Resistive Ternary Content Addressable Memory Systems for Data-Intensive Computing

This article examines the integration of gigascale ternary content addressable memory (TCAM) systems based on resistive memories within a general-purpose computing platform. TCAM density is improved by novel resistive memory cells that exploit phase change and spin-torque transfer magnetoresistive RAM technologies. The proposed TCAM system achieves average speedups of 3 to 4.5 times and average energy reductions of 5 to 8 times as compared to a conventional RAM-based system.

Qing Guo
Xiaochen Guo
Yuxin Bai
Ravi Patel
Engin Ipek
Eby G. Friedman
University of Rochester

Multicore processors have been widely hailed as the future of microprocessor performance scaling. The popular conception is appealing: while power is proportional to clock rate, and thermal budgets are limited to less than 200 watts with air cooling, one can continue to ride Moore's law by embracing parallelism and placing twice as many cores on a chip every few years, keeping the clock rate constant.

This conception regrettably presents a false hope. To maintain constant dynamic power at a given clock rate, supply and threshold voltages must scale in proportion to feature size. However, this approach induces an exponential rise in leakage power, which is fast approaching dynamic power in magnitude. As a consequence, the semiconductor industry roadmap projects a supply voltage reduction of only 1.2× by 2022, despite a 16× increase in transistor count. These projections suggest that a 2022 microprocessor will exhaust its power budget with less than 10 percent of the chip in active use.[1] Multicore processors—or any other circuits based on a large number of active transistors—appear to be doomed.

Power dissipation, however, is only part of the problem. Scaling system performance also requires increasing available off-chip memory bandwidth commensurately with growing transistor budgets. Unfortunately, the benefits of Moore's law are unavailable to conventional packaging technologies; consequently, both the speed and the number of signaling pins grow at a much slower rate.

Important data-intensive applications such as information retrieval, data mining, and multimedia processing demand significant computational power and generate substantial memory traffic, which places a heavy strain on both overall system power and off-chip

Published by the IEEE Computer Society

bandwidth. Device, circuit, and architecture innovations are needed to surmount this problem.

Ternary content addressable memories

A promising approach to addressing power and bandwidth limitations on many data-intensive workloads is to process the data directly within a memory chip. This requires augmenting a memory chip with useful but simple logic circuits such that data manipulation can be accomplished directly on the memory chip, while the hardware cost is kept acceptable. A content addressable memory (CAM) is one type of memory with processing capability that can benefit many different data-intensive applications. A CAM compares a given search key to a set of stored keys in parallel. If a match is found, the index (address) of the matching entry is reported. A ternary CAM (TCAM) is a special type of CAM that allows for both storing and searching with a wildcard in addition to a logic 0 or 1. A wildcard, when part of either the search key or the stored data word, matches against both binary states (as well as another wildcard). A surprisingly large number of applications can exploit this flexibility.

Figure 1 shows two example applications of TCAM. In IP lookup (Figure 1a), an incoming network packet's IP address is compared against a list of registered IP domains, and the domain that the IP address belongs to is selected. In word search (Figure 1b), a stored word that includes the given search key as a substring is found. In both cases, wildcards are used to mask off the irrelevant bit positions in either the stored or the search keys.

Overcoming the power and bandwidth challenges presented by emerging data-intensive applications requires architecting TCAM systems with high energy efficiency and capacity. Unfortunately, CMOS-based state-of-the-art TCAM devices suffer from 8× higher cost per bit and 10× higher power dissipation than SRAM.[2] These drawbacks restrict the capacity of existing TCAM devices to a few megabytes and confine the commercial use of TCAMs to niche networking applications.

Resistive memory technologies

Resistive memories, which include phase-change memory (PCM) and spin-torque transfer magnetoresistive RAM (STT-MRAM), store information by modulating the resistance of nanoscale storage elements, and are expected to scale to much smaller geometries than charge-based memories. Resistive memories are nonvolatile, free of leakage power, and immune to radiation-induced transient faults.

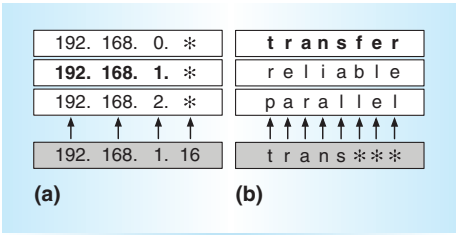


Figure 1. Example applications of ternary content addressable memories (TCAM). (a) IP lookup. (b) Word search. Search keys are in gray, and matching entries are in bold. An asterisk denotes a wildcard.

PCM

Among all resistive memory technologies, PCM is arguably the most mature: 128-Mbyte parts are in production, and gigabit-array prototypes have been demonstrated.[3] PCM exhibits a programmable resistance that varies from less than 10 KΩ in the crystalline state to greater than 1 MΩ in the amorphous state.[1] On a write, a high-amplitude current pulse is applied to the cell to induce Joule heating. Slowly reducing the write current causes the device to revert to a crystalline state, whereas an abrupt reduction in the current causes the device to retain its amorphous state. On a read, a small current passes through the cell, and the cell resistance is sensed by measuring the current.

STT-MRAM

Data is stored by modulating the magnetoresistance of a magnetic tunnel junction (MTJ) in STT-MRAM. An MTJ contains a pinned layer, which has a fixed magnetic spin, and a free layer in which the magnetic spin can be altered by applying a high-amplitude current pulse through the MTJ. Depending on the direction of the current, the free layer's magnetic polarity can be made parallel or antiparallel to that of the pinned layer, which causes the MTJ to exhibit a low (5 KΩ) or a high (12.5 KΩ) resistance,

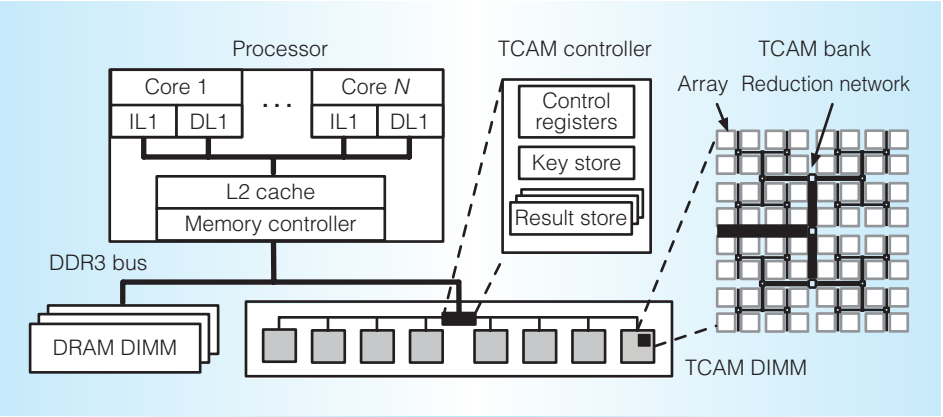


Figure 2. TCAM system architecture. The TCAM DIMM is a DDR3-compatible module with an on-DIMM controller that interfaces to the memory controller, which translates standard DDR3 commands into TCAM DIMM operations.

respectively.[1] Recently, a 64-Mbit DDR3-compatible STT-MRAM part has been made commercially available.[4]

Overview of the proposed TCAM system

As Figure 2 shows, the proposed TCAM system is built on a DDR3-compatible DIMM. The system supports one or more TCAM DIMMs on the memory bus, each comprising an on-DIMM controller that serves as the interface to the DDR3 bus, translating conventional DRAM commands into TCAM operations, and ensuring that DDR3 timing constraints are not violated. The processor communicates with the controller through a set of memory-mapped control registers (for configuring functionality), a key store (for buffering the search key), and a result store (for buffering the results). All of the chips share the on-DIMM command, address, and data buses; however, a search operation is restricted to operating on a single chip because of constraints on the maximum power dissipation. Each chip has eight banks; a bank comprises a set of arrays that are searched against the search key, as well as a hierarchical reduction network for aggregating the results. The resulting memory system's modularity lets the TCAM be selectively included in systems running workloads that are amenable to TCAM-based acceleration; moreover, when executing an application or a program phase that does not benefit from associative search capability, TCAM DIMM can be configured to provide ordinary RAM functionality.

Mapping an application to the proposed TCAM system is a three-step process:

- partition the workload between the processor and the TCAM system,
- define the relevant data structures for storing the keys, and
- specify the TCAM operations to be performed on the keys.

Figure 3 demonstrates this mapping process via an example image histogram application,[5] which creates a histogram of the pixel values in an RGB image. In a RAM-based system, the image is scanned pixel by pixel, and the appearance of each pixel value is recorded in an integer array (Figure 3a). Notably, the two operators in the RAM-based implementation, *scan* (of the image) and *count* (of the pixel values), can be replaced by searching the pixels and counting the number of matches using peripheral circuits in a TCAM-based system.

To facilitate searching the pixel values in the TCAM-based implementation (see Figure 3b), the raw image data and the search key are organized in a 4-tuple, in which an image ID accompanies each pixel (Figure 4); this ensures that the matching entries belong to the target image. In addition, a *mask* is used in conjunction with the search *key* (wherein a "1" denotes a valid bit in the search key and a "0" indicates a wildcard), such that only a single color is checked in

each search. Once the mask is stored in a memory-mapped register on the DIMM controller (lines 6, 14, and 22), the processor iteratively sends the search keys with different values to be compared against the stored keys, and reads the number (count) of matches from the result store (lines 10, 18, and 26). *Store* and *search* are API functions provided by a user-level library to facilitate programming the TCAM system.

Compared to the RAM-based solution, which requires streaming the image through the processor pipeline, the TCAM-based system can reduce power dissipation by eliminating data movement and instruction processing overheads. Bandwidth demand can also be lowered by processing data directly on the TCAM chip, thereby reducing off-chip traffic. In this article, we present two instantiations of the TCAM system—TCAM-DIMM[6] and AC-DIMM[7]—which provide different speed, capacity, and programmability tradeoffs.

TCAM-DIMM

TCAM-DIMM implements a novel TCAM cell and array architecture that can scale TCAM capacity to gigabytes.

TCAM cell structure

Figure 5 illustrates the proposed TCAM cell, which comprises three pairs of access transistors and resistive storage elements (3T-3R). The first two resistors store the data bit (D) and its complement ($\overline{D}$); the third resistor is permanently programmed to the high-resistance state.

Writing. To store a logic 1, D is programmed to the high-resistance state (R_{HI}), and $\overline{D}$ is programmed to the low-resistance state (R_{LO}). The states of D and $\overline{D}$ are flipped when storing a logic 0. To store a wildcard, both D and $\overline{D}$ are programmed to R_{HI}.

Searching. To search for a logic 1 or 0, SL and ($\overline{SL}$), respectively, are driven with the search bit and its complement, turning one of the access transistors on and the other off. A match decision is made based on the effective resistance between the matchline and ground. If a resistor with R_{HI} is in series with

```
for (i = 0; i < IMAGE_SIZE; i += 3) {
        /* Raw image data in the blue-green-red order */
        ++hist_blue[image[i]];
        ++hist_green[image[i + 1]];
        ++hist_red[image[i + 2]];
}
```

(a)

```
/* Data structure and layout */
typedef struct {
        uint8_t imgID;
        uint8_t blue;
        uint8_t green;
        uint8_t red;
} Bits32;

/* TCAM-based code */
 1  /* the search key in 32 bits */
 2  Bits32 key = {imgID, 0, 0 ,0};
 3  /* the mask in 32 bits */
 4  uint32_t mask = (255 << 24) | (255 << 16);
 5  /* send the mask for Blue to the TCAM controller */
 6  store(MASK, sizeof(uint32_t), mask);
 7  for(i = 0; i < 256; i++) {
 8      key.blue = i;
 9      /* send the search key for Blue and read the count */
10      hist_blue[i] = search(COUNT, sizeof(Bits32), key);
11  }

12  mask = (255 << 24) | (255 << 8);
13  /* send the mask for Green to the TCAM controller */
14  store(MASK, sizeof(uint32_t), mask);
15  for(i = 0; i < 256; i++) {
16      key.green = i;
17      /* send the search key for Green and read the count */
18      hist_green[i] = search(COUNT, sizeof(Bits32), key);
19  }

20  mask = (255 << 24) | 255;
21  /* send the mask for Red to the TCAM controller */
22  store(MASK, sizeof(uint32_t), mask);
23  for(i = 0; i < 256; i++) {
24      key.red = i;
25      /* send the search key for Red and read the count */
26      hist_red[i] = search(COUNT, sizeof(Bits32), key);
27  }
```

(b)

Figure 3. Image histogram with two implementations. (a) RAM-based solution. (b) TCAM-based solution. The RAM-based solution necessitates a scan of the entire image, whereas the TCAM-based solution iteratively searches with each of the pixel values. A mask is used to indicate the valid bits within the search key.

Pixel 0	imgID	blue	green	red
Pixel 1	imgID	blue	green	red
Pixel 2	imgID	blue	green	red

Figure 4. Data layout in the TCAM. Each pixel of the image is stored in a single row of TCAM. To confine the matches to the pixels within the target image, an image ID (`imgID`) is attached to each pixel of the image and is searched in conjunction with the pixel values.

the on transistor, the search results in a match; conversely, a resistance of R_{LO} connected to the matchline indicates a mismatch. To search for a wildcard, SL and $\overline{SL}$ are disabled and SX is driven high; hence, a resistor in the R_{HI} state is connected to the matchline regardless of the value stored in the cell.

The TCAM cell consumes an area of 27 F^2, where F is the feature size (3× the area of the one-transistor, one-resistor PCM cell projected by *ITRS*[1]), which is 20× smaller than an SRAM-based TCAM cell.

Row organization

Searching a TCAM row involves distinguishing the effective matchline-to-ground resistance on a word match—wherein each cell connects the matchline to ground through an R_{HI} path—from the effective resistance on a word mismatch—wherein at least one cell provides an R_{LO} path to ground. Hence, the effective parallel resistance to ground is always higher in the case of a match. A resistive memory technology with a greater difference between its high and low resistances (such as PCM) allows a larger number of bits to be sensed in parallel, whereas a technology with a lower R_{HI}/R_{LO} ratio (such as STT-MRAM) has a tighter constraint on the number of bits that can be searched at the same time.

Array architecture

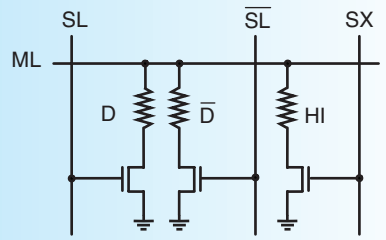
The size of an array is an important design decision that affects the area efficiency, power, speed, and reliability. Larger arrays improve area efficiency by amortizing the area overhead of the peripheral circuitry over more cells; however, the reliability of sensing deteriorates with the number of bits that are searched in parallel. To address this challenge, we leverage matchline segmentation; each row contains 1,024 cells, only a segment of which (that is, 128 columns) can be searched simultaneously. The peripheral circuitry supports iterative search for keys that are larger than a segment.

The system can provide two different results on a search operation: a *population count* indicating the total number of matching rows, and a *priority index* indicating the address of the matching row with the lowest index. Once a search operation is complete, the peripheral logic attached to each array computes the highest-priority matching row and the number of matches within the array, which are forwarded to a reduction network. The reduction network is a quad tree whose leaves accept the local count and indexing results from different arrays. Each internal node of the quad tree takes the outputs of its four children, processes them by counting or priority encoding, and then forwards the result to its parent. Once a search's results propagate to the root of the quad tree, the final results are obtained and placed in an SRAM-based result store that resides with the TCAM controller.

System interface

TCAM-DIMM's memory bus interface is fully compatible with the DDRx standard and its timing constraints. The minimum number of cycles between a write and a consecutive read in DDR3 is less than the time needed to perform a search; hence, without additional support, the memory controller can issue a read from the result store before the corresponding search operation is complete. To avert this problem, the processor inserts an appropriate number of dummy writes between the search and the subsequent read from the result store, generating the required number of idle cycles on the DDR3 bus for the search operation to complete. The TCAM controller detects the dummy write and silently drops it upon receipt.

Software is given access to the TCAM system by mapping the TCAM address range to the system's physical address space. A user-



D	$\overline{D}$	SL	$\overline{SL}$	SX	Match
HI	LO	**1**	**0**	**0**	yes
HI	LO	**0**	**1**	**0**	no
LO	HI	**1**	**0**	**0**	no
LO	HI	**0**	**1**	**0**	yes
HI	HI	—	—	—	yes
—	—	**0**	**0**	**1**	yes

Figure 5. The proposed 3T-3R TCAM cell and its truth table. (3T-3R: three pairs of access transistors and resistive storage elements.) The leftmost and middle resistors store the data bit and its complement, respectively, whereas the rightmost resistor is permanently programmed to R_{HI} to support searching with a wildcard.

level software library serves as the API to the programmer and implements four functions.

Create. This function is called at the beginning of an application to map a virtual address range to a portion of the TCAM physical address space. Allocated physical pages are marked as uncacheable, which prevents the memory requests to the TCAM address range from being reordered.

Destroy. This function releases the allocated TCAM space.

Store. This function writes into the memory-mapped control registers on TCAM-DIMM, including configuration registers and the search key buffer. To update the content in a TCAM array, the function sends the address and the new content to the on-DIMM controller, which then writes the data into the designated location.

Search. This function performs a search operation in three steps: it stores the search key into the memory-mapped key buffer; issues a sufficient number of dummy writes to pad the waiting period between the search operation and the subsequent read from the result store; and then reads the results from the memory-mapped result store.

AC-DIMM

Despite the performance and energy-efficiency improvements achieved by searching through a large volume of data in situ, TCAM-DIMM suffers from two shortcomings. First, the limited functionality of the peripheral circuits (that is, counting and indexing) diminishes the benefits of using TCAM-DIMM in data manipulation tasks that require going beyond searching. To calculate the sum of the matching entries, for instance, the processor must fetch the matching rows and conduct the additions in its ALUs, resulting in data movement overheads. Second, the 3T-3R cell structure requires a resistive memory technology with a high dynamic resistance range (R_{HI}/R_{LO}), such as PCM.

AC-DIMM addresses both of these limitations via a novel cell design and the integration of simple programmable microcontrollers to execute user-defined kernels on the search results. This flexibility and rich functionality let AC-DIMM cater to a much more diverse set of applications than the TCAM-DIMM system.

Cell topology

AC-DIMM implements a two-transistor, one-resistor (2T-1R) cell using STT-MRAM (see Figure 6a). Compared to PCM, STT-MRAM has shorter access latency, lower write energy, and practically unlimited write endurance. In each cell, a single MTJ behaves as a storage element; one transistor (M_{BL}), controlled by the wordline (WL), serves as a gating element between two vertical bitlines (BL and $\overline{BL}$), and a second transistor (M_{ML})

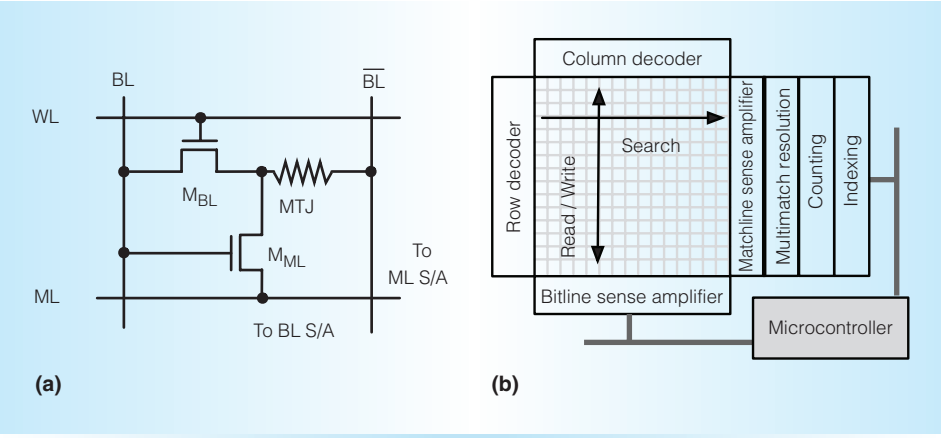


Figure 6. AC-DIMM cell and array architectures. (a) Cell circuit. (b) Array organization. Transistor M_{BL} and the bitline sense amplifier (BL S/A) act as the read port of the cell, and transistor M_{ML} and the matchline sense amplifier (ML S/A) act as the search port of the cell. Data in the array is addressed by the row and the column decoders.

connects the MTJ to the horizontal matchline (*ML*). $\overline{BL}$ and *ML* are terminated by sense amplifier (*SA*) circuits.

Writing. Programming a cell requires supplying a write current through the MTJ; the current's direction determines the written value. To store a logic 0, *BL* is connected to ground and $\overline{BL}$ is driven to V_{DD}; a write current flows from $\overline{BL}$ to *BL* and programs the MTJ to the high-resistance state (R_{HI}). In the reverse direction, both *ML* and *BL* are driven to V_{DD} and $\overline{BL}$ is connected to ground, which programs the MTJ to the low-resistance state (R_{LO}).

Reading. A read operation is initiated by setting *BL* to ground and *WL* to V_{DD}, connecting the MTJ to both bitlines and isolating it from the matchline. A bias circuit then drives $\overline{BL}$. The voltage that appears on $\overline{BL}$ depends on the resistive state of the MTJ (R_{HI} or R_{LO}) and is sensed by the sense amplifier (*SA*) at the end of $\overline{BL}$.

Searching. Searching a cell is equivalent to reading the stored bit and comparing it to the search bit. Specifically, both *WL* and $\overline{BL}$ are set to ground, *BL* is connected to V_{DD}, and *ML* is biased for sensing. The stored data is compared against the searched bit by an XNOR gate to produce the final search result.

Wildcards. AC-DIMM can achieve wildcard functionality by storing a bit (D) and its complement ($\bar{D}$) in two adjacent columns of a row. Searching for a 1 or a 0 requires biasing D or $\bar{D}$, respectively. AC-DIMM stores a wildcard by setting both D and $\bar{D}$ to R_{HI}. It searches with a wildcard by simply skipping the relevant column. This flexibility lets AC-DIMM bear the area overhead of a full TCAM only when wildcard storage is needed.

Array organization

AC-DIMM lets each memory row be read and written through vertical bitlines and searched through horizontal matchlines (Figure 6b). Because of the low R_{HI}/R_{LO} ratio of STT-MRAM, searching multiple bits simultaneously in a row is difficult and unreliable. Instead, AC-DIMM implements a bit-serial search scheme, wherein a search operation progresses iteratively, first sensing the cell on the initial column, and progressively searching column by column across the data array. As each stored bit is iteratively compared against the corresponding bit of the search word, the partial search result is stored in a peripheral helper flip-flop.

The match results are processed by a counting logic that computes the number of matches, a priority indexing logic that generates the address of the selected matching row,

and a multimatch resolution circuit that selects the first match among all of the matching rows and drives the wordline so that the selected row can be read or written in the following cycle, thereby eliminating the need for address calculation and decoding.

AC-DIMM provides user-programmable control over the reduction operations to be performed on the search results by placing a specialized microcontroller at the center of the four arrays (Figure 6b). Once search results are produced, the multimatch resolution circuit progressively selects the next matching row, which lets the microcontroller iteratively apply user-defined operators on the set of matching rows.

Evaluation

We performed architectural- and circuit-level evaluations of TCAM-DIMM and AC-DIMM. We augmented the SuperESCalar (SESC) simulator (http://sesc.sourceforge.net) to model an eight-core superscalar system. A 1-Gbyte TCAM-DIMM or AC-DIMM connects to the memory controller through a DDR3-1067 bus. We conducted Cadence Spectre circuit simulations for all of the constituent parts of the data array. The CMOS transistor model is based on the predictive technology model for the 22-nm node.[8]

We evaluated a set of benchmarks that represent a wide range of data-intensive applications from NU-MineBench,[9] Phoenix,[5] MiBench,[10] and SPEC CINT2000[11] suites. We evaluated TCAM-DIMM and AC-DIMM systems with the single-threaded version of each benchmark, whereas the baseline DRAM-based system ran the parallel applications (Phoenix benchmarks and NU-MineBench) with eight threads.

Performance

Over a set of eight benchmarks, AC-DIMM and TCAM-DIMM outperformed the DRAM-based system by 3.8 and 3.2×, respectively (Figure 7). (The remaining five workloads are not portable to TCAM-DIMM due to the limited flexibility of TCAM-DIMM; these workloads are shown on the right side of Figure 7.) The performance improvement comes from using a CAM (in this case, AC-DIMM and TCAM-DIMM), which permits simultaneous comparison of a search key against all the stored keys. This approach surpasses the software algorithms because retrieving the data out of the memory is not required, which not only improves performance but also eliminates unnecessary power consumption and bus traffic.

AC-DIMM achieved an additional 19 percent performance improvement over TCAM-DIMM. Note that the STT-MRAM-based AC-DIMM searches in a bit-serial fashion, which completes more search operations under a fixed power budget than TCAM-DIMM (which searches 128 bits in parallel).

Beyond the associative search capability, AC-DIMM can apply a rich set of user-defined operations to the search results and can process the search results in situ. Over a set of five applications exclusive to AC-DIMM acceleration, AC-DIMM outperformed the eight-threaded RAM-based multicore system by an average of 4.5×. Because data-intensive applications often suffer from insufficient off-chip memory bandwidth, these applications naturally benefit from AC-DIMM's processing-in-memory capability.

Energy

Figure 7 evaluates the system energy, including the eight applications that are portable to both AC-DIMM and TCAM-DIMM and the five applications that are portable only to AC-DIMM. AC-DIMM saves more energy than TCAM-DIMM on Histogram, Reverse Index, Bitcount, and Dijkstra. TCAM-DIMM achieves higher energy efficiency when the search key is long (as in string match, for example). Beyond a certain search key length (that is, 32 bits), the bit-serial search scheme used by AC-DIMM consumes more energy, because it must search each bit position progressively.

With the exception of Matrix Multiply, the five applications that are exclusively portable to AC-DIMM exhibit energy savings over the baseline. Matrix multiplication is highly computing intensive; consequently, running Matrix Multiply on AC-DIMM consumes the same amount of energy as it does on a multicore processor.

Both TCAM-DIMM and AC-DIMM achieve significant gains in execution

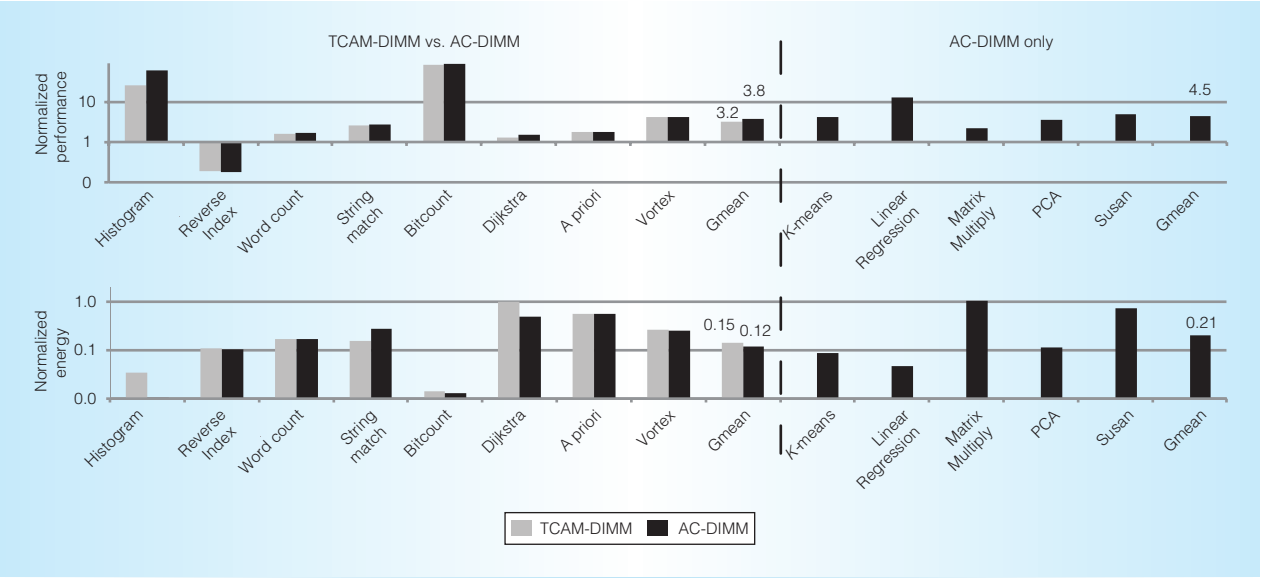


Figure 7. Performance and energy comparisons between TCAM-DIMM and AC-DIMM (normalized to the DRAM-based system). Among the 13 evaluated benchmarks, eight are mapped to both the TCAM-DIMM and the AC-DIMM implementations. The other five benchmarks require postprocessing of the search results, and hence benefit from only the AC-DIMM acceleration.

time and energy efficiency over a conventional RAM-based system. Resistive TCAM accelerators have significant potential to improve the performance and energy efficiency of future computer systems.

Datacenters today face increasingly demanding data-intensive processing workloads. It is therefore promising to deploy the TCAM-DIMM and the AC-DIMM architectures at scale within the context of a datacenter to help increase the data processing throughput and energy efficiency. Integrating the TCAM- DIMM and the AC-DIMM architectures into the MapReduce computing paradigm is another promising direction for future research. MICRO

References

1. *International Technology Roadmap for Semiconductors*, 2013; www.itrs.net/ITRS%20 1999-2014%20Mtgs,%20Presentations%20 &%20Links/2013ITRS/Summary2013.htm.
2. A. Goel and P. Gupta, "Small Subset Queries and Bloom Filters Using Ternary Associative Memories, with Applications," *Proc. ACM Sigmetrics Int'l Conf. Measurement and Modeling of Computer Systems*, 2010, pp. 143–154.
3. H. Chung et al., "A 58nm 1.8V 1Gb PRAM with 6.4MB/s program BW," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2011, pp. 500–502.
4. Everspin Technologies, *Spin-Torque MRAM Technical Brief*, 2013.
5. C. Ranger et al., "Evaluating MapReduce for Multi-core and Multiprocessor Systems," *Proc. 13th Int'l Symp. High-Performance Computer Architecture*, 2007, pp. 13–24.
6. Q. Guo et al., "A Resistive TCAM Accelerator for Data-Intensive Computing," *Proc. 44th Ann. IEEE/ACM Int'l Symp. Microarchitecture*, 2011, pp. 339–350.
7. Q. Guo et al., "AC-DIMM: Associative Computing with STT-MRAM," *Proc. 40th Ann. Int'l Symp. Computer Architecture*, 2013, pp. 189–200.
8. W. Zhao and Y. Cao, "New Generation of Predictive Technology Model for Sub-45nm Design Exploration," *Proc. 7th Int'l Symp. Quality Electronic Design*, 2006, pp. 585–590.
9. J. Pisharath et al., *NU-MineBench 2.0*, tech. report CUCIS-2005-08-01, Center for Ultra-Scale Computing and Information Security, Northwestern Univ., 2005; http://cucis.ece.northwestern.edu/techreports/pdf/CUCIS-2004-08-001.pdf.

10. M.R. Guthaus et al., "MiBench: A Free, Commercially Representative Embedded Benchmark Suite," *Proc. IEEE Int'l Workshop Workload Characterization*, 2001, pp. 3–14.
11. J.L. Henning, "SPEC CPU2000: Measuring CPU Performance in the New Millennium," *Computer*, vol. 33, no. 7, 2000, pp. 28–35.

Qing Guo is a PhD student in the Department of Computer Science at the University of Rochester. His research interests are in energy-efficient computing and architectures exploiting resistive memory technologies. Guo has an MS in computer science from the University of Rochester. Contact him at qguo@cs.rochester.edu.

Xiaochen Guo is an assistant professor in the Department of Electrical and Computer Engineering at Lehigh University. Her research focuses on leveraging resistive memories to build energy-efficient processors, memory systems, and accelerators. Guo has a PhD in electrical and computer engineering from the University of Rochester, where she completed the work for this article. Contact her at xig515@lehigh.edu.

Yuxin Bai is a PhD student in the Department of Electrical and Computer Engineering at the University of Rochester. Her research focuses on energy-efficient microarchitectures, including MOS current-mode computing and machine-learning-based power management techniques. Bai has an MS in electrical and computer engineering from the University of Rochester. Contact her at yuxin.bai@rochester.edu.

Ravi Patel is a PhD student in the Department of Electrical and Computer Engineering at the University of Rochester. His research interests include memristor, STT-MRAM, and high-performance memories. Patel has an MS in electrical and computer engineering from the University of Rochester. Contact him at rapatel@ece.rochester.edu.

Engin Ipek is an associate professor in the Department of Electrical and Computer Engineering and the Department of Computer Science at the University of Rochester. His research interests include energy-efficient architectures, high-performance memory systems, and the application of emerging memory technologies to computer systems. Ipek has a PhD in electrical engineering from Cornell University. Contact him at ipek@cs.rochester.edu.

Eby G. Friedman is a distinguished professor in the Department of Electrical and Computer Engineering, and the director of the High Performance VLSI/IC Design and Analysis Laboratory, at the University of Rochester. He is also a visiting professor at the Technion–Israel Institute of Technology. His research interests include high-performance synchronous digital and mixed-signal microelectronic design and analysis with application to high-speed portable processors and low-power wireless communications. Friedman has a PhD in electrical engineering from the University of California, Irvine. He is a Senior Fulbright Fellow and an IEEE Fellow. Contact him at friedman@ece.rochester.edu.