Received 4 January 2024, accepted 16 January 2024, date of publication 19 January 2024, date of current version 30 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3355961

RESEARCH ARTICLE

FASTA: Revisiting Fully Associative Memories in Computer Microarchitecture

ESTEBAN GARZÓN[1], (Member, IEEE), ROBERT HANHAN[2], MARCO LANUZZA[1], (Senior Member, IEEE), ADAM TEMAN[2], (Member, IEEE), AND LEONID YAVITS[2], (Member, IEEE)

[1]Department of Computer Engineering, Modeling, Electronics and Systems, University of Calabria, 87036 Arcavacata di Rende, Italy
[2]EnICS Labs, Faculty of Engineering, Bar-Ilan University, Ramat Gan 5290002, Israel

Corresponding author: Esteban Garzón (esteban.garzon@unical.it)

This work was supported in part by the European Union's Horizon Europe Program for Research and Innovation under Grant 101047160, in part by the Israeli Ministry of Science and Technology under Lise Meitner grant for Israeli-Swedish Research Collaboration under Grant 1001569396, and in part by the Israeli Ministry of Science and Technology grant for Groundbreaking Research under Grant 1001702600. The work of Esteban Garzón was supported by the Italian Ministry for Universities and Research (MUR) under the Call "Horizon Europe 2021–2027 Program under Grant H25F21001420001".

ABSTRACT Associative access is widely used in fundamental microarchitectural components, such as caches and TLBs. However, associative (or content addressable) memories (CAMs) have been traditionally considered too large, too energy-hungry, and not scalable, and therefore, have limited use in modern computer microarchitecture. This work revisits these presumptions and proposes an energy-efficient fully-associative tag array (FASTA) architecture, based on a novel complementary CAM (CCAM) bitcell. CCAM offers a full CMOS solution for CAM, removing the need for time- and energy-consuming precharge and combining the speed of NOR CAM and low energy consumption of NAND CAM. While providing better performance and energy consumption, CCAM features a larger area compared to state-of-the-art CAM designs. We further show how FASTA can be used to construct a novel aliasing-free, energy-efficient, Very-Many-Way Associative (VMWA) cache. Circuit-level simulations using 16 nm FinFET technology show that a 128 kB FASTA-based 256-way 8-set associative cache is 28% faster and consumes 88% less energy-per-access than a same sized 8-way (256-set) SRAM based cache, while also providing aliasing-free operation. System-level evaluation performed on the Sniper simulator shows that the VMWA cache exhibits lower Misses Per Kilo Instructions (MPKI) for the majority of benchmarks. Specifically, the 256-way associative cache achieves 17.3%, 11.5%, and 1.2% lower average MPKI for L1, L2, and L3 caches, respectively, compared to a 16-way associative cache. The average IPC improvement for L1, L2, and L3 caches are 1.6%, 1.4%, and 0.2%, respectively.

INDEX TERMS Fully associative cache, CAM, content addressable memory, memory architecture, aliasing, associative memory, 256-way associative cache.

I. INTRODUCTION

Associative access, i.e., access by content rather than by explicit address, is widely used across computer microarchitecture. This type of access is required for caches of all levels, translation lookahead buffers (TLB), reservation stations and register renaming in out-of-order execution engines, branch target buffers (BTB), and other fundamental microarchitectural components [1], [2], [3], [4], [5], [6], [7], [8]. Caches in typical processors are limited up to 8-way or 16-way associativity [9], [10], [11], [12], [13]. However, higher associativity has at least two distinct advantages over lower way-associative options:

1) Higher associativity leads to fewer conflict misses, with fully associative caches entirely devoid of such misses, thus reducing the overall miss rate.

The associate editor coordinating the review of this manuscript and approving it for publication was Mario Donato Marino.

2) Higher associativity mitigates the size limitation of aliasing-free virtually-indexed physically-tagged (VIPT) caches [14].

High associativity generally requires Content Addressable Memory (CAM). However, CAM has been historically considered too expensive to become a universal choice in computer microarchitecture [15], [16], [17], [18], [19]. The traditional presumption about CAMs is that they are:

1) Too power-hungry [15], [17], [19], [20], [21], [22]
2) Slower than conventional SRAM-based caches [23], [24]
3) Not scalable, making the implementation of large caches, many-way associativity, or other associative memories impractical [16], [18], [20], [25], [26]
4) Too expensive (in terms of silicon area) [18], [19], [25]

In this paper, we revisit the use of associative memories in computer microarchitecture, focusing on caches, but with relevance to other microarchitectural components that could benefit from full-associativity. This is done based on (1) a novel Complementary CAM (CCAM) bitcell that enables low-power CAM implementation, (2) a fully-associative tag array (FASTA) architecture, constructed with the CCAM bitcell, providing low-latency, precharge-free operation, and (3) a novel Very-Many-Way Associative (VMWA) cache architecture, built using FASTA. Therefore, we address the three first assumptions, i.e., power, performance, and scalability, and show that they can be overcome. We further show that the area overhead (the fourth assumption) in applications, such as cache, is moderate and hence might be affordable.

Our study includes detailed circuit design of FASTA, including custom layout of the CCAM bitcell and peripheral circuitry. The power, performance, and area of the proposed solution are evaluated and compared with SRAM-based options through post-layout circuit simulations using a commercial 16 nm technology. Functional performance is evaluated through system-level simulations using the Sniper CPU simulator [27].

This work makes the following contributions:

1) We revisit and reevaluate the feasibility of implementing associative memories in computer microarchitecture.
2) We propose a novel CAM bitcell (CCAM) that enables energy-efficient CAM operation.
3) We introduce FASTA, a fully associative memory design with a wide spectrum of applications in computer microarchitecture that benefit from high-associativity, including cache, TLB, BTB, reservation stations, register renaming, etc. FASTA overcomes the traditional drawbacks of fully associative memories, such as high energy consumption and scalability issues, making it practical to implement many-way fully associative memories.
4) We introduce a FASTA based *very-many-way associative cache* architecture that enables the construction of scalable (very large), aliasing-free, energy-efficient, low-latency caches.
5) We compare the proposed architecture with a baseline way-associative cache at the circuit- and architectural-levels to show the effectiveness of the solution.

II. BACKGROUND AND MOTIVATION

In this section, we overview the architectures and common implementations of traditional content addressable memory (CAM) arrays, introducing the challenges and tradeoffs that have limited their use in computer architecture.

A. CONTENT ADDRESSABLE MEMORIES

A CAM is a memory that, in addition to write and read, enables simultaneous comparison of a query (search) pattern with the entire content of the memory [19]. The top-level scheme of a CAM array is presented in Fig. 1(a). The array is based on bitcells (CAM BCs) with the same basic structure of a 6T-SRAM bitcell. This memory structure enables read and write operations to a cross-coupled inverter core through a set of wordlines (WLs) and complementary bitlines (BLs). In the context of CAM arrays, these BLs can be denoted as searchlines (SLs) [28]. Write and read operations are implemented similarly to SRAM. A set of search data registers (SDRs) drive the data onto the SLs during write and sample the data from the SLs following a read.

The search or compare operation is where CAM deviates from random-access memories [19], [29]. To carry out a compare operation, the search pattern is loaded into the SDRs and driven onto the SLs. If the search pattern matches the pattern stored in a certain row, the matchline (ML) of that row signals a `hit`. In the mismatching rows, a `miss` is signaled by the ML.

Logically, every i_{th} row in an m entry CAM contains n-bits of stored data and an n-bit comparator, whose output is ML_i. The comparator is typically implemented by one of the following two identical functions:

$$\overline{\mathrm{ML}_i} = \mathrm{NAND}\left(S \bar{\oplus} D\right), \quad (1)$$

$$\mathrm{ML}_i = \mathrm{NOR}\left(\overline{S \bar{\oplus} D}\right), \quad (2)$$

where S is the search (query) pattern, D is the pattern stored in the CAM row, and $\bar{\oplus}$ represents a bitwise XNOR operation between the $S_{i,j}$ and $D_{i,j}$ values broadcast and stored respectively in the column j.

Accordingly, there are two types of CAMs: NOR and NAND. The bitcells implementing NOR and NAND CAM arrays are presented in Fig. 1(b) and Fig. 1(c), respectively. The internal data nodes (D and $\overline{D}$) of the cross-coupled inverter core of both cells are connected to the access transistors (denoted "MT" in the figure). The cell is accessed for write and read operations by asserting the WL, and driving BL and $\overline{\mathrm{BL}}$ to opposite logic values for write, or precharging them for read. The data nodes are also connected to

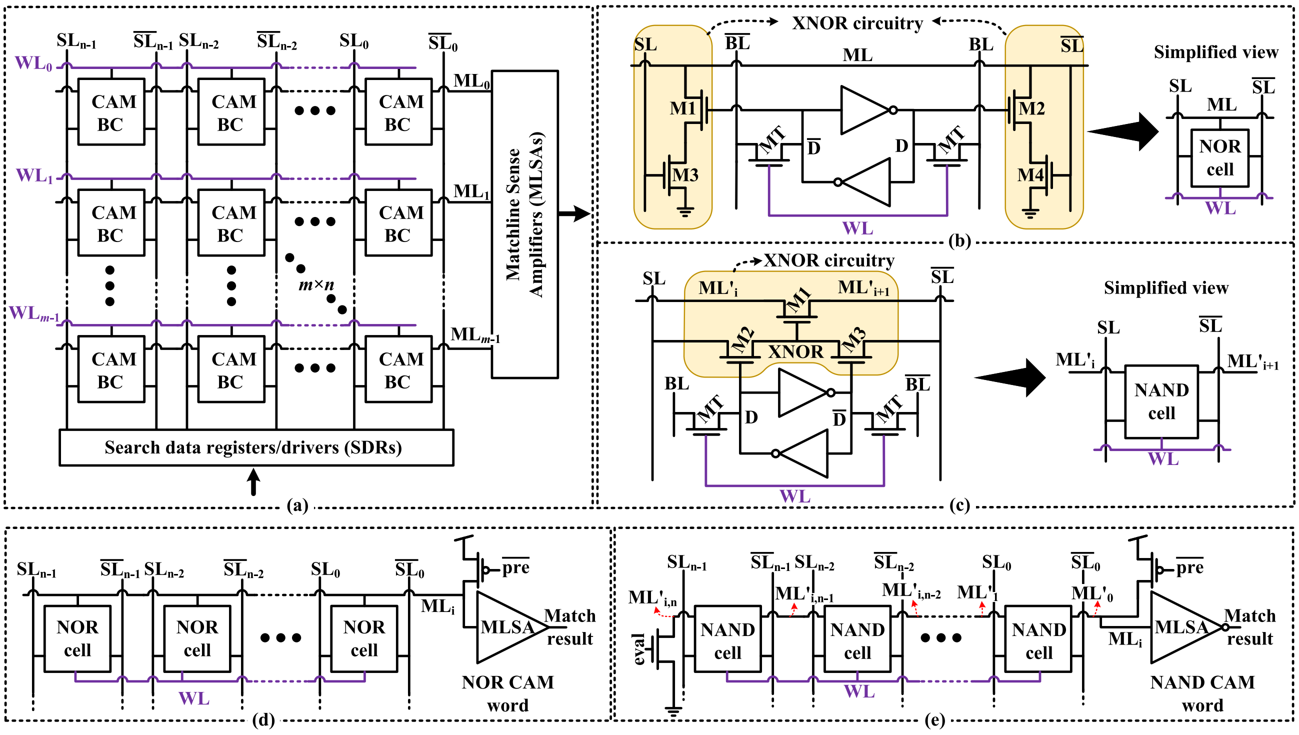


FIGURE 1. Associative memory (adopted from [19]): (a) top-level view, (b) NOR cell, (c) NAND cell, (d) NOR Matchline, (e) NAND Matchline. Bitlines (BLs) are not shown in (a), (d), and (e) for simplicity.

XNOR circuitry that is implemented differently for the two types of cells. The NOR CAM cell implements the XNOR by conditionally pulling down the ML through M1/M3 and M2/M4. The NAND CAM cell implements the XNOR by the M2/M3 transistor pair, conditionally enabling a pass gate (M1). The NOR and NAND CAM row designs are illustrated in Fig. 1(d) and Fig. 1(e), respectively.

Both the NOR and the NAND CAMs carry out a compare operation in two phases: precharge and evaluation. During the precharge phase, the $\overline{\text{pre}}$ signal is asserted to pull the ML high in both configurations. However, during the evaluation phase, the behavior of the NOR and NAND CAMs is complementary. In the NOR CAM, an **inverted** search pattern is asserted on the SLs. If the stored bit equals the **inverted** search bit in *even one* cell, i.e., $D_{i,j} = \overline{S_{i,j}}$ for row i and any column j, the ML discharges through either M1/M3 or M2/M4. Consequently, the value of ML_i at the end of the compare cycle is '0', signaling a **mismatch**. Otherwise, if the stored bit does **not** equal the **inverted** search bit in **every** cell, the ML has no discharge path. Therefore, its value remains high at the end of the compare cycle, signaling a **match**. For the NAND CAM, if the stored bit equals the search bit in *every* cell (i.e., $D_{i,j} = S_{i,j}$ for a row i and every column j), the ML discharges through the series of M1_i transistors, dropping to '0' and signaling a **match** at the end of the compare cycle. If *even a single* NAND CAM cell mismatches, the ML remains high, signaling a **mismatch**. While both options provide the required functionality, they present very different design tradeoffs when implementing an associative memory.

1) OTHER TECHNOLOGY ALTERNATIVES FOR ASSOCIATIVE MEMORIES

a: CNTFET-BASED CAM

Carbon Nanotube Field-Effect Transistor (CNTFET) technology has been proposed as a potential alternative to Silicon transistors. With high mobility and a fabrication process compatible with various substrates, CNTFETs have demonstrated the capability to build high-speed CMOS digital circuits and microprocessors, providing as much as a 10× improvement in performance over silicon counterparts [30]. Various CNTFET-based CAMs have been proposed in the literature [30], [31], [32], [33], [34], showing that CNTFET technology can excel in terms of search power and delay as compared to conventional silicon-transistors-based CAM. Some CAM cells utilize different techniques such as top-gated CNTFET (TG-CNTFET) and gate-all-around CNTFET (GAA-CNTFET) to improve performance [31], [32]. Such approaches have been shown to be efficient in reducing search delay and power consumption, while improving the noise immunity of CNTFET-based CAMs. Other types of optimizations include shorted gate CNTFET (SG-CNTFET) and independent gate CNTFET (IG-CNTFET). Unfortunately, CNTFET technology suffers from yield issues in large-scale manufacturing [35], which currently limits the size of CNTFET-based CAMs.

b: FeRAM-BASED CAM

Ferroelectric Random Access Memory (FeRAM) is a type of capacitor-based memory, where the ferroelectric material is used as the dielectric layer between two electrodes [36]. FeRAM has been in the market for some time and has niche usage in low-power-embedded systems, thanks to its speed, low power access, and low voltage operation. However, it has not reached widespread adoption in CAMs mainly due to its moderate area footprint and its destructive read operation [37], [38]. To address this issue, a novel approach has been developed utilizing quasi-nondestructive readout (QNRO) of the capacitor polarization. By selectively switching a small portion of polarization, enough to activate a read transistor channel, this technique allows for multiple read cycles exceeding 10^6 cycles before necessitating a write-back operation [37]. This enhances the feasibility of integrating FeRAM technology into CAM systems, presenting a viable solution to its previous limitations.

c: FeFET-BASED CAM

As in FeRAM technology, the ferroelectric field-effect transistor (FeFET) uses the polarization of ferroelectric materials. Toggling between the two polarities, FeFET can be used as memory storage. Several FeFET-based CAMs have been proposed [39], [40], [41], [42], [43]. FeFET CAMs mainly offer significant advantages over traditional SRAM-based CAMs, demonstrating improved memory density and energy efficiency [41]. These designs include the conventional 2FeFET-based CAM, multi-level cell architectures, and single FeFET CAMs [42], [43]. The latter has very high density, but requires several steps to carry out a search operation. Recent advancements include single FeFET CAM designs that exploit the ambipolar transport of devices, such as Schottky barrier FeFET or ambipolar ferroelectric tunnel FET (FeTFET), allowing pattern searching in just one step [44]. However, the asymmetry in ambipolar FeFETs impacts the CAM functionality and necessitates careful tuning [44]. Furthermore, FeFET-based CAMs use two different voltage levels for programming and search operation, which leads to an increase in design complexity and area overhead of the overall memory architecture.

d: MRAM-BASED CAM

Magnetoresistive Random-Access Memory (MRAM) is a spintronics memory technology that exploits the intrinsic spin of the electron to store information. MRAM technology has seen significant advancements in the last decade, with the two latest generations, namely Spin-Transfer Torque (STT) and Spin-Orbit Torque (SOT), gaining great interest. These MRAM technologies have found been proposed as memory cells in CAMs [45], [46], [47], [48], [49], [50], including a hybrid SOT/STT CAM cell [51]. However, the widespread adoption of MRAM faces challenges, such as the delicate balance between high programming current density and fast access speed, as well as concerns related to reliability arising from operational disturbance and process variations. Other magnetic memory technologies, including skyrmions and Domain-Wall "racetrack" Memory (DWM), have also been proposed for the construction of associative memories [52], [53].

e: RRAM-BASED CAM

Resistive Random-Access Memory (RRAM) stores data in the form of resistance. RRAM is built using oxide-based material, which changes its ion distribution in the presence of heat or electric field. Several RRAM-based CAMs have been proposed [54], [55], [56], [57], [58]. Among the most important issues of RRAM technology are the uniformity and reliability of resistive switching. This impacts the RRAM-based CAM performance, leading to false mismatch results [56].

f: FLASH-BASED CAM

Flash is based on moving the electric charge on and off of a floating-gate structure, thereby modifying the effective threshold voltage of the device. A number of CAM designs based on both NOR and NAND Flash alternatives have been proposed [59], [60], [61], [62], [63], [64]. In addition to the small area footprint and reduced power consumption compared to conventional CMOS-based CAM designs, Flash technology is mature enough for large-scale integration [60]. NAND Flash, employing 3D stacking and multilevel cells, has achieved significant density improvements. However, this has introduced reliability issues necessitating advanced wear leveling and error-correction mechanisms in modern Flash controllers. Moreover, 3D NAND flash CAM performance is limited by the charge trapping mechanism, and presents a high area-cost peripheral mainly due to the large voltage required for write operations [60].

B. TRADEOFFS AND CHALLENGES IN CAM DESIGN

The CAM circuits, described above, face several design tradeoffs that have limited their use in many microarchitectural components [16], [26], [65], instead giving preference to alternative implementations based on SRAM or register files [66]. These tradeoffs and challenges, including speed, energy consumption, scalability, and area overhead, are elaborated upon hereafter.

1) SPEED

The worst-case discharge path of the ML in NOR CAM is a pair of serially connected transistors (M1/M3 or M2/M4 of a single mismatching bit). In comparison, the ML discharge path of a NAND CAM goes through w serially connected transistors, where w is (tag_size + number_of_auxiliary_bits) in the case of a cache tag array. Therefore, NOR CAM has a clear speed advantage over NAND. For many high-speed applications [24], [67], [68], and especially when the associative array is on the critical path as in a TLB [69], [70], [71], the speed penalty of a NAND implementation is too high, leaving the NOR implementation as a likely option. The novel CCAM bitcell and a FASTA tag array architecture, introduced in Section III, do not require precharge and feature a tree-like logic for resolving the long resistive paths.

2) ENERGY CONSUMPTION

Unfortunately, the speed advantage of a NOR CAM comes at a very significant power consumption cost [24], [26], [66]. In the majority of CAM applications in computer microarchitecture (including cache and TLB), only a single CAM row matches in a compare cycle in the case of a `hit` and no rows match in the case of a `miss` [25], [72]. Therefore, at best, all but one of the MLs in a NOR CAM discharge during every compare cycle, and hence, need to be fully charged again before the next access. The resulting power consumption is excessive and is often the reason to prefer an alternative solution that is based on a non-associative memory. In contrast, a NAND CAM has lower energy consumption, as only the matching ML discharges, while the rest remain high and likely do not need to be re-charged for the next comparison (access). However, as mentioned above, the slow operation of a NAND CAM strongly limits its use in most microarchitectures. The proposed CCAM bitcell and FASTA architecture resolve this challenge, as well, by enabling low-energy CAM operation, comparable to NAND CAM.

3) SCALABILITY

Another concern in CAM-based fully-associative cache is its limited scalability [73], [74]. The reason is as follows. Large random-access memory structures are typically implemented hierarchically, for example with hierarchy levels such as ranks, banks, and subarrays in dynamic random-access memory (DRAM). A subarray size is limited due to physical restrictions, such as wordline and bitline lengths. Address decoding in such memories is also hierarchical, with a global decoder or pre-decoder selecting the subarray or a limited set of subarrays, and a local (row) decoder selecting the memory row. Because of that, only the subarrays where the data is located are activated, while the rest might be idle, saving energy. In CAMs, such a hierarchical approach is typically impossible, since the searched-for pattern might be located anywhere in the memory. Therefore, the entire memory needs to be searched, for which the entire memory needs to be active (i.e., all searchlines must be asserted), thus making truly large CAMs energetically impractical. The FASTA-based VMWA cache architecture, presented in Section IV, trades off full associativity for a slightly lower, but still very high associativity, while idling the non-accessed portions of a large cache. This feature enables the construction of large and energy-efficient caches.

4) AREA OVERHEAD

A final challenge that limits the use of CAMs is their area overhead [75]. CAMs are well-known to be larger than SRAMs of the same dimensions. Both NOR and NAND CAM cells have at least 50% more transistors than a 6T-SRAM cell. This has been another factor that traditionally turned away computer architects in times of scarcity.

However, in most computer microarchitecture applications, associative access is not used standalone. An associative memory array is typically coupled with an SRAM. For example, in cache, while the tag array might be implemented with CAM, the cache blocks are stored in an SRAM, controlled by the tag array. The data SRAM is of the same height as the tag array, but is much wider, for example, 64-bit tag array and 512-bit data SRAM. Therefore, even if the tag array carries a significant area overhead, it becomes much less substantial when amortized over the entire cache area. Accordingly, we believe that a certain degree of area overhead can be tolerated. This is because of the benefits provided by full (and very-many-way) associativity, as well as the relative size of the associative tag memory as compared to adjacent random access data memories.

C. RELATED WORK

Fully associative structures in computer microarchitecture, in general, and cache architecture, in particular, have not been a major focus of academic research in recent years. Therefore, positioning and evaluating the proposed FASTA-based VMWA cache design vis-a-vis recent developments in cache architecture is challenging. In this fairly limited related work section, we focus on several recent works that attempted to improve the associativity of cache, while optimizing the associativity vs. power consumption trade-off.

Increasing the associativity is a common technique for conflict miss reduction in caches [76], [77]. Consequently, VMWA eliminates conflict misses almost entirely as shown in Section V.

Non-temporal Streaming, a conflict miss rate optimization technique, uses a separate fully associative buffer in parallel with the main direct-mapped cache [78]. In [79] and [80], the conflict miss reduction is achieved by optimally selecting address bits for the cache index, for a fixed set of applications. MCC-DB [81] uses the prior knowledge of data access patterns to minimize LLC conflicts in multi-core systems through an enhanced OS facility of cache partitioning. CCProf [82] is a lightweight measurement-based profiler that identifies conflict cache misses and associates them with program source code and data structures.

Higher associativity often results in increased cache power consumption. Sub-banking was proposed in [83] as a method to reduce power consumption in way-associative caches by activating only parts of the cache while keeping the rest idle. Somewhat similarly, the proposed VMWA cache resolves the power consumption bottleneck and the scalability limitation of many-way associative caches by limiting the associativity level to the size of a single memory subarray. However, the proposed VMWA cache is built with an actual associative memory designed upon a novel CCAM cell, including additional circuit techniques. This approach enables very high associativity (e.g., 256-way), while at the same time featuring much lower energy consumption.

A B-cache was introduced as a technique for conflict miss reduction [84]. One disadvantage of this method is high per-access power consumption. An enhanced B-Cache [85] reduced the total access energy consumption; however, it remained higher than that of a lower associativity cache. The FASTA-based VMWA cache enables almost complete conflict miss elimination, and at the same time, significantly reduces the cache access energy consumption compared to a typical way-associative cache.

Another work that discusses the concept of many-way associative cache is [86], as one of the applications of a programmable resistive address decoder. However, the solution proposed there is based on emerging resistive memories, and therefore, its application in traditional computer microarchitecture is limited.

III. PROPOSED CCAM BITCELL AND FASTA ARCHITECTURE

The previous section overviewed CAM and introduced the various design tradeoffs and challenges that have limited its use in modern microarchitectures. In this section, we introduce FASTA, an energy-efficient and scalable associative memory architecture based on CCAM, a novel CAM bitcell that enables FASTA to meet the performance and power requirements of cache memories and other microarchitectural components.

A. COMPLEMENTARY CAM BITCELL

The basic building block of the energy-efficient FASTA architecture is the complementary CAM bitcell, shown in Fig. 2. While the storage mechanism of the CCAM bitcell is identical to standard CAM and SRAM cells (M1, M2, and the cross-coupled inverters of Fig. 2), the comparison logic is a combination of the NOR and NAND CAM schemes. This complementary (full CMOS) design enables static precharge-free operation with low power consumption, as explained hereafter.

The storage nodes of the CCAM bitcell (D and $\overline{\mathrm{D}}$) are connected to the gates of a pair of nMOS pass transistors (M3 and M4). These devices are connected to the searchlines (SL and $\overline{\mathrm{SL}}$) on one side and to each other on the other side, similar to a NOR CAM cell. The node connecting M3 and M4 represents an XNOR operation between the stored bit and the bitline data, as noted in Fig. 2. The XNOR node is connected to the gates of two additional transistors: a pMOS ($\mathrm{M_{NOR}}$) and an nMOS ($\mathrm{M_{NAND}}$). Similar to the NAND CAM configuration of Fig. 1(e), $\mathrm{M_{NAND}}$ is connected in series to the $\mathrm{M_{NAND}}$ transistors of the adjacent bitcells in the same row, i.e., the $\mathrm{ML_{NAND}}$ line of each CCAM cell is connected to the left end of the $\mathrm{M_{NAND}}$ transistor in the neighboring CCAM cell. The source of the $\mathrm{M_{NAND}}$ of leftmost bitcell in a row is connected to ground, such that if every XNOR node in a row is high, the $\mathrm{ML_{NAND}}$ line is discharged. This is illustrated in Fig. 3(a) for a 3-bit row of CCAM cells with all query patterns

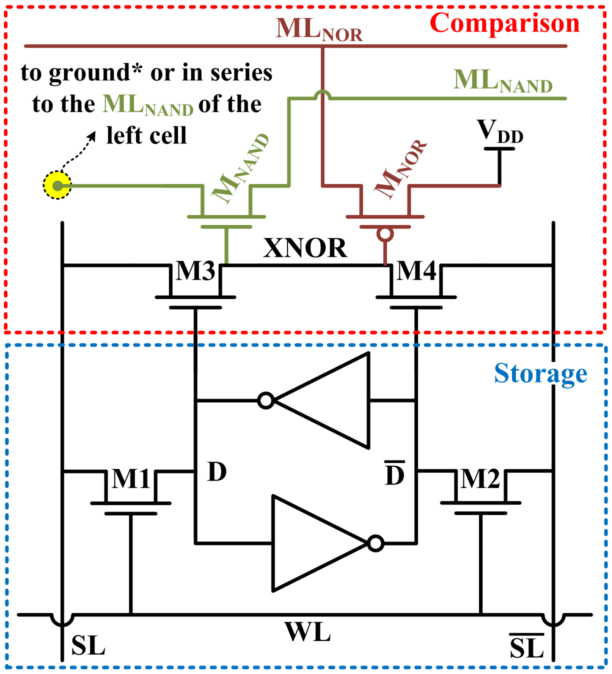


FIGURE 2. Schematic of the ten-transistor CCAM bitcell. *Only the source of the $\mathrm{M_{NAND}}$ of leftmost bitcell in a row is connected to ground.

(driven onto SL_j and $\overline{\mathrm{SL}}_j$) matching the data stored in the bitcells.

To complement this, the source of $\mathrm{M_{NOR}}$ is connected to the power supply (V_{DD}), while its drain is connected to the $\mathrm{ML_{NOR}}$ signal, shared by all bitcells in a row. Therefore, it is sufficient for the XNOR node of one cell in a row to be low, i.e., a single mismatching bit, to charge $\mathrm{ML_{NOR}}$. This is illustrated in Fig. 3(b), where $\mathrm{SL_1}$=‘0’ and $\mathrm{D_1}$=‘1’, such that $\mathrm{XNOR_1}$ is pulled down, opening a pull-up path to $\mathrm{ML_{NOR}}$ through $\mathrm{M_{NOR}}$. By shorting together the $\mathrm{ML_{NOR}}$ and $\mathrm{ML_{NAND}}$ lines at the drain node of the rightmost bitcell in a row, a row of CCAM cells makes up an n-input CMOS NAND gate. This structure benefits from the statically driven full-swing output characteristics of CMOS digital logic.

The complementary functionality of the CCAM ML dramatically reduces the switching activity of this high capacitive signal, because all but one ML (in the case of cache/TLB hit), or all MLs (in the case of cache/TLB miss) are statically held at V_{DD}. This is in contrast to a conventional NOR CAM, where all MLs (or all MLs but one) discharge during every compare cycle, and hence, need to be precharged before the next cache access [87]. Table 1 summarizes the electrical simulation results in terms of search latency and energy consumption of NOR CAM and NAND CAM relative to CCAM. All CAM designs were organized as 256 64-bit words built using a 16 nm FinFET technology operating at a V_{DD} of 0.8 V. CCAM has the lowest search energy but the highest delay among the three. The latter is due to the long pull-down path, similar to NAND CAM, exacerbated in CCAM by the additional capacitance of the pull-up PMOS transistors absent in NAND CAM. However, the resulting

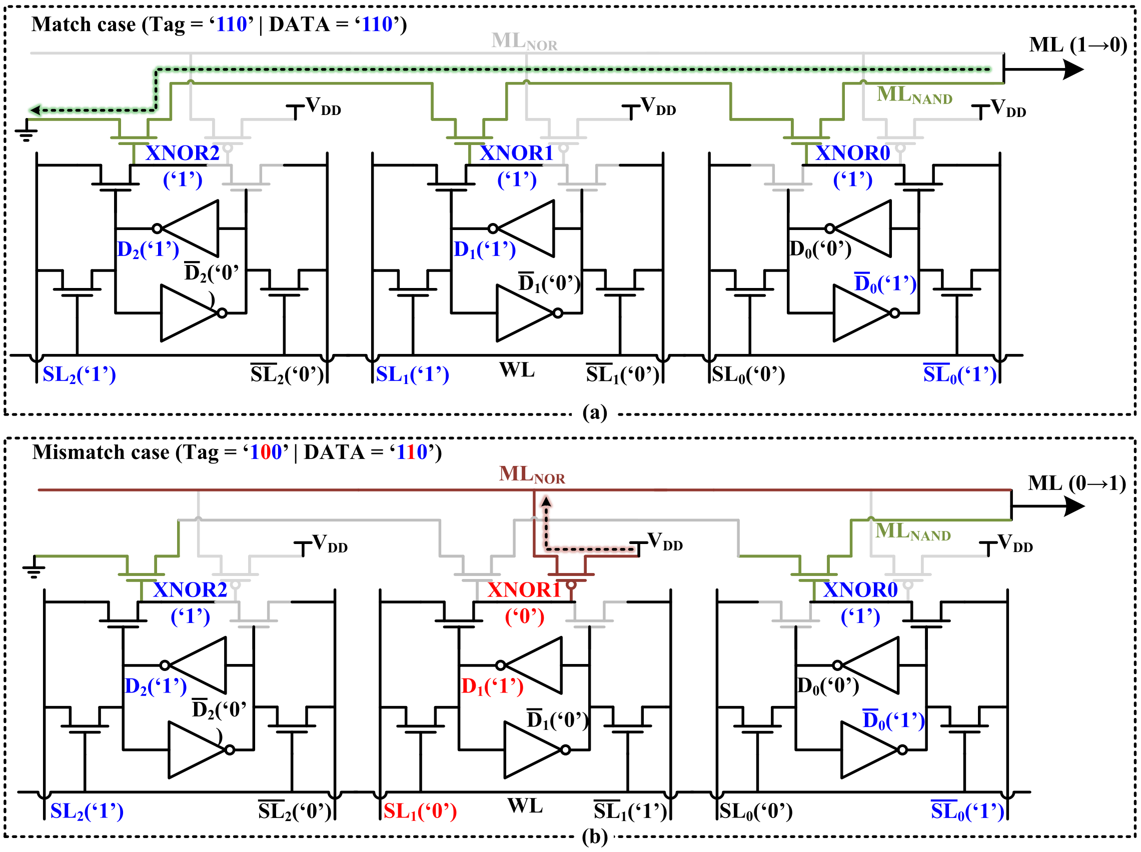


FIGURE 3. Match and mismatch cases for a 3-bit wide row of CCAM cells.

TABLE 1. Normalized search latency and energy consumption of a 256×64 NOR CAM and NAND CAM relative to CCAM. The reported values were obtained by means of circuit-level simulations using 16 nm FinFET technology, operating at V_{DD} of 0.8 V.

Type	Search Delay	Search Energy	Operating Frequency*
NOR CAM	0.58×	58.8×	0.5×
NAND CAM	0.75×	2.5×	0.5×

**Frequency reduction is due to precharge mandatory in NAND and NOR CAMs.*

CCAM frequency is still higher than that of NAND and NOR CAM, because those require a precharge cycle. CCAM-based FASTA requires no precharge, as elaborated upon in the next subsection.

B. FULLY-ASSOCIATIVE TAG ARRAY (FASTA)

Using the CCAM cell as the basis for data storage and comparison, a memory architecture is proposed to implement a fully associative tag array (FASTA). As shown in Fig. 4(a), FASTA comprises three components: **(1)** an array of CCAM cells, **(2)** a column of matchline sense amplifiers (MLSAs), and **(3)** hit/miss logic. The array is organized as m rows of w CCAM bitcells, which output m ML signals that are sensed by the MLSA circuitry. A w-bit wide row with its MLSA is shown in Fig. 4(b). For the current study, a simple inverter was used to implement the MLSAs; however, a faster sensing circuit could be used at the expense of increased area. The output signals of the MLSAs (hit $[0:m-1]$) typically enable the wordlines of an adjacent SRAM data array, e.g., in cache, where the SRAM row contains a cache block.

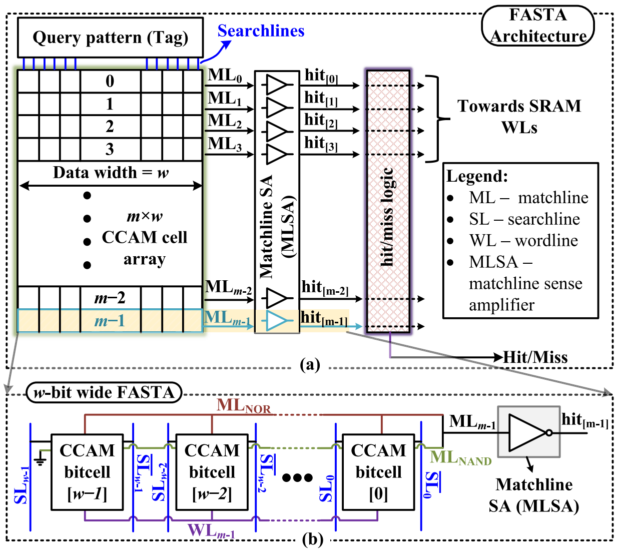


FIGURE 4. FASTA Architecture: (a) Top-level schematic diagram of the CCAM array, and (b) Zoom in on w-bit wide row of the array.

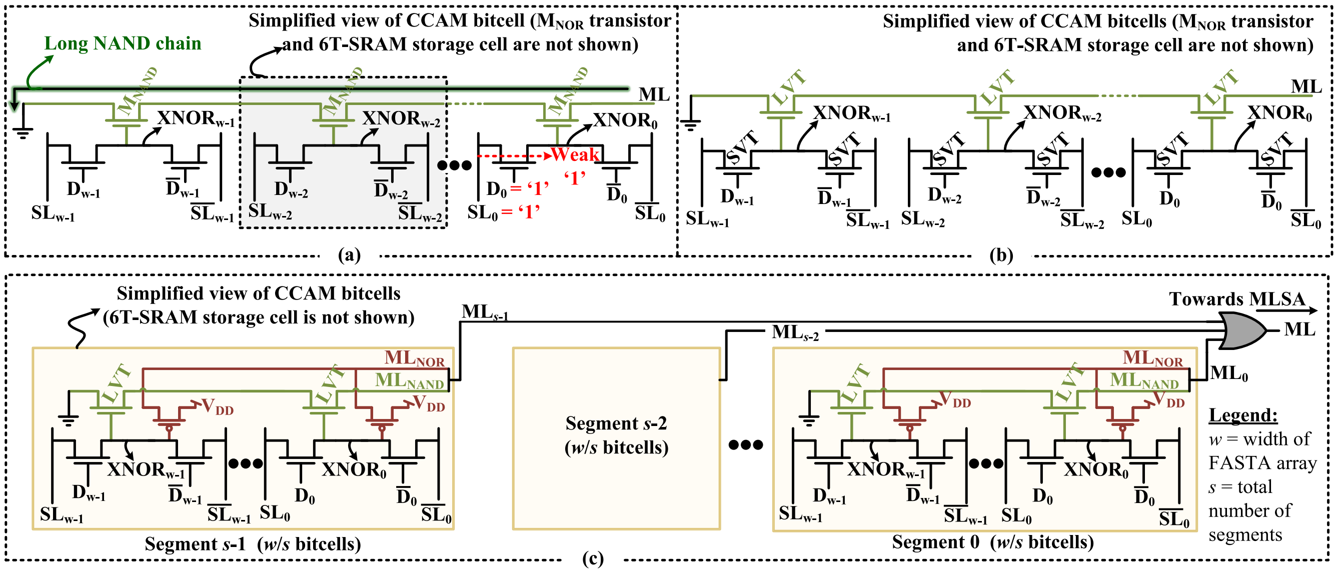

Long NAND chain
Simplified view of CCAM bitcell (MNOR transistor and 6T-SRAM storage cell are not shown)
ML
XNORw-1
XNORw-2
XNOR0
Weak '1'
D0 = '1'
SL0 = '1'
(a)
Simplified view of CCAM bitcells (MNOR transistor and 6T-SRAM storage cell are not shown)
LVT
SVT
(b)
Simplified view of CCAM bitcells (6T-SRAM storage cell is not shown)
MLs-1
MLs-2
Towards MLSA
ML0
MLNOR
MLNAND
VDD
Segment s-2 (w/s bitcells)
Segment s-1 (w/s bitcells)
Segment 0 (w/s bitcells)
Legend:
w = width of FASTA array
s = total number of segments
(c)

FIGURE 5. FASTA NAND chain discharge path in the case of a match operation. (a) Simplified scheme of the FASTA row. (b) FASTA NAND chain built from different transistor flavors. (c) w-bit wide FASTA NAND chain divided in s segments of w/s bitcells.

The hit/miss logic is essentially an m-input OR gate that signals a `hit` if there is a match in (at least) one of the rows and a `miss` otherwise. This logic can be implemented using various design approaches, such as a wired-OR or a NOR-tree, which is the structure used in this work.

The Query pattern (Tag) register of Fig. 4(a) is used by all three basic operations (write, read and compare) in the FASTA architecture. During writes, this register drives the data onto the CCAM searchlines (SLs), and during memory reads (if optionally implemented in the array), the data that is read out is sampled by that register. For a compare operation, the query pattern (a tag in the case of a cache) is written to this register and driven onto the SLs. XNOR between the search pattern bit and the stored data bit drives the ML transistors in the CCAM cells (see Fig. 2). This, in turn, discharges (**match**) or charges (**mismatch**) ML_i as demonstrated in Fig. 3. For a matching row, when the ML voltage discharges past the trip point of the MLSA inverter, the hit $[i]$ signal of that row rises. This signal is then output from the array and transmitted through the hit/miss logic to generate a global `hit` signal.

Unlike conventional NOR and NAND CAMs, the compare operation in the CCAM-based FASTA does not require precharge. This saves the delay of the precharge phase, which can be substantial, as it has to be designed assuming all MLs are fully discharged before the compare operation. However, the pull-down path still needs to traverse through w nMOS devices to signal a match, as illustrated in Fig. 5(a). One option to reduce the delay of this path is to use low or ultra-low threshold voltage (LVT or ULVT) transistors (Fig. 5(b)), which are typically provided as part of any modern process design kit, to implement M_{NAND}. While these transistors are much faster than the standard or high threshold (SVT or HVT) devices, they suffer from subthreshold leakage several orders of magnitude higher than that of an SVT device, leading to excessive leakage power. However, in the case of multiple serially connected cutoff transistors, the so-called "stack-effect" [88], [89] reduces this leakage to a tolerable level.

For FASTA arrays with large w (wide tags), the discharge delay may be excessive, even with ULVT transistors. To mitigate, the comparison logic of the FASTA row is segmented to limit the number of serially connected nMOS devices in the discharge path. Specifically, the w-bit row is divided into s comparison segments, such that each pull-down path has w/s transistors in series. In each such segment, the pull-down (ML_{NAND}) and the pull-up (ML_{NOR}) paths are connected to the output of the segment, as shown in Fig. 5(c). All segment outputs are propagated through an s-input CMOS OR gate. Such OR functionality can be further implemented by an OR logic tree.

These two methods (LVT devices and segmentation) can be combined to create an optimal trade-off between speed and leakage power (while also taking into consideration the area overhead of the segmentation option).

IV. VERY-MANY-WAY ASSOCIATIVE CACHE

A. MANY-WAY VS. FULLY-ASSOCIATIVE

The FASTA architecture, presented above, implements an energy-efficient associative memory, based on the CCAM bitcell, that mitigates the energy bottleneck of traditional CAM. While this solution is suitable for relatively small associative memories, the implementation of a very large cache (MBs or tens of MBs) would lead to excessive energy consumption, as compared to a conventional SRAM. This is because in a conventional SRAM, a hierarchy of many memory banks and subarrays would be employed, enabling the pre-decoding mechanism to disable large portions of the memory during access, thereby saving power. An associative memory, on the other hand, has to search the entire memory content, and therefore, all parts of the memory need to be accessed on every compare operation. For an application that requires large memories, such as a last-level cache, excessive energy consumption might be prohibitive. We propose resolving this by reducing the way-associativity to the level equal to the vertical dimension of a typical memory subarray (i.e., 256–1024-way associative), as presented below.

B. ALIASING-FREE CACHE

Before going into the proposed *very-many-way associative* cache architecture, let us revisit one of the main advantages of high associativity. In a virtual memory-based processor, a straightforward physically-indexed physically-tagged cache access requires translating the virtual tag into a physical tag by searching the TLB and only then accessing the cache with the full physical address. Since this type of access is slow, a popular solution is virtually-indexed physically-tagged (VIPT) cache access, which overlaps the TLB lookup with the cache access, thereby reducing the access latency. However, the size of a VIPT cache is limited by aliasing [90], [91]. Aliasing can occur when the index extends beyond the page offset, such that a part of the index requires virtual to physical translation. This limits the index size to (page_offset − block_offset), which in turn, limits the cache size to (page_size × associativity). So a processor with a 4 kB page size (page_offset = 12) and 64B block size (block_offset = 6) can have, at the most, 2^6=64 sets, and with 16-way associativity, the maximum aliasing-free cache size is 64 kB.

FASTA-enabled 256–1,024 way-associativity allows constructing a 1MB–4MB cache with 64 sets, such that the index would require no virtual to physical translation, thus both preserving the low latency and eliminating the aliasing. The VMWA approach detailed below supports such levels of associativity, such that large, low-latency, energy-efficient, and aliasing-free caches can be built.

C. VMWA ARCHITECTURE

The VMWA cache is implemented by several separate FASTA subarrays, each coupled with corresponding data SRAM modules that store the cache blocks (lines). Each of the FASTA subarrays is a tag array, with its hit$[i]$ signals enabling the wordlines of the adjacent data SRAM modules. Each individual FASTA is a **set**, and each FASTA row is a **way**. The index bits of the virtual address are used to select one of the FASTA arrays (sets), such that all the others can be idle during access to save energy.

Enabling the idling of all but one of the FASTA tag arrays (sets) is a very important advantage of the VMWA cache. In a conventional VIPT way-associative cache, all tag and all data SRAM arrays should be accessed in parallel to synchronize the arrival of physical tags from the tag arrays and the TLB. While this allows access time reduction, it also leads to significant energy consumption overhead. Such overhead is avoided in a VMWA cache because the working FASTA array (set) is selected by the virtual index, hence only one of many sets needs to be active.

Fig. 6 visualizes the implementation of an aliasing-free 1MB VIPT cache based on the proposed VMWA architecture. The cache is implemented with 64 sets, which are 256-row (i.e., 256-way) FASTA tag arrays and corresponding SRAM data modules. The VMWA address decoder activates one of the 64 sets. If the physical tag is found in the selected set, the corresponding data line is read out and passed through a 64-1 multiplexer as the cache output.

Revisiting the area overhead of the FASTA-based solution: the tag arrays are made up of 256 × 53 CCAM bits (including valid bit), while the adjacent cache line arrays are 256 × 512 SRAM bits, so there are approximately 10× more (small area) SRAM bits than (large area) CCAM bits. While this overhead is not negligible, it is not unique. For example, similar or larger area overheads are applied to enable Error Correction Code (ECC) protection in SRAM memories [92], [93] (12.5% for 64-data bits [93]) and fault-tolerance in Phase-Change Memory, PCM, (from 11.9% [94] to 12.5% [95]). We argue that such overhead might be afforded in microarchitectures, where performance is among the most important requirements.

Similarly to how the tag array is ECC protected in conventional cache, FASTA can also support ECC and provide tag protection by slightly adjusting the sense amplifier design, as suggested in [29] and silicon-proven by [28].

D. REPLACEMENT POLICY OVERHEAD

VMWA cache may support a variety of replacement policies, however, their overhead may be different from that of a conventional cache. The hardware overhead of Least Recently Used (LRU) is $\mathcal{O}(number\_of\_sets \times associativity \times \log_2(associativity))$. For a 256-way 8-set associative VMWA cache, such overhead is 1.56%, while for the same size conventional 256-set 8-way associative cache, the overhead is 0.58%. However, for replacement policies such as Dynamic Insertion Policy (DIP) [96], Static Re-Reference Interval Prediction (SRRIP), and Dynamic Re-Reference Interval Prediction (DRRIP) [97] or Not Recently Used (NRU), the hardware complexity is $\mathcal{O}(number\_of\_sets \times associativity)$. Therefore, the overhead should be very similar for the same size X-way Y-set associative VMWA cache and Y-way X-set associative conventional cache.

To conclude, a VMWA cache provides three major advantages:

1) It resolves the scalability limitation of VIPT caches and enables the construction of very large aliasing-free caches.
2) It enables a lower miss rate and performance improvement, as shown further in Section VI.
3) It saves energy by accessing only a single tag/data array during a search operation, as compared to an SRAM-based cache, which needs to access and compare the entire memory (all ways) in parallel.

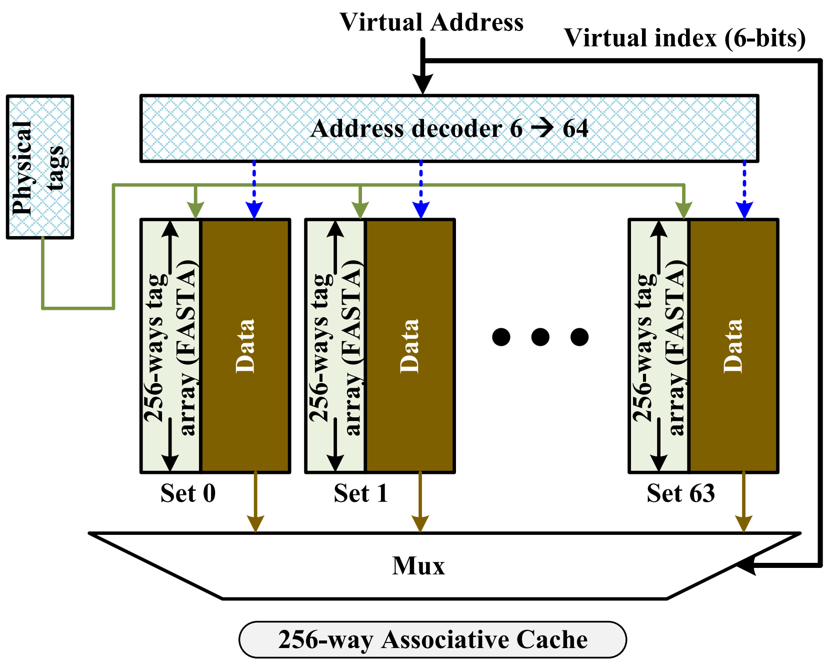


FIGURE 6. Top level view of a VMWA 256-way associative cache. Note: contrary to the convention, tag arrays are the sets, and memory rows are the ways.

V. EVALUATION AND RESULTS

A FASTA-based 256-way 8-set associative cache was custom-designed, including layout, using a commercial 16 nm FinFET PDK featuring a nominal supply voltage of 0.8 V. The results are obtained from post-layout simulations carried out within the Cadence Virtuoso environment, using the Spectre family of circuit simulators. Therefore, the results take into account the impact of the RC parasitics. For comparison, a baseline 8-way 256-set associative cache was also custom-designed using the same PDK under the same assumptions. The comparison between the 256-way 8-set associative cache and its baseline 8-way 256-set associative counterpart is presented below.

A. EVALUATION SETUP

The architecture of the baseline 8-way 256-set associative cache is shown in Fig. 7(a). The choice of 8-way set associative is based on reported associativity typically found in modern processors, such as the AMD Zen-3 [12] and Intel Skylake quad-core or Intel i7-6700 CPU [11], [13]. Such associativity requires 256 sets to provide a cache size of 128KB; however, note that when indexed virtually and tagged physically, this configuration is prone to aliasing.

The SRAM tag arrays are 256 rows × 51 columns (64-bit virtual address, less 6-bit block offset, less 8-bit virtual index, plus one valid bit), as illustrated in Fig. 7(b). The outputs of eight comparators are OR-ed to generate the global `hit` signal. The outputs of the sense amplifiers of the data SRAM arrays (eight ways) are fed into an 8-to-1 multiplexer that outputs the 64B cache line.

The structure of the 256-way 8-set FASTA-based VMWA cache is shown in Fig. 7(c). Note that, contrary to the convention of Fig. 7(a), FASTA arrays are *sets* while FASTA rows are *ways*. Each FASTA block (set) has 256 rows × 56 columns (64-bit virtual address, less 6-bit block offset, less 3-bit virtual index, plus one valid bit), as shown in Fig. 7(d). The three-bit index activates one of the eight sets (FASTA-based tag arrays), while the 55-bit physical tag is fed to the selected FASTA as the search (query) pattern. The Hit/Miss signal outputs from each set are OR-ed to provide the global `hit` signal, while the data outputs of each set are multiplexed to output the cache line.

B. ACCESS TIME AND ENERGY SIMULATION

To compare the proposed 256-way 8-set VMWA architecture with the baseline SRAM-based 8-way 256-set cache, we evaluate the access time and energy consumption. A FASTA row is divided into 14 segments of 4 cells. The evaluation is done by simulation, however below we detail the main delay and energy components. For the baseline cache, an access comprises the following (refer to Fig. 7(a) and (b) for the annotations):

1) **Index Decoding**: The time to decode the 8-bit index is denoted as t_{INDEX}=140 ps. This is essentially the delay of an 8:256 SRAM address decoder (we assume the 2 MSBs of the index are not translated).
2) **Tag and Data Readout**: The time to read out the 50-bit physical tag, valid bit, and 64B cache line, denoted as t_{DATA}=128 ps. This time includes the bitline discharge and sense amplifier delay of the 512+51=563-bit wide

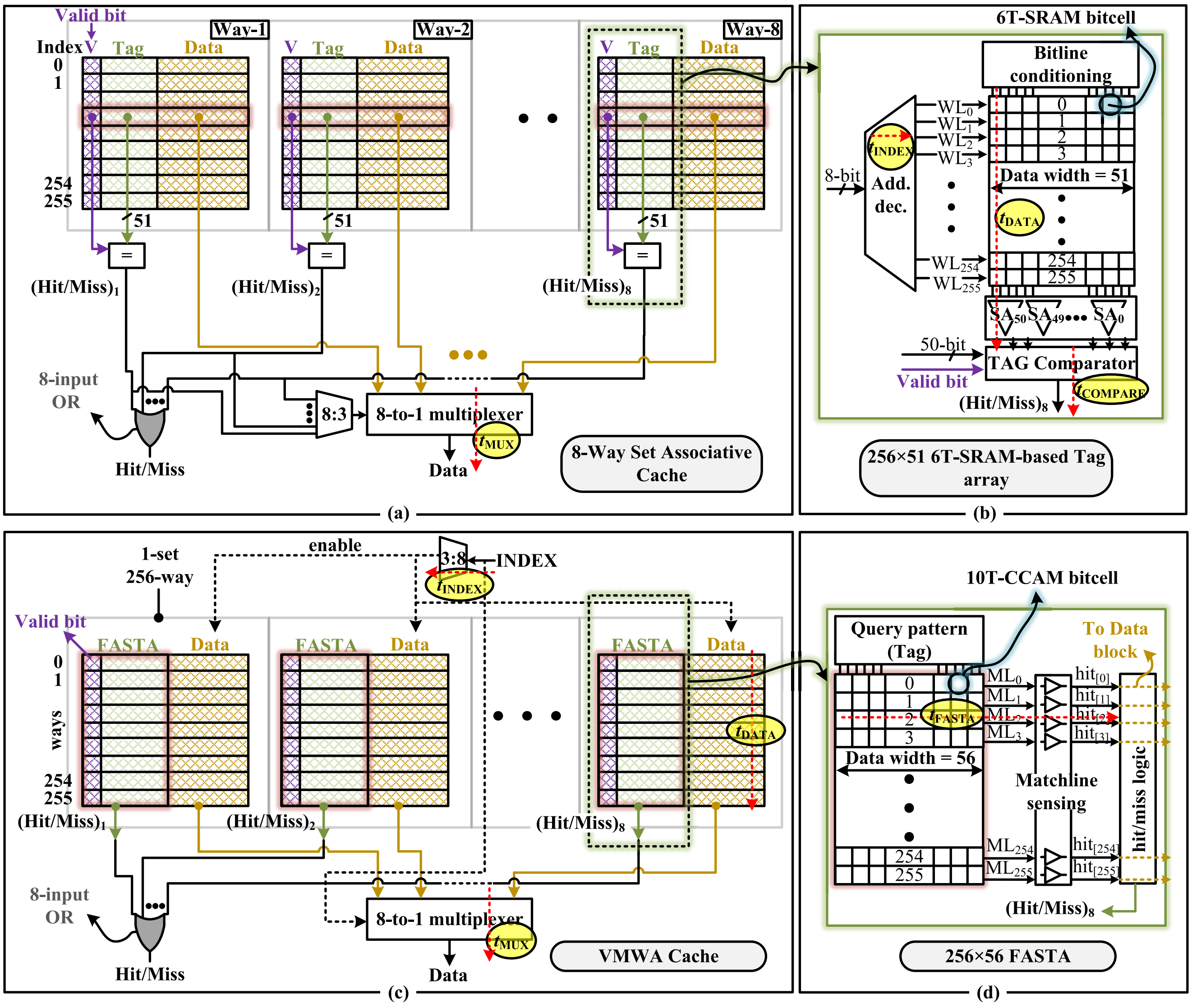


FIGURE 7. High-level architecture of an (a) 8-way 256-set associative cache, with the (b) standard 6T SRAM-based tag memory array. (c) very-many-way (256-way) 8-set associative cache, along with (d) Fully associative tag array (FASTA).

SRAM (way) and is applied in parallel to all 8 ways (to ensure the simultaneous readiness of the data and all physical tags).

This operation is energy-expensive since all 8 ways of the baseline cache are accessed in parallel, both for data and tags. An alternative is to access only the tags (while keeping the data parts of SRAM idle), and once the way is resolved, read the cache block from the relevant way only. This option is slower but less energy-demanding. For delay evaluation purposes, we select the more aggressive first (parallel access) alternative.

3) **Tag Comparison**: The time to compare the physical tags within a single way, denoted as $t_{COMPARE}$=163 ps.
4) **Multiplexing**: The time it takes to multiplex out the valid data from all 8 ways, denoted as t_{MUX}=25 ps. This operation takes the one-hot encoded Hit/Miss vector from the 8 ways and selects one to output from the cache.

Altogether, the baseline VIPT cache hit latency is calculated as:

$$t_{HIT} = t_{INDEX} + t_{DATA} + t_{COMPARE} + t_{MUX}. \quad (3)$$

For the 256-way 8-set VMWA cache, the hit latency comprises the following components (refer to Fig. 7(c) and (d) for the annotated timing):

1) **Index Decoding**: The time to decode the 3-bit index to select one of eight 256-way FASTA sets, while disabling the others. This time is denoted as t_{INDEX}=10 ps.
2) **FASTA Search**: The time to compare the physical tag within the selected set (FASTA). This delay is denoted t_{FASTA}=166 ps. Note that this delay ends with the assertion of the matched row's hit[i] signal, without

the need to wait for the global Hit/Miss signal of the set.
3) **Data Readout**: The time to readout the 64B cache line, initiated once the matched row's hit[i] signal enables the assertion of a wordline in the SRAM array. This delay is denoted t_{DATA}=128 ps.
4) **Multiplexing**: The time it takes to multiplex out the valid data from one of the 8 sets, denoted as t_{MUX}=25 ps. Note that this delay is slightly shorter than the t_{MUX} of the baseline cache, as the multiplexer is directly controlled by the 3-bit index (which requires no translation).

Altogether, the VMWA hit latency is calculated as:

$$t_{\text{HIT}} = t_{\text{INDEX}} + t_{\text{FASTA}} + t_{\text{DATA}} + t_{\text{MUX}}. \quad (4)$$

For the energy consumption comparison, the baseline 8-way 256-set associative cache includes the following components:
1) **Tag Readout**: The energy to readout a 50-bit tag and valid bit from a 256-row SRAM tag array, denoted as E_{TAG}= 0.71 pJ. This figure also includes the energy of the tag comparator and the Hit/Miss generation circuitry.
2) **Cache Line Readout**: The energy to read out a 64B cache line from the data SRAM array, denoted E_{DATA}= 3.88 pJ. This includes precharging the bitlines, charging a single wordline, and sensing the bitlines. Both tag and cache line readouts are performed in parallel in all eight ways.
3) **Multiplexing**: The energy required to multiplex out the valid data from eight ways, denoted as E_{MUX}= 0.19 pJ.

Altogether, the baseline 8-way 256-set associative cache energy is calculated as:

$$E_{\text{HIT}} = (E_{\text{TAG}} + E_{\text{DATA}}) \cdot 8 + E_{\text{MUX}}. \quad (5)$$

The overall energy consumption of the 256-way 8-set VMWA cache includes:
1) **Tag Search**: The energy to apply a 56-bit search within a 256-row FASTA, denoted as E_{SEARCH} = 0.43 pJ. This includes decoding the index to select the set, driving the searchlines, discharging and sensing a single matchline, and generating the global Hit/Miss signal of the set. Since only one set is selected and the others are disabled, the search energy applies to a single FASTA array.
2) **Cache Line Readout**: The energy to read out a 64B cache line from the FASTA-coupled SRAM array, denoted E_{DATA} = 3.88 pJ. This includes precharging the bitlines, charging a single wordline, and sensing the bitlines, but does not include address decoding, as the wordline is selected based on the hit[i] outputs of the active set.
3) **Multiplexing**: The energy required to multiplex out the data from the selected set to the cache output, denoted E_{MUX} = 0.19 pJ.

Altogether, the VMWA energy is calculated as:

$$E_{\text{HIT}} = E_{\text{SEARCH}} + E_{\text{DATA}} + E_{\text{MUX}}. \quad (6)$$

Note that the energy during a cache miss is slightly lower since no ML is discharged; however, this small deviation is omitted from the results.

C. POST-LAYOUT ACCESS TIME, ENERGY AND AREA RESULTS

Table 2 presents the results of the post-layout comparison between the proposed 256-way 8-set VMWA cache and the baseline SRAM-based 8-way 256-set design for any microarchitectural application that requires associative access, as presented in Fig. 7. The energy consumption and delay figures are calculated at a supply voltage of 0.8 V and during a search operation resulting in a `hit`. All memory peripherals are taken into account in the calculations.[1]

a: ACCESS TIME

FASTA-based 256-way 8-set cache provides 28% faster access when compared with an SRAM-based 8-way 256-set associative cache. The access time advantage is due to the optimized division of the match line (into comparison segments), and the number of fins and low threshold voltage devices in comparison transistors.

b: ENERGY CONSUMPTION

The energy consumption of a single FASTA array is 0.43 pJ (Table 2). This is 40% lower than the energy consumption of the building block of the SRAM-based tag array, which is a 256-row SRAM bank. However, during a search operation, all ways (i.e., individual Tag/Data arrays) of the baseline cache are accessed in parallel, while the VMWA activates only a single set (i.e., only a single FASTA+data array). This leads to energy savings of 88% for the VMWA as compared to the baseline, when considering all of the energy components of equations (5) and (6). Leakage power consumption is also reported in Table 2 showing that the baseline consumes 70% lower standby power than VMWA.

Fig. 8 shows the breakdown of the energy consumption during a `hit` case for a single way of the SRAM-based 8-way 256-set baseline cache and a single set of the 256-way 8-set VMWA proposed cache. The SRAM data array and the multiplexing present about the same energy consumption for both designs. The main difference is in the SRAM tag and FASTA array. As mentioned above, FASTA achieves 40% less energy consumption as compared to the SRAM baseline. This energy gain is mainly due to the lack of address decoders and differential sense amplifiers, among others.

Fig. 9 shows the 'pie-to-pie' energy consumption breakdown for a `hit` case for the proposed and baseline caches. SRAM tag and FASTA arrays consume about 15% and 9.5% of the total energy respectively, during a `hit`. Out of

[1]I/O peripherals (e.g., BL drivers, buses, etc.), as well as the control circuitry, are not taken into account.

TABLE 2. Post-layout comparison results. 256-way 8-set associative VMWA cache vs. 8-way 256-set baseline cache. Reported energy and access time results are obtained for a Hit case during a search operation.

Figures of merit	8-way 256-set associative (Baseline)	256-way 8-set associative (VMWA)
Supply voltage (V)	0.8	0.8
Total cache access time (ps)	455	328
Energy per single tag array (pJ)	0.71	0.43
Total energy per cache access (pJ)	36.9	4.50
Standby Leakage Power (µW)	0.90	3.06
Area (Normalized to baseline)	1×	1.09×
Aliasing	YES	NO

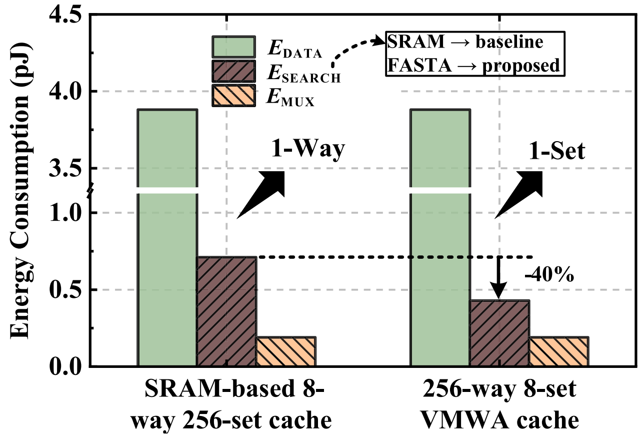


FIGURE 8. Breakdown of the energy consumption during a Hit case for an SRAM-based 8-way 256-set cache and the proposed 256-way 8-set VMWA cache. The energy figures are referred to as 1-way and 1-set of the baseline and proposed designs, respectively.

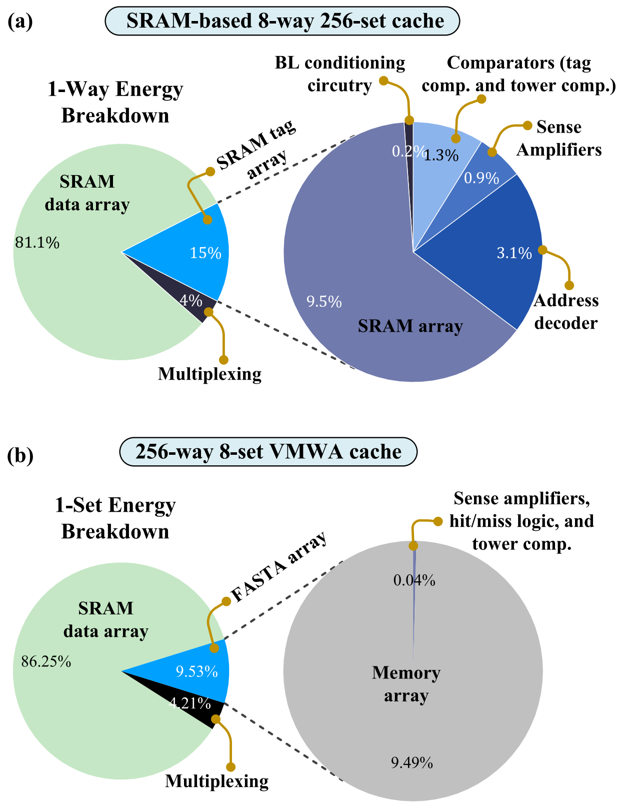


FIGURE 9. 'Pie-to-pie' chart energy breakdown for (a) the SRAM-based 8-way 256-set cache and (b) the proposed 256-way 8-set VMWA cache. The energy breakdown of the SRAM tag and FASTA arrays are highlighted in the right charts.

15% SRAM tag consumption, about 9.5% is consumed by the SRAM array, while the remaining energy consumption is distributed mainly between BL conditioning circuitry, comparison circuitry, sense amplifiers, and address decoders. Out of 9.5% energy consumption of FASTA array, its memory core consumes most of the energy, while sensing and hit/miss logic consumes only about 0.04%. This is mainly because inverters are used as sense amplifiers in each row, and only one row is active/toggled during a hit, thereby reducing the overall energy consumption per operation.

c: AREA OVERHEAD

The primary contributor to area overhead is the memory array. The 10-transistor-CCAM cell is 1.86× larger than the 6T-SRAM baseline cell. Additional area is needed for the per-row segmented NAND/OR-tree, and the Hit/Miss logic. However, in most FASTA applications, each tag array is coupled with an SRAM data array. While these two components feature the same number of rows, a typical cache tag array has approximately 10× fewer columns, e.g., 56-bit FASTA vs. 512-bit cache line in VMWA sets. Fig. 10 shows the relative area footprint of the baseline and proposed cache designs, where the data array is (on average) about 85% of the total area. The overall area footprint of a single baseline way and VMWA set are about 0.075 mm^2 and 0.082 mm^2, respectively. Therefore, the total area increase for the standalone FASTA and VMWA implementations is limited to 9%.

It is noteworthy that there is negligible hardware overhead if the proposed FASTA is compared with a fully associative tag array built with conventional NAND CAM cells. The proposed 10T-CCAM cell features an extra transistor (M_{NOR}) as compared to the conventional 9T-NAND CAM. Despite this, the layout footprint of both CCAM and NAND CAM cells remains nearly identical. This equivalency is primarily attributed to the alignment of the cell pitch (or cell height) with the CAM row pitch, which corresponds to the standard cell pitch. In this way, the vacant space within the NAND CAM cell macro can be filled with the M_{NOR} pMOS transistor. Consequently, the M_{NOR} area overhead in the CCAM cell is negligible or non-existent.

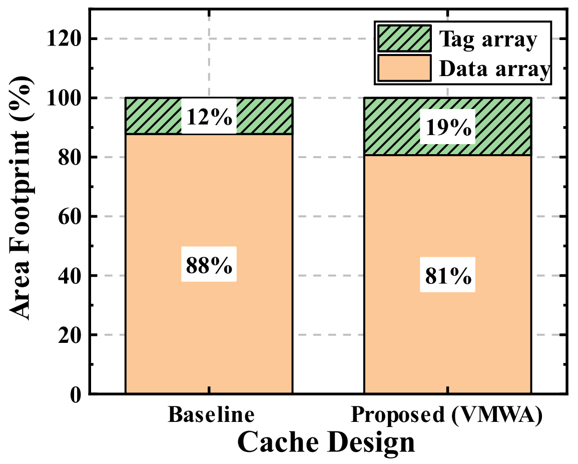


FIGURE 10. Relative area footprint of the baseline and proposed cache designs. The total area of a single way (for baseline) and set (for VMWA) are about 0.075 mm^2 and 0.082 mm^2, respectively.

d: DESIGN TRADE-OFFS

When compared to conventional SRAM-based associative memory, the proposed FASTA-based VMWA cache features the following advantages:

- Cost-effective very high associativity
- Aliasing-free large-scale cache design
- Improved search access time
- Lower energy consumption

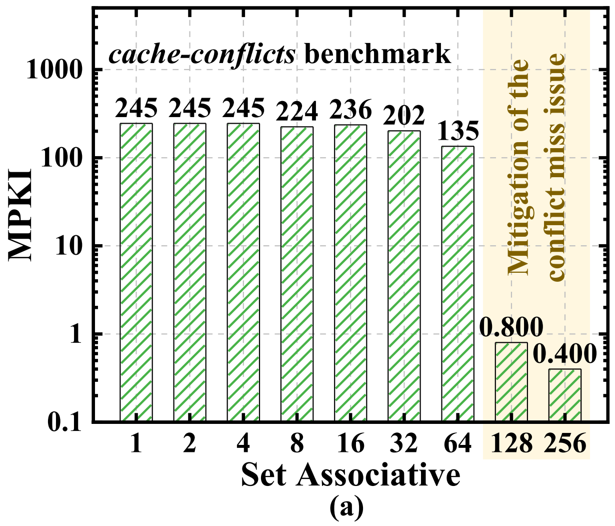
Overall, the above benefits are obtained at a moderate area overhead of 9%. Such overhead is not extraordinary in contemporary memory designs. For example, a 12.5% overhead is typical in ECC-protected memories.

e: FASTA ARRAY SIZE CONSIDERATIONS

- FASTA associativity increases linearly with its height (number of rows). Increasing FASTA height allows efficient associativity scaling, infeasible in SRAM-based design. However, increasing the array height may adversely affect the FASTA performance and energy. Therefore a tradeoff that balances associativity, performance, and energy efficiency exists.
- FASTA width is defined by the tag size, which in turn is determined by the memory address space size (64-bit). Therefore changing the width is not a design choice.
- Number of FASTA arrays (number of sets) is determined by the desired cache size given the constant FASTA height.

VI. VMWA CACHE PERFORMANCE EVALUATION

We focus the performance evaluation on the FASTA-based VMWA cache presented in Section 4. Other associative structures of computer microarchitecture, such as TLB, BTB, etc. are not evaluated since popular public domain architecture simulators (e.g., GEM5 [98]) do not easily support user-defined associativity for any of these structures but cache. Sniper [27], another commonly used CPU simulator,



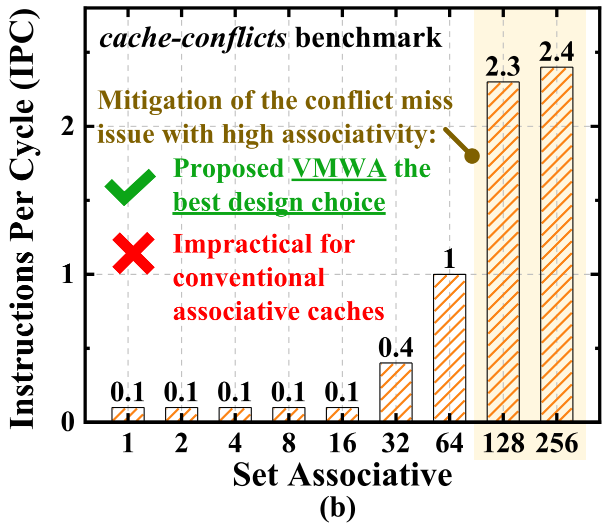


FIGURE 11. (a) MPKI and (b) IPC as a function of associativity for *cache-conflicts* benchmark. The experiment emphasizes the practical implications of the proposed VMWA design on cache performance optimization.

supports user-defined associativity only for cache and small TLBs.

The VMWA cache performance is evaluated using Sniper version 7.4 [99], which is arguably the fastest, yet sufficiently accurate tool for system-level analysis. Sniper supports variable associativity in three-level cache memory hierarchy, including L1, L2, and Last-Level L3 caches.

The figures of merit in our performance evaluation are Miss Per Kilo Instructions (MPKI), Instructions Per Cycle (IPC), and execution time. We study their behavior as a function of the associativity and replacement policy. While the relations between miss rate and cache size and associativity have been extensively investigated in the past [100], our evaluation differs in that we extend it to a very high associativity level (256).

We performed three distinct experiments: 'cache-conflicts', 'cache associativity', and 'energy consumption'. Further details on each experiment are provided below.

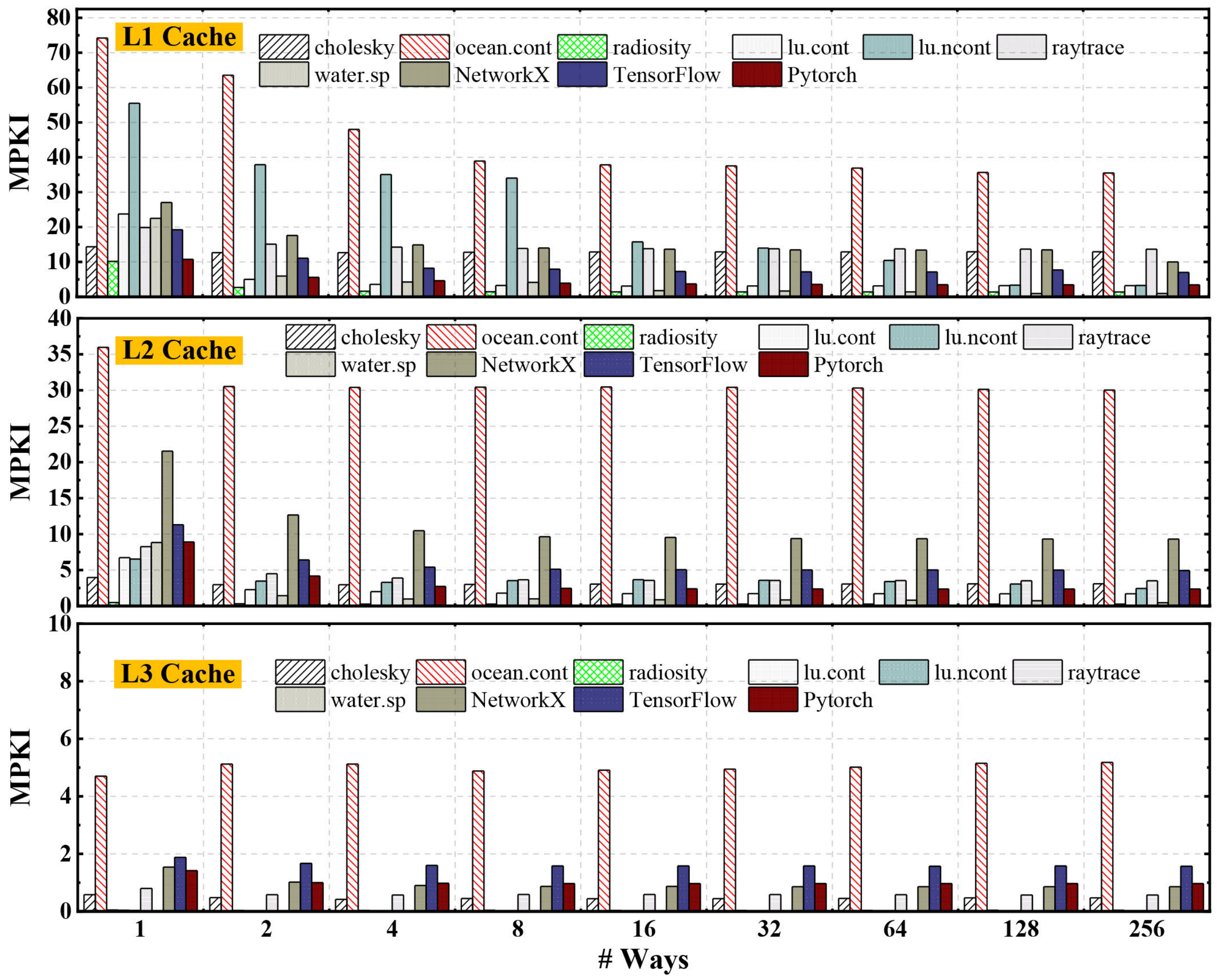


FIGURE 12. Misses Per Kilo Instructions (MPKI) as a function of the associativity for L1, L2, and L3 caches. The legend shows the workloads: cholesky, ocean.cont, radiosity, lu.cont, lu.ncont, raytrace, water.sp, NetworkX, Tensorflow, Pytorch.

A. CACHE-CONFLICTS

The purpose of this first experiment is to stress and highlight the adverse effect of conflict misses on MPKI and IPC. We use the *cache-conflicts* benchmark from hardware-effects suite [101] which is a part of the recent DAMOV benchmark [102].

The cache-conflicts benchmark stresses the cache associativity by periodically writing to the same set, causing quick thrashing in low-associativity caches. For example, in a CPU with 32 kB 8-way associative L1 cache and 64-byte cache blocks, a conflict miss-caused cache thrashing occurs when the CPU continuously writes to memory addresses with the interval of 4096. The simulator was set in *Gainestown* configuration, with one logical and physical core and with 32 kB L1 cache and block size of 64 bytes. We run the cache-conflicts benchmark while varying the L1 associativity. The memory access interval is set to vary according to the associativity. The overall number of writes is set such that the capacity miss component remains negligible, to allow the focus on conflict miss. The resulting MPKI and IPC are presented in Fig. 11(a) and (b) respectively. The continuous write access to the same set causes a significant MPKI and the resulting drop in low associativity configurations. The increase of associativity to 128 and 256 resolves this problem almost entirely (the small MPKI is due to capacity miss), allowing a significant improvement in IPC.

B. CACHE ASSOCIATIVITY

This second experiment is devised to study the typical impact of high associativity on MPKI, IPC, and execution time using more general-purpose workloads. This experiment involves several workloads from the splash2 [103] benchmark as well as the NetworkX (running large-scale graph traversal), Pytorch, and Tensorflow (processing large-scale tensor datasets). The splash2 workloads are specified in the legends of Figs. 12–14.

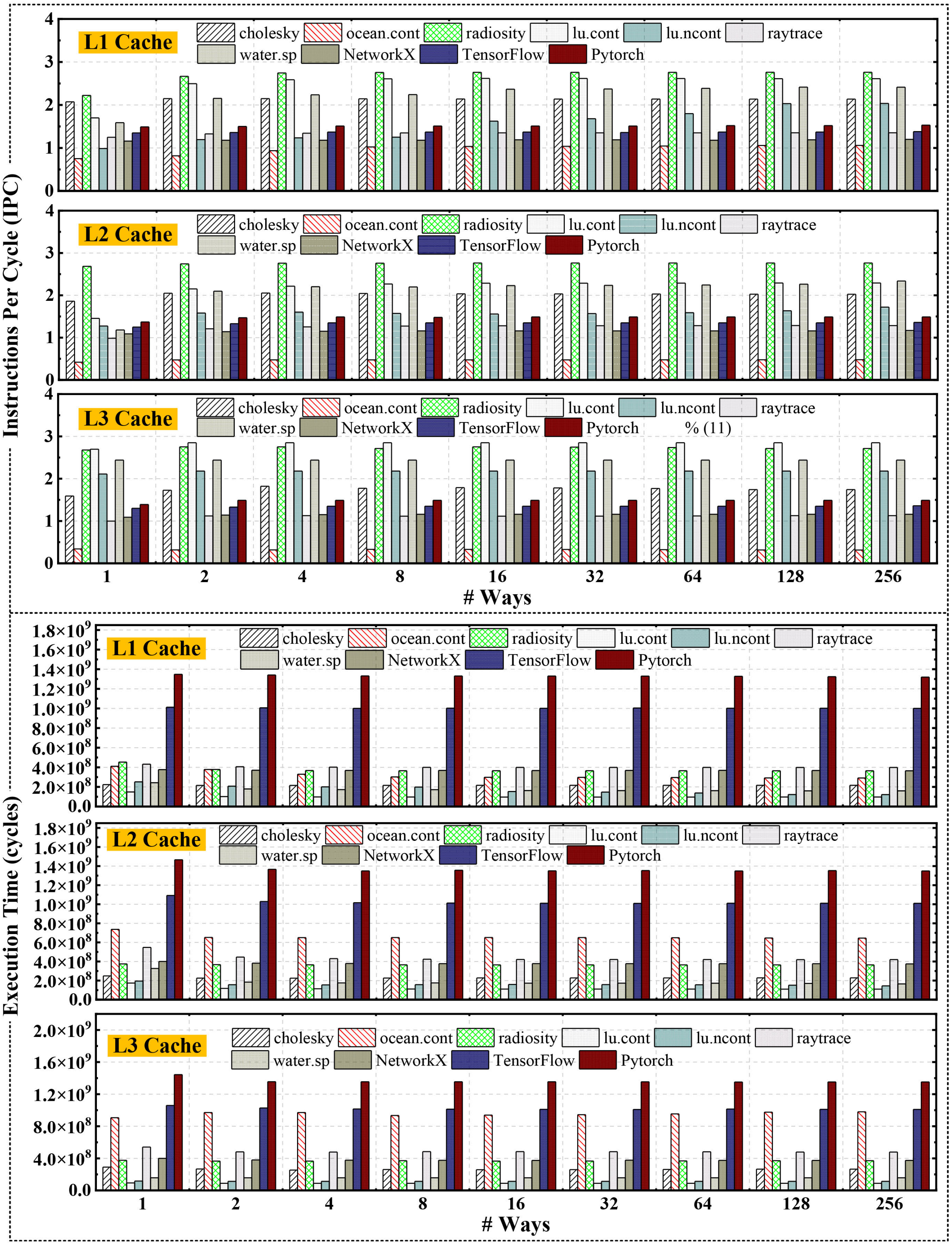


FIGURE 13. Instructions per cycle (IPC) and Execution time as function of associativity for L1, L2, and L3 caches. The legend shows the workloads: cholesky, ocean.cont, radiosity, lu.cont, lu.ncont, raytrace, water.sp, NetworkX, Tensorflow, Pytorch.

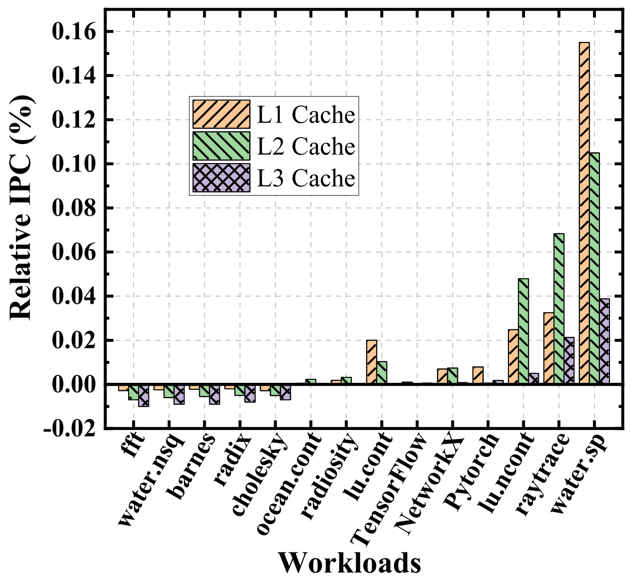


FIGURE 14. S-curve for VMWA relative to 16-way associative baseline for L1, L2 and L3 for several splash2 workloads, NetworkX, Pytorch and Tensorflow.

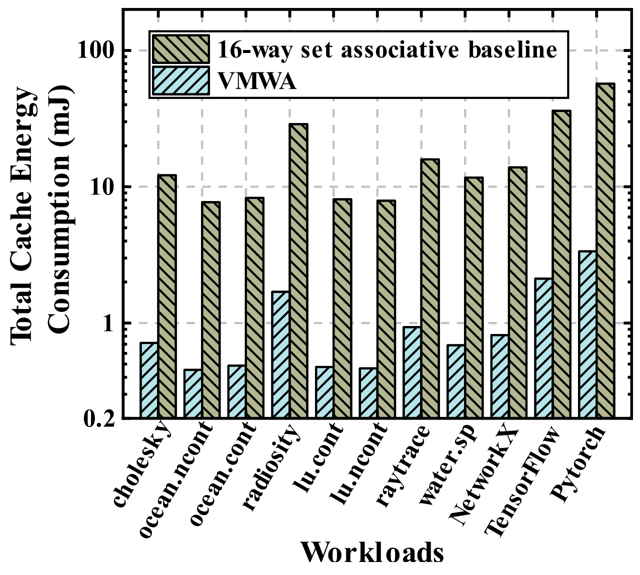


FIGURE 15. Total cache energy consumption for different workloads.

In each run, we vary the cache associativity of one of the cache hierarchy levels (L1, L2, or L3) while the rest are set to Sniper default (32 kB, 256 kB, and 8 MB for L1, L2, and L3, respectively). The LRU and random replacement policies were tested. At high associativity levels, random performs slightly worse than LRU hence we show here only the random replacement results.

The MPKI, IPC, and execution time results are presented in Fig. 12 and Fig. 13, respectively. The following are the main results of our analysis:

- The VMWA cache exhibits lower MPKI in the majority of the benchmarks. Specifically, the average MPKI improvement achieved by the 256-way associative L1 compared to 16-way associative L1 is 17.3%. The average IPC improvement is 1.6%.
- The average MPKI improvement achieved by the 256-way associative L2 compared to 16-way associative L2 is 11.5%. The average IPC improvement is 1.4%.
- The average MPKI improvement achieved by the 256-way associative L3 compared to 16-way associative L3 is 1.2% which translates to 0.2% IPC improvement on average.
- The s-curve summarizing the IPC gains of the VMWA cache compared to 16-way associative L1, L2, and L3 cache is presented in Fig. 14.

C. ENERGY CONSUMPTION

This third experiment aims to functionally evaluate energy consumption. Fig. 15 shows the total cache energy consumption for different workloads and for the proposed VMWA and a 16-way set associative cache. The VMWA presents about 94% lower energy consumption (on average) compared to the 16-way set associative cache. This energy gain is achieved mainly because during a search operation, the VMWA activates a single set while the baseline has to access all 16 ways in parallel.

VII. CONCLUSION

Associative access is widely employed across modern computer microarchitecture. In this work, we revisit the convention surrounding associative memory (CAM) and propose a set of solutions that enable the reintroduction of CAM to computer microarchitecture. Specifically, we focus on developing a fully associative tag array (FASTA), which enables full associativity, while providing better access time and lower energy consumption compared with a way-associative SRAM-based solution. We also introduce, design, and evaluate a FASTA-based Very-Many-Way Associative (VMWA) cache, which enables very high associativity, scalability, and aliasing-free large-scale cache implementation. The benefits of VMWA cache associativity, scalability, and reduced energy consumption come with an area overhead of around 9%. We argue that such an overhead is quite typical in contemporary memory designs and hence might be afforded in applications where high associativity plays a critical role in enabling performance improvement.

Performance evaluation shows that for many benchmarks and cache configurations (sizes and levels), VMWA achieves lower MPKI and shorter execution time compared to 16-way associative cache. Looking beyond cache, the VMWA architecture may deliver significant functional upside to a variety of associative access memory structures widely deployed across computer microarchitecture.

APPENDIX A GLOSSARY

Acronyms used in this manuscript are given in Table 3.

TABLE 3. Glossary.

Acronyms	
CAM	Content-Addressable Memory
TLB	Translation Lookahead Buffer
BTB	Branch Target Buffers
VIPT	Virtually-Indexed Physically-Tagged
FASTA	Fully-Associative Tag Array
SRAM	Static Random-Access Memory
DRAM	Dynamic Random-Access Memory
RRAM	Resistive Random-Access Memory
CNTFET	Carbon Nanotube Field-Effect Transistor
TG-CNTFET	Top-Gated CNTFET
GAA-CNTFET	Gate-All-Around CNTFET
SG-CNTFET	Shorted Gate CNTFET
IG-CNTFET	Independent Gate CNTFET
FeRAM	Ferroelectric Random Access Memory
QNRO	Quasi-Nondestructive ReadOut
STT-MRAM	Spin-Transfer Torque Magnetic RAM
SOT-MRAM	Spin-Orbit Torque Magnetic RAM
DWM	Domain-Wall Memory
CCAM	Complementary Content-Addressable Memory
VMWA	Very-Many-Way Associative
WL	Wordline
SL	Searchline
BL	Bitline
SDRs	Search Data Registers
ML	Matchline
D	Data bit
$\overline{\text{pre}}$	Precharge
MLSA	Matchline Sense Amplifier
LVT	Low Threshold Voltage
ULVT	Ultra-Low Threshold Voltage
SVT	Standard Threshold Voltage
HVT	High Threshold Voltage
ECC	Error Correction Code
LRU	Least Recently Used
DIP	Dynamic Insertion Policy
SRRIP	Static Re-Reference Interval Prediction
DRRIP	Dynamic Re-Reference Interval Prediction
NRU	Not Recently Used
MPKI	Miss Per Kilo Instructions
IPC	Instructions Per Cycle

REFERENCES

[1] Q. Ge, Y. Yarom, D. Cock, and G. Heiser, "A survey of microarchitectural timing attacks and countermeasures on contemporary hardware," *J. Cryptograph. Eng.*, vol. 8, no. 1, pp. 1–27, Apr. 2018, doi: 10.1007/s13389-016-0141-6.

[2] M. Ewais and P. Chow, "Disaggregated memory in the datacenter: A survey," *IEEE Access*, vol. 11, pp. 20688–20712, 2023, doi: 10.1109/ACCESS.2023.3250407.

[3] S. Liu, P. Reviriego, and F. Lombardi, "Protection of associative memories using combined tag and data parity (CTDP)," *IEEE Trans. Nanotechnol.*, vol. 20, pp. 1–9, 2021, doi: 10.1109/TNANO.2020.3042114.

[4] Y. Chen, J. Mu, H. Kim, L. Lu, and T. T. Kim, "BP-SCIM: A reconfigurable 8T SRAM macro for bit-parallel searching and computing in-memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 5, pp. 2016–2027, May 2023, doi: 10.1109/TCSI.2023.3240303.

[5] A. Basu, J. Gandhi, J. Chang, M. D. Hill, and M. M. Swift, "Efficient virtual memory for big memory servers," *ACM SIGARCH Comput. Archit. News*, vol. 41, no. 3, pp. 237–248, Jun. 2013, doi: 10.1145/2508148.2485943.

[6] G. Gracioli, A. Alhammad, R. Mancuso, A. A. Fröhlich, and R. Pellizzoni, "A survey on cache management mechanisms for real-time embedded systems," *ACM Comput. Surveys*, vol. 48, no. 2, pp. 1–36, Nov. 2015, doi: 10.1145/2830555.

[7] W. Zhao, D. Feng, W. Tong, J. Liu, Z. Chen, B. Wu, and C. Wang, "APPcache+: An STT-MRAM based approximate cache system with low power and long lifetime," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 11, p. 1, Apr. 2023, doi: 10.1109/TCAD.2023.3267713.

[8] J. Nian, Z. Liang, H. Liu, and M. Yang, "An efficient fault-tolerant protection method for l0 BTB," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 3, pp. 1284–1297, Mar. 2023, doi: 10.1109/TCSI.2022.3228774.

[9] E. Cheshmikhani, H. Farbeh, and H. Asadi, "3RSeT: Read disturbance rate reduction in STT-MRAM caches by selective tag comparison," *IEEE Trans. Comput.*, vol. 71, no. 6, pp. 1305–1319, Jun. 2022, doi: 10.1109/TC.2021.3082004.

[10] G. Modi, A. Bagchi, N. Jindal, A. Mandal, and P. R. Panda, "CABARRE: Request response arbitration for shared cache management," *ACM Trans. Embedded Comput. Syst.*, vol. 22, no. 5s, pp. 1–24, 2023, doi: 10.1145/3608096.

[11] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx, "Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution," in *Proc. 27th USENIX Secur. Symp.*, 2018, pp. 991–1008. [Online]. Available: https://dl.acm.org/doi/abs/10.5555/3277203.3277277

[12] M. Evers, L. Barnes, and M. Clark, "The AMD next-generation-"Zen 3" core," *IEEE Micro*, vol. 42, no. 3, pp. 7–12, May 2022, doi: 10.1109/MM.2022.3152788.

[13] J. Kim, H. Jang, H. Lee, S. Lee, and J. Kim, "UC-check: Characterizing micro-operation caches in ×86 processors and implications in security and performance," in *Proc. MICRO-54: 54th Annu. IEEE/ACM Int. Symp. Microarchit.*, Oct. 2021, pp. 550–564, doi: 10.1145/3466752.3480079.

[14] T. Zheng, H. Zhu, and M. Erez, "SIPT: Speculatively indexed, physically tagged caches," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2018, pp. 118–130, doi: 10.1109/HPCA.2018.00020.

[15] S. Vakili and A. Zarei, "DSCAM: Latency-guaranteed and high-capacity content-addressable memory on FPGAs," *IEEE Embedded Syst. Lett.*, early access, p. 1, 2023, doi: 10.1109/LES.2023.3334288.

[16] H. Wong, V. Betz, and J. Rose, "Comparing FPGA vs custom CMOS and the impact on processor microarchitecture," in *Proc. 19th ACM/SIGDA Int. Symp. Field Program. Gate Arrays*, Feb. 2011, pp. 5–14, doi: 10.1145/1950413.1950419.

[17] P. Papavramidou and M. Nicolaidis, "Reducing power dissipation in memory repair for high fault rates," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 31, no. 12, pp. 2112–2125, Dec. 2023, doi: 10.1109/TVLSI.2020.3046605.

[18] R. Nadkarni, I. Arsovski, R. Wistort, and V. Chickanosky, "Improved match-line test and repair methodology including power-supply noise testing for content-addressable memories," in *Proc. IEEE Int. Test Conf.*, Oct. 2006, pp. 1–9, doi: 10.1109/TEST.2006.297700.

[19] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," *IEEE J. Solid-State Circuits*, vol. 41, no. 3, pp. 712–727, Mar. 2006, doi: 10.1109/JSSC.2005.864128.

[20] W. Chen, X. Yang, and F. Z. Wang, "Memristor content addressable memory," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NANOARCH)*, Jul. 2014, pp. 83–87, doi: 10.1109/NANOARCH.2014.6880476.

[21] M. Somasundaram, "Memory and power efficient mechanism for fast table lookup," U.S. Patent 7 162 572, Jan. 9, 2007.

[22] T. V. Mahendra, S. W. Hussain, S. Mishra, and A. Dandapat, "Energy-efficient precharge-free ternary content addressable memory (TCAM) for high search rate applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 7, pp. 2345–2357, Jul. 2020, doi: 10.1109/TCSI.2020.2978295.

[23] V. Chaudhary and L. T. Clark, "Low-power high-performance NAND match line content addressable memories," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 8, pp. 895–905, Aug. 2006, doi: 10.1109/TVLSI.2006.878476.

[24] S. W. Hussain, T. V. Mahendra, S. Mishra, and A. Dandapat, "Match-line control unit for power and delay reduction in hybrid CAM," *IET Circuits, Devices Syst.*, vol. 15, no. 3, pp. 272–283, May 2021, doi: 10.1049/cds2.12024.

[25] K. Mohammad, A. Qaroush, M. Washha, and B. Mohammad, "Low-power content addressable memory (CAM) array for mobile devices," *Microelectron. J.*, vol. 67, pp. 10–18, Sep. 2017, doi: 10.1016/j.mejo.2017.07.001.

[26] E. Garzón, L. Yavits, A. Teman, and M. Lanuzza, "Approximate content-addressable memories: A review," *Chips*, vol. 2, no. 2, pp. 70–82, Mar. 2023, doi: 10.3390/chips2020005.

[27] T. E. Carlson, W. Heirman, and L. Eeckhout, ''Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation,'' in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, Nov. 2011, pp. 1–12, doi: 10.1145/2063384.2063454.

[28] E. Garzón, R. Golman, M. Lanuzza, A. Teman, and L. Yavits, ''A low-complexity sensing scheme for approximate matching content-addressable memory,'' *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 10, p. 1, Jun. 2023, doi: 10.1109/TCSII.2023.3286257.

[29] K. Pagiamtzis, N. Azizi, and F. Najm, ''A soft-error tolerant content-addressable memory (CAM) using an error-correcting-Match scheme,'' in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2006, pp. 301–304, doi: 10.1109/CICC.2006.320887.

[30] Y. Li, J. Tang, B. Gao, J. Yao, A. Fan, B. Yan, Y. Yang, Y. Xi, Y. Li, J. Li, W. Sun, Y. Du, Z. Liu, Q. Zhang, S. Qiu, Q. Li, H. Qian, and H. Wu, ''Monolithic three-dimensional integration of RRAM-based hybrid memory architecture for one-shot learning,'' *Nature Commun.*, vol. 14, no. 1, pp. 7140–7023, Nov. 2023, doi: 10.1038/s41467-023-42981-1.

[31] S. S. Ganie and A. Singh, ''Gate all around CNTFET based ternary content addressable memory,'' *ECS J. Solid State Sci. Technol.*, vol. 11, no. 6, Jun. 2022, Art. no. 061006, doi: 10.1149/2162-8777/ac77bc.

[32] V. Prasad and D. Das, ''Design of ternary content-addressable memory using CNTFET,'' in *Proc. Int. Conf. Frontiers Comput. Syst. (COMSYS)*. Cham, Switzerland: Springer, 2020, pp. 853–858, doi: 10.1007/978-981-15-7834-2_80.

[33] P. Kumar, K. A. Kumar Jain, A. Ram P S, and M. S. Sunita, ''Low power and high-performance ternary content addressable memory using carbon nanotube FET,'' in *Proc. IEEE 8th Int. Conf. Converg. Technol. (I2CT)*, Apr. 2023, pp. 1–6, doi: 10.1109/I2CT57861.2023.10126270.

[34] A. Gangadhar and K. Babulu, ''Design of low-power and high-speed CNTFET-based TCAM cell for future generation networks,'' *J. Supercomput.*, vol. 77, no. 9, pp. 10012–10022, Sep. 2021, doi: 10.1007/s11227-021-03657-z.

[35] M. Beste, S. Kiamehr, and M. B. Tahoori, ''Physical design of CNTFET-based circuits for yield improvement,'' in *Proc. IEEE 12th Int. New Circuits Syst. Conf. (NEWCAS)*, Jun. 2014, pp. 65–68, doi: 10.1109/NEWCAS.2014.6933986.

[36] J. Yoo, H. Song, H. Lee, S. Lim, S. Kim, K. Heo, and H. Bae, ''Recent research for HZO-based ferroelectric memory towards in-memory computing applications,'' *Electronics*, vol. 12, no. 10, p. 2297, 2023, doi: 10.3390/electronics12102297.

[37] K. Ni, Y. Xiao, S. Deng, and V. Narayanan, ''Computational associative memory powered by ferroelectric memory,'' in *Proc. Device Res. Conf. (DRC)*, Jun. 2023, pp. 1–2, doi: 10.1109/DRC58590.2023.10187048.

[38] M. Tarkov, F. Tikhonenko, V. Popov, V. Antonov, A. Miakonkikh, and K. Rudenko, ''Ferroelectric devices for content-addressable memory,'' *Nanomaterials*, vol. 12, no. 24, p. 4488, Dec. 2022, doi: 10.3390/nano12244488.

[39] L. Liu, A. F. Laguna, R. Rajaei, M. M. Sharifi, A. Kazemi, X. Yin, M. Niemier, and X. S. Hu, ''A reconfigurable FeFET content addressable memory for multi-state Hamming distance,'' *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 6, pp. 1–14, Mar. 2023, doi: 10.1109/TCSI.2023.3259940.

[40] K. Ni, X. Yin, A. F. Laguna, S. Joshi, S. Dünkel, M. Trentzsch, J. Müller, S. Beyer, M. Niemier, X. S. Hu, and S. Datta, ''Ferroelectric ternary content-addressable memory for one-shot learning,'' *Nature Electron.*, vol. 2, no. 11, pp. 521–529, Nov. 2019, doi: 10.1038/s41928-019-0321-3.

[41] O. Bekdache, H. N. Eddine, M. Al Tawil, R. Kanj, M. E. Fouda, and A. M. Eltawil, ''Scalable complementary FeFET CAM design,'' in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2023, pp. 1–5, doi: 10.1109/ISCAS46773.2023.10181788.

[42] C. Li, F. Müller, T. Ali, R. Olivo, M. Imani, S. Deng, C. Zhuo, T. Kämpfe, X. Yin, and K. Ni, ''A scalable design of multi-bit ferroelectric content addressable memory for data-centric computing,'' in *IEDM Tech. Dig.*, Dec. 2020, pp. 2931–2934, doi: 10.1109/IEDM13553.2020.9372119.

[43] Z. Zhang, S. Mao, G. Xu, Q. Zhang, Z. Wu, H. Yin, and T. Ye, ''An ultra-dense one-transistor ternary-content-addressable memory array based on non-volatile and ambipolar fin field-effect transistors,'' *IEEE Trans. Electron Devices*, vol. 70, no. 3, pp. 1029–1033, Mar. 2023, doi: 10.1109/TED.2023.3239330.

[44] H. Xu, J. Yang, T. Kämpfe, C. Zhuo, K. Ni, and X. Yin, ''On the challenges and design mitigations of single transistor ferroelectric content addressable memory,'' *IEEE Electron Device Lett.*, vol. 45, no. 1, pp. 112–115, Jan. 2024, doi: 10.1109/LED.2023.3334756.

[45] E. Garzón, M. Lanuzza, A. Teman, and L. Yavits, ''AM4: MRAM crossbar based CAM/TCAM/ACAM/AP for in-memory computing,'' *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 1, pp. 408–421, Mar. 2023, doi: 10.1109/JETCAS.2023.3243222.

[46] C. Zhuo, Z. Yang, K. Ni, M. Imani, Y. Luo, S. Wang, D. Zhang, and X. Yin, ''Design of ultracompact content addressable memory exploiting 1T-1MTJ cell,'' *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 5, pp. 1450–1462, May 2023, doi: 10.1109/TCAD.2022.3204515.

[47] S. Matsunaga, A. Katsumata, M. Natsui, T. Endoh, H. Ohno, and T. Hanyu, ''Design of a nine-transistor/two-magnetic-tunnel-junction-cell-based low-energy nonvolatile ternary content-addressable memory,'' *Jpn. J. Appl. Phys.*, vol. 51, no. 2S, p. 02, 2012.

[48] B. Song, T. Na, J. P. Kim, S. H. Kang, and S.-O. Jung, ''A 10T-4MTJ nonvolatile ternary CAM cell for reliable search operation and a compact area,'' *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 6, pp. 700–704, Jun. 2017, doi: 10.1109/TCSII.2016.2594827.

[49] E. Garzón, L. Yavits, G. Finocchio, M. Carpentieri, A. Teman, and M. Lanuzza, ''A low-energy DMTJ-based ternary content- addressable memory with reliable sub-nanosecond search operation,'' *IEEE Access*, vol. 11, pp. 16812–16819, 2023, doi: 10.1109/ACCESS.2023.3245981.

[50] S. Narla, P. Kumar, A. F. Laguna, D. Reis, X. S. Hu, M. Niemier, and A. Naeemi, ''Design of a compact spin-orbit-torque-based ternary content addressable memory,'' *IEEE Trans. Electron Devices*, vol. 70, no. 2, pp. 506–513, Feb. 2023, doi: 10.1109/TED.2022.3231569.

[51] D. Zhang, B. Wu, H. Zhu, and W. Liu, ''Capacity-oriented high-performance NV-TCAM leveraging hybrid MRAM scheme,'' in *Proc. 17th ACM Int. Symp. Nanosc. Architectures*, Dec. 2022, pp. 1–6, doi: 10.1145/3565478.3572315.

[52] J. Lai, J. Cai, L. Liu, and Z. Huang, ''A design and analysis perspective on architecting memory using domain-wall memory,'' in *Proc. IEEE Int. Conf. Smart Internet Things (SmartIoT)*, Aug. 2019, pp. 454–458, doi: 10.1109/SMARTIOT.2019.00082.

[53] R. Zhang, C. Tang, X. Sun, M. Li, W. Jin, P. Li, X. Cheng, and X. S. Hu, ''Sky-TCAM: Low-power skyrmion-based ternary content addressable memory,'' *IEEE Trans. Electron Devices*, vol. 70, no. 7, pp. 3517–3522, Jul. 2023, doi: 10.1109/TED.2023.3274506.

[54] I. Merlin, E. Garzón, A. Fish, and L. Yavits, ''DIPER: Detection and identification of pathogens using edit distance-tolerant resistive CAM,'' *IEEE Trans. Comput.*, early access, pp. 1–12, Sep. 2023, doi: 10.1109/TC.2023.3315829.

[55] K. Pan, A. M. S. Tosson, N. Wang, N. Y. Zhou, and L. Wei, ''A novel cascadable TCAM using RRAM and current race scheme for high-speed energy-efficient applications,'' *IEEE Trans. Nanotechnol.*, vol. 22, pp. 214–221, 2023, doi: 10.1109/TNANO.2023.3271308.

[56] L. Brackmann, T. Ziegler, A. Jafari, D. J. Wouters, M. Tahoori, and S. Menzel, ''Design limitations in oxide-based memristive ternary content addressable memories,'' in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2023, pp. 1–5, doi: 10.1109/ISCAS46773.2023.10181488.

[57] S. Agwa, G. Papandroulidakis, and T. Prodromakis, ''A 1T1R+2T analog content-addressable memory pixel for online template matching,'' in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2023, pp. 1–5, doi: 10.1109/ISCAS46773.2023.10181451.

[58] D. R. B. Ly, J-P. Noel, B. Giraud, P. Royer, E. Esmanhotto, N. Castellani, T. Dalgaty, J.-F. Nodin, C. Fenouillet-Beranger, E. Nowak, and E. Vianello, ''Novel 1T2R1T RRAM-based ternary content addressable memory for large scale pattern recognition,'' in *IEDM Tech. Dig.*, Dec. 2019, pp. 3551–3554, doi: 10.1109/IEDM19573.2019.8993621.

[59] H. Yang, P. Huang, R. Han, X. Liu, and J. Kang, ''An ultra-high-density and energy-efficient content addressable memory design based on 3D-NAND flash,'' *Sci. China Inf. Sci.*, vol. 66, no. 4, Apr. 2023, Art. no. 142402, doi: 10.1007/s11432-021-3502-4.

[60] C. Jin, J. Xu, J. Zhao, J. Gu, J. Chen, H. Liu, H. Qian, M. Zhang, B. Chen, R. Cheng, Y. Liu, X. Yu, and G. Han, ''A multi-bit CAM design with ultra-high density and energy efficiency based on FeFET NAND,'' *IEEE Electron Device Lett.*, vol. 44, no. 7, pp. 1104–1107, Jul. 2023, doi: 10.1109/LED.2023.3277845.

[61] H. Z. Yang, P. Huang, R. Z. Han, Y. C. Xiang, Y. Feng, B. Gao, J. Z. Chen, L. F. Liu, X. Y. Liu, and J. F. Kang, ''A novel high-density and low-power ternary content addressable memory design based on 3D NAND flash,'' in *Proc. IEEE Silicon Nanoelectronics Workshop (SNW)*, Jun. 2020, pp. 39–40, doi: 10.1109/SNW50361.2020.9131657.

[62] A. Kazemi, S. Sahay, A. Saxena, M. M. Sharifi, M. Niemier, and X. S. Hu, ''A flash-based multi-bit content-addressable memory with Euclidean squared distance,'' in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design (ISLPED)*, Jul. 2021, pp. 1–6, doi: 10.1109/ISLPED52811.2021.9502488.

[63] H. Yang, P. Huang, R. Li, N. Tang, Y. Zhang, Z. Zhou, L. Liu, X. Liu, and J. Kang, ''Flash-based content addressable memory with l2 distance for memory-augmented neural network,'' *IScience*, vol. 26, no. 12, Dec. 2023, Art. no. 108371, doi: 10.1016/j.isci.2023.108371.

[64] Y.-H. Lin, P.-H. Tseng, F.-M. Lee, M.-H. Lee, C.-C. Hsieh, D.-Y. Lee, K.-C. Wang, and C.-Y. Lu, ''NOR flash-based multilevel in-memory-searching architecture for approximate computing,'' in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2022, pp. 1–4, doi: 10.1109/IMW52921.2022.9779250.

[65] P. W. Cook, M. Gupta, V. Zyuban, J. Wellman, A. Buyuktosunoglu, P. N. Kudva, H. Jacobson, S. E. Schuster, P. Bose, and D. M. Brooks, ''Power-aware microarchitecture: Design and modeling challenges for next-generation microprocessors,'' *IEEE Micro*, vol. 20, no. 6, pp. 26–44, Nov. 2000, doi: 10.1109/40.888701.

[66] H. Wong, V. Betz, and J. Rose, ''Quantifying the gap between FPGA and custom CMOS to aid microarchitectural design,'' *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 10, pp. 2067–2080, Oct. 2014, doi: 10.1109/TVLSI.2013.2284281.

[67] S. W. Hussain, T. V. Mahendra, S. Mishra, and A. Dandapat, ''Low-power content addressable memory design using two-layer P-N match-line control and sensing,'' *Integration*, vol. 75, pp. 73–84, Nov. 2020, doi: 10.1016/j.vlsi.2020.06.001.

[68] S. Mishra, T. V. Mahendra, J. Saikia, and A. Dandapat, ''A low-overhead dynamic TCAM with pipelined read-restore refresh scheme,'' *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 5, pp. 1591–1601, May 2018, doi: 10.1109/TCSI.2017.2756662.

[69] N. C. Papadopoulos, V. Karakostas, K. Nikas, N. Koziris, and D. N. Pnevmatikatos, ''A configurable TLB hierarchy for the RISC-V architecture,'' in *Proc. 30th Int. Conf. Field-Program. Log. Appl. (FPL)*, Aug. 2020, pp. 85–90, doi: 10.1109/FPL50879.2020.00024.

[70] R. Jeyapaul and A. Shrivastava, ''Code transformations for TLB power reduction,'' *Int. J. Parallel Program.*, vol. 38, nos. 3–4, pp. 254–276, Jun. 2010, doi: 10.1007/s10766-009-0123-8.

[71] D. Fan, Z. Tang, H. Huang, and G. R. Gao, ''An energy efficient TLB design methodology,'' in *Proc. Int. Symp. Low Power Electron. Design*, 2005, pp. 351–356, doi: 10.1109/lpe.2005.195546.

[72] A. Efthymiou and J. D. Garside, ''An adaptive serial–parallel CAM architecture for low-power cache blocks,'' in *Proc. Int. Symp. Low Power Electron. Design*, 2002, pp. 136–141, doi: 10.1109/lpe.2002.1029577.

[73] S. Li, L. Liu, P. Gu, C. Xu, and Y. Xie, ''NVSim-CAM: A circuit-level simulator for emerging nonvolatile memory based content-addressable memory,'' in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2016, pp. 1–7, doi: 10.1145/2966986.2967059.

[74] C. Zhang and B. Xue, ''Two dimensional highly associative level-two cache design,'' in *Proc. IEEE Int. Conf. Comput. Design*, Oct. 2008, pp. 679–684, doi: 10.1109/ICCD.2008.4751934.

[75] L. Trupti Ranjan, M. Durgamadhab, and L. Fu, ''Recent research for HZO-based ferroelectric memory towards in-memory computing applications,'' *Micro Nanoelectronics Devices, Circuits Syst.–Select Proc. MNDCS*. Cham, Switzerland: Springer, 2023, doi: 10.1007/978-981-19-2308-1.

[76] X. Shang, W. Jia, J. Shan, X. Ding, and C. Borcea, ''Reestablishing page placement mechanisms for nested virtualization,'' *IEEE Trans. Cloud Comput.*, vol. 11, no. 3, pp. 1–12, May 2023, doi: 10.1109/TCC.2023.3276368.

[77] M. D. Hill and A. J. Smith, ''Evaluating associativity in CPU caches,'' *IEEE Trans. Comput.*, vol. 38, no. 12, pp. 1612–1630, Dec. 1989, doi: 10.1109/12.40842.

[78] J. A. Rivers and E. S. Davidson, ''Reducing conflicts in direct-mapped caches with a temporality-based design,'' in *Proc. ICPP Challenges for Parallel Process.*, vol. 1, pp. 154–163, Aug. 1996, doi: 10.1109/ICPP.1996.537156.

[79] N. Ho, P. Kaufmann, and M. Platzner, ''Optimization of application-specific L1 cache translation functions of the LEON$_3$ processor,'' in *Proc. 11th Int. Conf. Soft Comput. Pattern Recognit. (SoCPaR)*, Cham, Switzerland: Springer, 2021, pp. 266–276, doi: 10.1007/978-3-030-49345-5_28.

[80] T. Givargis, ''Improved indexing for cache miss reduction in embedded systems,'' in *Proc. 40th Annu. Design Autom. Conf.*, 2003, pp. 875–880, doi: 10.1145/775832.776052.

[81] R. Lee, X. Ding, F. Chen, Q. Lu, and X. Zhang, ''MCC-DB: Minimizing cache conflicts in multi-core processors for databases,'' *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 373–384, 2009, doi: 10.14778/1687627.1687670.

[82] P. Roy, S. L. Song, S. Krishnamoorthy, and X. Liu, ''Lightweight detection of cache conflicts,'' in *Proc. Int. Symp. Code Gener. Optim.*, Feb. 2018, pp. 200–213, doi: 10.1145/3168819.

[83] M. B. Kamble and K. Ghose, ''Energy-efficiency of VLSI caches: A comparative study,'' in *Proc. 10th Int. Conf. VLSI Design*, 1997, pp. 261–267, doi: 10.1109/ICVD.1997.568087.

[84] C. Zhang, ''Balanced cache: Reducing conflict misses of direct-mapped caches,'' in *Proc. 33rd Int. Symp. Comput. Archit. (ISCA06)*, 2006, pp. 155–166, doi: 10.1109/ISCA.2006.12.

[85] C. Zhang, ''Reducing cache misses through programmable decoders,'' *ACM Trans. Archit. Code Optim.*, vol. 4, no. 4, pp. 1–31, Jan. 2008, doi: 10.1145/1328195.1328200.

[86] L. Yavits, U. Weiser, and R. Ginosar, ''Resistive address decoder,'' *IEEE Comput. Archit. Lett.*, vol. 16, no. 2, pp. 141–144, Jul. 2017, doi: 10.1109/LCA.2017.2670539.

[87] L. Yavits, R. Kaplan, and R. Ginosar, ''Enabling full associativity with memristive address decoder,'' *IEEE Micro*, vol. 38, no. 5, pp. 32–40, Sep. 2018, doi: 10.1109/MM.2018.053631139.

[88] L. Deng, K. Li, and W. Shan, ''IVATS: A leakage reduction technique based on input vector analysis and transistor stacking in CMOS circuits,'' in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2023, pp. 1–5, doi: 10.1109/ISCAS46773.2023.10182138.

[89] S. Narendra, V. De, D. Antoniadis, A. Chandrakasan, and S. Borkar, ''Scaling of stack effect and its application for leakage reduction,'' in *Proc. Int. Symp. Low Power Electron. Design-(ISLPED)*, 2001, pp. 195–200, doi: 10.1145/383082.383132.

[90] Y. Xu, Z. Yu, D. Tang, Y. Cai, D. Huan, W. He, N. Sun, and Y. Bao, ''Towards developing high performance RISC-V processors using agile methodology,'' in *Proc. 55th IEEE/ACM Int. Symp. Microarchit. (MICRO)*, Oct. 2022, pp. 1178–1199, doi: 10.1109/MICRO56248.2022.00080.

[91] S. Kaxiras and A. Ros, ''A new perspective for efficient virtual-cache coherence,'' in *Proc. 40th Annu. Int. Symp. Comput. Archit.*, Jun. 2013, pp. 535–546, doi: 10.1145/2485922.2485968.

[92] H. Farbeh, L. Delshadtehrani, H. Kim, and S. Kim, ''ECC-united cache: Maximizing efficiency of error detection/correction codes in associative cache memories,'' *IEEE Trans. Comput.*, vol. 70, no. 4, pp. 640–654, Apr. 2021, doi: 10.1109/TC.2020.2994067.

[93] R. Naseer and J. Draper, ''DEC ECC design to improve memory reliability in sub-100nm technologies,'' in *Proc. 15th IEEE Int. Conf. Electron., Circuits Syst.*, Aug. 2008, pp. 586–589, doi: 10.1109/ICECS.2008.4674921.

[94] S. Schechter, G. H. Loh, K. Strauss, and D. Burger, ''Use ECP, not ECC, for hard failures in resistive memories,'' *ACM SIGARCH Comput. Archit. News*, vol. 38, no. 3, pp. 141–152, Jun. 2010, doi: 10.1145/1816038.1815980.

[95] D. H. Yoon, N. Muralimanohar, J. Chang, P. Ranganathan, N. P. Jouppi, and M. Erez, ''FREE-p: Protecting non-volatile memory against both hard and soft errors,'' in *Proc. IEEE 17th Int. Symp. High Perform. Comput. Archit.*, Feb. 2011, pp. 466–477, doi: 10.1109/HPCA.2011.5749752.

[96] M. K. Qureshi, A. Jaleel, Y. N. Patt, S. C. Steely, and J. Emer, ''Adaptive insertion policies for high performance caching,'' *ACM SIGARCH Comput. Archit. News*, vol. 35, no. 2, pp. 381–391, May 2007.

[97] A. Jaleel, K. B. Theobald, S. C. Steely, and J. Emer, ''High performance cache replacement using re-reference interval prediction (RRIP),'' *ACM SIGARCH Comput. Archit. News*, vol. 38, no. 3, pp. 60–71, Jun. 2010, doi: 10.1145/1816038.1815971.

[98] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, ''The gem5 simulator,'' *ACM SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, 2011.

[99] W. Heirman, T. Carlson, and L. Eeckhout, "Sniper: Scalable and accurate parallel multi-core simulation," in *Proc. 8th Int. Summer School Adv. Comp. Arch. Compilation High-Perform. Embedded Syst. (ACACES)*, 2012, pp. 91–94.

[100] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Sixth Edition: A Quantitative Approach*, 6th ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2017. [Online]. Available: https://dl.acm.org/doi/10.5555/3207796

[101] *Hardware-Effects*. Accessed: Jun. 15, 2023. [Online]. Available: https://github.com/Kobzol/hardware-effects

[102] G. F. Oliveira, J. Gómez-Luna, L. Orosa, S. Ghose, N. Vijaykumar, I. Fernandez, M. Sadrosadati, and O. Mutlu, "DAMOV: A new methodology and benchmark suite for evaluating data movement bottlenecks," *IEEE Access*, vol. 9, pp. 134457–134502, 2021, doi: 10.1109/ACCESS.2021.3110993.

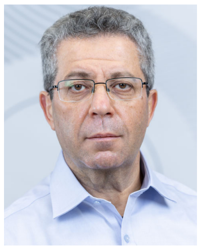
[103] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," *ACM SIGARCH Comput. Archit. News*, vol. 23, no. 2, pp. 24–36, 1995.

ESTEBAN GARZÓN (Member, IEEE) received the Ph.D. degree (Hons.) in electronics engineering from the University of Calabria (UNICAL), Italy, in 2022. In 2022, he won a highly competitive research fellowship funded by the Italian Ministry for Universities and Research (MUR), under the call "Horizon Europe 2021–2027 Programme". He is currently a Research Fellow with the Department of Computer Engineering, Modeling, Electronics, and Systems Engineering, UNICAL. He has authored/coauthored more than 40 scientific papers in international peer-reviewed journals and conferences, and has participated in several IC tapeouts. His research interests include domain-specific hardware accelerators, spintronics, cryogenic embedded memories, standard and emerging technologies for logic and memory, and low-power applications.

ROBERT HANHAN received the B.Sc. degree in computer engineering from the Technion—Israel Institute of Technology, Haifa, Israel, in 2021, where he is currently pursuing the M.Sc. degree in computer engineering. In addition, he is with Bar-Illan University, as a Front-End SoC Engineer with the SoC Laboratory, to provide SoC solutions to specific domain applications, in terms of architecture and design. His research interests include hardware acceleration for genome analysis, domain specific acceleration, and processing in memory.

MARCO LANUZZA (Senior Member, IEEE) received the Ph.D. degree in electronic engineering from the Mediterranea University of Reggio Calabria, Reggio Calabria, Italy, in 2005. Since 2006, he has been with the University of Calabria, Rende, Italy, where he is currently an Associate Professor. He has authored over 100 publications in international journals and conference proceedings. His current research interests include the design of ultralow voltage circuits and systems, the development of efficient models and methodologies for leakage- and variability-aware designs, and the design of digital and analog circuits in emerging technologies. He is an Associate Editor of *Integration, the VLSI Journal*.

ADAM TEMAN (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Ben-Gurion University (BGU) of the Negev, Be'er Sheva, in 2014. He was a Design Engineer with Marvell Semiconductors, from 2006 to 2007. From 2014 to 2015, he was a Postdoctoral Researcher with École Polytechnique Fédérale de Lausanne (EPFL), Switzerland, under a Swiss Government Excellence Scholarship. Since 2015, he has been with Bar-Ilan University, where he is currently an Associate Professor and the Co-Director of Emerging Nanoscaled Integrated Circuits and Systems (EnICS) Laboratories. He has authored over 100 scientific articles and ten patents. His research interests include embedded memories, energy-efficient circuit design, hardware for artificial intelligence, open source processor platforms and accelerators, and methodologies for physical implementation. He is a member of the technical and review boards of several conferences and journals. He is an Associate Editor of the *Microelectronics Journal*.



LEONID YAVITS (Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from Technion, Israel. He is currently with the Faculty of Engineering, Bar-Ilan University, Israel. He is a serial entrepreneur, who was involved in Co-Founding and Successful Management (from a concept to M&A) of several start-ups in the field of ASICs. His research interests include bioinformatics, domain specific accelerators, and processing in memory.

Open Access funding provided by 'Università della Calabria' within the CRUI CARE Agreement