INVITED PAPER

Emerging Trends in Design and Applications of Memory-Based Computing and Content-Addressable Memories

Logical organization of memory to suit tasks—analytics, mining, pattern recognition—benefits by removing several inefficiencies that arise in the extraction of data. This paper reviews content-addressable and associative memories and discusses challenges and opportunities with reference to the variety of device forms in exploration.

By Robert Karam, *Student Member IEEE*, Ruchir Puri, *Fellow IEEE*, Swaroop Ghosh, *Senior Member IEEE*, and Swarup Bhunia, *Senior Member IEEE*

ABSTRACT | Content-addressable memory (CAM) and associative memory (AM) are types of storage structures that allow searching by content as opposed to searching by address. Such memory structures are used in diverse applications ranging from branch prediction in a processor to complex pattern recognition. In this paper, we review the emerging challenges and opportunities in implementing different varieties of CAM/AM structures. Beyond-CMOS silicon and nonsilicon memory technologies hold significant promise in implementing dense, fast, and energy-efficient CAM/AM structures. We describe circuit/architecture level implementations of CAM/AM using these technologies, as well as novel applications in different domains, including informatics, text analytics, data mining, and reconfigurable computing platforms.

KEYWORDS | Associative memory (AM); binary CAM; computing with memory; content-addressable memory (CAM); nano-CAM; ternary CAM; transactional memory (TM)

Manuscript received December 23, 2014; revised April 30, 2015; accepted May 6, 2015. Date of current version July 15, 2015. This work was supported in part by the National Science Foundation (NSF) under Grants CNS-1054744, 1441757, and 1002237; and by the Semiconductor Research Corporation (SRC) under Grants 2014-EP-2575 and 2013-HJ-2442.

R. Karam is with Case Western Reserve University, Cleveland, OH 44106, USA.
R. Puri is with the IBM Watson Research Center, Yorktown Heights, NY 10598, USA (e-mail: ruchir@us.ibm.com).
S. Ghosh is with the University of South Florida, Tampa, FL 33620, USA.
S. Bhunia is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA.

Digital Object Identifier: 10.1109/JPROC.2015.2434888

I. INTRODUCTION

TRADITIONAL memory structures, such as static random access memory (SRAM) or dynamic random access memory (DRAM) arrays, store data at a unique address and can recall the data upon presentation of the complete unique address. On the other hand, memory with content-based access, which we hereafter refer to as content-operated memory (COM), are specialized memory structures that access the stored information based on input data (either completely or in part), rather than an input address. Like other forms of memory, a COM allows two basic operations: write (store or update a value) and read (search and retrieve a value on a match). COMs are used in a computer system when an input address is not available for information being searched/stored and/or when very-high-speed parallel searching is required. Conceptually, COMs perform the inverse function of traditional random access memory (RAM) [1], which makes such memories well suited for fast parallel search of all stored data and comparison with the provided input. This has resulted in their widespread use in many application domains—from branch prediction tables or cache controllers in processors, to lookup tables in high-speed routers—all requiring content-based search of the memory. Over the past several decades, one of the primary uses of COM has been for packet forwarding and classification in high-speed network systems [60], [61]. There are, however, many emerging

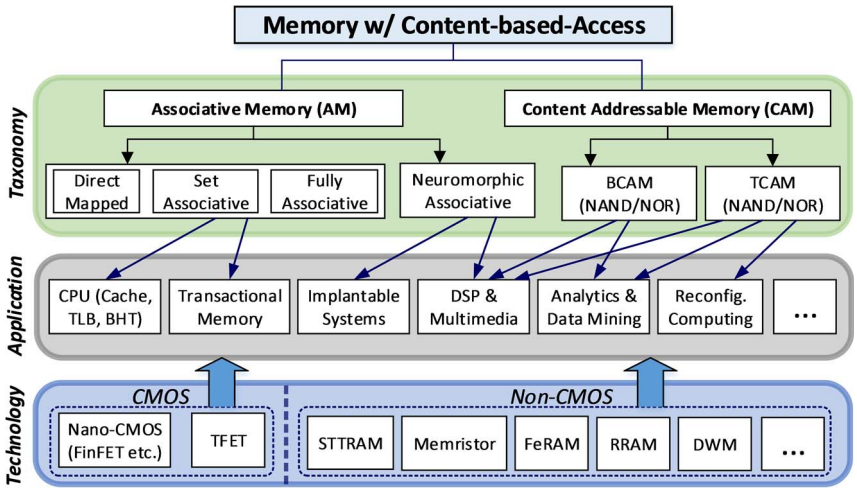

Fig. 1. ***Overview of the COM space: Associative memory and CAM make up the two primary forms, with several varieties of each; arrows represent typical use cases for the specific memory device, discussed further in Section IV. These memories play key roles in current and emerging applications, including analytics, implantable/wearable systems, signal processing systems, processors, and hardware transactional memories. As technology moves beyond CMOS, novel nanoscale devices are being considered for implementing these ubiquitous memories.***

applications and architectures of COMs using both complementary metal–oxide–semiconductor (CMOS) and non-CMOS devices, bringing lower power, higher speed parallel search capabilities to next-generation electronics.

Fig. 1 shows an overview of the COM technology space, in which content-addressable memory (CAM) and associative memory (AM) are the two primary forms. While they both perform similar functions, they do so in different ways. In either case, an access returns the address of the matching data, and null if no match is found. In general, AM does this by placing restrictions on where the data may be stored based on the content. In the three forms of AM: 1) direct-mapped, 2) set associative, and 3) fully associative, the limitations on where data may be stored are different, with direct-mapped AM resulting in the most stringent rules (only one location possible for the data). A fourth kind of AM called neuromorphic memory is based on a neural network architecture and enables associative recall of data even under noisy conditions.

In many ways, CAM is similar to direct-mapped AM: both require update policies (such as least recently used) for overwriting data when it is no longer useful, and both allow immediate retrieval of an address given the input data. However, the mechanism enabling this rapid search is very different between the two. With direct-mapped AM, the storage location is based on the content, so there is only one place for a given data word to be stored. With CAM, there are no restrictions on where the data may be stored. Instead, each entry in the memory contains search hardware that activates in parallel; when a search pattern is supplied to the CAM, every entry simultaneously compares itself to the input pattern, and if a match is found, the address is returned. Furthermore, ternary CAMs (TCAMs) enable "don't care" matches, which can test for partial matches or range matches.

Related to CAM and AM, there is a new emerging application: transactional memory (TM). While CAMs and AMs address the question of *where* data are stored, TMs deal with *how* it is accessed and shared among different threads and cores in a multi-processor, multithreaded environment. It is not, therefore, related to associative or content addressable memories in operation, but actually uses them as the underlying storage mechanism for the different hardware components that enable its operation. This storage can include a transaction log, register contexts, or the cache itself. While software transactional memory (STM) has existed for some time, hardware transactional memories (HTMs) have only recently emerged [3], [4] in enterprise and consumer electronic devices, and promise to make development of parallel programs more intuitive and efficient.

Fig. 2 shows high-level block diagrams for these three kinds of memory structures (CAM, AM, and HTM) with supporting external hardware. Historically, these memory structures have been implemented using CMOS processes, but recent advances have enabled the use of non-CMOS nanoscale technologies for denser and more efficient COM implementations. These emerging nanoscale devices hold tremendous potential for higher integration density (on the order of 10^{10} devices/cm^2) [5], lower power consumption, and higher switching speeds compared with traditional CMOS devices. Although many of these devices are not amenable for implementing cascaded, irregular logic, their bistable nature and ability to realize dense periodic structures make them ideal candidates for next-generation

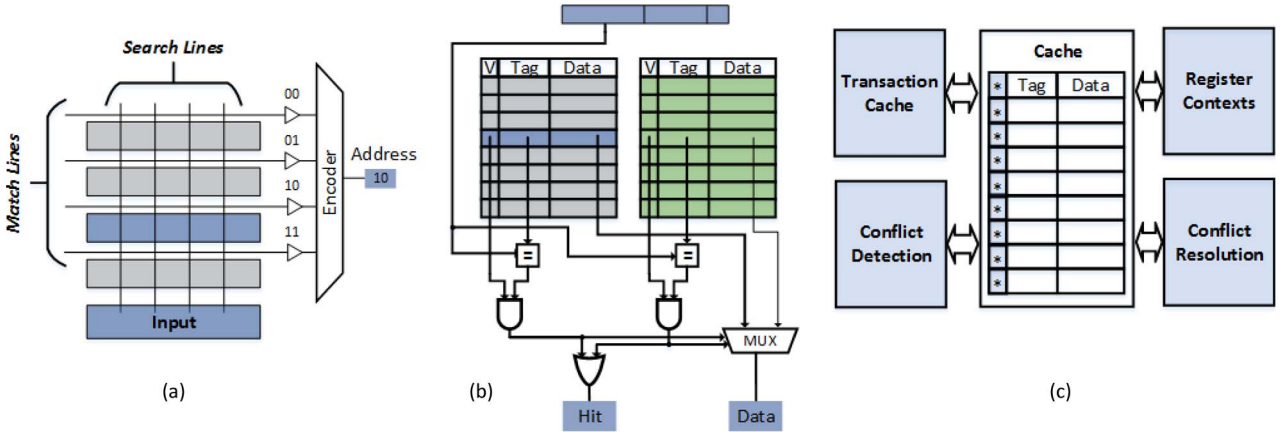

Fig. 2. *High-level block diagram for (a) CAM; (b) AM; and (c) HTM.*

memories. Moreover, unlike conventional CMOS-based SRAM and DRAM arrays, these memory forms are suitable for nonvolatile operation, which is attractive in many applications, including reconfigurable computing frameworks [6]. Some of the promising memory devices include: 1) ferroelectric random access memory (FeRAM) [7]; 2) spin torque transfer random access memory (STTRAM) [8]; 3) phase change random access memory (PCRAM) [9]; 4) nanoelectromechanical memory (NEMM) [10]; 5) molecular memory [11]; and 6) resistive random access memory (ReRAM) [37]. Most of these emerging devices are also highly attractive for efficient CAM or AM implementation, which provides higher density, lower search and update latency/power, lower leakage current, and nonvolatility of stored information.

In this paper, we focus on the emerging trends in CAM/AM design, highlighting the shift in circuit-architecture level design of CAM with the emergence of new device technologies. In particular, beyond-CMOS memory technologies provide new opportunities and challenges with respect to various forms of CAM/AM implementations. We analyze the CAM designs using these technologies, and discuss emerging applications of CAM in diverse fields, including analytics, reconfigurable computing platforms, signal processing in wearable/implantable systems, and neuromorphic AM, which enables energy-efficient non-Boolean computing. We also discuss a novel application of AM in the implementation of hardware transactional memory (HTM). Although reviews of CAM design have been addressed earlier in open literature, to the best of our knowledge, this is the first attempt to analyze CAM/AM design with promising novel post-CMOS nanoscale devices as well as their emerging applications.

The rest of the paper is organized as follows. Section II provides background on content-addressable and associative memory describing different classes of COM implementations and major operations in COM. Section III describes emerging COM implementations with novel post-CMOS devices. Section IV focuses on architecture and various applications of CAM, and Section V presents the application of AM in HTM. Section VI discusses future directions in COM design and applications. Finally, Section VII presents a summary on emerging trends in the field of CAM and AM.

II. BACKGROUND

In this section, we describe different types of conventional CMOS-based CAMs, their design challenges, and how they have evolved over time.

A. Existing CAM Implementations

At the core of a CAM cell is a conventional SRAM bitcell, as illustrated for the NAND topology in Fig. 3(a). Extra transistors are added to implement the pulldown path of the XNOR operation, used for comparing the input (i.e., the search pattern) with the stored entry. A typical CAM supports four basic operations: 1) precharge; 2) set searchline (SL); 3) evaluate; and 4) match/mismatch. First, the matchline (ML) is precharged, and search pattern is placed on the SL. Next, a comparison is performed (evaluate), which discharges the ML if the SL pattern matches with the bitcell. There are two prominent CAM topologies, the NOR and NAND cells, shown in Fig. 3. READ and WRITE access circuitry is omitted for clarity in this and subsequent figures of CAM cells.

1) NOR Cell: The NOR cell compares the true and complement of the stored data D (and $\overline{D}$), with the true and complement of the search pattern on SL (and $\overline{SL}$). This is done using transistors M1–M4. Transistors M1/M3 (M2/M4) implement the pulldown path of a dynamic XNOR logic gate with inputs D and SL ($\overline{D}$ and $\overline{SL}$). A match disables both pulldown paths, disconnecting the ML from ground. Multiple cells share the ML in parallel to form a CAM word and implement the NOR function, as shown in Fig. 3(b). The ML voltage depends on the match of individual cells in the word, with a complete match maximizing the ML swing.

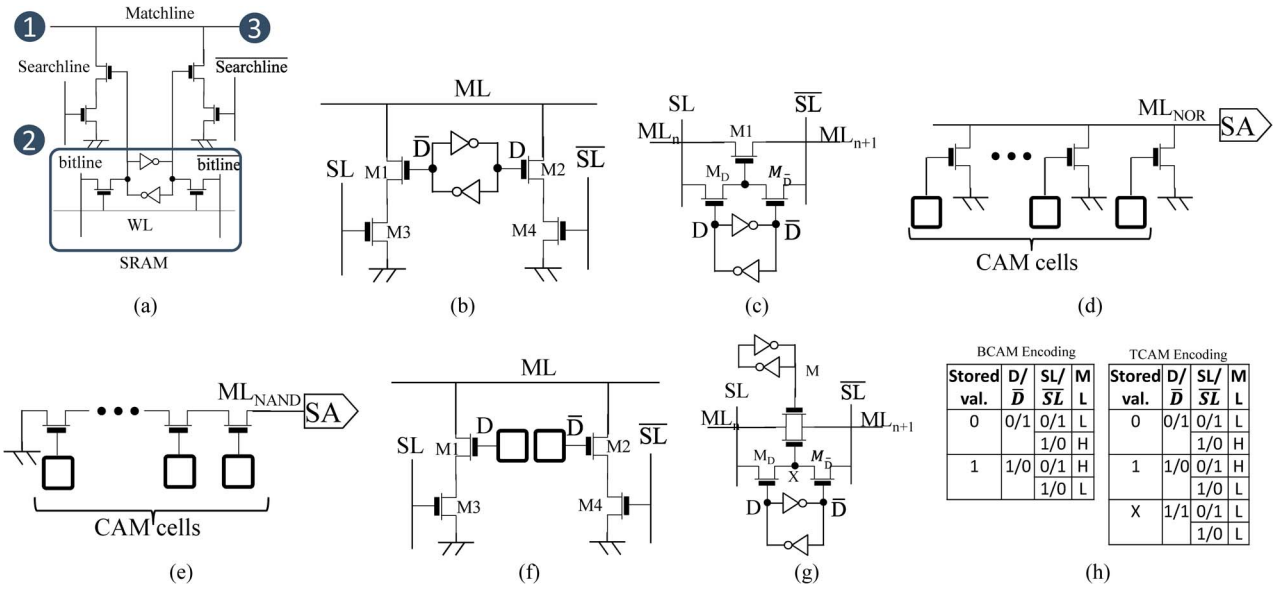

Fig. 3. ***(a) Basic structure of a CAM. (b) and (c) Two types (NOR and NAND) of binary CAMs. (d) and (e) Single word of NOR and NAND CAM. (f) and (g) Two types of ternary CAMs. (h) Truth tables for BCAM and NOR TCAM.***

2) NAND Cell: The NAND cell compares the true and complement of the stored data and the search pattern using transistors M1, M_D, and $M_{\overline{D}}$. During a match, either pass transistor M_D or $M_{\overline{D}}$ is ON and passes the SL or $\overline{SL}$ value (which will be logic "1") to node N. This turns ON transistor M1, connecting ML_n with ML_{n+1}. In the mismatch cases, pass transistor M_D or $M_{\overline{D}}$ is ON and passes the SL or $\overline{SL}$ value (which will be logic "0") to node N, turning OFF transistor M1. Meanwhile, ML_n remains disconnected from ML_{n+1}, implementing the NAND function, as shown in Fig. 3(c). A match condition for the entire word connects ML to ground and creates a discharge path.

NAND Versus NOR Cell: A NOR cell provides a full rail swing at the gates of all comparison transistors whereas a NAND cell provides a degraded logic "1" voltage at an intermediate node due to the threshold voltage drop across the pass transistor. The presence of series connected transistors makes the NAND topology slower than NOR; still, the NAND cell is more energy efficient than the NOR since the ML discharges only when there is a complete match. The ML power is given by $pC_{ML}V_{DD}^2f$ where p is the switching probability of the ML, C_{ML} is the switching capacitance, V_{DD} is the voltage swing, and f is the search frequency. Fig. 3(d) and (e) shows one word of NOR and NAND CAM, respectively. In an N-bit CAM word, the ML switching probability is $(N-1)/N$ for NOR whereas it is only $1/N$ for NAND CAM. Furthermore, the ML capacitance of the NOR ML increases faster than that of the NAND ML as more bits are added in the CAM word. This provides a tradeoff between speed and energy efficiency of CAM design. Mixed NAND–NOR cells have been proposed to explore this tradeoff. Different variants of the NOR cell (9T versus 10T) and NAND cell (10T versus 9T) are also possible; however, they are not preferred due to area and delay penalties [12].

B. Types of CAM

CAMs are further divided into two categories: binary CAM (BCAM) and ternary CAM (TCAM). While both can perform the basic search function, the BCAM [Fig. 3(b) and (c)] cannot natively handle partial matches using "don't care," denoted by an "X." Compared with BCAM, TCAM requires an extra bit per cell to allow storage of three states: 0, 1, and X. For example, a match with "X" is often required in network routers to mask certain bits of an IP address. Fig. 3(f) and (g) shows the formation of TCAM based on NOR and NAND cells, respectively. If a TCAM is preferred in a design, but only BCAM is available, it can still be used implicitly by using two BCAM cells per TCAM symbol, at the cost of half of the original storage capacity. Generally, NOR cell TCAMs store X with values $D = 11$, while the state $D = 00$ is undefined and not used; however, other designs with different encodings are possible [70]. The full truth tables for BCAM and NOR TCAM are shown in Fig. 3(h).

C. CAM Operation

1) Types of Search: Many applications [13] require partial matching with "don't cares," which in turn may result in multiple matches being made [12]. Typically, the match addresses are required in priority order from the CAM; thus, a priority encoder is employed at the output to handle this requirement. For example, in the classless

Table 1 Example Forwarding Table

Entry	Prefix	Next Hop Info
1	110	Interface #3
2	10001	Interface #1
3	11011	Interface #2
4	1101	Interface #4

interdomain routing (CIDR) environment, the IP search table performs longest prefix matching (LPM) [14], [15]. The LPM is explained with an example in Table 1 that shows the prefix stored in CAM and information on the next destination. An LPM searches for the longest prefix among those that match the given input. Here, input string "1101111" matches entries 1, 3, and 4, but as the longest entry, only #3 will be selected. The priority can be static (e.g., low to high address yields increasing or decreasing priority) or dynamic (e.g., runtime modifications/updating the data according to priority).

2) Types of Update: The instabilities of the internet require a forwarding table to be updated (either adding or deleting a prefix) frequently to reflect its route changes [16]. A backbone router may experience an average of 100 updates to its routing table per second, potentially reaching upwards of 1000 updates/s [17]. When a TCAM is used to accommodate the forwarding table, a single update involves multiple TCAM entry moves in order to maintain the order constraint of the TCAM entries. While updating, search operations over the updating range are frozen until the update completes. Consequently, frequent updates limit a router's lookup performance considerably. TCAM update algorithms have been proposed to reduce the number of entry moves per update [17], [18]. Although the goal of their design is to minimize the TCAM locking time for updates, the entire TCAM still needs to be blocked from lookups until an update operation has finished. Update algorithms such as the one in [15] have been proposed to avoid locking the TCAM during update while ensuring the consistency of the table by eliminating direct TCAM entry overwriting.

D. CAM Design Challenges

The primary design challenges associated with CAM are: 1) energy; 2) area; 3) speed; 4) reliability; and 5) noise margin. Here, we discuss these challenges in detail.

1) Energy: The search energy comprises dynamic energy dissipated in MLs, SLs, and the priority encoder, as well as leakage energy of the core bitcell, access transistors, decoder, wordline driver, and ML precharge circuit [19]. Energy-efficient CAM design is a highly researched area, and techniques have been proposed to lower the power consumption of each of the above components. ML power reduction using a low-swing ML is proposed in [20]. Selective precharge CAM [21] and current-race sensing scheme are developed to reduce SL switching activity and ML voltage swing [22]. A dual ML TCAM is proposed to reduce the active ML capacitance [23]. Other techniques include adding a capacitor in each ML to limit the number of charges per search transaction [20], injecting a small current to each ML prior to the sensing phase [24], exploiting the input data pattern to reduce the ML segment switching probability [25], using a self-reference sense amplifier to stop the ML charging around the threshold voltage of the output inverter [26], and uneven injection of current in missed row versus matched row [27]. Shutting down redundant comparisons using pipelining [28], automated background checking [29], and precomputation-based CAM [30] to limit the number of searches have also been proposed to lower the power consumption.

2) Area: The footprint of the CAM is primarily dependent on the area of the storage bitcell. CAMs based on dynamic storage, similar to the storage mechanism in DRAM, have been proposed to lower the footprint at the cost of refresh power and an expensive process technology. Emerging technologies such as STTRAM, ReRAM, memristor, and DWM have been proposed to address the energy and footprint challenges [31]–[39].

3) Speed: The delay of the CAM is composed of two components: match delay and priority encoder delay. The match delay is a function of the bitcell topology (NAND versus NOR), ML capacitance, array size, and the search pattern. Techniques to improve the match delay include low swing ML, pipelined ML, and low capacitance ML [20], [22], [23], [28]. The priority encoder delay depends on the number of matches, and in the worst case, the match information must ripple through the encoder from the first entry to the last entry of the array. Multiple match resolvers and parallel-prefix techniques have been proposed to improve the speed of the priority encoder [40], [41].

4) Reliability: The reliability of the CAM is coupled with the underlying memory technology. For example, the reliability of fully CMOS CAM is limited by the aging mechanisms of transistors, specifically, negative bias temperature instability (NBTI), positive bias temperature instability (PBTI), hot-carrier injection (HCI), and time-dependent dielectric breakdown (TDDB). For the nano-CAMs described in Section III, the reliability is primarily governed by the characteristics of the memory technology employed for storage. The memristor, resistive RAM, and phase-change RAM typically suffer from poor read/write endurance which limits the reliability of the CAM cell.

5) Noise Margin: The robustness of sensing depends on the search pattern, with the worst case due to a single bit mismatch. Background leakage lowers the noise margin further. Soft errors—the inadvertent flipping of stored bits

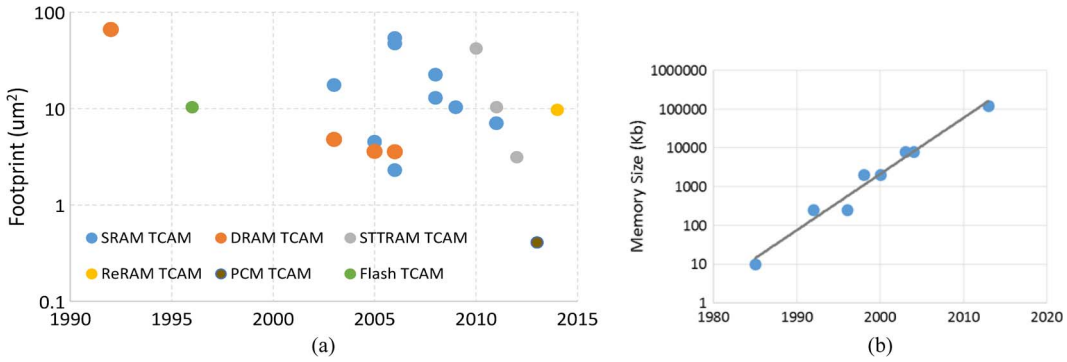

Fig. 4. *Evolution of CAM in terms of (a) area; and (b) memory size over years as reported in open literature.*

due to radiation-induced charge accumulation—are another reliability issue for memories in general that may corrupt the CAM content.

E. Evolution of CAM

Fig. 4 shows the landscape of CAMs over time with respect to their relative footprint and memory capacity. It can be noted that CAM density has improved significantly over the past two decades, a result that can be jointly attributed to aggressive CMOS technology scaling, emerging high-density nonvolatile storage technologies enabling nano-CAMs, and new circuit structures capable of exploiting the special properties of nano-CAMs.

III. EMERGING CAMS: CIRCUIT-LEVEL IMPLEMENTATIONS

This section introduces emerging device technologies and circuit designs to realize nano-CAMs. The emerging nano-CAMs are compared to each other with respect to various design metrics.

A. Types of Nano-CAMs

Nano-CAMs are realized using emerging nonvolatile memory technologies such as memristors, ReRAM, spintronic memories—e.g., STTRAM, magnetic RAM (MRAM), domain wall memory (DWM), and FeRAM. For nano-CAMs, the state of the core cell is stored in terms of either charge, or magnetic polarity, or resistance. Note that replacing the core cell with other forms of storage technology is not sufficient to ensure functionality and meet the design goals. Instead, nano-CAMs need to exploit the features of emerging technologies and employ judicious techniques to improve the footprint, power, speed, and robustness. Several emerging technologies such as STTRAM, FeRAM, and DWM are also capable of holding multiple states in a bitcell, a feature which has been exploited to efficiently design multivalued CAM structures in order to improve density and power [6], [42], [43].

B. STTRAM-Based CAM

1) Technology: STTRAM [44] is a promising memory technology for embedded memory (e.g., processor cache) due to its high-density, low standby power, and high-speed operation. The bitcell leakage is eliminated because the storage element, a magnetic tunnel junction (MTJ), is nonvolatile. The MTJ contains a free layer and a pinned magnetic layer [a schematic is shown in Fig. 5(a)]. The resistance of the MTJ stack is high (low) if the free layer magnetic orientation is antiparallel (parallel) compared to the fixed layer. The configuration of the MTJ can be changed from parallel to antiparallel (or *vice versa*) by injecting current from the sourceline to the bitline (or *vice versa*).

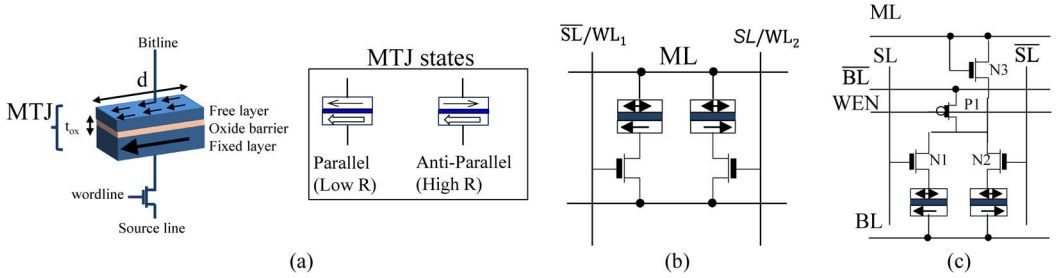

Fig. 5. *(a) Schematic of MTJ. The resistance states are also shown; (b) and (c) two flavors of MTJ CAMs.*

2) CAM Bitcell: Fig. 5(b) shows a two-transistor–two-MTJ TCAM cell [45] where MTJ resistance is used for storing the bitcell content. The robustness of the MTJ TCAM design is sensitive to the tunneling magnetoresistance (TMR) between the high (R_{high}) and low resistances (R_{low}) of the MTJ. TMR is given by $(R_{\text{high}} - R_{\text{low}})/R_{\text{low}}$ and is roughly in the order of 100%–150% in today's technology, which in turn results in a small ML voltage differential between the match and mismatch states. Therefore, STTRAM TCAM can only operate in a word-parallel, bit-serial fashion, resulting in long search time and high energy consumption [31]. A four-transistor–two-MTJ TCAM cell is proposed in [46] where transistors P1 and N3 are added to widen the ML voltage differential and enable full parallel search even for long word lengths.

The STTRAM TCAM operates in two phases: precharge and evaluate. In precharge, the ML is charged to V_{DD}, whereas in evaluate, the ML is pulled down through the TCAM cells similar to a conventional TCAM. The logic state of the selected TCAM cell, which is stored as MTJ resistance, determines the ML voltage. During a match, the discharge of the ML is relatively slow, and it stays at a relatively high voltage level. During mismatch, the discharge of the ML is relatively fast which pulls it down to a low voltage level. Higher TMR (200%–400%) widens the ML differential by 2x, while larger transistors also enhance the margin at the cost of density.

C. Memristor-Based CAM

1) Technology: Memristors [32]–[35] are two-terminal circuit elements in which the magnetic flux ϕ between the terminals is a function of the amount of electric charge q that can pass through the device, given by the equation $d\phi = M(q)dq$. Using Lenz's rule that expresses voltage $v = d\phi/dt$ and current $i = dq/dt$, a memristor is defined by the relationship $v = M(q)i$. Fig. 6(a) shows a schematic of a memristor demonstrated by HP labs, which was built with TiO_{2-x} and TiO_2 sandwiched between two platinum wires. TiO_{2-x} has an oxygen deficiency that makes it a conductor, whereas TiO_2 is an insulator. Application of bias voltage pushes the oxygen vacancies to the other terminal, reducing the resistance of the switch.

2) CAM Bitcell: Memristors can provide a R_{high} to R_{low} ratio of up to 1000x [32]–[35] which enables high ML voltage differentials between match and mismatch states. Fig. 6(b) shows the two-transistor–two-memristor nonvolatile TCAM cell. The memristors can be programmed by asserting their wordline signal and applying appropriate voltages to their bitlines (BL and $\overline{\text{BL}}$), while the value of the resistance being programmed is determined by the applied voltage direction (BL = 0/$\overline{\text{BL}}$ = 1 or BL = 1/$\overline{\text{BL}}$ = 0). Note that the memristance (resistance) of the memristor is determined by the amplitude and pulse width of the programming voltage/current. This offers a tradeoff opportunity between the operating speed and the programming energy.

Like other nanotechnologies, memristors also suffer from process variations, which can affect its resistance level and R_{high} to R_{low} ratio. This ratio is very sensitive to the variability of the lowest and highest resistance levels of the memristor device. Regardless, increasing the size of select transistors (N1–N2) can improve the ML differential. In optimizing the TCAM cell design, it is important to note that each design parameter affects the other parameters. For example, when the select transistor width changes from 120 to 480 nm, the minimum required R_{high} to R_{low} ratio ensuring an ML voltage of 0.2 V reduces from 1000× to 300× for a word length of 64 b. This technique allows higher tolerance to the memristor resistance variations while lowering programming energy and improving programming speed of the memristors.

D. FeRAM-Based CAM

1) Technology: FeRAM is a special transistor structure with a gate array that includes a ferroelectric film as a capacitor, as shown in Fig. 7(a) [36]. Data are stored as the polarization direction of the ferroelectric film on the gate stack. WRITE operations are performed by applying different

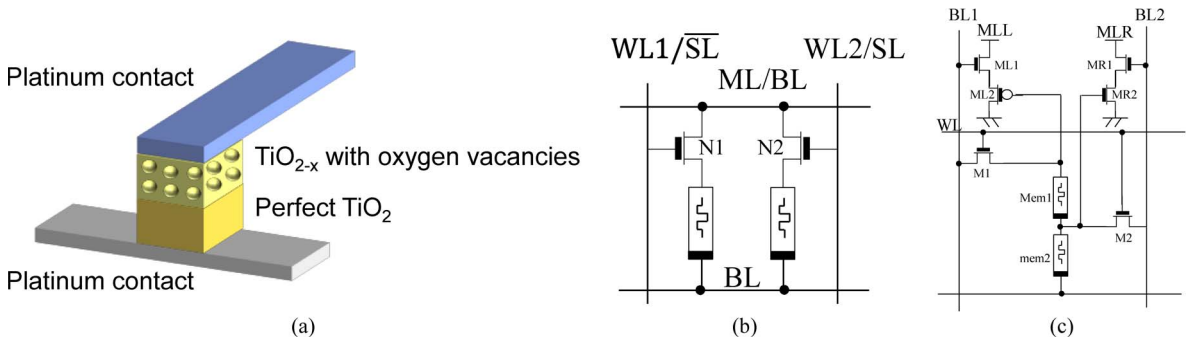

Fig. 6. *(a) Schematic of a memristor device; (b) and (c) two flavors of memristor CAMs.*

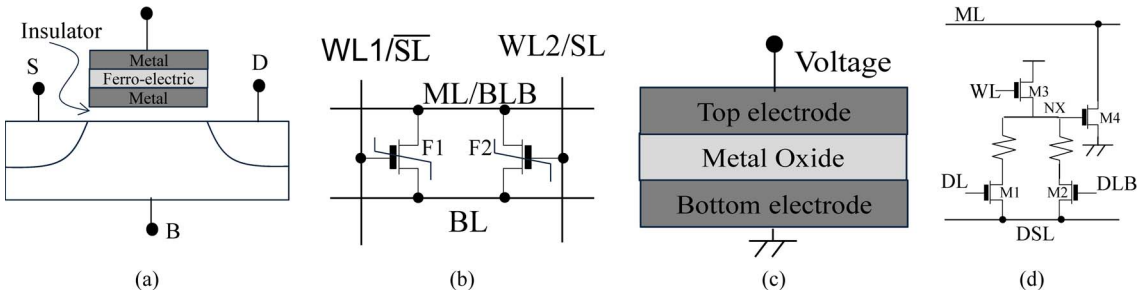

Fig. 7. *(a) and (b) Schematic of FeRAM and the corresponding CAM. (c) and (d) Schematic of ReRAM and the corresponding CAM.*

voltage pulses to the ferroelectric capacitor, and the polarization of the film switches depending on the direction of the applied voltage, shifting the threshold voltage of the transistor. For example, if the polarization direction is opposite to the applied READ voltage, the threshold voltage of the transistor increases, indicating a logic "0." Otherwise, the threshold voltage of the transistor decreases, indicating a logic "1." FeRAM is a promising technology for high-density, high-speed memory applications due to its simple structure, nonvolatility, easy CMOS integration, and scalability [36].

2) CAM Bitcell: ML discharging-based mechanisms limit the speed of the TCAM design. In [31], a simple, high-speed, and nonvolatile FeRAM cell is proposed [Fig. 7(b)]. WRITE operations are performed by applying the appropriate voltages to the bitlines (BL and BLB) and the wordline (WL). The true and complementary data are then stored in FeRAMs F1 and F2, respectively. Conversely, the search operation has two phases. If the stored data is logic "1," then F1 (F2) is programmed with a high (low) threshold voltage. During precharge, the ML is charged to a high voltage level, and during evaluate, a high search voltage level corresponding to logic "1" is applied to WL1, while its complement corresponding to logic "0" is applied to WL2. Since the threshold voltage of the F1 is high, the READ voltage does not turn ON F1 while F2 is OFF. As a result, the FeRAMs in the TCAM cell are in the cutoff state, and the precharged voltage on the ML is retained. If the stored data is logic "0," F2 is turned ON and the ML is pulled down to ground. The application of FeRAMs in the TCAM cell offers not only a full VDD swing ML voltage difference between match and mismatch states, but also negligible performance degradation when the word length increases.

E. Resistive RAM-Based CAM

1) Technology: The ReRAM device structure is simply an oxide material sandwiched between two metal electrodes, called the metal–insulator–metal (MIM) structure [37] [Fig. 7(c)]. The recent advancements in resistive switching using complex metal oxides such as the pervoskite oxides of $SrTiO_3$ and $SrZrO_3$, and the binary metal oxides, such as NiO and TiO_2, have revived interest in ReRAM. These devices exhibit resistive switching between a high-resistance state (HRS) and a low-resistance state (LRS). The switching event from HRS to LRS is called the "set" process, while the switching event from LRS to HRS is called the "reset" process.

2) CAM Bitcell: Fig. 7(d) shows the circuit of a ReRAM CAM cell [47] consisting of two ReRAMs and two comparison transistors (M1/M2), a WRITE-control transistor (M3), and an ML-driver transistor (M4). The gate and source of the comparison transistor are connected to datalines (DL/DLB) and a dynamic sourceline (DSL), respectively. The gate and drain of the M3 are connected to the wordline and WRITE voltage control, respectively. The ReRAM performs two overwrite operations, set and reset, which alter the ReRAM from high resistance to low resistance and *vice versa*. In standby mode, the ML is kept at precharge levels, while wordline, write voltage, and sourceline are kept at 0 V. For writing a logic "1" in the first phase, the right-hand side ReRAM is set by turning DLB on, applying set voltage to the wordline and WRITE voltage control, and, 0 V to sourceline and DL. In phase two, the left-hand side ReRAM is reset by turning DL on, applying reset voltage to the wordline and sourceline, and 0 V to the WRITE voltage control and DLB.

During a search operation, the CAM decouples the ReRAM cells from the ML current path using transistor M4, which is controlled by an RC-filtered pulse. In a search-1 operation, (DL = 1 and DLB = 0), a sourceline pulse is passed to node NX through transistor M1 and the left-hand side ReRAM. If the ReRAM is in a low-resistance state (mismatch) due to a short RC delay, the voltage at NX will exceed the threshold voltage of transistor M4 for a given period, enabling it to generate a large mismatch current to pull down the ML voltage. If the ReRAM is in a high-resistance state (match), a long delay, short pulsewidth, and low peak voltage are generated at node NX. Since the voltage at NX is below the threshold voltage of M4, it is cut off with a very small average match current so that the ML stays near the precharged value. If both ReRAMs are high

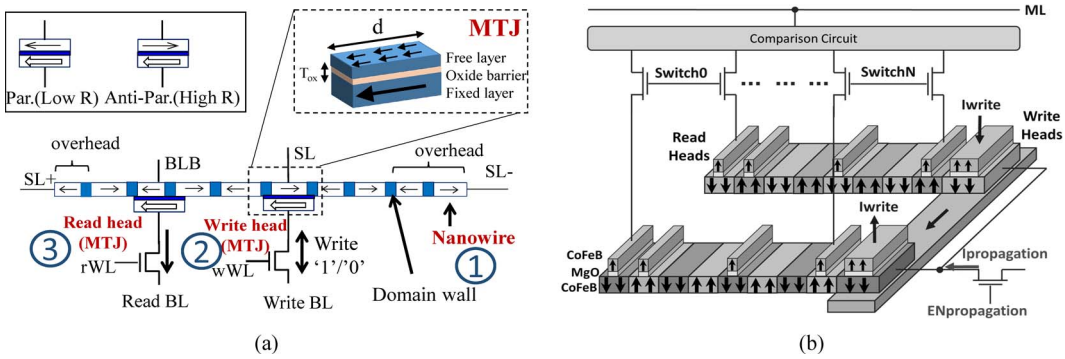

Fig. 8. ***(a) Schematic of DWM cells. (b) The corresponding CAM realization.***

resistance (don't care) or masked inputs (DL = DLB = 0), the CAM behaves similar to the match case. After a specific, self-timed period, the sense amplifier is turned on to detect the ML voltage and generate a digital output.

F. DWM-Based CAM

1) Technology: Domain wall memory (DWM) is a promising memory technology due to its multilevel cell capability, allowing it to store multiple bits per cell [38]. Additionally, it provides low standby power, nonvolatility, fast access time, good endurance, and good retention. DWM consists of three components: 1) the READ head; 2) the WRITE head; and 3) a magnetic nanowire. The READ and WRITE heads are similar to conventional MTJs, while the nanowire holds the bits in terms of magnetic polarity. The most interesting component of the nanowire is the formation of domain walls (DWs) between domains of opposite polarities [Fig. 8(a)]. The DWs can be shifted forward and backward by injecting charge current from left-shift and right-shift contacts, causing the nanowire to operate as a shift register. The new domains are injected by first pushing current through shift contacts to move the bits in lockstep until the desired bit is under WRITE head. Next, spin-polarized current is injected through the write MTJ (using wWL, wBL, and SL) in a positive or negative direction to write a "1" or "0" (up-spin or down-spin) in the nanowire. A READ is performed by bringing the desired bit under READ head using a shift and sensing the resistance of the MTJ formed by the new bit (after the DW crosses the READ head) using rBL and rWL.

2) CAM Bitcell: The DWM CAM [Fig. 8(b)] is composed of comparison circuits, magnetic nanowires, and DW nucleation/propagation circuits [39]. A complimentary pair of magnetic nanowires are used to present one word at a time in order to obtain the most reliable and fast access operation for CAM applications. The comparison circuit is designed based on a precharge sense amplifier [39]. The CAM includes two MTJs connected together, forming the write heads. Due to the opposite directions of the WRITE current pulse through these two MTJs, complementary polarities are nucleated in the nanowire. One of the critical challenges for complementary magnetic nanowires is to synchronize the domain wall positions. The current pulse is kept the same for both nanowires to address this challenge. Identical physical notches are built in the nanowires to hold or pin the DWs and enable their synchronization. A pair of READ MTJs are used for reading each bit of the storage element.

G. Comparative Analysis of Nano-CAMs

Table 2 qualitatively compares the major properties of CAM structures implemented in CMOS and the emerging memory technologies, as discussed above. Other emerging technologies such as phase-change memory (PCM) [48], carbon nanotube field-effect transistors (CNTFETs) [49], [50], multigate transistor technologies such as FinFET, as well as single electron transistors have also been envisioned for CAM realization [51], [52]. Table 3 presents the quantitative comparison of the nano-CAMs in terms of area, speed, energy, and endurance [53]. It can be noted that ReRAM CAM can achieve better density, energy, and delay compared

Table 2 Qualitative Comparison of Key Properties of CAM Structures Realized With Nanoscale Devices

Nano-CAM Type	Non-volatile?	Area (Density)	Power		Latency	Scalability (size)
			(Search/Update)	Leakage		
CMOS-CAM [1]	No	Mod. (Mod.)	High	High	Moderate	Poor
STTRAM-CAM [45] [31] [46]	Yes	Low (High)	Moderate	Negligible	Low	Good
ReRAM-CAM [47]	Yes	Low (High)	Moderate	Negligible	Low	Good
DWM-CAM [39]	Yes	Low (High)	Moderate	Negligible	Low	Good
FeRAM-CAM [31]	Yes	Mod. (Mod.)	Moderate	Negligible	Moderate	Moderate

Table 3 Quantitative Comparison of Different Nanoscale Memory Based CAMs

Nano-CAM Type	Search delay	Area	Power		Endurance
			(Search/Update)	Leakage	
CMOS-CAM [54]	1.077X	2.27X	1.33X	0.15X	10^{16}
MTJ-CAM [55]	2.24X	2.34X	1.49X	5.7	10^{16}
ReRAM-CAM [53]	1X	1X	1X	1X	10^{6}
FeRAM-CAM [31]	NA	NA	NA	NA	10^{14}
PCRAM-CAM [56]	1.81X	1.12X	1.48X	21X	10^{9}

to MTJ, PCM, and CMOS CAMs at the cost of poor endurance.

IV. EMERGING CAM ARCHITECTURES AND APPLICATIONS

With the emergence of new device and integration technologies in the nanometer technology era, the traditional CAM/AM architectures described in Section II are beginning to give way to more densely packed memory arrays and more energy-efficient cells. COM implementations described in Section III offer greater potential for advancements in density, speed, and energy efficiency. Recent developments in 3-D CAM architecture, including 3-D partitioned MLs and SLs, result in reduced wordline capacitance and overall power consumption in ternary CAMs [57], [58]. In addition, recent advances in 3-D multigate transistor technology (FinFET) have been applied to CAM architectures, aiming to reduce the leakage power inherent to deep submicrometer processes, while reducing the dynamic power primarily contributed by the parallel search hardware in CAMs [59]. Just as new memory devices and circuit/architecture level design are being explored for use in CAM, many new applications are emerging today, more than 50 years after its invention. Table 4 summarizes some of the major existing and emerging applications of CAM, as well as the type of CAM used (binary or ternary) and the role the CAM plays in the application.

Networking is among the most common uses for TCAM. It is one of the critical components that help the modern routing hardware to keep pace with the ever-growing demand for faster networking speed. CAMs have also been used frequently in search engine or database acceleration, as well as for lossless compression in multimedia applications. However, the use of CAMs in reconfigurable computing to store minimally redundant function outputs is relatively new. Using CAMs in neuromorphic associative memories is also a significant emerging application of fast parallel-search-compatible information storage with specific application in various forms of pattern recognition. Text mining applications using term frequency counting can also make effective use of binary CAMs. Finally, using CAM in wireless implants and wearables as part of a lossy data compression scheme or real-time pattern recognition promises to reduce power consumption while improving data throughput.

A. Networking

One of the most common uses for CAM is in network routers [60], [61]. Routers receive data packets and must forward them to the appropriate output based on the destination address. Routers keep track of addresses and corresponding ports in a routing table, which can contain thousands of entries. An iterative search over the entire address space would take prohibitively long time and even a faster method like binary search can be unacceptably slow, in addition to the cost of updating the table (i.e., inserting or deleting from a binary tree). Moreover, the latency increases significantly with larger tables. Over time, the size of each routing table entry has grown from 32 b for Internet Protocol version 4 (IPv4) to 128 b for Internet Protocol version 6 (IPv6).

Ternary CAMs were proposed as a way to handle the ever-growing demand of networking speed, even as routing tables and entry sizes grow. By leveraging the parallel search capabilities, the router can rapidly identify the output port given the destination address. Using the TCAM's "don't care" capability, network address ranges can be specified, e.g., 1XX1 will match 1001, 1011, 1101, and

Table 4 Summary of Common and Emerging CAM Applications, Including the Most Commonly Used CAM Type and the Role It Plays in the Application

Application	Type of CAM	Role
Networking	Ternary	Routing Tables
Neuromorphic Associative Memory	Binary/Ternary/Multi-valued	Associative search, primarily for pattern recognition
Reconfigurable Computing	Binary/Ternary or Multi-valued	Larger lookup tables, improving density / performance / energy
Analytics	Binary/Ternary	Search / Database acceleration
Text Mining	Binary	Hash table / frequency count
Implants/Wearables	Binary	Data compression / pattern recognition
Multimedia	Binary	Lossless compression

1111, making it possible to represent the address space with far fewer entries than a BCAM. Finally, unlike software data structures, a lower penalty is incurred for updating the table, an operation that can happen frequently, depending on where in the network the router is located.

CAMs have also been used to accelerate network security applications, specifically network intrusion detection systems (NIDSs) [62], [63]. These systems typically test network packets against a set of rules, requiring a large number of string matching operations. Using TCAM, patterns can be matched against incoming packets, enabling networks to operate more securely without a NIDS bottleneck.

B. Neuromorphic Associative Memory

Associative memory extends the concept of associative recall from biology. Many animals, including humans, are capable of recalling things—people, places, phrases, sounds, and smells, to name a few—given a small hint. For example, a person can easily match a cropped or degraded photograph to the original, or recall a well-known phrase even if many of the words are omitted. Neuromorphic associative memories can be used to mimic the associative recall function in hardware [64]. These biologically inspired systems can be trained for computer vision/perception, motor control, or integrating the outputs of multiple sensors to provide feedback about a system's current physical environment [65].

Such trainable networks are referred to as artificial neural networks (ANNs), and they are attractive for AM applications because of their massively parallel computing capability. These networks operate using neurons for computation, and synapses for connections, and are usually trained by iterative methods. At each synapse, input and intermediate data are weighted by a certain amount which varies between synapses and changes with training. In many cases, these networks are inherently fault-tolerant; for example, if an input image is corrupted by noise or displaced pixels, the network can still respond with the correct output, as shown in Fig. 9(a) [93].

Different ANN models exist, including recurrent and feedforward, each with its own strengths and weaknesses; examples of the first include Hopfield networks [66] or its variant, bidirectional associative memory (BAM) [67], and examples of the latter include the multilayer perceptron (MLP) [68] and cellular neural network (CNN) [69]. Still others, such as spiking neural networks (SNNs), model plasticity, or the ability to modify the strength between neuronal connections, through the use of neural spikes [spike-timing-dependent plasticity (STDP)] [70], emulating biological processes within the brain.

Both Hopfield and BAM networks enable recall of data, though BAMs allow the output size to differ from the input size, while Hopfield networks are restricted to equal input and output sizes. These networks are relatively robust despite their simplicity, but can become confused if two inputs that differ slightly map to very different outputs. Another drawback of this approach is that neurons at each stage must be connected to every neuron in the stages surrounding it. This fully connected architecture makes scalability an issue, since the total number of synapses grows exponentially with increasing numbers of neurons. Another approach, the cellular neural network (CNN), builds on the ANN model but reduces the number of synapses by allowing individual neurons to connect only to neighbors within a given radius. CNN, as illustrated in Fig. 9(b), is attractive in many application domains, and has been successfully applied as AM for problems of image recognition and classification. Fig. 9(c) and (d) shows CNN performance for a higher neighborhood radius (NR) design with a significantly higher synapse count that is more tolerant to quantization, which simplifies the distribution and storage of synaptic weights.

While these networks can be implemented in CMOS technology, many factors, especially leakage power, limit

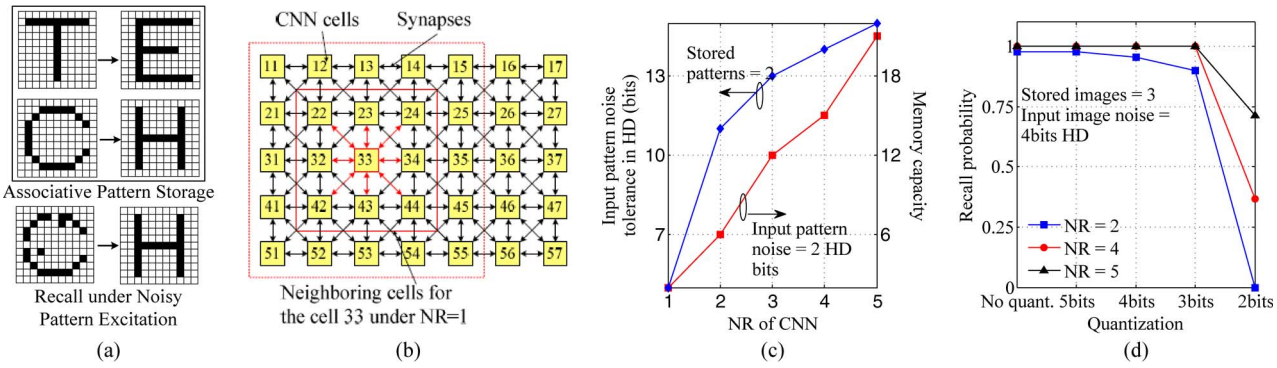

Fig. 9. *Cellular neutral network. (a) Example heteroassociative storage of patterns in an 11 × 11 CNN array, and successful recall with noisy input "C." (b) CNN array with identical locally interconnected cells, neighborhood radius (NR) =1 (NR defines the extent of local connectivity). (c) Memory capacity and input noise tolerance [in Hammimg Distance (HD) bits], at varying NR (results are for 11 × 11 CNN–AM). (d) Recall probability at varying degree of quantization and for varying NR CNN–AM. (All subfigures are reproduced with permission from [93].)*

their scalability. Instead, advances in nanoscale memory technologies are enabling development of lower power, more robust ANNs in hardware. Memristors have been used, both in hardware emulation [65] and experiments using true memristive devices [71], [72], to demonstrate the ANNs ability to learn simple associations based on learning methods akin to those used within mammalian brains. Similarly, advances in tunnel FET (TFET) technology have improved CNN-based associative memories by enabling the synapse to operate in an ultralow-power state [93]. The use of PCM-based synapses for high-density spike neural network architectures has also been explored [73]. In these cases, the differences between network types, the number of neurons and synapses simulated, emulated, or fabricated, the sizes of the data sets and methods used for training, as well as differing process technologies, make a direct quantitative comparison difficult. In general, however, leveraging the unique properties of these nanoscale memory devices can lead to low-power operation, faster switching times, and higher output resistance, paving the way for higher connectivity networks with greater noise tolerance and the potential to model entire mammalian brains with billions of neurons and trillions of synapses.

C. Reconfigurable Computing with CAM

Reconfigurable computing architectures can utilize properties of CAM for efficient function representation and evaluation. Traditional reconfigurable architectures such as FPGA make extensive use of multiple-input–single-output lookup tables (LUTs) with programmable interconnects (PIs) for spatial evaluation of a function. However, such a model requires an elaborate programmable interconnect network, which leads to significant design overhead and poor scalability across process technology. Moreover, the memory required to represent function outputs grows exponentially with increasing input sizes, making space-saving alternatives attractive.

To mitigate the limitation imposed by programmable interconnects, the concept of PLA-based logic evaluation has been explored which can drastically reduce the memory requirements in a reconfigurable computing framework [1]. Specifically, this method has been applied in the context of memory-based computing (MBC), a mixed spatial/temporal reconfigurable computing paradigm which uses memory for traditional data storage, as well as for computation using LUTs. The latter is achieved by representing a function in terms of its input space instead of its output space; function evaluation is performed for a given input by searching the input space and returning logic "1" on a match and logic "0" otherwise. This parallel search requirement makes CAMs an attractive option for storing function results. Fig. 10 shows an example using a three-literal sum of products for an arbitrary Boolean function. With a normal LUT representation, the three terms can form 2^3 combinations, requiring eight LUT entries. On the other hand, using a binary CAM, only combinations resulting in a true output are stored; therefore, finding a match or not results in an equivalent output of logic "1" or "0," respectively. This method removes redundancy from the table storage and effectively prevents exponential growth of table size.

$$f(X_1, X_2, X_3) = X_1X_2X_3 + X_1\overline{X_2}X_3 + \overline{X_1}X_2X_3$$

$X_1X_2X_3$	Z
000	0
001	0
010	0
011	1
100	0
101	1
110	0
111	1

LUT

$X_1X_2X_3$
011
101
111

BCAM

$X_1X_2X_3$
011
1X1

TCAM

Fig. 10. *Example application of BCAM and TCAM for storing an arbitrary logic function. Unlike the LUT, the CAM size does not increase exponentially with the input bit width.*

In many cases, such as the BCAM, two or more of the entries may differ by only one bit, as illustrated in Fig. 10. In such cases, further optimizations can be made using ternary CAMs to exploit the "don't care" state. For example, if input combinations 101 and 111 both result in a true output, a ternary CAM can simply store 1X1, reducing the number of entries in the table. In a more extreme example, a four-input OR gate can be represented by only four terms: 1XXX, X1XX, XX1X, and XXX1, as opposed to the 16 entries required in a LUT-based representation. The TCAM representation necessary for this space reduction is illustrated in Fig. 11(a). Fig. 11(b) shows the exponential savings in memory requirement achieved by this approach over a conventional LUT-based implementation for several benchmark designs.

Although the viability of such a framework was only verified for SRAM, we note that the same concept can be extended to nanoscale devices, which are amenable to binary CAM design. Moreover, it is worth noting that the emerging nanoscale devices, which hold promise for implementing three or multivalued logic can further reduce the number of bits required to implement the CAM. For example, a three-valued CAM can reduce the memory requirement, as shown in Fig. 11(b), by half.

D. CAM in Analytics and Data Mining

Given their rapid search capabilities, CAMs have been applied extensively to problems requiring fast text search and pattern matching. Over the years, this has included text search for database coprocessors [74], data compression [28], [75], [76], and even search engine accelerators [77], [78]. Building on this, another emerging application for CAM is in the field of analytics and data mining. One common kernel found in nearly all data mining applications relies on the counting of terms in a set of documents.

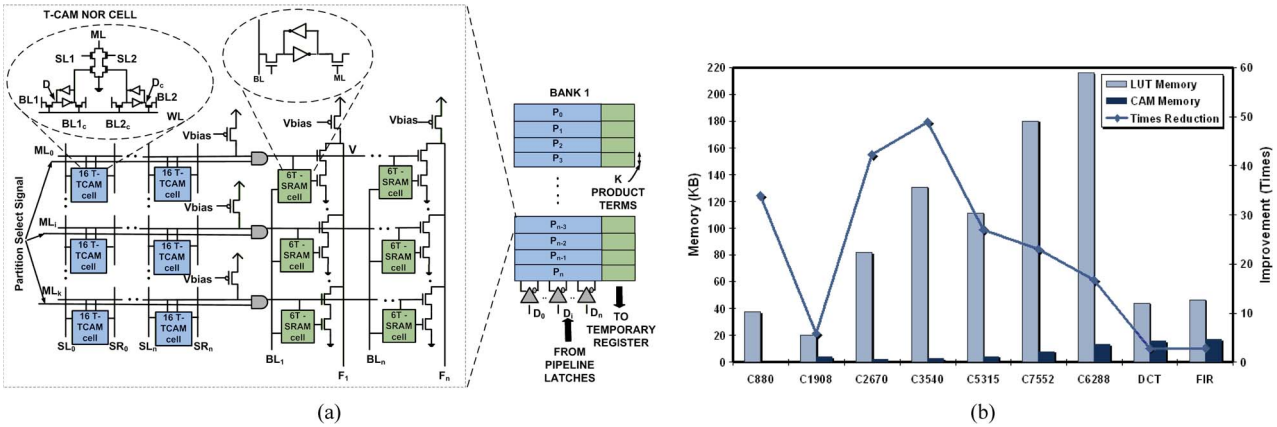

Fig. 11. *(a) Ternary CAM-based realization of the partitions in a memory-based computing (MBC) framework. (b) Comparison of memory requirements between LUT and CAM-based implementations [1].*

CAMs are uniquely suited to such a task, enabling fast search times for existing terms in the memory. Fig. 12 demonstrates the application of CAM to the term frequency, inverse document frequency (TF–IDF) kernel, which attempts to rank the importance of words based on their frequency: words appearing commonly in one document but rarely in others are considered more important than a word that appears frequently in all documents.

To implement the TF–IDF frequency count in hardware, the CAM operates as a kind of dictionary. If the CAM is searched for a given term and no match is found, it can be added to an available memory location. If instead a match is found, the CAM responds with the address. This is transferred to a control unit, which interfaces the CAM with a standard SRAM array; the address selects the location in the SRAM storing the frequency counts for the term. This value is automatically read, incremented using a shared adder, and written back to the same location.

Before counting terms, however, it is common to remove stems or suffixes, like "-ing" or "-ment" from terms, as the root generally retains a similar meaning. By limiting the bit-width for each CAM entry to 30 b, enabling the storage of six ASCII letters (5 b/character), any words longer than six characters are truncated, resulting in a rudimentary form of stemming. While a more complex stemming algorithm will generally do a better job, preliminary analysis with rank biased overlap (RBO) [79] shows little difference in TF–IDF score lists generated from input text based on this method, and the more complex Porter stemming algorithm [80] for a large body of English text. With a persistence parameter of 0.95 (that is, a 95% chance a person comparing the lists will compare two entries and continue on to the next line), the lists share a similarity index of 92%; for $p = 0.90$, this increases to 95%; for $p = 0.75$, they are nearly identical, at 99% similarity. In many applications, especially mining large data sets on a low power budget, this method offers excellent returns with little tradeoff in accuracy.

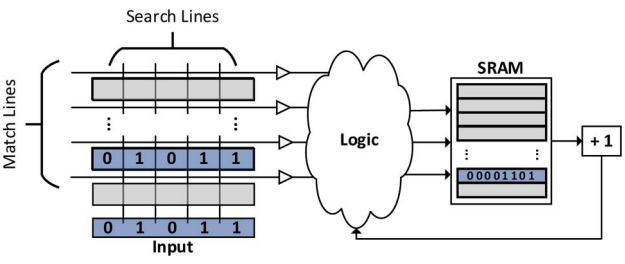

Fig. 12. *Overview of the text mining accelerator hardware for counting term frequency. A binary CAM is used like a hash table; logic connects this to an SRAM array, which tracks the count of a given term. Upon finding a match, the corresponding count increments.*

E. CAM in Wearable/Implantable Systems

The explosion of wearables, implants, and the Internet of Things (IoT) has made sensors a ubiquitous part of life. Data acquisition from such devices may outpace on-chip compute or wireless telemetry capabilities under a practical power budget, especially in the case of implants, where reducing power consumption is a critical design driver. For example, implants designed to record neural signals may need to transmit data to an external device for processing, but transmission of the entire signal often consumes a prohibitively large amount of power, requiring a high degree of compression. Using pattern recognition techniques, different forms of neural spikes can be referenced with a specific sequence of bits. This vocabulary-based data compression before wireless transmission has been explored earlier for neural signal compression [81]. In the context of data compression in a brain implant, use of a CAM can be highly effective when attempting to match extracted features to a given pattern to identify specific sequences in brain activity. Moreover, for some lossless compression

hardware, CAMs are considered among the most efficient architectures for efficient dictionary-based pattern matching [28]. CAMs can simplify compression hardware, and reduce latency and power consumption, making them attractive for use in implants, wearables, and IoT devices.

CAM structures also allow efficient hardware implementation of closed-loop neural prosthesis systems, which rely on detecting behaviorally meaningful patterns from recorded neural data to trigger specific preventive/corrective/assistive actions (e.g., drug delivery). Such a computational task for online, real-time neural pattern recognition requires both efficient signal processing algorithms and special purpose customized hardware. A neural pattern recognition algorithm stores a set of behaviorally meaningful patterns (called alphabets) in a vocabulary, then searches for a match with an input recorded waveform [82]. An efficient implementation of such an algorithm realizes the pattern search and update module with a modified version of the hierarchical SL-based CAM [83] architecture, as shown in Fig. 13 [95]. It uses a tunable "distance metric" based on equality of variable number of parameters with different precision; the "first match" strategy by implementing a priority encoder; the "least recently used replacement"; and "binary encoding" options. By arranging the vocabulary tables as a memory array, it significantly saves die area and power. Though the incorporation of matching functionality increases the area by almost 2x per cell (compared to a 6-T SRAM cell), the overall savings in terms of power and area compared to an array of registers is nearly 10x. The data in the memory need to be updated as new alphabet symbols are generated. Upon detection of a match between the incoming symbol and the stored data, it returns the memory address, which is essentially the alphabet symbol. Since the match does not need to be perfect, a hierarchical search can be used, where local SLs are used to search only a part of the array, which is selected based on the duration of the spike and channel number. In order to incorporate tunability of the distance metric, a ternary CAM principle can be used. By setting the less significant bits to "don't care," the distance metric threshold can be varied easily. This optimization is particularly suitable for partial matching of certain features during neural pattern recognition.

Furthermore, CAM is being increasingly employed in the domain of multimedia applications. Compression plays a major role in multimedia applications such as audio and video. Dictionary-based lossless compression routines such as LZ77 benefit greatly from the rapid parallel lookup operations offered by CAMs [28]. Similar approaches have been applied to lossy JPEG compression as well.

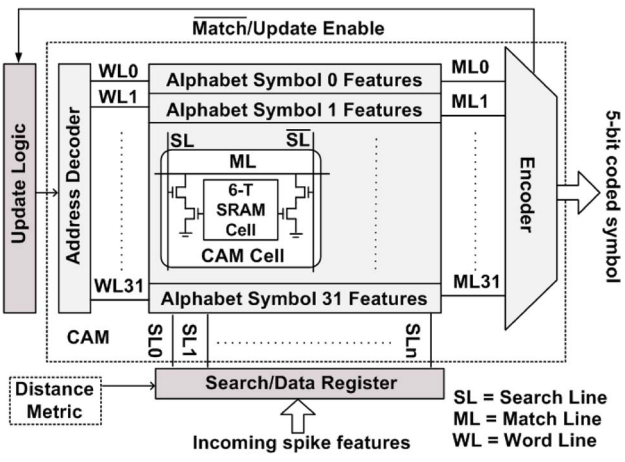

Fig. 13. *Vocabulary search/update module in a neural pattern recognition algorithm implemented using a modified version of CAM [95].*

V. TRANSACTIONAL MEMORY

A. Background

The concept of transactional memory (TM) has been around for over 20 years [84]. Initially implemented in software (i.e., STM), it has only recently entered mainstream computing devices directly in hardware, referred to as HTM [3], [4]. Unlike the ubiquitous random access memory, TM is optimized for use in multicore or multithreaded environments. With the rapid increase in core count and the push toward parallel programming, complexities associated with concurrent thread execution have made it difficult for software developers to create efficient, highly multithreaded applications using only explicitly defined memory locks. The primary concern is the use of shared memory, and ensuring that the same state is visible to all threads, an issue referred to as memory consistency. Using locks, developers can make certain that no two threads can modify the same memory location simultaneously, thereby maintaining memory consistency. However, this can be problematic when the lock granularity is coarse, e.g., along-running function might hold the lock until completion, rather than only during memory accesses. By introducing more fine-grained locks, the likelihood of stalling another thread for a long time decreases, but the complexity of the code and possibility of inadvertent deadlocks (when two threads block each other in stalemate) increases.

With a TM system, there is no need for the programmer to lock memory regions. In its place, the concept of atomic execution has been introduced, which defines regions of code with load and store operations that are grouped and executed together, in isolation from other threads. Such regions represent a single transaction, which can result in either success or failure when executed. In the first case, memory operations complete and any resulting variable changes are written to the memory, changing the system state. In the second, the transaction fails due to a conflict with other transactions, and no changes are made to the system state.

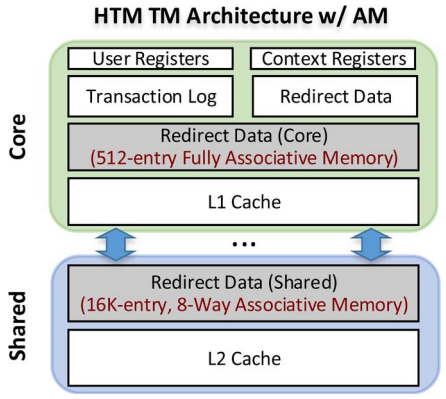

Fig. 14. *Architecture of SUV–TM [72], which makes use of AM in the critical path of the memory access.*

Though the underlying register memory is still standard SRAM, extra hardware is introduced, which greatly simplifies concurrent access to the shared memory locations. Depending on specific implementations, different hardware modules are used for tracking transactions. In general, a set of transaction registers is used to store the current system state prior to execution. A transaction log maintains a list of transactions in an AM, which is used by the abort and commit logic [84]. A prominent example is the SUV–TM [85] system, which makes use of core-specific and shared redirect tables, including a 512-entry fully associative cache, and a 16K (16 384) entry, eight-way associative cache shared between cores, as illustrated in Fig. 14. The core-specific cache lies in the critical path for memory accesses, making the zero-latency access important for efficient program execution. In other implementations, different methods of conflict detection can be used, including eager (or active) and lazy, and different methods of conflict management and resolution.

B. Recent Trends in HTM

Moving toward HTM has allowed significant performance improvements for highly concurrent software programs compared with STM. The implementation of HTM in the IBM BlueGene/Q processors promise to improve software scalability for concurrent programs running on a 20-petaflop supercomputer [3]. More recently, the fourth-generation Intel Core (Haswell) processors introduce HTM to mainstream consumer electronics in the form of transactional synchronization extensions (TSX), including one form, hardware lock elision (HLE), that is backward compatible with current processors, and a second form, restricted transactional memory (RTM), which offers a full hardware implementation but does not support backward compatibility [4].

However, several challenges with HTM remain to be addressed in future implementations. Table 5 notes the benefits and drawbacks of both HTM and STM. As a compromise, new research efforts have been exploring hybrid transactional (HyTM) memories [86], [87] with built-in redundancies to ensure that all transactions eventually execute and complete. This deviates from HTM-only implementations like Haswell's RTM, which does not provide a guarantee that all transactions will execute to completion [86].

VI. FUTURE DIRECTIONS

A. Scalability

As with any memory structure, scaling issues in terms of size exist for content-addressable memory and AM. The addition of the parallel search requires high hardware overhead and reduces the cell density compared to standard RAMs. With parallel search, there arises problems of significant dynamic power consumption, in addition to the leakage power increase (due to additional device counts) with decreasing feature sizes. The highly capacitive SLs typically used in CAM architectures are not very amenable to scaling to larger sizes due to their large negative impact on energy efficiency. Consequently, a large body of research has focused on reducing the energy consumption without compromising the throughput using circuit/architecture-level, or codesign approaches [3]–[7]. In the past, dynamic CMOS circuit techniques have been explored in the context of designing low-power and low-cost CAMs of large size [5]. However, these designs can suffer from low noise margins and charge sharing leading to poor robustness of operation [5]. On the other hand, emerging nanoscale devices enable higher density, larger size CAMs. The nonvolatile nature of most non-CMOS memory devices can drastically reduce leakage power. Moreover, CAM implementations with most of these devices feature lower dynamic power during search/update operations than their CMOS counterparts. The combined effect of higher density and lower power creates the possibility of larger size CAM

Table 5 Comparison of Key Advantages and Disadvantages Between STM and HTM

	Advantages	Disadvantages
STM	• No practical limit	• High overhead • Level of concurrency dependent on lock granularity
HTM	• Low overhead • More concurrency (w/o extra software complexity)	• Limited hardware resource

implementations, which can be beneficial in many application domains.

B. Application Space

Despite the design tradeoffs involving energy, performance and area, CAM/AMs play a pivotal role in many speed-critical applications. Emerging applications, including neuromorphic AM, reconfigurable computing platforms, big data analytics, and even wearables/implants will continue to grow and rely on ever-faster search and pattern matching capabilities of nanoscale CAM/AM architectures without a significant increase in die area and power consumption. Innovations in integration technologies (e.g., 3-D integration) are expected to benefit CAM implementations—in particular improving the energy efficiency and integration density—similar to their SRAM counterparts. However, the positive impact of integration and device technologies needs to be complemented with efficient circuit-architecture level design. Hence, there is a pressing need to explore new design methodologies for CAM to cater to next-generation hardware and software applications in these growing sectors.

Memory-based computing models and architectures appear very promising for reconfigurable computing with the emerging nanoscale devices. Both purely spatial reconfigurable computing platforms like field-programmable gate arrays (FPGAs) [1], [91], [92] as well as various temporal computing platforms [2], [88], [89] can leverage the unique properties of nanoscale CAM structures and their associative search capabilities. Compared to the implementation of standard island-style FPGA architecture using nanoscale devices, time-multiplexed memory-based computing that uses dense CAM structures to implement logic functions (in contrast with 1-D LUTs stored in SRAM arrays) can be more effective in preserving the high integration density of nanoscale memory and improving performance due to a large reduction in programmable interconnect resources [2], as shown in Fig. 15(a). Fig. 15(b) and (c) shows comparison of performance and energy-delay product (EDP), respectively, between an STTRAM-based MBC and a CMOS FPGA for several benchmark applications. Without the bottleneck of a vast PI network, the MBC architecture is free to leverage the unique properties of emerging nonvolatile memory (NVM) technologies, not only for CAM-based function evaluation, but also for general data storage. In several case studies, circuit/architecture/application mapping co-optimization approaches were found to significantly improve the energy-delay product (EDP) by exploiting the READ- or recall-dominant access pattern in a reconfigurable computing platform [90], [94]. The general principle followed here is to optimize an STTRAM or ReRAM-based CAM cell for READ energy and/or READ noise margin, while affecting infrequent WRITE/UPDATE operation. Multivalued CAM cells [43], [52], which are easier to realize in many emerging memory technologies than in CMOS, hold promise for reconfigurable computing frameworks as well as other applications. As nanoscale memories continue to grow in reliability, density, and energy efficiency, frameworks

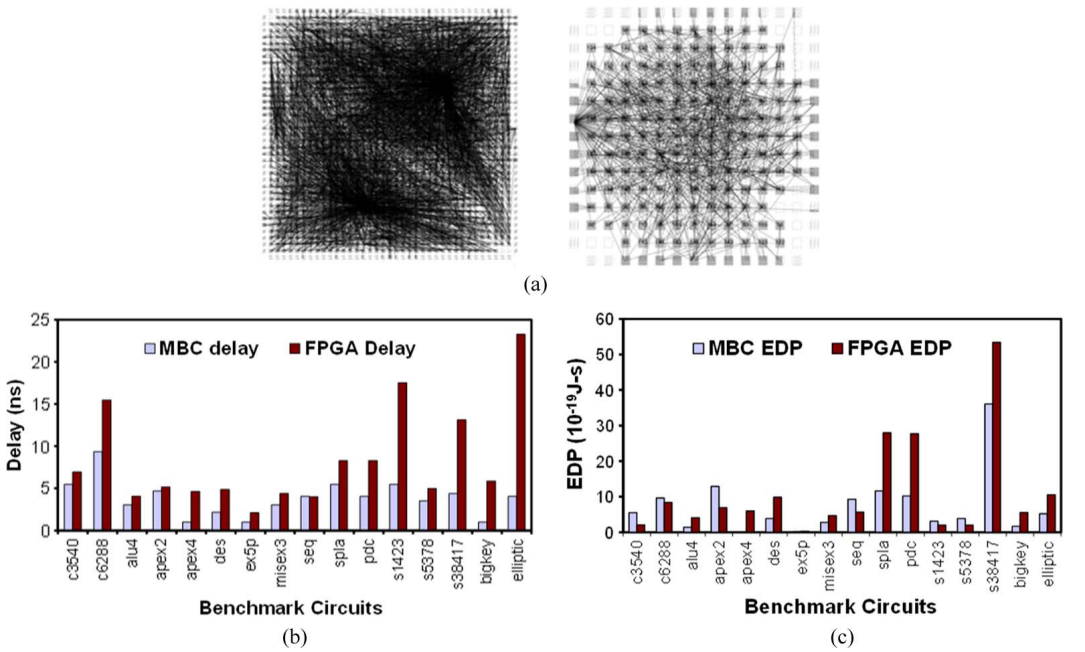

Fig. 15. *(a) Efficient combination of spatial and temporal computing enables vast reduction in programmable interconnect overhead [6]. Improvements in (b) delay and (c) EDP for the memory-based computing framework over a baseline FPGA for a number of benchmark circuits [90].*

similar to MBC, which heavily rely on computations mapped to memory, will see proportional improvements in these critical design drivers.

As COM implementations in various forms and sizes with emerging NVM technologies mature over time, they will be considered powerful building blocks in the hardware implementations of various computer systems. The feasibility of integrating permanent nonvolatile storage with computing technologies will undoubtedly bring about a major paradigm shift. As the distinction between computation and storage is blurred, the prospect of computing in memory appears realistic, especially for data-intensive applications, where energy dissipated in on-chip computation constitutes only a small fraction of the total energy consumption [2]. The primary contribution comes from transporting data between off-chip memory and on-chip computing elements—a limitation referred to as the Von-Neumann bottleneck. An effective solution that has been explored to mitigate this bottleneck is in-memory computing that enables off-chip computing in NVM arrays, where the data reside permanently. The concept of in-memory computing has been proposed in numerous earlier works [8], [9] in the context of computing in the main memory. However, with the emergence of fast and reliable nonvolatile storage, recent in-memory compute architectures have targeted these NVM technologies [1], [2], [6]. The benefits of such architectures are particularly noteworthy for pattern recognition, data mining, and other analytics applications, which traditionally involve moving a large volume of data through the memory hierarchy. For these applications and others, LUT-dominated function mapping can leverage multivalued CAM implementations for dramatic reductions in memory size, as mentioned in Section V. Many data-intensive kernels such as pattern recognition, filtering, or text analytics can exploit the fast, parallel search of information stored in associative arrays, making content-based addressing and associative search highly attractive in the realm of in-memory computing.

VII. SUMMARY

CAM/AM are specialized memory arrays, where READ/WRITE access is guided by content. They are suitable for fast parallel search of stored information. They come in many forms and are attractive in many application domains. With the emergence of novel nanoscale devices, in particular, an array of promising nonvolatile information storage devices such as STTRAM, DWM, ReRAM, and FeRAM, new, often highly energy-efficient and scalable CAM/AM implementations using these devices are emerging. In this paper, we have presented a survey and comparative analysis of emerging CAM/AM implementations. We have also discussed the application space that can greatly benefit from these implementations.

While emerging nanoscale devices can potentially extend the realization of CAM/AM in beyond-CMOS computing platforms, they also show tremendous promise in extending their application into new domains. They include low-power, memory-centric reconfigurable computing, signal processing in wearables/implants, neuromorphic AM—suitable for non-Boolean computing, HTM using AM structures that allow concurrent access to the same memory locations, and various types of analytics tasks. The application space of CAM/AM is growing at a steady pace due to the increasing demand of content-based information retrieval and fast associative search in many areas. The combination of emerging opportunities for CAM/AM applications and innovations in nanoscale devices, integration technologies, as well as circuit-architecture level design is expected to fuel significant future research in these areas.

Acknowledgment

The authors would like to thank Dr. S. Mukhopadhyay (Georgia Institute of Technology), Dr. S. Paul (Intel Corporation) and as well as researchers in the Nanoscape Research Lab in the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH, USA, for their valuable inputs.

REFERENCES

[1] S. Paul and S. Bhunia, "Reconfigurable computing using content addressable memory for improved performance and resource usage," in *Proc. 45th Annu. Design Autom. Conf.*, 2008, pp. 786–791.

[2] S. Paul, A. Krishna, W. Qian, R. Karam, and S. Bhunia, "MAHA: An energy-efficient malleable hardware accelerator for data-intensive applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 6, pp. 1005–1016, Jun. 2015.

[3] P. Bright, "IBM's new transactional memory: Make-or-break time for multithreaded revolution," *ARS Technica*, Aug. 2011. [Online]. Available: http://arstechnica.com/gadgets/2011/08/ibms-new-transactional-memory-make-or-break-time-for-multithreaded-revolution/

[4] P. Bright, "Transactional memory going mainstream with Intel Haswell," *ARS Technica*, Feb. 2012. [Online]. Available: http://arstechnica.com/business/2012/02/transactional-memory-going-mainstream-with-intel-haswell/

[5] M. R. Stan, P. D. Franzon, S. C. Goldstein, J. C. Lach, and M. M. Ziegler, "Molecular electronics: From devices and interconnect to circuits and architecture," *Proc. IEEE*, vol. 91, no. 11, pp. 1940–1957, Nov. 2003.

[6] S. Paul and S. Bhunia, "Computing with nanoscale memory: Model and architecture," in *Proc. IEEE/ACM Int. Symp. Nanoscale Architect.*, 2009, DOI: 10.1109/NANOARCH.2009.5226362.

[7] K. Kim and Y. J. Song, "Current and future high density FRAM technology," *Integr. Ferroelectr.*, vol. 61, no. 1, pp. 3–15, 2004.

[8] T. W. Andre *et al.*, "A 4-Mb 0.18-μm 1T1MTJ toggle MRAM with balanced three input sensing scheme and locally mirrored unidirectional write drivers," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 301–309, Jan. 2005.

[9] W. Y. Cho *et al.*, "A 0.18-μm 3.0-v 64-mb nonvolatile phase-transition random access memory (PRAM)," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 293–300, Jan. 2005.

[10] J. Ward *et al.*, "A nonvolatile nanoelectromechanical memory element utilizing a fabric of carbon nanotubes," in *Proc. Non-Volatile Memory Technol. Symp.*, 2004, Nov. 2004, pp. 34–38.

[11] T. Rueckes *et al.*, "Carbon nanotube-based nonvolatile random access memory for molecular computing," *Science*, vol. 289, no. 5476, pp. 94–97, 2000.

[12] T. Kohonen, *Content-Addressable Memories*. New York, NY, USA: Springer-Verlag, 1980.

[13] A. Hanlon, "Content-addressable and associative memory systems," *IEEE Trans. Electron. Comput.*, vol. EC-15, no. 4, pp. 509–521, Aug. 1966.

[14] W. R. Stevens, *TCP/IP Illustrated (1): The Protocols*. Boston, MA, USA: Addison-Wesley, 1993.

[15] V. Fuller, T. Li, J. Yu, and K. Varadhan, "Classless inter-domain routing (CIDR):

An address assignment and aggregation strategy," 1993.

[16] A. T. Do *et al.*, "0.77 fj/bit/search content addressable memory using small match line swing and automated background checking scheme for variation tolerance," *IEEE J. Solid-State Circuits*, vol. 49, no. 7, pp. 1487–1498, Jul. 2014.

[17] G. Wang and N.-F. Tzeng, "TCAM-based forwarding engine with minimum independent prefix set (MIPS) for fast updating," in *Proc. IEEE Int. Conf. Commun.*, 2006, vol. 1, pp. 103–109.

[18] D. Shah and P. Gupta, "Fast updating algorithms for TCAMs," *IEEE Micro*, vol. 21, no. 1, pp. 36–47, 2001.

[19] P. Gillingham, "Circuit and method for performing variable width searches in a content addressable memory," U.S. Patent 6 708 250, Mar. 16, 2004.

[20] G. Kasai, Y. Takarabe, K. Furumi, and M. Yoneda, "200 MHz/200 MSPS 3.2 W at 1.5 V Vdd, 9.4 Mbits ternary CAM with new charge injection match detect circuits and bank selection scheme," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2003, pp. 387–390.

[21] C. A. Zukowski and S.-Y. Wang, "Use of selective precharge for low-power content-addressable memories," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1997, vol. 3, pp. 1788–1791.

[22] I. Arsovski, T. Chandler, and A. Sheikholeslami, "A ternary content-addressable memory (TCAM) based on 4T static storage and including a current-race sensing scheme," *IEEE J. Solid-State Circuits*, vol. 38, no. 1, pp. 155–158, Jan. 2003.

[23] N. Mohan and M. Sachdev, "Low power dual matchline ternary content addressable memory," in *Proc. Int. Symp. Circuits Syst.*, 2004, vol. 2, DOI: 10.1109/ISCAS.2004.1329351.

[24] O. Tyshchenko and A. Sheikholeslami, "Match sensing using match-line stability in content-addressable memories (CAM)," *IEEE J. Solid-State Circuits*, vol. 43, no. 9, pp. 1972–1981, Sep. 2008.

[25] I. Arsovski, T. Hebig, D. Dobson, and R. Wistort, "1 Gsearch/sec ternary content addressable memory compiler with silicon-aware early-predict late-correct single-ended sensing," in *Proc. Symp. VLSI Circuits*, Jun. 2012, pp. 116–117.

[26] I. Arsovski and R. Wistort, "Self-referenced sense amplifier for across-chip-variation immune sensing in high-performance content-addressable memories," in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2006, pp. 453–456.

[27] I. Arsovski and A. Sheikholeslami, "A mismatch-dependent power allocation technique for match-line sensing in content-addressable memories," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1958–1966, Nov. 2003.

[28] K.-J. Lin and C.-W. Wu, "A low-power CAM design for LZ data compression," *IEEE Trans. Comput.*, vol. 49, no. 10, pp. 1139–1145, Oct. 2000.

[29] P.-T. Huang and W. Hwang, "A 65 nm 0.165 fj/bit/search 256 144 TCAM macro design for IPv6 lookup tables," *IEEE J. Solid-State Circuits*, vol. 46, no. 2, pp. 507–519, Feb. 2011.

[30] C.-S. Lin, J.-C. Chang, and B.-D. Liu, "A low-power precomputation-based fully parallel content-addressable memory," *IEEE J. Solid-State Circuits*, vol. 38, no. 4, pp. 654–662, Apr. 2003.

[31] I. Bayram and Y. Chen, "NV-TCAM: Alternative interests and practices in NVM designs," in *Proc. IEEE Non-Volatile Memory Syst. Appl. Symp.*, 2014, DOI: 10.1109/NVMSA.2014.6927206.

[32] L. O. Chua, "Memristor—The missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, Sep. 1971.

[33] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[34] R. Williams, "How we found the missing memristor," *IEEE Spectrum*, vol. 45, no. 12, pp. 28–35, Dec. 2008.

[35] Y. N. Joglekar and S. J. Wolf, "The elusive memristor: Properties of basic electrical circuits," *Eur. J. Phys.*, vol. 30, no. 4, pp. 661–675, 2009.

[36] Y. Arimoto and H. Ishiwara, "Current status of ferroelectric random-access memory," *MRS Bull.*, vol. 29, no. 11, pp. 823–828, 2004.

[37] H.-S. Wong *et al.*, "Metal-oxide RRAM," *Proc. IEEE*, vol. 100, no. 6, pp. 1951–1970, Jun. 2012.

[38] S. S. Parkin, M. Hayashi, and L. Thomas, "Magnetic domain-wall racetrack memory," *Science*, vol. 320, no. 5873, pp. 190–194, 2008.

[39] Y. Zhang, W. Zhao, J.-O. Klein, D. Ravelsona, and C. Chappert, "Ultra-high density content addressable memory based on current induced domain wall motion in magnetic track," *IEEE Trans. Magn.*, vol. 48, no. 11, pp. 3219–3222, Nov. 2012.

[40] C.-C. Wang, J.-S. Wang, and C. Yeh, "High-speed and low-power design techniques for TCAM macros," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 530–540, Feb. 2008.

[41] W. W. Fung, "Low power circuits for multiple match resolution and detection in ternary CAMs," M.S. thesis, Dept. Electr. Comput. Eng., Univ. Waterloo, Waterloo, ON, Canada, 2004.

[42] T. Uemura and M. Yamamoto, "Three-valued magnetic tunnel junction for nonvolatile ternary content addressable memory application," *J. Appl. Phys.*, vol. 104, no. 12, 2008, Art. ID. 123911.

[43] A. Kamisawa, "Ferroelectric memory with multiple-value storage states," U.S. Patent 5 291 436, Mar. 1, 1994.

[44] M. Hosomi *et al.*, "A novel nonvolatile memory with spin torque transfer magnetization switching: Spin-RAM," in *Tech. Dig. IEEE Int. Electron Devices Meeting*, 2005, pp. 459–462.

[45] S. Matsunaga *et al.*, "Fine-grained power-gating scheme of a metal-oxide-semiconductor and magnetic-tunnel-junction-hybrid bit-serial ternary content-addressable memory," *Jpn. J. Appl. Phys.*, vol. 49, no. 4S, 2010, Art. ID. 04DM05.

[46] S. Matsunaga, A. Katsumata, M. Natsui, and T. Hanyu, "Design of a low-energy nonvolatile fully-parallel ternary CAM using a two-level segmented match-line scheme," in *Proc. 41st IEEE Int. Symp. Multiple-Valued Logic*, 2011, pp. 99–104.

[47] L.-Y. Huang *et al.*, "ReRAM-based 4T2R nonvolatile TCAM with 7× NVM-stress reduction, 4× improvement in speed-wordlength-capacity for normally-off instant-on filter-based search engines used in big-data processing," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2014, DOI: 10.1109/VLSIC.2014.6858404.

[48] H. Wu, F. Lombardi, and J. Han, "A PCM-based TCAM cell using NDR," in *Proc. IEEE/ACM Int. Symp. Nanoscale Architect.*, Jul. 2013, pp. 89–94.

[49] S. Murotiya and A. Gupta, "CNTFET based design of content addressable memory cells," in *Proc. 4th Int. Conf. Comput. Commun. Technol.*, Sep. 2013, DOI: 10.1109/ICCCT.2013.6749593.

[50] D. Das, A. Roy, and H. Rahaman, "Design of content addressable memory architecture using carbon nanotube field effect transistors," in *Progress in VLSI Design and Test*, vol. 7373, H. Rahaman, S. Chattopadhyay, and S. Chattopadhyay, Eds. Berlin, Germany: Springer-Verlag, 2012, pp. 233–242, ser. Lecture Notes in Computer Science. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31494-0_27

[51] S. Kang *et al.*, "A 0.1-μm 1.8-V 256-MB phase-change random access memory (PRAM) with 66-MHz synchronous burst-read operation," *IEEE J. Solid-State Circuits*, vol. 42, no. 1, pp. 210–218, Jan. 2007.

[52] K. Degawa *et al.*, "A high-density ternary content-addressable memory using single-electron transistors," in *Proc. IEEE 36th Int. Symp. Multiple-Valued Logic*, 2006, DOI: 10.1109/ISMVL.2006.6.

[53] M.-F. Chang *et al.*, "17.5 A 3T1R nonvolatile TCAM using MLC ReRAM with sub-1ns search time," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2015, DOI: 10.1109/ISSCC.2015.7063054.

[54] K. Nii *et al.*, "13.6 A 28 nm 400 MHz 4-parallel 1.6 Gsearch/s 80MB ternary CAM," in *IEEE Int. IEEE Solid-State Circuits Conf. Dig. Tech. Papers*, 2014, pp. 240–241.

[55] S. Matsunaga *et al.*, "A 3.14 um 2 4T-2MTJ-cell fully parallel TCAM based on nonvolatile logic-in-memory architecture," in *Proc. IEEE Symp. VLSI Circuits*, 2012, pp. 44–45.

[56] J. Li, R. K. Montoye, M. Ishii, and L. Chang, "1 MB 0.41μm^2 2T-2R cell nonvolatile TCAM with two-bit encoding and clocked self-referenced sensing," *IEEE J. Solid-State Circuits*, vol. 49, no. 4, pp. 896–907, Apr. 2014.

[57] Y.-J. Hu, J.-F. Li, and Y.-J. Huang, "3-D content addressable memory architectures," in *Proc. IEEE Int. Workshop Memory Technol. Design Testing*, 2009, pp. 59–64.

[58] E. C. Oh and P. D. Franzon, "Design considerations and benefits of three-dimensional ternary content addressable memory," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2007, pp. 591–594.

[59] D. Bhattacharya, A. Bhoj, and N. Jha, "Design of efficient content addressable memories in high-performance FinFET technology," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 5, pp. 963–967, May 2015.

[60] A. McAuley and P. Francis, "Fast routing table lookup using CAMs," in *Proc. IEEE 12th Annu. Joint Conf. IEEE Comput. Commun. Soc., Netw.: Found. Future*, 1993, vol. 3, pp. 1382–1391.

[61] F. Yu, R. Katz, and T. Lakshman, "Gigabit rate packet pattern-matching using TCAM," in *Proc. 12th IEEE Int. Conf. Netw. Protocols*, Oct. 2004, pp. 174–183.

[62] I. Sourdis and D. Pnevmatikatos, "Pre-decoded CAMs for efficient and high-speed NIDS pattern matching," in *Proc. 12th Annu. IEEE Symp. Field-Programm. Custom Comput. Mach.*, Apr. 2004, pp. 258–267.

[63] S. Yusuf and W. Luk, "Bitwise optimised CAM for network intrusion detection systems," in *Proc. IEEE Int. Conf. Field Programm. Logic Appl.*, 2005, pp. 444–449.

[64] Y. V. Pershin and M. Di Ventra, "Neuromorphic, digital, quantum computation with memory circuit elements," *Proc. IEEE*, vol. 100, no. 6, pp. 2071–2080, Jun. 2012.

[65] Y. Pershin and M. D. Ventra, "Experimental demonstration of associative memory with memristive neural networks," *Neural Netw.*, vol. 23, no. 7, pp. 881–886, 2010.

[66] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci.*, vol. 79, no. 8, pp. 2554–2558, 1982.

[67] B. Kosko, "Bidirectional associative memories," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-18, no. 1, pp. 49–60, 1988.

[68] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, classification," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 683–697, Sep. 1992.

[69] L. O. Chua and T. Roska, *Cellular Neural Networks and Visual Computing: Foundations and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[70] S. Song, K. D. Miller, and L. F. Abbott, "Competitive Hebbian learning through spike-timing-dependent synaptic plasticity," *Nature Neurosci.*, vol. 3, no. 9, pp. 919–926, 2000.

[71] S. Yu, Y. Wu, R. Jeyasingh, D. Kuzum, and H.-S. Wong, "An electronic synapse device based on metal oxide resistive switching memory for neuromorphic computation," *IEEE Trans. Electron Devices*, vol. 58, no. 8, pp. 2729–2737, Aug. 2011.

[72] S. H. Jo *et al.*, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Lett.*, vol. 10, no. 4, pp. 1297–1301, 2010.

[73] M. Suri *et al.*, "Phase change memory as synapse for ultra-dense neuromorphic systems: Application to complex visual pattern extraction," in *Proc. IEEE Int. Electron Devices Meeting*, 2011, DOI: 10.1109/IEDM.2011.6131488.

[74] J. P. Wade and C. G. Sodini, "A ternary content addressable search engine," *IEEE J. Solid-State Circuits*, vol. 24, no. 4, pp. 1003–1013, Apr. 1989.

[75] C.-Y. Lee and R.-Y. Yang, "High-throughput data compressor designs using content addressable memory," *Inst. Electr. Eng. Proc.—Circuits Devices Syst.*, vol. 142, no. 1, pp. 69–73, 1995.

[76] S. Jones, "100 mbit/s adaptive data compressor design using selectively shiftable content-addressable memory," *Inst. Electr. Eng. Proc.—Circuits Devices Syst.*, vol. 139, no. 4, pp. 498–502, 1992.

[77] K. Eshraghian, "Memristormos content addressable memory (MCAM): Hybrid architecture for future high performance search engines," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 8, pp. 1407–1417, Aug. 2011.

[78] Q. Guo, X. Guo, Y. Bai, and E. Ipek, "A resistive TCAM accelerator for data-intensive computing," in *Proc. 44th Annu. IEEE/ACM Int. Symp. Microarchitect.*, 2011, pp. 339–350.

[79] W. Webber, A. Moffat, and J. Zobel, "A similarity measure for indefinite rankings," *ACM Trans. Inf. Syst.*, vol. 28, no. 4, p. 20, 2010.

[80] M. F. Porter, "An algorithm for suffix stripping," *Program, Electron. Library Inf. Syst.*, vol. 14, no. 3, pp. 130–137, 1980.

[81] S. Narasimhan, M. Tabib-Azar, H. J. Chiel, and S. Bhunia, "Neural data compression with wavelet transform: A vocabulary based approach," in *Proc. 3rd Int. IEEE/EMBS Conf. Neural Eng.*, 2007, pp. 666–669.

[82] S. Narasimhan, M. Cullins, H. J. Chiel, and S. Bhunia, "Wavelet-based neural pattern analyzer for behaviorally significant burst pattern recognition," in *Proc. IEEE 30th Annu. Int. Conf. Eng. Med. Biol. Soc.*, 2008, pp. 38–41.

[83] 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.

[84] M. Herlihy and J. E. B. Moss, "Transactional memory: Architectural support for lock-free data structures," *ACM SIGARCH Comput. Architect. News*, vol. 21, no. 2, pp. 289–300, 1993.

[85] Z. Yan, H. Jiang, D. Feng, L. Tian, and Y. Tan, "SUV: A novel single-update version-management scheme for hardware transactional memory systems," in *Proc. IEEE 26th Int. Parallel Distrib. Process. Symp.*, May 2012, pp. 131–143.

[86] L. Dalessandro *et al.*, "Hybrid NOREC: A case study in the effectiveness of best effort hardware transactional memory," *ACM SIGPLAN Notices*, vol. 46, no. 3, pp. 39–52, 2011.

[87] I. Calciu, J. Gottschlich, T. Shpeisman, G. Pokam, and M. Herlihy, "Invyswell: A hybrid transactional memory for Haswell's restricted transactional memory," in *Proc. 23rd Int. Conf. Parallel Architect. Compilat.*, 2014, pp. 187–200.

[88] E. Mirsky and A. DeHon, "Matrix: A reconfigurable computing architecture with configurable instruction distribution and deployable resources," in *Proc. IEEE Symp. FPGAs Custom Comput. Mach.*, 1996, pp. 157–166.

[89] H. Singh *et al.*, "Morphosys: An integrated reconfigurable system for data-parallel and computation-intensive applications," *IEEE Trans. Comput.*, vol. 49, no. 5, pp. 465–481, May 2000.

[90] S. Paul, S. Chatterjee, S. Mukhopadhyay, and S. Bhunia, "Nanoscale reconfigurable computing using non-volatile 2-D STTRAM array," in *Proc. 9th IEEE Conf. Nanotechnol.*, Jul. 2009, pp. 880–883.

[91] S. A. Guccione, D. Levi, and D. Downs, "A reconfigurable content addressable memory," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2000, pp. 882–889.

[92] J. M. Ditmar, "A dynamically reconfigurable FPGA-based content addressable memory for IP characterization," M.S. thesis, KTH/Royal Inst. Technol., Stockholm, Sweden, 2000.

[93] A. R. Trivedi, S. Datta, and S. Mukhopadhyay, "Application of silicon-germanium source tunnel-FET to enable ultralow power cellular neural network-based associative memory," *IEEE Trans. Electron Devices*, vol. 61, no. 11, pp. 3707–3715, Nov. 2014.

[94] H. Hajimiri *et al.*, "Content-aware encoding for improving energy efficiency in resistive random access memory," in *Proc. IEEE/ACM Int. Symp. Nanoscale Architect.*, 2013, pp. 76–81.

[95] S. Narasimhan, "Ultralow-power and robust implantable neural interfaces: An algorithm-architecture-circuit co-design approach," Ph.D. dissertation, Dept. Electr. Eng. Comput. Sci., Case Western Reserve Univ., Cleveland, OH, USA, 2012.

ABOUT THE AUTHORS

Robert Karam (Student Member, IEEE) received the B.S.E. degree in computer engineering from Case Western Reserve University, Cleveland, OH, USA, in 2012, where he is currently working toward the Ph.D. degree in computer engineering.

His research interests include reconfigurable computing architectures, in-memory computing, and digital signal processing (DSP) algorithms.

Ruchir Puri (Fellow, IEEE) is an IBM Fellow at IBM Thomas J Watson Research Center, Yorktown Heights, NY, USA, where he leads high-performance design and methodology solutions for all of IBM's enterprise server and system chip designs. Most recently, he led the design methodology innovations for IBM's latest Power7 and zEnterprise microprocessors and is currently leading design methodology research efforts on future processors. He has been an Adjunct Professor at the Department of Electrical Engineering, Columbia University, New York, NY, USA, and was also honored with John Von-Neumann Chair at Institute of Discrete

Mathematics at Bonn University, Germany. He is an inventor of over 50 U.S. patents (both issued and pending) and has authored over 100 publications on the design and synthesis of low-power and high-performance circuits.

Dr. Puri is a member of the IBM Academy of Technology, IBM Master Inventor, an ACM Distinguished Speaker, and an IEEE Distinguished Lecturer. He has received numerous accolades, including the highest technical position at IBM, the IBM Fellow, which was awarded for his transformational role in microprocessor design methodology. In addition, he has received ''Best of IBM'' awards in both 2011 and 2012 and an IBM Corporate Award from IBM's CEO, and several IBM Outstanding Technical Achievement awards. He is also a recipient of an SRC outstanding mentor award. He also received the 2014 Asian American Engineer of the Year Award. He has delivered numerous keynotes and invited talks at major VLSI design and automation conferences, National Science Foundation, and U.S. Department of Defense Research panels. He has been an editor of the IEEE Transactions on Circuits and Systems.

Swaroop Ghosh (Senior Member, IEEE) received the B.E. degree (honors) from the Indian Institute of Technology Roorkee (IIT Roorkee), Roorkee, India, in 2000, the M.S. degree from the University of Cincinnati, Cincinnati, OH, USA, in 2004, and the Ph.D. degree in computer engineering from Purdue University, West Lafayette, IN, USA, in 2008.

He joined the University of South Florida, Tampa, FL, USA, in Fall 2012. He was a Senior Research and Development Engineer in Advanced Design, Intel Corporation, from 2008 to 2012. His research interests include energy-efficient, secure, and robust circuit/system design and digital testing for nanometer technologies. Dr. Ghosh is an Associate Editor of the IEEE Transactions on Circuits and Systems-Part I: Regular Papers and a Guest Editor of the IEEE Journal on Emerging and Selected Topics in Circuits and Systems. He serves in the technical program committees of ACM/IEEE conferences such as the Design Automation Conference (DAC), the International Conference on Computer-Aided Design (ICCAD), Design, Automation and Test in Europe (DATE), and the International Symposium on Low Power Electronics and Design (ISLPED).

Swarup Bhunia (Senior Member, IEEE) received the B.E. degree (honors) from Jadavpur University, Kolkata, India, the M.Tech. degree from the Indian Institute of Technology (IIT), Kharagpur, India, and the Ph.D. degree in computer engineering from Purdue University, West Lafayette, IN, USA in 2005.

Currently, he is a Professor in the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA. Earlier, he has served as the T. and A. Schroeder Associate Professor of Electrical Engineering and Computer Science at Case Western Reserve University, Cleveland, OH, USA. He has over ten years of research and development experience with over 200 publications in peer-reviewed journals and premier conferences. His research interests include hardware security and trust, adaptive nanocomputing, and novel test methodologies.

Dr. Bhunia received the 2013 IBM Faculty Award, the 2011 National Science Foundation Career Development Award, the 2009 Semiconductor Research Corporation Inventor Recognition Award, the 2005 SRC Technical Excellence Award, and several best paper awards/nominations. He has been serving as an Associate Editor of the IEEE Transactions on Computer-Aided Design, the IEEE Transactions on Multi-Scale Computing Systems, *ACM Journal of Emerging Technologies*, and *Journal of Low Power Electronics*. He served as Guest Editor of *IEEE Design & Test of Computers* (2010, 2013) and IEEE Journal on Emerging and Selected Topics in Circuits and Systems (2014). He has served as co-Program Chair of the IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH) 2013, and the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST) 2015, and in the program committee of many IEEE/ACM conferences.