Proposed**
Total Transistors	6T	1T+1MTJ	1T+1PCM	1T+1C	2T	3T	4T	2T
Technology	32	32	32	32	32	32	32	28
Area (um^2)	**0.64**	0.16	0.16	0.16	0.24	0.38	**0.54**	0.24
Data Storage	Latch	Magnetization	Phase	Capacitor	MOS	MOS	MOS	MOS
Read/Write Time	Short/Short	Short/**Long**	Short/**Long**	Short/Short	Short/Short	Short/Short	Short/**Long**	Short/Short
Read/Write Energy	X/Y	X/**5Y**	X/**5Y**	0.5X/0.5Y	0.5X/0.5Y	0.5X/0.5Y	0.5X/0.5Y	0.5X/0.5Y
Leakage/Yield	**High**/High	Low/High	Low/High	Low/**Low**	Low/High	Low/High	Low/High	Low/High
Retention Time	-	-	-	**20 us**	**20 us**	**20 us**	1.6 ms	350 us
Destructive Read/ Decoupled Bitline	No/**No**	No/Yes	No/Yes	**Yes**/**No**	No/Yes	No/Yes	No/Yes	No/No

Apart from energy overheads, an eDRAM row, and hence a portion of the cache, is unavailable for the duration of the refresh period, leading to performance overheads. To counter the shortcomings of traditional eDRAM, another eDRAM variant, Gain Cell (GC) has been proposed [22, 65], which has multiple advantages over 1T1C eDRAM. These include a cheaper, logic-compatible fabrication process and the absence of a dedicated capacitor per cell; GCs use the transistor's parasitic capacitance for data storage. Finally, GCs provide non-destructive reads [13] and decoupled read/write bitlines, resulting in lower access latency and energy consumption [65].

Despite these advantages, traditional 2T and 3T GCs have not found adoption widespread since they suffer from low DRTs, hence requiring frequent refreshes. While 4T GCs with higher DRTs have been proposed [20], they suffer from higher write energies and lower density, making them unattractive as 1T1C eDRAM replacements. In any case, the presence of refresh makes existing GCs useless at any level of cache, other than LLCs [13].

As a result, even though each one of these technologies has its pros and cons, they cannot be used as a *drop-in* replacement for SRAM caches, especially for levels closer to the CPU. However, in the presence of a suitable SRAM replacement, many of their pros can be combined to architect high capacity caches that have similar latency profiles as that of an SRAM based hierarchy. In this paper, we attempt such a design, starting from ground up. First, we propose a novel half V_{DD} precharge 28 nm bulk technology based *2T* Gain Cell, which provides high storage density, low access latencies, and a high DRT. This makes the cell amenable for use *at all levels of the cache hierarchy*. Even after accounting for peripheral circuitry overheads, compared to 6T SRAM, the GC array offers a 50% reduction in read/write energies, 50% reduction in the area while keeping access latency unchanged. It exhibits low leakage energy and has modest refresh requirements. A comparison of advantages of the proposed GC over competing memory technologies for caches is presented in Table 1. The main contributions of this work are summarized below:

- We propose a novel 2T Gain Cell and use in on-chip caches, with 99% lower leakage power than traditional 6T SRAM and ~17.5× higher data retention time compared to conventional GC and eDRAM, reducing the need for frequent refreshes. We combine this already large refresh window with optimizations like staggered refresh, to design practically refresh-free GC caches. As a result, GCs can be used at *all* levels of the cache hierarchy.
- We show that GC based sub-arrays exhibit 2× area advantage and similar latency characteristics as compared to SRAM, even after accounting for overheads of peripheral circuits. This helps architect higher capacity caches within the same area and latency budgets. As a result, for single-programmed workloads, iso-area caches architected using proposed GCs at all levels of the hierarchy exhibit a 42% reduction in dynamic energy and a 30% increase in IPC as compared to SRAM. Multi-programmed workloads exhibit similar behavior.
- Finally, to create high capacity, low latency caches, we evaluate several hybrid cache hierarchies by incorporating emerging technologies like STT-RAM with GCs to design caches that can provide up to 4×



(a) Proposed 2T GC (b) 6T SRAM Cell.

Fig. 2. Schematic diagram of the memory cell.

higher capacity, compared to iso-area SRAM caches. For multi-core workloads, hybrid caches architected with GCs can provide 43% performance and 44% energy benefits as compared to iso-area SRAM caches.

The remainder of this paper is organized as follows. We provide the circuit level implementation details for proposed GC in Section 2 and details of the architectural implementations of GC caches in Section 3. Section 4 presents the experimental evaluation setup, while Section 5 analyses the energy and performance implications of the implementations. Section 6 evaluates hybrid cache hierarchies. Finally, we discuss related work in Section 7 and conclude in Section 8.

2 HALF VDD PRECHARGE BASED GAIN CELL

Gain Cells have started gaining traction owing to their logic-compatible fabrication process, small area footprint, and low energy requirements [20, 22, 65, 69]. GCs have been fabricated and tested in FinFET [61], bulk [8], and FDSOI [20] processes. However, these proposals still suffer from low DRT leading to enormous refresh energy, as shown in Table 1. We implement an NMOS-PMOS based 2T GC on an in CMOS 28nm where we use 2x reduction in leakage current for PMOS write transistor and VDD/2 precharge, thus improving the DRT.

2.1 Using Half VDD Precharge to increase DRT

Data retention time (DRT) of a transistor is directly linked with the leakage current of the transistor in the OFF condition. The leakage current (I_{OFF}), in turn, exponentially depends on the threshold voltage (V_{TH}) of the transistor. In the recent past, with technology scaling, leakage has prohibitively increased in bulk-MOSFETs necessitating significant manufacturing efforts and additional fabrication steps to control the leakage current. We propose a 2T Gain Cell with Half VDD Precharge in this work. The W1 transistor is taken as PMOS to reduce the leakage compared to NMOS. However, it increases the write access time, and to take care of that ULVT (Ultra Low (V_{TH})) transistor is used. We have also used Half VDD Precharge to reduce leakage to a minimum during the hold state. The above two methodology increases the DRT of the transistor. To further increase the DRT a 1fF MOM capacitor is added at storage node without any area overhead. We have implemented the 2T GC using UMC 28nm CMOS technology and simulated it using Cadence Virtuoso.

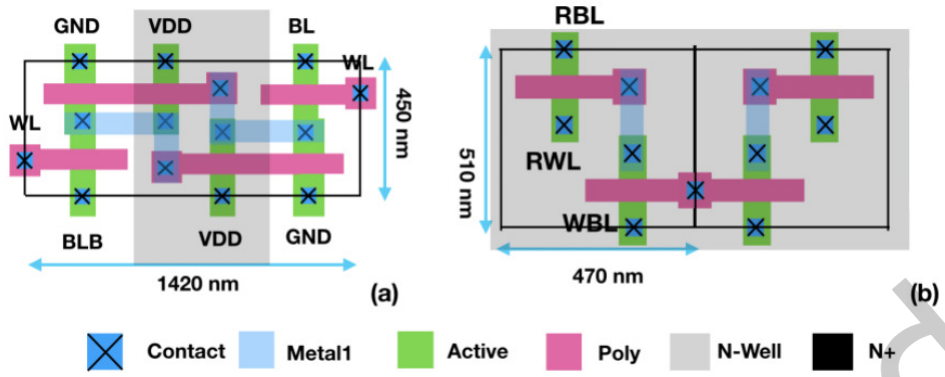


Fig. 3. Layout of (a) 6T SRAM Cell (b) Proposed 2T Gain Cell.

Table 2. Working of the Proposed Gain Cell

Operation	WBL	WWL	RBL	RWL
Read	$V_{DD}/2$	0	V_{DD}(floating)	0
Write	Data	V_{DD}	V_{DD}	V_{DD}
Hold	$V_{DD}/2$	0	V_{DD}	V_{DD}

2.2 Working of Gain Cell (GC)

Figure 2(a) shows the schematic of the proposed 2T GC. GC has decoupled read and write operations and has non-destructive reads, unlike 1T1C eDRAM. The input signals for these operations are illustrated in Table 2. Working of proposed 2T GC is similar to conventional 2T GC, except that $V_{DD}/2$ precharge WBL during hold condition.

2.2.1 Write Operation. Write Bitline (WBL) is shared across an entire column in an array and has a large capacitance. Write Wordline (WWL) runs along the row in the array. To write to the cell, data is first transferred to WBL, and then a row-signal (WWL) is used to transfer data to the Q node.

2.2.2 Read Operation. For a read operation, Read Bitline (RBL), which is a shared signal across the column, is first precharged to V_{DD}. Then, to read data, RBL is kept floating. Active low signal to Read Wordline (RWL) is used to read the data. If data stored in Q is 1, RBL discharges; otherwise it remains at V_{DD}, which is sensed by a sense amplifier. During the read operation, energy is consumed in the switching of RBL and RWL. Most importantly, read operation in GC is non-destructive, since RBL is decoupled from the Q node [65].

2.2.3 Hold Condition. Periods where the cell is neither read nor written to is known as the hold condition. This is important since the cell still needs to retain data during this period, unlike SRAM where data is retained due to cross-coupled inverters shown in Figure 2(b). The charge in Q node leaks from W1 over time, necessitating refresh operations for data restoration, before the DRT window closes. The WBL is is precharged to $V_{DD}/2$ in hold state.

2.3 GC comparison with 6T SRAM

2.3.1 Data Retention Time (DRT). The proposed GC uses $V_{DD}/2$ precharge WBL during hold operation and PMOS write transistor, leading to reduction in leakage current from the W1 transistor, improving DRT. To further increase the DRT a 1fF MOM capacitor is added at storage node without any area overhead. To quantify DRTs, we performed Monte Carlo simulations considering the standard 6-σ local and 1-σ global process variations. Figure 4 shows the data degradation of conventional and proposed 2T GC, for 10K M-C simulations. DRT is measured at a point where the data can be read without error. Typically, 100mV is a sufficient margin to read data properly, and we consider the worst-case DRT as the refresh interval, making these results more pessimistic than usual. For conventional 2T GC [65], DRT obtained is 19μs by considering 100% yield (Figure 4(a)), which is consistent with [20]. For the proposed GC, the data decay has significantly slowed down, as seen from Figure 4(b). We note that the worst-case DRT obtained for the proposed GC *improves to 350 us*, which is ∼**18.5**× higher than the traditional GC. Simulations are done at V_{DD} = 0.9V, Temperature = 27°C, and at TT (typical NMOS and typical PMOS) corner.

2.3.2 Leakage Power. Since the proposed 2T GC has a smaller number of leakage paths compared to SRAM (schematic in Figure 2), it inherently has lower leakage power. Additionally, we have used $V_{DD}/2$ precharge and PMOS write transistor to further reduce the leakage current significantly. We compare the leakage power of 6T SRAM, GC Traditional, and proposed GC in Table 3. We show that proposed GC has ∼**99%** reduction in leakage power as compared to 6T SRAM.

2.3.3 Area. Figure 3 compares the layout of the 6T SRAM cell and 2T GC at 28 nm technology. SRAM cell takes 0.64 μm^2, whereas the proposed GC takes 0.24 μm^2, which is 40% of the SRAM cell. The smaller size of the proposed GC allows for much higher density for the same area.

At the cell level, the proposed GC takes only 0.4× area compared to the 6T SRAM cell. Layout of the SRAM cell is drawn in a very efficient way and have area efficiency of 80-90% [32, 57, 75]. Considering this, at the cache level, GC can have ∼2× capacity as compared to SRAM cache. Even though GC has 2.5× benefits at the cell level, at the cache level, it reduces to 2.0× due to peripheral circuitry overhead.

In the rest of the paper, for the iso-area comparison, we have considered 2× capacity of GC compared to the SRAM.

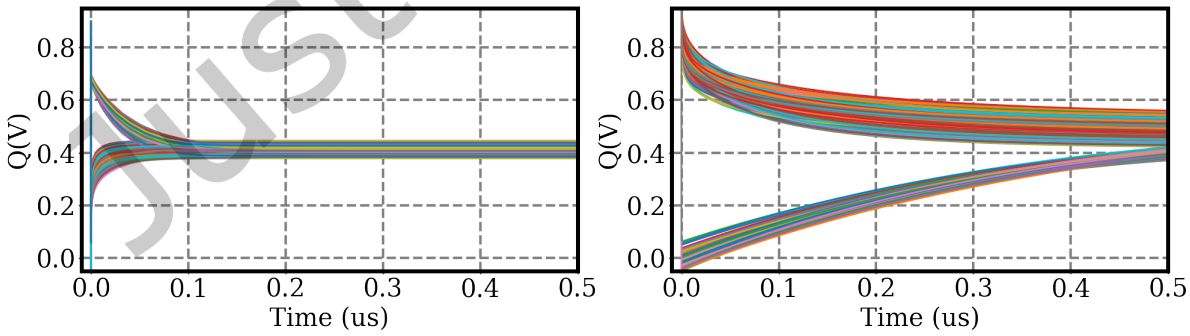


Fig. 4. 10K, Monte-Carlo waveforms of 1 and 0 decay (a) Traditional 2T Gain Cell (b) Proposed 2T Gain Cell.

Table 3. Leakage Power of 6T SRAM, Traditional GC & Proposed GC

Cell	SRAM	Traditional GC	Proposed GC
Leakage Power (pW)	629	0.304	0.0013

3 ARCHITECTING GAIN CELLS FOR CACHES

First, we describe the sub-array construction of proposed GCs, and the mechanism by which it maps to various cache configurations. Then we compare the architectural benefits of GCs over SRAM.

GCs are arranged as sub-arrays, in a typical row-column fashion. A cache can then be mapped to multiple sub-arrays, as dictated by its capacity. From an extensive design space exploration, we conclude that the sweet spot for minimum latency and peripheral circuitry overheads lie at a sub-array size of 256×512 bits, or 16 KB. Hence, GC caches can be architected such that each way, across all cache sets, maps to one sub-array. As a result, looking up a cacheline (64B) is the same as looking up a row of this sub-array, as shown in the Set0-Way0 to Row0 mappings in Figure 5. In cases where combined size of a way is >16 KB, we keep adding sub-arrays, until all the sets have been accounted for. For example, the right hand side of Figure 5 depicts a cache where one way is mapped to two 256×512 sub-arrays.

However, this prohibits mapping of any cache configuration where the combined capacity for one way is smaller than 16 KB, as illustrated in the left half of Figure 5. For these caches, we keep the design choice of mapping an entire way to one sub-array, while reducing the size of the sub-array. For example, in the case of a 64KB, 16-way cache, an entire way (4KB) is mapped to a 64×512 bit sub-array. This ensures that a 64 B cacheline lookup is not spread across multiple sub-arrays.

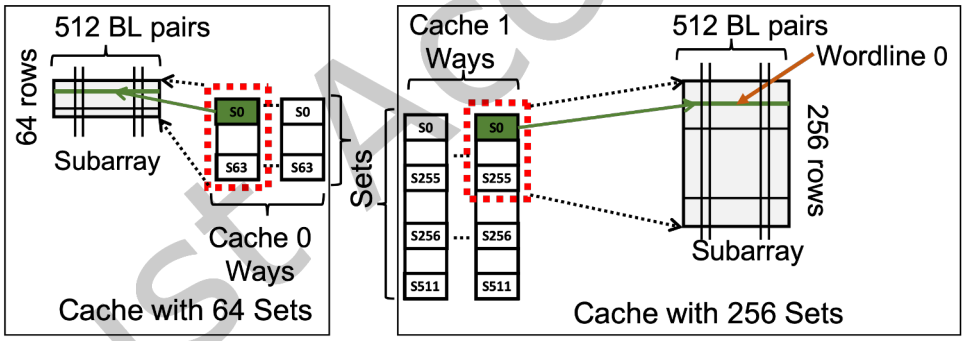


Fig. 5. Mapping of Caches to GC Subarrays for different caches

Figure 6 shows the block diagram of the proposed GC cache. Compared to SRAM cache, GC has additional refresh counters at each level of cache. As per concurrent refresh, the refresh counter will generate signal to refresh the same row in all subarrays at the same time. For staggering the refresh across different rows, the counter times out after every DRT/N time (N is number of rows in subarray).

Next, we compare and contrast the architecture level characteristics of on-chip caches devised using SRAM and GCs. We extract SRAM and GC energy and latency parameters using an enhanced CACTI [51] model and present these results in Table 4. These results were also validated using SPICE simulations using UMC 28nm CMOS technology. We observe that for every cache level, the dynamic read and write energies for a GC cache are reduced by *at least* 50%, as compared to an SRAM one. This is because the proposed GC requires just one bitline

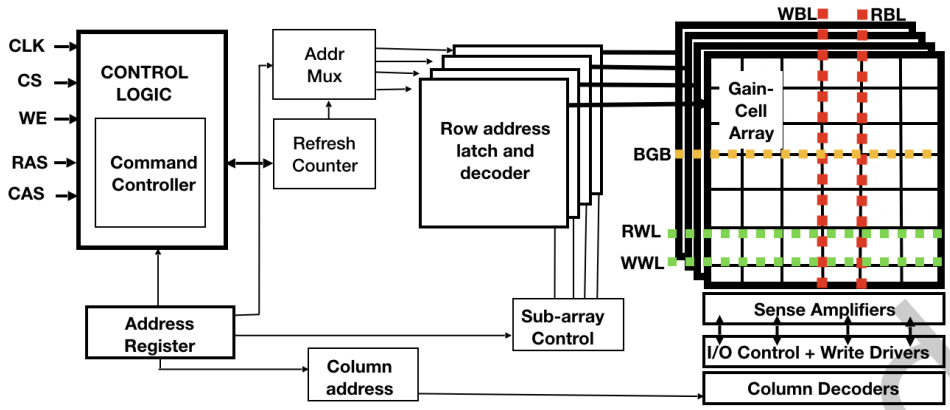


Fig. 6. Block diagram of the Proposed Gain Cell based Cache

Table 4. Cache Characteristics - SRAM v/s Proposed GC

Cache Level		32kB L1	256kB L2	8MB L3
Latency (ns) **(in cycles)**	SRAM	0.49**(2)**	1.48**(5)**	2.83**(10)**
	GC	0.43**(2)**	1.42**(5)**	2.42**(9)**
Read/Write Energy per bit (pJ)	SRAM	0.264	0.367	0.693
	GC	0.182	0.25	0.455
Leakage/bit (pW) (SRAM/GC)		629/0.0013		
Refresh Interval(us)/ Period per line(ns)		350/1.5		
Refresh Energy/bit (pJ)		1.87		

per read or write access, thereby reducing a significant fraction of the dynamic energy consumed in switching of bitlines and word lines [43, 68].

Additionally, owing to decoupled read and write like 8T SRAM [57] operations, GC has slightly lower access latencies as compared to SRAM, allowing for similar cycle time access as that of a SRAM cache, for a given processor frequency. Another trait of GC is high density, which enables us to fit a similar capacity cache in *half the area*. Alternatively, in a given area budget, we can implement a higher capacity cache by increasing associativity. As shown in Figure 7, iso-area access latencies of GC based caches at all levels of the hierarchy are similar to SRAM caches, with additional benefit of GC caches having twice the capacity. Doubling the capacity of an SRAM based cache increases the area by 2.0×. Not only that, it also increases access latency of caches by at least 30%. As a result, GC based caches allow for twice the capacity in the same latency **and** area budget, for every level of cache.

3.1 GC Based Cache Proposals

Using these observations, we propose the use of GCs at various levels of caches, from all on-chip caches architected using GCs (**ALL-GC**) to just last-level cache (**LLC-GC**) or L1 cache (**L1-GC**) being GC, and compare with the baseline case where all caches are implemented with SRAM (**ALL-SRAM**). Since GCs provide excellent density benefits over SRAM, we examine the energy and performance implications of GC over SRAM for both iso (cache) capacity and iso-area. For iso-capacity (**-CAP**), SRAM and GC caches are compared with the same cache size, which indicates lower on-chip area usage by GC caches. While in the case of iso-area (**-AREA**), the GC caches

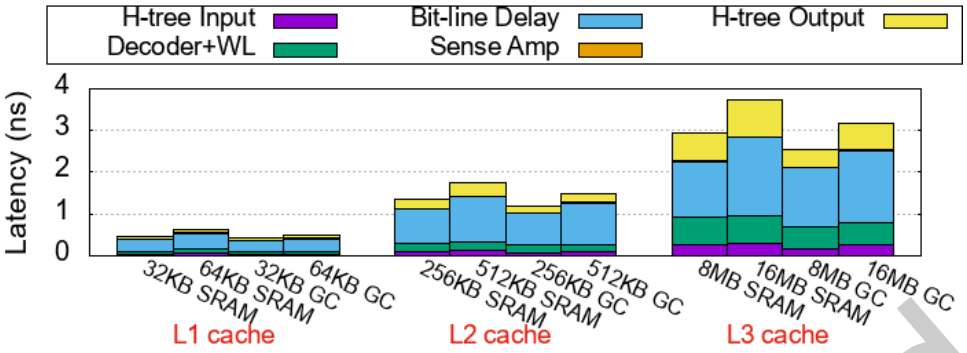


Fig. 7. Latency comparison of proposed GC and conventional SRAM caches

are doubled in capacity by increasing their associativity, while retaining latency characteristics. We maintain the tag array in SRAM; only the data arrays are replaced with GCs.

3.2 Handling Refresh in Gain Cell Caches

One of the biggest challenges in GC caches is the need to refresh. In addition to adding energy overheads, the cache is made unavailable for access during refresh operations, which adversely affects performance. We use a staggered, concurrent refresh mechanism [39] to reduce unavailability of GC cache.

As explained earlier in this section, one way of the cache is mapped to a row in the GC sub-array. Refreshing one row of the sub-array takes 3 ns. We refresh one row in a sub-array at a time and iterate over all the rows in a round-robin fashion in the course of the 350 μs refresh window, which is the DRT of an individual cell.

A refresh is done by reading the sub-array in the first 1.5 ns of the refresh window. Data is written back to the row in the second half of the window. As a result, the sub-array is available for write in the first half and a read in the second half. This is made possible due to the presence of separate read and write bitlines. This optimization increases the availability of the sub-array and hence, the associated way – it is now unavailable only for 1.5 ns every 1.36 μs. Since the tags are maintained in SRAM, the cache can still be accessed to check for hits/misses.

We carry out a detailed analysis of performance and energy implications of this refresh policy, in Section 5.3, and conclude that the performance overheads are minimal, since a tiny fraction (0.003%) of cache accesses happen concurrently with refresh, leading to a *worst-case* 1.7% reduction in performance, as compared to the SRAM baseline.

Table 5. System Configuration (ALL-SRAM)

Processor	8-core, 3.4GHz, x86_64 ISA, 19-stage OOO
Decode, Rename, Fetch Width	4-7 fused, 4, 6 instructions per cycle
Issue, Dispatch, Commit width	4, 6, 4 fused μ-ops per cycle
ROB/Branch misprediction	168 entries/8 cycles penalty
L1-I/L1-D cache	32 KB, 8-way & 2 cycles. 64B line
L2 cache	256 KB, 8-way & 5 cycles. 64B line
L3 cache	Shared 8 MB, 16-way & 10 cycles. 64B line
Main Memory	4096MB DDR3, 100 ns access, Read/Write Energy per 64B (nJ) = 41.6/54.4

Table 6. Workloads

Single-Programmed
SPEC CPU2017
perlbench_r, gcc_r, bwaves_r, mcf_r, cactuBSSN_r, parest_r, povray_r, lbm_r, omnetpp_r, wrf_r, xalancbmk_r, cam4_r, deepsjeng_r, imagick_r, nab_r, roms_r, xz_r
PARSEC
canneal, dedup, facesim, ferret, fluidanimate, freqmine, raytrace, streamcluster

Multi-Programmed	
MEM_HIGH	**MEM_MED**
cactuBSSN_r, mcf_r, streamcluster, gcc_r,canneal, omnetpp_r, facesim, perlbench_r	xalancbmk_r, ferret, cam4_r, bwaves_r, deepsjeng_r, wrf_r, povray_r, freqmine
MEM_LOW	**mix1**
raytrace, parest_r, fluidanimate, nab_r, dedup, imagick_r, roms_r, lbm_r	bwaves_r, xz_r, wrf_r, raytrace, roms_r, dedup, lbm_r, freqmine
mix2	**mix3**
perlbench_r, ferret, parest_r, canneal, omnetpp_r, cam4_r, nab_r, streamcluster	mcf_r, cactuBSSN_r, povray_r, xalancbmk_r, deepsjeng_r, imagick_r, facesim, fluidanimate
8x* - Running 8 copies of the same benchmark	

4 EVALUATION METHODOLOGY

We evaluate our proposed architectures by using an 8-core system with configuration listed in Table 5, simulated using Sniper [12]. This configuration is used as the baseline for evaluation (ALL-SRAM). For iso-area GC caches (-AREA), cache capacity is doubled by doubling associativity. We test our proposals against 25 benchmarks from the SPEC CPU2017 [62] and PARSEC [11] suites and study energy and performance implications in Sections 5.1 and 5.2, respectively. These workloads, listed in Table 6, are simulated for 2 billion instructions each, after a 500 million warmup period. Additionally, to include variations in the application behavior and test against multi-programmed workloads, we divide the workloads in six sets, each consisting of 8 benchmarks, which represents: memory-intensive applications (**MEM_HIGH**), applications with average memory access (**MEM_MED**), applications with sparse memory access (**MEM_LOW**), and three random mixes of 8-workloads (**mix1-3**). This classification is done based on memory accesses per kilo instructions to caches - higher accesses means more memory intensive. Also, we create six homogeneous multi-programmed workloads (**8x***) - 8 copies of the same benchmark, each per core.

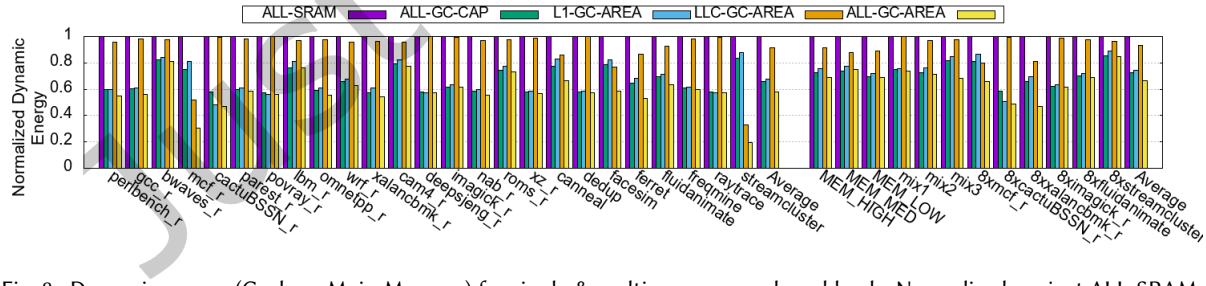


Fig. 8. Dynamic energy (Cache + Main Memory) for single & multi-programmed workloads. Normalized against ALL-SRAM.

5 EVALUATION OF GC CACHES

In this section, we quantify the benefits of various GC based architectures and compare them with SRAM based caches. The evaluation includes a study of memory subsystem energy, performance, and the impact of refresh operations.

5.1 Energy Analysis

While most emerging memory technologies exacerbate energy requirements of the memory subsystem [13, 41, 81, 82], GCs, on the contrary, provide significant energy savings over SRAM. In comparison with the baseline SRAM, at the array level, proposed GC design consumes ~46% less energy per read and 40-50% less energy per write, as shown in Table 4.

The ALL-SRAM case, where all levels of caches are assumed to be SRAM, is used as the baseline. We first study ALL-GC-CAP, where we replace all SRAM caches with the same capacity GC caches (iso-capacity). Next, we evaluate iso-area GC caches with double capacity. For this, we evaluate three configurations: (a) L1-GC-AREA, where we replace L1 cache in ALL-SRAM with a double capacity GC cache. (b) LLC-GC-AREA, where we replace last-level cache in ALL-SRAM with double capacity GC, and (c) ALL-GC-AREA, where we replace *all* levels of caches in ALL-SRAM with double capacity, iso-area GCs.

Figure 8 compares the dynamic energy consumed by the memory subsystem in proposed architectures, for both single and multi-programmed workloads. We calculate the dynamic energy consumption of caches and main memory by taking the product of the total number of accesses to each level with the energy consumption of per access using the per bit cache access energy (mentioned in Table 4). Access energies of main memory are obtained from [2] and are presented in Table 5. We observe that any cache hierarchy devised using GCs exhibits savings in dynamic energy. In L1-GC-AREA, where only the L1 is architected using proposed GCs, results in a 36% reduction in dynamic energy, averaged across all the single programmed benchmarks. The LLC-GC-AREA configuration, which replaces the SRAM LLC with a double capacity GC cache, also exhibits a 4% average reduction in dynamic energy. Similar results are obtained for multi-programmed workloads as well. For the memory-intensive mix (MEM-HIGH), the energy savings of L1-GC-AREA and LLC-GC-AREA stand at 25% and 9%, respectively.

Finally, using GC for *all* levels of the cache increases these gains tremendously. On average, across the single programmed workloads, ALL-GC-AREA achieves **42%** (**34**% in the case of multi-programmed) reduction in dynamic energy consumption as compared to the ALL-SRAM baseline. Even in cases where the area density benefits of proposed GC are not being utilized, i.e., in the sub-optimal configurations of ALL-GC-CAP, where all SRAM caches are replaced with *equal* capacity GC caches, we observe an average reduction in the dynamic energy of 34% and 28% for single and multi-programmed workloads respectively.

Compared to the baseline, applications like *streamcluster* and *mcf_r*, that have large number of memory accesses, achieve up to 80% reduction in dynamic energy for iso-area GC LLCs (LLC-GC-AREA). Increasing LLC capacity allows the working set of these applications to reside in the cache, reducing the number of off-chip accesses substantially (by ~99%). Reduced off-chip accesses reduce the high off-chip dynamic energy, resulting in massive energy savings. Additionally, in a large, many-core CPU running at a low voltage, leakage from on-chip caches contributes substantially to the chip's power draw [5]. Proposed GC, with a large savings of 99.3% in leakage energy, as depicted in Table 3, helps reduce these costs substantially.

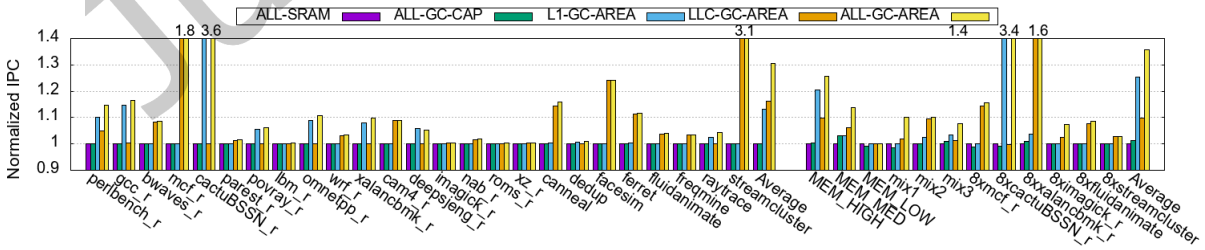


Fig. 9. System performance for single & multi-programmed (IPC is added across all cores) workloads. Normalized against ALL-SRAM.

5.2 System Performance

Figure 9 illustrates the system performance in terms of instructions per cycle (IPC) for various proposals, using single and multi-programmed workloads. We observe that the performance difference between iso-capacity SRAM caches (ALL-SRAM) and GC caches (ALL-GC-CAP) is negligible - 0.1% drop in IPC for GC caches on average, with respect to ALL-SRAM. All GC based iso-area caches (ALL-GC-AREA) exhibit *performance gains* as compared to the baseline. Average performance increase of **30%** and **36%** is observed across single and multi-programmed workloads, respectively. In the case of multi-programmed workloads, memory-intensive workloads tend to benefit most. We observe a 25% performance increase for MEM_HIGH workloads mix, as compared to the baseline. We verified our results on an aggressive processor configuration, with better prefetcher, replacement policy and DRAM access latency [27, 52, 63], and observed similar benefits (<4% IPC drop from above reported benefits).

Applications like *streamcluster*, *canneal*, and *mcf_r* have working set sizes that exceed the capacity of baseline SRAM caches. Hence, when the LLC size is doubled (LLC-GC-AREA), they see large performance improvements (>200%) as working sets can reside on caches. While, many applications like *cactuBSSN_r* and *gcc_r* have working sets that reside in the on-chip memory and leverage larger cache size to fit working sets in the L1 cache. As a result, they achieve significant performance improvements in L1-GC-AREA implementation (drop-in accesses to next level caches by >90%) but not in LLC-GC-AREA. On average, L1-GC-AREA and LLC-GC-AREA achieve IPC improvements of 13% and 16%, respectively.

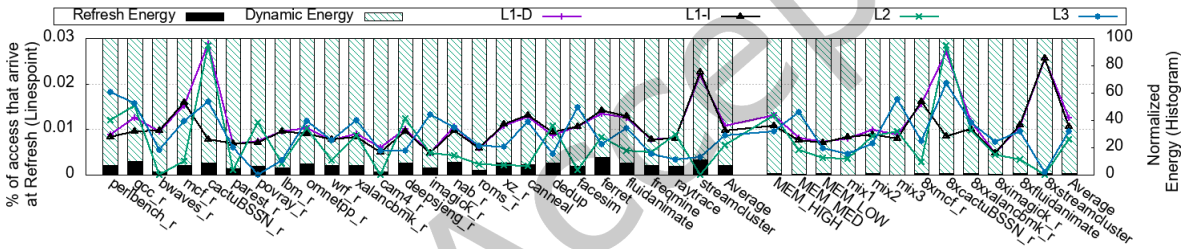


Fig. 10. Percentage of accesses to the cache which arrive when the cacheline is being refreshed *(y1-axis)*. Breakup of energy consumption of caches, normalized to total energy (Dynamic + Refresh) *(y2-axis)*. Configuration used is ALL-GC-AREA.

5.3 Impact of Refresh

Regular GC based caches are required to refresh cells at regular intervals. Unfortunately, this has adverse effects on both energy and performance. 1T1C eDRAM and traditional 2T, and 3T GCs can have huge refresh energy overheads, accounting for up to 97% of total LLC energy, as was observed in the experimental results presented in Figure 1.

However, for caches designed using the proposed GC, owing to high DRTs and staggered refresh mechanisms, we observe that the refresh energy consumption is minimal, assuming the most pessimistic scenarios. For experiments carried out with an all GC based cache subsystem (ALL-GC-AREA), which should exhibit the worst case refresh energy consumption profile, we observe that on average, across all single programmed benchmarks, refresh energy contributes <**8%** (13%, at max for *ferret*) of the total energy consumption of all caches. We illustrate these observations in Figure 10. For this worst case, we show that refresh energy has an insignificant contribution to the total energy, as depicted in the histogram on y2-axis.

Besides, the refresh operations do not affect performance adversely, as demonstrated from ALL-GC-CAP results from Section 5.2. This is evidenced by the fact that caches spend only 0.02% of the time on refresh, on

Table 7. LLC parameters for different technologies

	eDRAM 32MB	**STTRAM** 32MB	**Hybrid** (8MB **GC** +16MB **STTRAM**)
Read Latency (ns) **(cycles)**	5.15 **(18)**	26 **(89)**	2.83 **(10)**, 26 **(89)**
Write Latency (ns) **(cycles)**	5.15 **(18)**	60 **(204)**	2.83 **(10)**, 60 **(204)**
Read/**Write** Energy / bit (pJ)	5.2/**6.12**	5.35/**7.85**	3.81/**5.52**, 5.35/**7.85**
Refresh Interval/**Period**	0.02ms/**4ns**	-/-	350us/**1.5ns**, -/-
Refresh Energy/bit (pJ)	3.5	-	1.87/ -

average. Our experiments show that, on average, ~0.011% of accesses to caches were made during refresh interval throughout the entire simulation (Figure 10, y1-axis).

6 HYBRID CACHE HIERARCHY

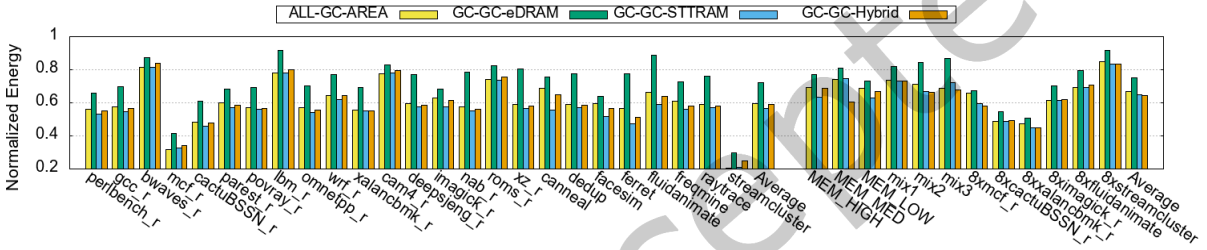


Fig. 11. Dynamic + Refresh Energy for single & multi-programmed Workloads. Normalized against ALL-SRAM.

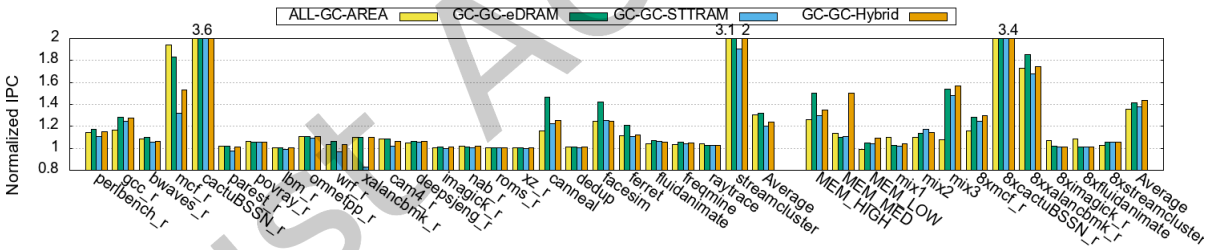


Fig. 12. System performance for single & multi-programmed (IPC is added across all cores) workloads. Normalized against ALL-SRAM.

A growing body of research has proposed either eDRAM or STT-RAM as a replacement for LLCs ([4–6, 13, 15, 31, 37, 44, 58, 64, 66, 80]). In this section, we build on prior work to evaluate hybrid cache hierarchies, in an effort to build efficient SRAM "free" on-chip caches.

First, we compare proposed GC based caches with other, state-of-the-art memory technologies. We consider the architectures, where L1 and L2 caches are kept as GC and use either eDRAM or STT-RAM in LLC, namely "*GC-GC-eDRAM*" and "*GC-GC-STTRAM*" respectively. STT-RAM parameters were taken from [44], while parameters for eDRAM are obtained from CACTI simulations, and are listed in Table 7. In our experiments, we consider state-of-the-art refresh-optimized eDRAM [4] which achieves ~20× reductions in the number of refreshes over regular eDRAM at 2-3% area overhead.

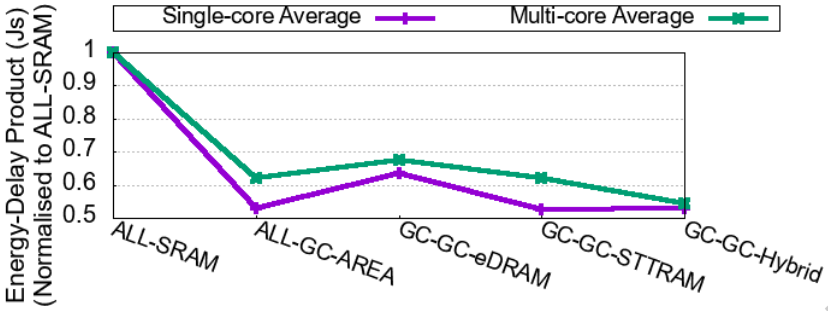


Fig. 13. Energy Delay Product (Js) of proposals, normalized to ALL-SRAM.

We compare these technologies with *ALL-GC-AREA* and present energy and performance comparisons in Figures 11 and 12, respectively. These experiments provide several interesting results. (a) eDRAM based LLC has 2× density benefits over GC, which helps it achieve 2.3% (4% for multi-core) improvement in IPC over ALL-GC-AREA, which makes a case for an eDRAM-based LLC. However, eDRAM has large refresh overheads. Even the refresh-optimized eDRAM [4] (GC-GC-eDRAM) results in 21% (12% for multi-core) more total energy consumption than ALL-GC-AREA. Traditional eDRAM results in much worse energy overheads – 6.8× higher energy consumption as compared to ALL-GC-AREA.

STT-RAM, with the same 2× density benefits over GC, suffers from much longer access latencies, which have been enumerated in Table 1. As a result, configurations with STT-RAM LLC experienced a performance drop of 7% with respect to ALL-GC-AREA, even though the number of off-chip requests actually dropped significantly by 13%. However, in realistic cases (multi-core runs) that take advantage of larger LLC, GC-GC-STTRAM performs slightly better than ALL-GC-AREA. Consequently, with smaller off-chip accesses, energy consumption reduces by 5%, compared to ALL-GC-AREA, concluding that STT-RAM LLC would be a better design.

In an effort to get the best of all worlds: utilize higher density of STT-RAM, and low latency of GC, we propose *hybrid LLC* designs of GC and STT-RAM. We selected STT-RAM to avoid the high energy overheads imposed by eDRAM. We carried out a design space exploration for the optimal size partitioning between GC and STT-RAM, while maintaining the same area budget as SRAM, and found that equal-area (8 MB GC, 16 MB STT-RAM) distribution results in the sweet spot of high performance and low energy. The parameters of this organization are listed in Table 7. The ways of each set of the hybrid cache are split between GC and STT-RAM cachelines, in the ratio of capacity. On a cache lookup, tags of both GC and STT-RAM ways are read and compared. If there is a hit in one of the GC ways, a read or write is carried out. On a miss in GC ways, but a hit an STT-RAM way, the cacheline is moved to the LRU position in the GC ways. Since GC ways tend to have "hot" data, in order to exploit temporal locality, the evicted cacheline from GC way is moved to the LRU position of the STT-RAM ways. In case of a miss in both GC and STT-RAM ways, the cacheline is fetched from the next level and placed in the LRU position of STT-RAM ways. The proposed hybrid cache architecture can be further optimized via novel replacement policies and prefetchers, which we leave for future work [7, 38]. With L1 and L2 cache as GC, and a hybrid LLC, we evaluate the architecture, results of which are compiled in orange bars of Figures 11 & 12, under "*GC-GC-Hybrid*".

As expected, the energy consumption of hybrid design is very close to that of ALL-GC-AREA: within 2%, on average, across both single and multi-core benchmarks. More importantly, the increased LLC accesses, due to larger cache, are performed with lower latencies of GC. As a result, we observe 5% improvement in IPC over ALL-GC-AREA baseline, averaged across multi-core simulations. Compared to the traditional SRAM hierarchy, our proposal shows **24%** better performance with **42%** less energy consumption (**43**% better with **36**% less energy,

in case of multi-core). In conclusion, while GC works best for L1 and L2 caches, real environments require large-capacity LLC with low latency, which can be addressed with our proposed GC-STTRAM hybrid LLC design. In the hybrid design, we move the recently accessed cachelines to the GC part of hybrid LLC, thus serving them with lower latency, if locality exists.

In summary, we present Energy-Delay Product (EDP) results of various architectures in Figure 13. As can be observed, for both single and multi-programmed workloads, *any* GC based hierarchy does better than the baseline SRAM one. The most favorable design point is obtained by utilizing optimized GC caches at all levels, and a hydrid STT-RAM - GC LLC. This architecture achieves an EDP which is **0.52**× of the baseline SRAM one.

7 RELATED WORK

Emerging Memory Technologies for Caches: Due to scalability and energy issues of traditional SRAM, several studies have been carried out to evaluate emerging memory technologies for caches. STT-RAM, owing to its low leakage energy and density benefits, has been viewed as a promising candidate. However, it suffers from inherent weaknesses - high write latency and write energy. Emerging memory technologies, like STT-RAM, PCRAM, ReRAM, etc, typically offer higher bit density over SRAM. However, they have higher write latency than SRAM and asymmetric read-write energies [13, 49, 53, 64, 67, 71, 79] which hinders their adoption. eDRAM is one among these promising emerging memory technologies identified for on-chip caches. Our work is based on Gain Cell which is a variant of eDRAM. There have been many proposals to alleviate these shortcomings [6, 15, 26, 30, 37, 58, 60, 64, 66, 72, 83].

Another potential replacement for SRAM are eDRAMs, which offer high density, low leakage, similar access latencies, and low dynamic energies. However, eDRAM requires refresh operations to preserve data integrity [29]. As cache size increases, each refresh requires more energy, and more lines need to be refreshed; thus, refresh can potentially become the main source of eDRAM power dissipation, as demonstrated in Figure 1. Many studies have been carried out to amortize this effect [5, 13, 23, 46, 70, 76, 78]. For instance, Refrint [5] proposes an algorithm that only refreshes the data that will most likely be used in the near future, thus obviating unnecessary refreshes. [4] showed that the DRTs of cells in large eDRAM modules exhibit spatial correlations, and exploit this behavior to reduce refresh energy. RANA [70] exploits the intrinsic error resiliency of CNNs to relax eDRAM refreshes while training. In contrast, the proposed GC already has insignificant refresh overheads.

Gain Cell: Gain Cell embedded DRAM (GC-eDRAM) has gathered tremendous interest as an alternative over to the 6T SRAM due to its high density, decoupled read-write operation, low leakage per bitcell [8, 16–18, 20–22, 34, 40, 65]. Moreover, Gain Cell's logic compatible fabrication process make them attractive for on-chip cache designs. However, in order to retain data, they need periodic refreshes as they store charge on the parasitic capacitance of the transistors.Many circuit and array level architectures have been proposed for GCs [8, 16–18, 20–22, 34, 40, 65]. Transistor's parasitic capacitor is used to store the data.

Architectural optimizations for SRAM caches: Many schemes have been looked into reducing SRAM's leakage energy [19, 33]. For instance, Drowsy caches [19] puts the cold cachelines into a state-preserving, low-power mode to cut leakage power. To catch up with the increasing demands for large capacity caches, numerous cache compression techniques have been looked into [10, 50, 54]. Panda and Seznec [54] propose DISH - simple dictionary based cache compression scheme. Arelakis et al. [10] present HyComp, a hybrid compression method tailored for caches, that selects the best compression method based on data-type prediction, to reduce decompression latency. There has also been a significant amount of work on achieving higher performance in caches. Studies have been carried out for efficient cache replacement and cache management policies [14, 27, 48, 55, 56], dead block predictions [36, 42, 47], or exploiting the differences between reads and writes in caches [35, 73].

3D integration tools promise high performance by enabling high bandwidth interconnects on-chip. Many works have explored integrating 3D caches over the logic layer, both industry [1, 77] and academia [28, 59, 66, 78]. Our proposed Gain Cell based caches are an implementation of eDRAM, and can benefit from similar benefits demonstrated by [1, 28, 77] – [28] observed 57% improvement by using a 512MB 3D die-stacked cache with respect to a baseline chip without die-stacking.

8 CONCLUSION

In this work, we propose a novel half VDD precharge 2T Gain Cell (GC) as a promising candidate for use in all levels of on-chip caches. The proposed GC has better energy efficiency, a much smaller area footprint, and better scalability as compared to 6T SRAM. We demonstrate high data retention time (~17.5× compared to exiting 2T Gain Cell and eDRAM), which minimizes contribution of refresh to the overall energy consumption of the cache hierarchy.

We evaluate various architectural implementations of GC, for all levels of on-chip caches and demonstrate that GC based caches substantially reduce the dynamic energy consumption of memory subsystem as compared to traditional SRAM caches. Finally, we show that proposed GC, in conjunction with emerging memory technologies like STT-RAM can be used to architect SRAM-free cache hierarchies with a much superior energy-delay product as compared to SRAM caches.

ACKNOWLEDGMENTS

This work is supported through grants received from Prime Minister Research Fellowship, Science and Engineering Research Board (SERB), Government of India, under CRG/2018/005013, MTR/2019/001605, SUPRA SPR/2020/000450, Ministry of Electronics and IT (MEITY) under YFRF Visvesvaraya PhD fellowship and Semiconductor Research Corporation (SRC) through contracts 2020-IR-2980 and 2020-IR-3005. This work is partially supported through Ashoka University startup and Huawei Technologies India grants to Manu Awasthi.

REFERENCES

[1] [n. d.]. 3D V-Cache. https://www.amd.com/en/campaigns/3d-v-cache.
[2] [n. d.]. Micron Technical Note TN-41-01. http://www.micron.com/products/support/power-calc/.
[3] James W Adkisson, Ramachandra Divakaruni, Jeffrey P Gambino, and Jack A Mandelman. 2002. Embedded DRAM on silicon-on-insulator substrate. US Patent 6,350,653.
[4] Aditya Agrawal, Amin Ansari, and Josep Torrellas. 2014. Mosaic: Exploiting the spatial locality of process variation to reduce refresh energy in on-chip eDRAM modules. In *Proceedings of the 20th International Symposium on High Performance Computer Architecture (HPCA)*.
[5] Aditya Agrawal, Prabhat Jain, Amin Ansari, and Josep Torrellas. 2013. Refrint: Intelligent Refresh to Minimize Power in On-chip Multiprocessor Cache Hierarchies. In *Proceedings of 19th International Symposium on High Performance Computer Architecture (HPCA)*.
[6] Junwhan Ahn, Sungjoo Yoo, and Kiyoung Choi. 2014. DASCA: Dead write prediction assisted STT-RAM cache architecture. In *Proceedings of 20th International Symposium on High Performance Computer Architecture (HPCA)*.
[7] Junwhan Ahn, Sungjoo Yoo, and Kiyoung Choi. 2015. Prediction hybrid cache: An energy-efficient STT-RAM cache architecture. *IEEE Trans. Comput.* (2015).
[8] Yoshiyuki Ando. 2003. Capacitorless DRAM gain cell. US Patent 6,560,142.
[9] Jeff Andrews and Nick Baker. 2006. Xbox 360 system architecture. *IEEE micro* 26, 2 (2006), 25–37.
[10] Angelos Arelakis, Fredrik Dahlgren, and Per Stenstrom. 2015. Hycomp: A Hybrid Cache Compression Method for Selection of Data-type-specific Compression Methods. In *Proceedings of the 48th International Symposium on Microarchitecture (MICRO)*.
[11] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. 2008. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *Proceedings of 17th International conference on Parallel Architectures and Compilation Techniques (PACT)*.
[12] Trevor E Carlson, Wim Heirman, and Lieven Eeckhout. 2011. Sniper: Exploring the Level of Abstraction for Scalable and Accurate Parallel Multi-core Simulation. In *Proceedings of International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*.

[13] Mu-Tien Chang, Paul Rosenfeld, Shih-Lien Lu, and Bruce Jacob. 2013. Technology comparison for large last-level caches (L3Cs): Low-leakage SRAM, low write-energy STT-RAM, and refresh-optimized eDRAM. In *Proceedings of 19th International Symposium on High Performance Computer Architecture (HPCA)*.

[14] M. Chaudhuri. 2009. Pseudo-LIFO: The Foundation of a New Family of Replacement Policies for Last-level Caches. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*.

[15] Xunchao Chen, Navid Khoshavi, Jian Zhou, Dan Huang, Ronald F DeMara, Jun Wang, Wujie Wen, and Yiran Chen. 2016. AOS: Adaptive Overwrite Scheme for Energy-Efficient MLC STT-RAM cache. In *Proceedings of the 53rd ACM/EDAC/IEEE Design Automation Conference (DAC)*.

[16] Woong Choi, Gyuseong Kang, and Jongsun Park. 2015. A refresh-less eDRAM macro with embedded voltage reference and selective read for an area and power efficient Viterbi decoder. *IEEE Journal of Solid-State Circuits* 50, 10 (2015), 2451–2462.

[17] Ki Chul Chun, Pulkit Jain, Tae-Ho Kim, and Chris H Kim. 2012. A 667 MHz Logic-Compatible Embedded DRAM Featuring an Asymmetric 2T Gain Cell for High Speed On-die Caches. *IEEE Journal of Solid-State Circuits* 47, 2 (2012), 547–559.

[18] Ki Chul Chun, Pulkit Jain, Jung Hwa Lee, and Chris H Kim. 2011. A 3T Gain Cell Embedded DRAM Utilizing Preferential Boosting for High Density and Low Power on-die Caches. *IEEE Journal of Solid-State Circuits* 46, 6 (2011), 1495–1505.

[19] Krisztián Flautner, Nam Sung Kim, Steve Martin, David Blaauw, and Trevor Mudge. 2002. Drowsy caches: simple techniques for reducing leakage power. In *Proceedings 29th Annual International Symposium on Computer Architecture (ISCA)*.

[20] R. Giterman, A. Fish, A. Burg, and A. Teman. 2018. A 4-Transistor nMOS-Only Logic-Compatible Gain-Cell Embedded DRAM With Over 1.6-ms Retention Time at 700 mV in 28-nm FD-SOI. *IEEE Transactions on Circuits and Systems I: Regular Papers* 65, 4 (2018), 1245–1256.

[21] R. Giterman, A. Fish, N. Geuli, E. Mentovich, A. Burg, and A. Teman. 2018. An 800-MHz Mixed-V_T 4T IFGC Embedded DRAM in 28-nm CMOS Bulk Process for Approximate Storage Applications. *IEEE Journal of Solid-State Circuits* 53, 7 (2018), 2136–2148.

[22] R. Giterman, A. Teman, P. Meinerzhagen, L. Atias, A. Burg, and A. Fish. 2016. Single-Supply 3T Gain-Cell for Low-Voltage Low-Power Applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24, 1 (2016), 358–362.

[23] Nagendra Gulur, R Govindarajan, and Mahesh Mehendale. 2016. MicroRefresh: Minimizing refresh overhead in DRAM caches. In *Proceedings of the Second International Symposium on Memory Systems*.

[24] Per Hammarlund, Alberto J Martinez, Atiq A Bajwa, David L Hill, Erik Hallnor, Hong Jiang, Martin Dixon, Michael Derr, Mikal Hunsaker, Rajesh Kumar, et al. 2014. Haswell: The fourth-generation Intel Core Processor. *IEEE Micro* 34, 2 (2014), 6–20.

[25] Lisa Hsu, Ravi Iyer, Srihari Makineni, Steve Reinhardt, and Donald Newell. 2005. Exploring the cache design space for large scale CMPs. *ACM SIGARCH Computer Architecture News* 33, 4 (2005), 24–33.

[26] Mohsen Imani, Abbas Rahimi, Yeseong Kim, and Tajana Rosing. 2016. A low-power hybrid magnetic cache architecture exploiting narrow-width values. In *Proceedings of the 5th Non-Volatile Memory Systems and Applications Symposium (NVMSA)*.

[27] Aamer Jaleel, Kevin B. Theobald, Simon C. Steely, Jr., and Joel Emer. 2010. High Performance Cache Replacement Using Re-reference Interval Prediction (RRIP). In *Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA)*.

[28] Djordje Jevdjic, Stavros Volos, and Babak Falsafi. 2013. Die-stacked dram caches for servers: Hit ratio, latency, or bandwidth? have it all with footprint cache. *ACM SIGARCH Computer Architecture News* (2013).

[29] Naifeng Jing, Yao Shen, Yao Lu, Shrikanth Ganapathy, Zhigang Mao, Minyi Guo, Ramon Canal, and Xiaoyao Liang. 2013. An Energy-efficient and Scalable eDRAM-based Register File Architecture for GPGPU. In *Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA)*.

[30] Adwait Jog, Asit K Mishra, Cong Xu, Yuan Xie, Vijaykrishnan Narayanan, Ravishankar Iyer, and Chita R Das. 2012. Cache Revive: Architecting Volatile STT-RAM caches for enhanced performance in CMPs. In *Proceedings of the DAC Design Automation Conference (DAC)*.

[31] SH Kang and C Park. 2017. MRAM: Enabling a sustainable device for pervasive system architectures and applications. In *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*.

[32] E. Karl, Z. Guo, J. W. Conary, J. L. Miller, Y. Ng, S. Nalam, D. Kim, J. Keane, U. Bhattacharya, and K. Zhang. 2015. 17.1 A 0.6V 1.5GHz 84Mb SRAM design in 14nm FinFET CMOS technology. In *Proceedings of the IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*. 1–3.

[33] Stefanos Kaxiras, Zhigang Hu, and Margaret Martonosi. 2001. Cache decay: exploiting generational behavior to reduce cache leakage power. In *Proceedings 28th International Symposium on Computer Architecture (ISCA)*.

[34] Amit Kazimirsky, Adam Teman, Noa Edri, and Alexander Fish. 2017. A 0.65-v, 500-mhz integrated dynamic and static ram for error tolerant applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 25, 9 (2017), 2411–2418.

[35] S. Khan, A. R. Alameldeen, C. Wilkerson, O. Mutluy, and D. A. Jimenezz. 2014. Improving Cache Performance Using Read-Write Partitioning. In *Proceedings of the 20th International Symposium on High Performance Computer Architecture (HPCA)*.

[36] Samira Manabi Khan, Yingying Tian, and Daniel A Jimenez. 2010. Sampling Dead Block Prediction for Last-Level Caches. In *Proceedings of the 43rd International Symposium on Microarchitecture (MICRO)*.

[37] Navid Khoshavi, Xunchao Chen, Jun Wang, and Ronald F DeMara. 2016. Read-tuned stt-ram and edram cache hierarchies for throughput and energy enhancement. *arXiv preprint arXiv:1607.08086* (2016).

[38] Namhyung Kim, Junwhan Ahn, Kiyoung Choi, Daniel Sanchez, Donghoon Yoo, and Soojung Ryu. 2018. Benzene: an energy-efficient distributed hybrid cache architecture for manycore systems. *ACM Transactions on Architecture and Code Optimization (TACO)* (2018).
[39] Toshiaki Kirihata, Paul Parries, David R Hanson, Hoki Kim, John Golz, Gregory Fredeman, Raj Rajeevakumar, John Griesemer, Norman Robson, Alberto Cestero, et al. 2005. An 800-MHz embedded DRAM with a concurrent refresh mode. *IEEE Journal of Solid-State Circuits* 40, 6 (2005), 1377–1387.
[40] Wolfgang H Krautschneider and Werner M Klingenstein. 1994. Process for the Manufacture of a High Density Cell Array of Gain Memory Cells. US Patent 5,308,783.
[41] Emre Kültürsay, Mahmut Kandemir, Anand Sivasubramaniam, and Onur Mutlu. 2013. Evaluating STT-RAM as an energy-efficient main memory alternative. In *Proceedings of International Symposium on Performance Analysis of Systems and Software (ISPASS)*.
[42] An-Chow Lai, Cem Fide, and Babak Falsafi. 2001. Dead-block prediction & dead-block correlating prefetchers. In *Proceedings 28th Annual International Symposium on Computer Architecture (ISCA)*.
[43] Donghyuk Lee, Yoongu Kim, Vivek Seshadri, Jamie Liu, Lavanya Subramanian, and Onur Mutlu. 2013. Tiered-latency DRAM: A low latency and low cost DRAM architecture. In *Proceedings of the 2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*.
[44] K Lee, R Chao, K Yamane, VB Naik, H Yang, J Kwon, NL Chung, SH Jang, B Behin-Aein, JH Lim, et al. 2018. 22-nm FD-SOI Embedded MRAM Technology for Low-Power Automotive-Grade-l MCU Applications. In *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*.
[45] Junmin Lin, Yu Chen, Wenlong Li, Aamer Jaleel, and Zhizhong Tang. 2009. Understanding the memory behavior of emerging multi-core workloads. In *Proceedings of the Eighth International Symposium on Parallel and Distributed Computing, ISPDC'09*.
[46] Prasanth Mangalagiri, Karthik Sarpatwari, Aditya Yanamandra, VijayKrishnan Narayanan, Yuan Xie, Mary Jane Irwin, and Osama Awadel Karim. 2008. A low-power phase change memory based hybrid cache architecture. In *Proceedings of the 18th ACM Great Lakes symposium on VLSI*. ACM, 395–398.
[47] R Manikantan, Kaushik Rajan, and Ramaswamy Govindarajan. 2011. NUcache: An efficient multicore cache organization based on next-use distance. In *Proceedings of the IEEE 17th International Symposium on High Performance Computer Architecture*. IEEE, 243–253.
[48] Raman Manikantan, Kaushik Rajan, and Ramaswamy Govindarajan. 2012. Probabilistic shared cache management (PriSM). In *2012 39th Annual International Symposium on Computer Architecture (ISCA)*.
[49] Asit K Mishra, Xiangyu Dong, Guangyu Sun, Yuan Xie, Narayanan Vijaykrishnan, and Chita R Das. 2011. Architecting on-chip interconnects for stacked 3D STT-RAM caches in CMPs. In *Annual International Symposium on Computer Architecture (ISCA)*.
[50] Sparsh Mittal and Jeffrey S Vetter. 2016. A survey of architectural approaches for data compression in cache and main memory systems. *IEEE Transactions on Parallel and Distributed Systems* 27, 5 (2016), 1524–1536.
[51] Naveen Muralimanohar, Rajeev Balasubramonian, and Norman P Jouppi. 2009. CACTI 6.0: A Tool to Model Large Caches. *HP laboratories* (2009), 22–31.
[52] Kyle J Nesbit and James E Smith. 2004. Data cache prefetching using a global history buffer. In *10th International Symposium on High Performance Computer Architecture (HPCA'04)*.
[53] Fabian Oboril, Rajendra Bishnoi, Mojtaba Ebrahimi, and Mehdi B Tahoori. 2015. Evaluation of hybrid memory technologies using SOT-MRAM for on-chip cache hierarchy. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2015).
[54] Biswabandan Panda and André Seznec. 2016. Dictionary Sharing: An Efficient Cache Compression Scheme for Compressed Caches. In *Proceedings of the 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*.
[55] Moinuddin K. Qureshi, Aamer Jaleel, Yale N. Patt, Simon C. Steely, and Joel Emer. 2007. Adaptive Insertion Policies for High Performance Caching. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA)*.
[56] Moinuddin K. Qureshi, Daniel N. Lynch, Onur Mutlu, and Yale N. Patt. 2006. A Case for MLP-Aware Cache Replacement. In *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*.
[57] Jan M Rabaey, Anantha P Chandrakasan, and Borivoje Nikolic. 2002. *Digital Integrated Circuits*. Vol. 2. Prentice hall Englewood Cliffs.
[58] Mitchelle Rasquinha, Dhruv Choudhary, Subho Chatterjee, Saibal Mukhopadhyay, and Sudhakar Yalamanchili. 2010. An Energy Efficient Cache Design Using Spin Torque Transfer (STT) RAM. In *Proceedings of the 16th International symposium on Low Power Dlectronics and Design (ISLPED)*.
[59] Paul Reed, Gus Yeung, and Bryan Black. 2005. Design aspects of a microprocessor data cache using 3D die interconnect technology. In *International Conference on Integrated Circuit Design and Technology*. IEEE.
[60] Mohammad Hossein Samavatian, Hamed Abbasitabar, Mohammad Arjomand, and Hamid Sarbazi-Azad. 2014. An Efficient STT-RAM Last Level Cache Architecture for GPUs. In *Proceedings of the 51st Annual Design Automation Conference (DAC)*.
[61] Amir Shalom, Robert Giterman, and Adam Teman. 2018. High Density GC-eDRAM Design in 16nm FinFET. In *Proceedings of the 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*.
[62] Sarabjeet Singh and Manu Awasthi. 2019. Memory Centric Characterization and Analysis of SPEC CPU2017 Suite. In *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*.
[63] Alan Jay Smith. 1982. Cache memories. *ACM Computing Surveys (CSUR)* (1982).

[64] Clinton W Smullen, Vidyabhushan Mohan, Anurag Nigam, Sudhanva Gurumurthi, and Mircea R Stan. 2011. Relaxing Non-Volatility for Fast and Energy-Efficient STT-RAM Caches. In *Proceedings of 17th High Performance Computer Architecture (HPCA)*.

[65] Dinesh Somasekhar, Yibin Ye, Paolo Aseron, Shih-Lien Lu, Muhammad M Khellah, Jason Howard, Greg Ruhl, Tanay Karnik, Shekhar Borkar, Vivek K De, et al. 2009. 2 GHz 2 Mb 2T gain cell memory macro with 128 GBytes/sec bandwidth in a 65 nm logic process technology. *IEEE Journal of Solid-State Circuits* 44, 1 (2009), 174–185.

[66] Guangyu Sun, Xiangyu Dong, Yuan Xie, Jian Li, and Yiran Chen. 2009. A Novel Architecture of the 3D stacked MRAM L2 Cache for CMPs. In *Proceedings of 15th International Symposium on High Performance Computer Architecture (HPCA)*.

[67] Zhenyu Sun, Xiuyuan Bi, Hai Li, Weng-Fai Wong, Zhong-Liang Ong, Xiaochun Zhu, and Wenqing Wu. 2011. Multi retention level STT-RAM cache designs with a dynamic refresh scheme. In *proceedings of the 44th annual IEEE/ACM international symposium on microarchitecture*. 329–338.

[68] N. Surana and J. Mekie. 2018. Energy Efficient Single-Ended 6-T SRAM for Multimedia Applications. *IEEE Transactions on Circuits and Systems II: Express Briefs* (2018).

[69] A. Teman, P. Meinerzhagen, R. Giterman, A. Fish, and A. Burg. 2014. Replica Technique for Adaptive Refresh Timing of Gain-Cell-Embedded DRAM. *IEEE Transactions on Circuits and Systems II: Express Briefs* 61, 4 (2014), 259–263.

[70] Fengbin Tu, Weiwei Wu, Shouyi Yin, Leibo Liu, and Shaojun Wei. 2018. RANA: Towards Efficient Neural Acceleration with Refresh Optimized embedded DRAM. In *Proceedings of the 45th Annual International Symposium on Computer Architecture (ISCA)*.

[71] Jue Wang, Xiangyu Dong, Yuan Xie, and Norman P Jouppi. 2013. i 2 WAP: Improving non-volatile cache lifetime by reducing inter-and intra-set write variations. In *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*.

[72] Zhe Wang, Daniel A Jiménez, Cong Xu, Guangyu Sun, and Yuan Xie. 2014. Adaptive Placement and Migration Policy for an STT-RAM Based Hybrid Cache. In *Proceedings of the 20th International Symposium on High Performance Computer Architecture (HPCA)*.

[73] Zhe Wang, Samira M. Khan, and Daniel A. Jiménez. 2012. Improving Writeback Efficiency with Decoupled Last-write Prediction. In *Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA)*.

[74] James Warnock, Brian Curran, John Badar, Gregory Fredeman, Donald Plass, Yuen Chan, Sean Carey, Gerard Salem, Friedrich Schroeder, Frank Malgioglio, et al. 2015. 4.1 22nm next-generation ibm system z microprocessor. In *Proceedings of the 2015 IEEE International Solid-State Circuits Conference-(ISSCC) Digest of Technical Papers*.

[75] Neil HE Weste and David Harris. 2015. *CMOS VLSI design: A Circuits and Systems Perspective*. Pearson Education India.

[76] Chris Wilkerson, Alaa R Alameldeen, Zeshan Chishti, Wei Wu, Dinesh Somasekhar, and Shih-lien Lu. 2010. Reducing Cache Power with Low-cost, Multi-bit Error-Correcting Codes. In *Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA)*.

[77] Matt Wordeman, Joel Silberman, Gary Maier, and Michael Scheuermann. 2012. A 3D system prototype of an eDRAM cache stacked over processor-like logic using through-silicon vias. In *IEEE International Solid-State Circuits Conference*.

[78] Xiaoxia Wu, Jian Li, Lixin Zhang, Evan Speight, Ram Rajamony, and Yuan Xie. 2009. Hybrid Cache Architecture with Disparate Memory Technologies. In *Proceedings of the 36th Annual International Symposium on Computer Architecture(ISCA)*.

[79] Yuan Xie. 2013. *Emerging memory technologies: design, architecture, and applications*. Springer Science & Business Media.

[80] Wei Xu, Hongbin Sun, Xiaobin Wang, Yiran Chen, and Tong Zhang. 2011. Design of last-level on-chip cache using spin-torque transfer RAM (STT RAM). *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 19, 3 (2011), 483–493.

[81] Yuang Zhang, Li Li, Zhonghai Lu, Axel Jantsch, Minglun Gao, Hongbing Pan, and Feng Han. 2014. A survey of memory architecture for 3D chip multi-processors. *Microprocessors and Microsystems* 38, 5 (2014), 415–430.

[82] Ping Zhou, Bo Zhao, Jun Yang, and Youtao Zhang. 2009. A durable and energy efficient main memory using phase change memory technology. In *Proceedings of the 36th Annual International Symposium on Computer Architecture (ISCA)*.

[83] Ping Zhou, Bo Zhao, Jun Yang, and Youtao Zhang. 2009. Energy reduction for STT-RAM using early write termination. In *Proceedings of the 2009 International Conference on Computer-Aided Design*.