Forecasting lifetime and performance of a novel NVM last-level cache with compression

Carlos Esquin*, Pablo Ibáñez*, Teresa Monreal†, Jose M. Llaberia†, Victor Viñals*

*University of Zaragoza  †Universitat Politècnica de Catalunya

I. INTRODUCTION

The goal of the last-level cache (LLC) in the memory subsystem of a chip multiprocessor is to filter main memory requests from the private levels and service them with a lower latency. The number of integrated cores on a chip is increasing and so is the LLC size. Most of LLCs are built using SRAM technology that does not scale well in terms of density and static power. Given this limitation, non-volatile (NV) memories are interesting alternatives to replace on-chip SRAM due to their higher density and lower static power [5]. However, write operations are costly in terms of latency and energy, and gradually wear out the bitcells until a hard fault occurs. Compared to SRAM, the write endurance in NV memories is much lower and is usually modeled by a normal distribution of mean $10^6$ write operations, $w$ depending on technology, manufacturer and target market [3], [6], [8].

Several works address the write endurance problem in complementary ways such as evenly distributing writes across the whole memory structure (wear-leveling [3], [9]), reducing the amount of written information [2], or using redundant storage [8]. In order to cope with hard faults, memory structures must include a way to detect and correct them, generally with error correction codes, ECCs. A bitcell failure requires deactivating the corresponding memory region it belongs to: a whole memory page [8], cache frame [1], or single byte [4].

In summary, to improve the state of the art in non-volatile LLCs (NV-LLC), proposals are needed that simultaneously reduce the number of writes, maintain uniform wear on all bit cells, and continue to operate even if some of the rated capacity is out of service. Thus, our first contribution is a novel NV-LLC design which combines byte-level disabling and compression. It can include any compression mechanism offering low decompression latency, high coverage and sufficient compression ratio. We select Base-Delta-Immediate (BDI) compression, which is well positioned in this tradeoff [7].

Cache block compression is used to achieve two effects, namely to reduce the amount of information written (to increase the cache lifetime) and to allow compressed blocks to be written to degraded cache frames (to increase the cache hit rate). To this end, we propose a circuitry that rearranges the bytes of a compressed block among the healthy bytes of a cache frame, while also guaranteeing uniform wear. A full-custom VLSI design demonstrates its feasibility in terms of area, latency and power consumption.

Previous work assesses lifetime improvement indirectly through relative reduction in the number of writes [9], or by using approximate aging models in the context of implementing main memory with NV technologies [8]. To the best of our knowledge, there is no model able to predict the time evolution of the capacity or performance of a NV-LLC that degrades as a consequence of writes. Therefore, our second contribution is such a forecasting procedure, suitable for multicore NV-LLC with or without compression. It proceeds through successive epochs. Within each epoch, a cycle-accurate simulation and byte failure prediction is performed.

II. COMPRESSION-BASED NV-LLC

We assume a SECDED mechanism, capable of correcting a single-bit fault and detecting up to two. The mechanism, upon detecting and correcting a single bit fault, triggers an operating system exception. In the absence of compression, the exception service routine has no choice but to disable the entire cache frame to avoid a second fault which could no longer be corrected. The baseline systems considered below are state-of-the-art proposals that implement this coarse-grained disabling [1], [8]. On the contrary, our proposal allows disabling at much finer byte granularity, taking advantage of compression to store cache blocks of reduced size.

To slow aging, it is necessary to avoid write concentration in specific regions of the cache. The conventional solution seeks an even distribution of writes among cache sets, and for this we index sets using a good hash function. However, since compression allows only part of the bytes of a frame to be written, our proposal also needs to ensure a balanced wear across all the bytes of the frame, and even when the capacity of the frames decreases. Therefore, we propose a new intra-line wear leveling mechanism based on byte rearrangement and a global counter that points out the starting point of frame writing.

Figure 1 explains the steps taken in a block write. First, the compression unit receives the block B, ① Compression. The result of each BDI compressor is: a) whether the compression is feasible and, if so, b) the compressed block, CB. Next, the corresponding ECC bits are computed from CB, ② ECC. The length of the compressed block and the ECC bits (ECB) determine the minimum capacity a frame must have to accommodate it. Accordingly, the replacement algorithm selects the victim block from the subset of frames with the required minimum capacity, ③ Replacement Algorithm.
Every frame has an associated bitmap that points out the faulty bytes. Finally, in relation to the intra-frame wear-leveling mechanism, a global counter points out the initial write position of the block within the frame, and from it the block is rearranged for selective writing (RECB) and the write control bits are generated. ④ Block rearrangement.

Fig. 1. Writing compressed blocks to the NV-LLC.

III. FORECASTING PROCEDURE

To forecast the evolution of the effective capacity over time, we propose to initialize a map with the number of remaining writes of each cache byte (RW map) using a statistical model of resilience of the selected memory technology ③, ⑧. A naive, but exact, approach is to simulate the progressive degradation of the NV-LLC: each time a write occurs, it is reflected in the RW map by degrading the corresponding bytes. When a byte suffers a hard fault it is deactivated and the simulation continues with the system a little more degraded. However, to get realistic results, the simulation must be cycle-accurate and driven by a realistic workload, but in doing so we could only forecast time lapses of a few milliseconds.

An approximate approach is to move faster in time by simulation-prediction epochs, see Figure 2. At each epoch, a short simulation calculates the number of writes per unit time to each byte, recording them in a per-byte write bandwidth map (WB map). At the end of the simulation it is calculated which byte will die next and in which time, updating the RW map with that byte deactivated. This approach needs as many simulations as the number of bytes in the LLC, which again means an unaffordable simulation time.

To decrease the number of simulations, we propose to approximate a little further by making K consecutive predictions from a single simulation, see blue text in Figure 2. For an uncompressed cache we compute the average WB in sets with A alive frames, $wb_{avg}(A)$, applying this aging rate to the RW map: to each byte, $wb_{avg}(A)$ is applied if its frame is in a set with A alive frames. The procedure can be extended to the cache with compression also taking into account the compression class (CC) each frame belongs to and calculating new averages $wb_{avg}(A,CC)$.

IV. EVALUATION

A multicore processor with the latency parameters of an STT-RAM LLC was simulated using the gem5 full-system simulator. The workload consists of application mixes taken from SPEC CPU 2006 and 2017. The bitcells write endurance follow a normal distribution of $\mu = 10^{11}$ maximum number of write operations and $\sigma = 0.2\mu$ ③, ⑧.

Figure 3 shows the evolution of the system performance and capacity of a 4MB LLC with the above characteristics, from the beginning of operation until reaching 50% of its effective capacity. The byte-disabling compression-based system (CMP) is compared against two baseline systems: Frame-Disabling (FD) ① and ECP6 which provides every frame with 6 ECPs (FD+6) ⑧. FD and FD+6 do not implement compression and disable whole cache frames when a byte experience a fault. Compared to FD, FD+6 and CMP multiply the time required to lose half the capacity by factors 1.47 and 6.2, respectively. CMP loses performance very gradually. For example, the time required to lose 15% of IPC is 3.1 and 2.1 times longer in CMP than in FD and FD+6, respectively.

Fig. 2. Forecasting procedure flow diagram.

Fig. 3. Lifetime and performance evolution until the NV-LLC reaches 50% of the effective capacity in a 4-core on-chip multiprocessor.

REFERENCES

[1] J. Chang et al., “The 65-nm 16-mb shared on-die l3 cache for the dual-core intel xeon processor 7100 series,” JSSC, 2007.
[2] D. B. Digen et al., “Compression architecture for bit-write reduction in non-volatile memory technologies,” in NANOARCH, 2014.
[3] H. Farbeh et al., “Floating-ec: Dynamic repositioning of error correcting code bits for extending the lifetime of stt-ram caches,” TomC, 2016.
[4] A. Ferreron et al., “Concertina: Squeezing in cache content to operate at near-threshold voltage,” Transactions on Computers, 2015.
[5] K. Korgaonkar et al., “Density tradeoffs of non-volatile memory as a replacement for sram based last level cache,” in ISCA, 2018.
[6] M. Natsui et al., “Dual-port sot-mram achieving 90-mhz read and 60-mhz write operations under field-assistance-free condition,” JSSC, 2020.
[7] G. Pekhimenko et al., “Base-delta-immediate compression: Practical data compression for on-chip caches,” in PACT, 2012.
[8] S. Schechter et al., “Use ecc, not ecc, for hard failures in resistive memories,” ACM SIGARCH Computer Architecture News, 2010.
[9] J. Wang et al., “2wap: Improving non-volatile cache lifetime by reducing inter-and intra-set write variations,” in HPCA, 2013.