Associative Memristive Memory for Approximate Computing in GPUs

Amirali Ghofrani, *Student Member, IEEE,* Abbas Rahimi, *Student Member, IEEE,* Miguel A. Lastras-Montaño, Luca Benini, *Fellow, IEEE* Rajesh K. Gupta, *Fellow, IEEE* Kwang-Ting Cheng *Fellow, IEEE*

***Abstract*—Associative memory, in the form of lookup tables, is a promising approach to improving energy efficiency by enabling *computing-with-memory*. A processing element can be tightly coupled with an associative memory where function responses are pre-stored. Associative memories can recall function responses for a subset of input values therefore avoiding the actual function execution on the processing element that leads to energy saving. One challenge however is to reduce the energy consumption of associative memory modules themselves.**

In this paper, we address the challenge of designing ultra-low-power associative memories. We first use memristive parts for memory implementation and demonstrate the energy saving potential of integrating associative memristive memory (AMM) into graphics processing units (GPUs). Next, we leverage *approximate computing* which takes advantage of application-level tolerance to errors, to enable voltage overscaling to further reduce energy consumption of an AMM module. Voltage overscaling deliberately relaxes the searching criteria of an AMM: The AMM module finds stored patterns matching an input search pattern with a Hamming distance of 0, 1, or 2. This controllable inexact matching introduces some errors to the computation, that are tolerable for the target application. The energy consumption is further reduced by employing a purely resistive crossbar architecture for the AMM module. To evaluate our solution, we tightly integrate AMM modules with floating point units (FPUs) in an AMD Southern Islands GPU. Then we run four image processing kernels on an AMM-integrated GPU to evaluate the proposed architecture. Our experimental results show that the use of the AMM modules reduces energy consumption of running these kernels on GPU by, on average, 23%–45%, compared to the baseline GPU without AMM modules. We also show that these image processing kernels can tolerate errors resulting from approximate search operations with an acceptable degradation of image quality, i.e., a PSNR greater than 30dB.

***Index Terms*—Associative memory, TCAM, memristor, approximate computing, GPUs, voltage overscaling, FPUs.**

I. Introduction

The scaling of physical dimensions in semiconductor circuits has reached an astonishing level of integration of over eight billion transistors in a single chip based on a 28-nm process. This gives a grand total of 3,072 processing cores in recent GPU chips [1] enforcing energy efficiency as a primary concern. Earlier work has suggested supply voltage overscaling (VOS) [2] to reduce energy consumption. However, reducing the operating voltage of a core beyond a *critical point* leads to the so-called "path walls" [3], [4]. Hitting the path walls results in a core failure or massive number of errors, which is not acceptable [5], [6].

A. Ghofrani is with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106, USA (e-mail: ghofrani@ece.ucsb.edu).

Approximate computing is another approach to increase energy efficiency that leverages application-level tolerance to few errors in many multimedia applications. In most multimedia applications, the final output is interpreted by human and thus does not have to be perfect [7]. Hence, such applications can generally tolerate a limited number of errors as long as the degraded output maintains an expected quality. However, approximate computing cannot support aggressive VOS beyond the critical point, due to the sudden increase in the number of errors [8], [9].

Associative computing using ternary content-addressable memories (TCAMs) has also emerged as a promising solution to improve energy efficiency [10], [11]. Associative memories can pre-store highly frequent computations and minimize their re-execution to save energy. Conventional TCAM designs based on CMOS suffer from low density and high power consumption. To address this, recent works propose TCAMs using memristive memories [12], [13]. Li *et al.* demonstrate a 1-Mb TCAM chip using a 2-transistor/2-resistive-memristor (2T-2R) cell that achieves a 10× smaller cell size than a CMOS-based TCAM [12]. This TCAM also enables low voltage search operation. A 1-Mb memristive memory (aka ReRAM) operating at a voltage as low as 270mV for ultra-low energy consumption has also been demonstrated [14].

The low voltage operation of memristive devices provides an opportunity to further reduce energy consumption in associative memories: aggressive VOS techniques can be applied to associative memristive memories, while managing errors to an acceptable level suitable for approximate computing in GPUs. This paper explores this opportunity and makes the following contributions:

(1) We propose an associative memristive memory (AMM) module to enable low-power *computing-with-memory*. An AMM, tightly integrated into a floating point unit (FPU), is a programmable module accessible by software to store computations that appear frequently. An AMM is composed of a memristive TCAM and a memristive memory block that together represent the pre-stored computations as partial functionality of the associated FPU. We implement the TCAM block in two flavors, 2T-2R and 0T-2R. The 0T-2R implementation leverages memristive devices with diode-like rectifying behavior to eliminate the access-transistors, needed in the 2T-2R TCAM. We show that the 0T-2R TCAM generally offers better energy saving due to the elimination of the static power consumption of the access-transistors.

(2) We explore the potential of the AMM module to enable "approximate computing-with-memory" under VOS: we carefully employ VOS in an AMM module which leads to approximate search operation in TCAMs (i.e., inexact

matching). Varying the degree of VOS, an AMM module exhibits a "controllable" inexact matching behavior: when we reduce the voltage from 1.0V down to 725mV (or 800mV), the 2T-2R (or 0T-2R) TCAM inside an AMM module is able to find stored patterns matching an input query pattern within a Hamming distance of 0, 1, or 2. Employing this inexact matching under VOS further lowers the power consumption at the cost of few errors in the output result. This level of approximate computing is often acceptable for most of image processing applications.

(3) We present an OpenCL execution flow, to profile GPU kernels, identify frequently repeated computations, and program the AMM module. The profiler extracts a set of values that occur frequently through searching the space of possible inputs provided by training samples. These key-computations are then pre-stored in the AMM module, which can be searched to avoid repeated executions in the FPU. We demonstrate the effectiveness of our AMM modules on the Southern Islands GPUs with four image processing kernels adopted from the AMD APP SDK v2.5 [15]. We use 10% of Caltech 101 computer vision dataset [16] for the training, and the full dataset for the testing. Our experimental results show that by integrating small AMM modules into FPUs, we can reduce the energy consumption of running these image processing kernel by, on average, 23%–45%. We also show that the image processing kernels can tolerate the error due to the inexact pattern matching for all test images with a peak signal-to-noise ratio (PSNR) no lower than 30dB.

The rest of the paper is organized as follows. Section II discusses the relevant research on computing-with-memory and approximate computing. Section III covers the necessary background on memristors. In Section IV, we discuss the AMM module in detail and describe how it enables approximate computing-with-memory in GPUs. Section V elaborates on the 0T-2R TCAM architecture and its operation. Section VI discusses execution flow and programming for AMM modules, and provides experimental results to validate the energy saving merits of the method. Section VII concludes the paper.

II. Related Work

Computing-with-memory has shown significant energy efficiency using emerging non-volatile memories [12], [17], [18], [19], [20]. For instance, spin-torque transfer RAM (STTRAM) has been used for reconfigurable frameworks which partition the entire input application into smaller representable lookup tables [17] or use a co-design approach for a better application mapping [18]. Compilers can further optimize the lookup table resource allocation among various functions used in a program [19]. However, these frameworks map the entire application [17], [18] or hot functions [19] to the non-volatile memory, hence limit their applicability to a subset of applications amenable to full memory-based computing. Others have proposed associative memories that use TCAMs with memristive elements to realize a low-power search operation [12], [13], [21], [22]. However, their exact search operation does not exploit the application-level tolerance to errors, leaving an untapped energy saving potential.

Approximate Computing has been also explored to reduce the energy consumption in the processing cores [23], [24], [25], [9]. In [23], approximate computing is applied to the floating point operations in GPUs by designing imprecise hardware blocks. An associative memory with approximate matching reduces energy of the FPUs by exploiting value similarity and locality inside GPU applications using tiny lookup tables [25]. However, it can afford to maintain very few contexts due to the limitations of conventional CMOS-based lookup tables. Moreover, these CMOS-based lookup tables and imprecise hardware blocks suffer from the common drawback of having a critical operating voltage during VOS [3], [4], [6].

We have earlier designed an AMM module that enables approximate computing-with-memory under aggressive VOS [9]. We improve its TCAM architecture by employing an access-transistor-free implementation that further reduces energy consumption and area. Our proposed programming flow also limits the write stress to the AMM modules, thus extending their lifetime, by allowing only a limited number of write operations at the beginning of the kernel.

III. Background on Memristors

Recent studies on metal oxide valence change ReRAMs [26], generally referred to as memristors [27], demonstrate their potential to offer ultra-small and low-power non-volatile memory elements [28]. A memristor is a two-terminal passive programmable resistor, the resistance of which is maintained in the absence of an electric field. This characteristic makes them an ideal candidate as a non-volatile memory. Memristive devices with high (or low) resistances, R_{OFF} (or R_{ON}), can be used to store logic value 0 (or 1). The stored logic can be read by applying a low voltage pulse to the device and sensing the resulting current: a high (or low) current indicates a low (or high) resistance state stored in the device, and thus a logic 1 (or 0).

Memristors typically have a metal/insulator/metal structure. The change in the resistance is due to the non-volatile formation of a conductive filament inside the insulating oxide layer. Such filament is formed by applying a voltage (or current) pulse across the device. The applied electric field forms a filament by mobilizing the conductive particles (e.g., metallic ions, oxygen vacancies, etc.) and making them drift inside the insulating oxide layer [29]. With the formation of such a highly conductive filament, the device is set to a low resistance state (ON state). To reset the device back to a high resistance state (OFF state), a pulse with an opposite voltage polarity is applied. Such pulse ruptures the filament by dispersing the conductive particles. Fig. 1(a) and Fig. 1(b) show the filament formation and rupture processes in a Pt/TaO_2/Pt memristor.

The change to the resistance has a strong non-linear dependency on the amplitude and the duration of the applied pulse; applying voltages above a write threshold, V_{thw}, effectively changes the resistance of the device, while applying voltages below V_{thw} has negligible impact on device's resistance, as is shown Fig. 2(a). This behavior is generally attributed to the super-exponential dependence of the ionic mobility on the applied voltage [30]. A second type of non-linearity is further

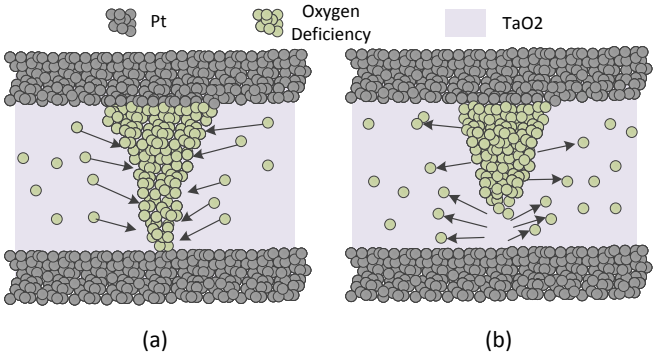


Fig. 1. A memristor's exemplar realization: (a)/(b) applying a positive/negative write voltage to the device can form/destroy a low-resistance filament, by attracting/dispersing conductive particles, thus writing logic 1/0 to the cell.

engineered in the "I-V characteristics" of several memristive devices by introducing extra layers in the device stack of memristors [31], [32], [33]. Researchers have achieved a variety of I-V behaviors; from rectifying devices that behave as a diode in the presence of negative biases, as shown in Fig. 2(b), to devices that exhibit a very high resistance $> 100\text{G}\Omega$ below a read threshold, for both ON and OFF states, as shown in Fig. 2(d).

The existence of such non-linearities in the dR/dt-V and I-V characteristics of memristive devices, enables the realization of crossbar-based memristive memory modules with no access-transistor per memory element [34]. Ultra-high density memory arrays can be realized with memristive devices due to the possible elimination of the access-transistors per memory cell [33], [35], [34], as well as the memristor's simple structure which enables feature sizes that can be shrunk to a sub-10nm scale [36], [37]. Both analysis and preliminary experimental measurements have also demonstrated the potential of memristive memory modules for lower power consumption than existing technologies [38], [39]. The main contributors to its power efficiency are the elimination of the access-transistors that have considerable static power consumption in small technology nodes, and the memristor's non-volatile attribute which requires no power to maintain the state. Several nanoscale memristive crossbars have been successfully demonstrated recently [33], [40], [41].

Memristive memories can also be utilized to implement power-efficient associative memories [9], [44]. Section IV describes the design and operation of associative memories using linear I-V memristive devices. Fig. 2(a) shows such a typical linear I-V behavior. In Section V, an access-transistor-free associative memory is presented that leverages the rectifying I-V behavior observed in W/SiGe/a-Si/Ag devices, as shown in Fig. 2(b) [33].

IV. Associative Memristive Memory (AMM)

In the following subsections, we describe 1) the integration of the AMM modules with the FPU pipelines commonly found in the state-of-the-art GPUs, 2) the architecture of the AMM modules, and 3) the realization of the inexact matching operation suitable for approximate computing.

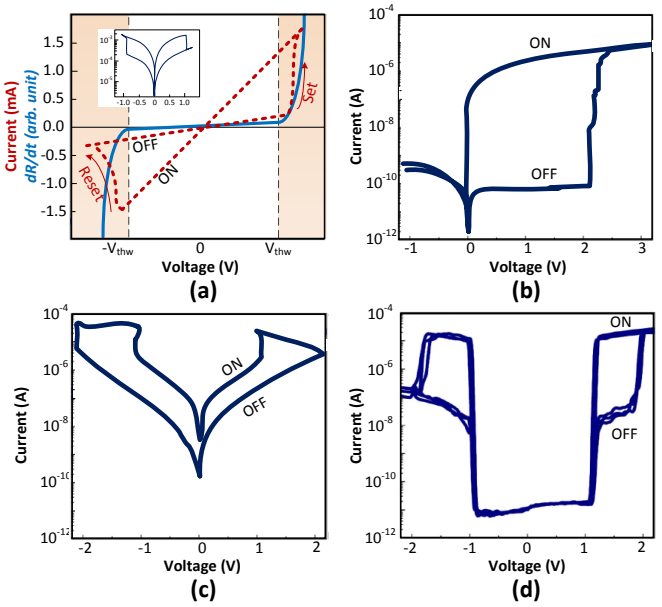


Fig. 2. A memristor's I-V and dR/dt-V characteristic. (a) Exemplar realization: the solid line shows the non-linearity in the rate of change for the resistance of the device based on the applied voltage. Applying voltages over a threshold voltage V_{thw} effectively changes the state of the device. The dashed red lines show the linear I-V characteristic observed in Pd/Ta_2O_{5-x}/TaO_y/Pd memristors [32]. (b) The rectifying I-V behavior observed in W/SiGe/a-Si/Ag memristors [33]. Application of negative voltages results in negligible currents for both ON and OFF devices. (c) Non-linear I-V characteristics of a Pt/TiO_x/TiO_{2-x}/Pt memristor [42]. (d) Non-linear I-V diagram of a memristor with a field assisted super-linear threshold selector (FAST^{TM}) [43]. The inset of sub-figure (a) as well as the sub-figures (b)-(d) show the current in log-scale, to highlight the non-linearity in the I-V characteristics.

A. Integration of the AMM modules with GPU Architecture

We focus on the integration of an AMM module on one of the most recent GPUs from AMD, the Southern Islands family (Radeon HD 7000-series). The Southern Islands is based on AMD's Graphics Core Next which is a RISC single instruction, multiple data (SIMD) architecture. We target the Radeon HD 7970 device which has 32 compute units as shown in Fig. 3. Every compute unit contains a scheduler and a set of four SIMD execution units. Each SIMD execution unit has 16 cores, which gives a total number of 64 cores per compute unit. The core executes the instructions using integer units (IUs) and floating point units (FPUs). A vector instruction is fetched once and executed in a SIMD fashion within the compute unit. After the fetch and decode stages, the source operands for each instruction are read from vector registers or local memory. When the source operands are ready, the execution stage starts to issue the operations into the IUs or FPUs. The execution stage of every FPU has a latency of six cycles and a throughput of one instruction per cycle [45]. Finally, the result of the computation is written back to the destination operands.

In order to fully exploit the energy saving potential of both partial computing-with-memory and approximate computing, we tightly integrate an AMM module to a FPU [21], [9]. For each type of FPU (e.g., ADD or SQRT), we first identify a set of highly frequent input operands. This is done during a profiling stage described in Section VI-B. Such highly frequent inputs and their pre-calculated results are then stored in the

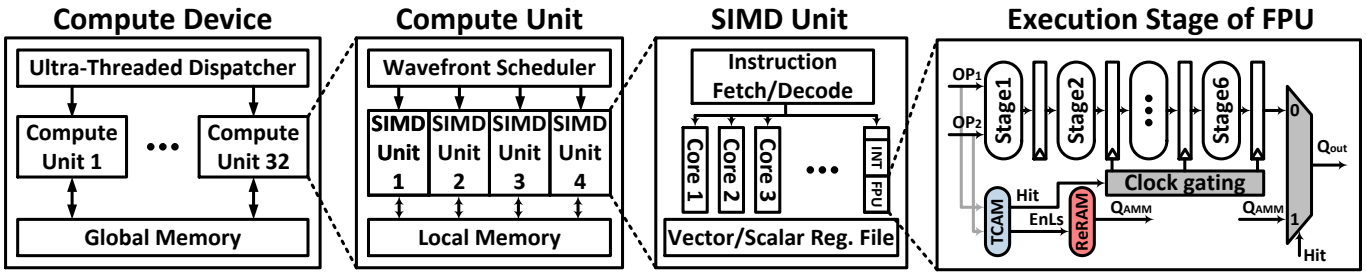


Fig. 3. Radeon HD 7970 GPU architecture. An AMM module (TCAM+MRAM) is integrated to the execution stage of each FPU, which is not present in the original GPU design.

AMM module. During the normal operation of the FPUs, input patterns to FPUs are searched for within the highly frequent input patterns already stored in the AMM module. In case of a match, the pre-computed result stored in the AMM module (Q_{AMM} in the rightmost part of Fig. 3) is read and propagated toward the end of the pipeline along with a hit signal. The propagated hit signal clock-gates the remaining stages of the FPU to save energy by avoiding unnecessary re-execution. In case of a miss, the FPU works normally, and its result (Q_{FPU}) is selected as the pipeline output. The hit signal selects either Q_{FPU} or Q_{AMM} as the output. The use of an AMM module enables significant power savings, as 1) it performs the match operation and returns the output at lower energy costs compared to the FPU, thanks to the ultra-low-power characteristics of the memristive memories, and 2) several stages of the FPU pipeline are clock-gated in case of match events. It is worth mentioning that the reduction in the power consumption is achieved without affecting the latency and the throughput of the FPU, as the AMM modules work under the same clock frequency and their outputs are pipelined to match the latency of the FPU pipeline. The rightmost part of Fig. 3 illustrates the integration of an AMM module with the FPU.

The AMM module consists of two pipelined stages: (1) a memristive TCAM which stores and searches for the highly frequent sets of input operands detailed in Section IV-B, and (2) a resistive random access memory (ReRAM) that maintains the pre-calculated output results for each set of such frequent operands discussed in Section IV-C. For each FPU, in the first stage, the inputs are also fed to the memristive TCAM. The TCAM searches to determine whether the input pattern belongs to the set of high frequency input patterns. In case of a match, the result of the operation is read from the ReRAM in the second stage of the pipeline.

B. Memristive TCAM

CMOS-based associative memories require a large number (10+) of transistors per bit to store and search for patterns [46], incurring significant penalty in terms of power and area [10]. In an attempt to improve the energy- and area-efficiency of the TCAM modules, *memristive TCAMs* have been proposed [12], [22] that employ nano-scale and ultra-low-power memristive devices instead of transistors that are larger than memristors and consume more energy.

Fig. 4(a) shows the structure of a memristive TCAM. Each TCAM row stores one pattern of the highly frequent input

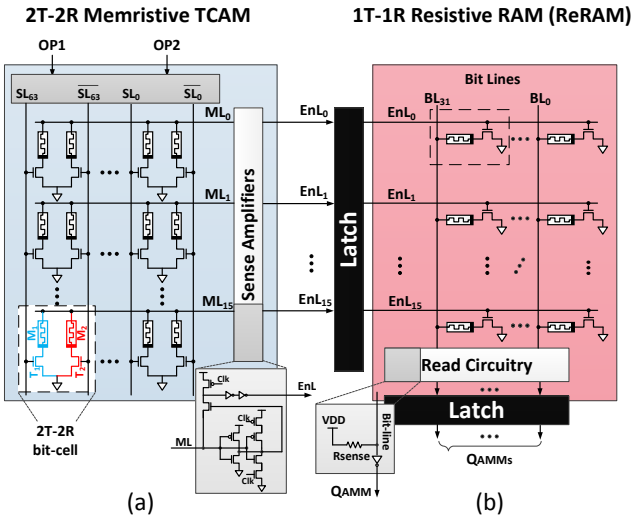


Fig. 4. A 16-row AMM module to represent the partial functionality of a FPU with two input operands, e.g., ADD or MUL. (a) Two 32-bit operands are stored on each row of the 2T-2R TCAM. The match lines are sensed to detect a match/mismatch and drive Enable Lines (EnLs). (b) EnLs activate a row in the 1T-1R resistive random access memory (ReRAM) to read the computation results stored in the ReRAM.

operands. We use a 2-transistor/2-memristor structure for each bit (i.e., a 2T-2R bit-cell) for the TCAM design inspired from [12]. Memristors store the data pattern according to the Table I, while the gates of the access transistors are driven by the search pattern (SL and $\overline{\text{SL}}$) to facilitate the search operation. All the 2T-2R bit-cells on each TCAM row are connected to a "match-line" (ML), the voltage value of which indicates a match/mismatch at the end of the search cycle. Each 2T-2R bit-cell provides two possible "discharge paths" from the ML to the ground, each consisting of a transistor and a memristor in series. The highlighted TCAM cell in Fig. 4(a) illustrates the two discharge paths per bit-cell in red and blue. To program the TCAM, individual memristive devices are written into by applying the write voltages on the MLs and connecting the access-transistors via the search lines (SL and $\overline{\text{SL}}$).

The search operation in a memristive TCAM consists of two phases, a *precharge phase* and an *evaluation phase*. During the precharge phase, the MLs are precharged, as illustrated in Fig. 5(a): both transistors in the 2T-2R cell are OFF, disconnecting the discharge paths, while the match line is being precharged. In the evaluation period, the search pattern and its complement are applied on SL and $\overline{\text{SL}}$ respectively. Hence, either T_1 or T_2 is turned OFF, leaving only one possible

TABLE I
RESISTANCE PATTERN STORED IN 2T-2R TCAM CELLS.

Logic Value	M_1	M_2
'1'	OFF	ON
'0'	ON	OFF
'X' (Don't care)	OFF	OFF

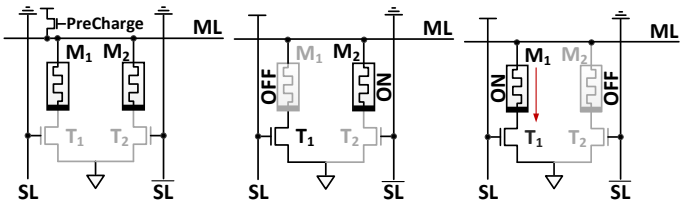
discharge path per bit-cell. In case of a match, the memristive device connected to the ON transistor is in the OFF state, which effectively disconnects the remaining discharge path, and thus prevents the discharge of the ML. A TCAM match is illustrated in Fig. 5(b): A logic '1' is searched for in a cell storing a logic '1'. Both discharge paths are disconnected via either an OFF memristor (M_1) or an OFF transistor (T_2). In case of a mismatch, however, the memristor connected to the ON transistor is also in the low-resistance ON state, providing a highly conductive path between ground and the match-line which discharges the ML quickly. A TCAM mismatch is shown in Fig. 5(c): A logic '1' is searched for in a cell storing a logic '0'. Both the transistor and the memristor on one discharge path are ON (M_1 and T_1), leading to the discharge of the ML. At the end of the evaluation phase, the ML is sampled to determine the "match" output.

In each TCAM row, a ML is shared among W 2T-2R cells, where W is the number of bits in each word. In case of an exact word match, i.e. bit-by-bit, the ML stays charged for an extended period of time as all the discharge paths are disconnected in every 2T-2R cell. If the pattern-under-search and the stored pattern mismatch by even a single bit, the ML is discharged quickly because of the existence of highly conductive path(s) between the ML and ground. This provides a clear margin between an exact match and mismatches. *As the number of bit-mismatches increases, the ML will be discharged even faster. We exploit this property to design inexact matching suitable for approximate computing described in Section IV-D.*

In order to increase the noise margin and provide a digital match/mismatch output signal, a clocked self-referenced sensing circuitry is utilized [12]. Fig. 7(a) illustrates the evolution of the digital "match" signal, i.e., the output of the clocked self-reference sensing circuitry, during the evaluation phase for different number of bit-mismatches based on SPICE simulations. As it is expected, this signal drops faster with more bit-mismatches. The digital match signals are sampled (i.e., latched) at the end of the evaluation phase. A logic '1' means that the line is not discharged yet, indicating a match. The latched match signals are then fed to the ReRAM as enable lines (EnL), to read the previously-computed and stored results from the corresponding word-line in the ReRAM. The logical OR of the EnLs provides a "hit signal" which indicates that the result is provided by the AMM module.

C. Resistive RAM (ReRAM)

Fig 4(b) shows the structure of the ReRAM module that stores the pre-computed results of the highly frequent input patterns. In our implementation, the ReRAM uses 1T-1R bit-cells to store each bit [47]. The memory is programmed by applying proper write voltages on the bit-lines, while



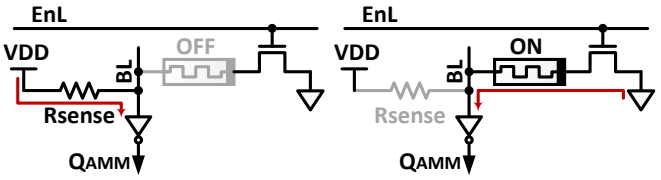
(a) Precharge (b) Search: Match (c) Search: Mismatch

Fig. 5. 2T-2R TCAM operation. (a) Precharge: Both SL and $\overline{\text{SL}}$ are driven to ground, disconnecting both discharge paths. A precharge circuity precharges the ML. (b) Match: {search pattern, stored pattern} = {1,1}. Both discharge paths are disconnected by either an OFF transistor or an OFF memristor, preventing the discharge. (c) Mismatch: {search pattern, stored pattern} = {1,0}. M_1 and T_1 are both ON, providing a low-resistance discharge path to discharge the ML. Transistors are not drawn to scale.

enable lines are used to select the target cell. During the read operation, the enable lines are driven by the EnLs provided by the TCAM. Either one or none of the EnLs are active based on a hit or a miss in the TCAM stage, respectively. The active EnL selects a row in the ReRAM: The access transistor in the 1T-1R cell is turned on, which provides a path from the bit-line to the ground, through the selected memristor. Each bit-line is connected to a read circuit consisting of a sense resistor R_{Sense} and a NOT gate. The read circuitry works as a voltage divider. If the selected memristor stores a high-resistance logic '0', $R_{Memristor} >> R_{Sense}$ and thus the voltage drop on the sense resistor is negligible and the output of the NOT gate will be a logic '0'. If the memristor stores a low-resistance logic '1', $R_{Sense} >> R_{Memristor}$, thus most of the voltage is dropped on the sense resistor and the output of the read circuitry is a logic '1'. Fig. 6 illustrates the ReRAM operation.

D. AMM Module with Inexact Matching

Fig. 7(a) shows the effect of the number of bit-mismatches on the discharge time. It can be observed that when the number of bit mismatches is small (e.g., 1 or 2), there is a clear difference in the drop time of the mismatched signals with distinct number of bit-mismatches. This clear margin allows us to provide a controllable "inexact" matching by shortening the evaluation period (i.e., faster sampling), or similarly by reducing the supply voltage (i.e., voltage overscaling or VOS) while preserving the same evaluation period. In both cases, a pattern with a Hamming distance of 1 or 2 (i.e., the number of bit-mismatches) is considered as a "match". This inexact matching causes approximations during search operation, which introduces a limited number of errors in the computation. We show that the quality degradation due to the incurred error is tolerable in several image processing applications, i.e., PSNR does not fall below a certain threshold. Hence, we enable approximate computing on AMM module



(a) Read '0' (b) Read '1'

Fig. 6. 1T-1R Memristive RAM operation.

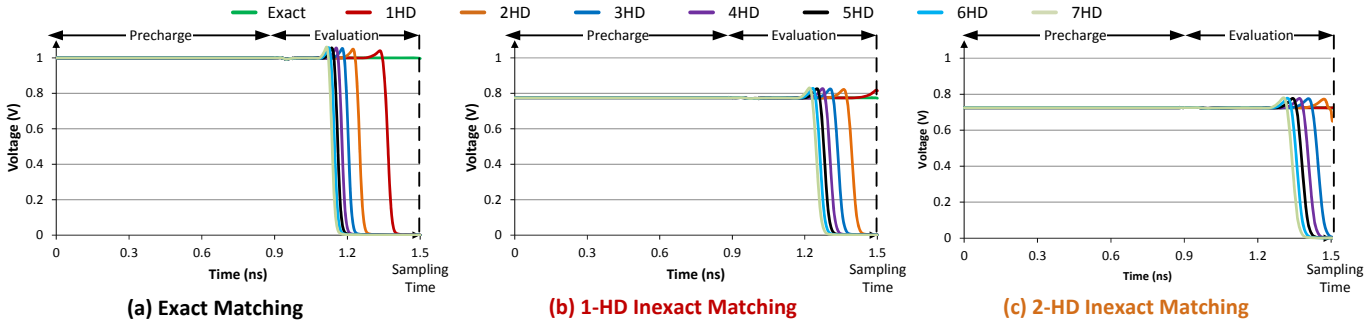


Fig. 7. 2T-2R TCAM match operation under VOS. Different x-HD lines show the drop time of the digital match signal for TCAM rows storing patterns with a Hamming distance of x from the pattern under search (i.e., the number of bit-mismatches). (a) Exact matching at 1V. (b) Lowering the V_{DD} to 775mV, increases the discharge time, thus patterns with 1-bit mismatch are still considered as matched. (c) Further lowering the V_{DD} to 725mV, enables inexact matching of patterns with 2 bit Hamming distance to the search pattern. 0T-2R TCAM exhibits a similar behavior with slightly different voltage levels and drop times.

by applying VOS which further lowers the power consumption [9].

Fig. 7 illustrates the effect of VOS on the matching operation of the TCAM module. Operating at the nominal voltage of 1V guarantees an exact matching with no errors as shown in Fig. 7(a). Decreasing the V_{DD} results in longer discharge times. Hence, given the same clock period, with a one-bit mismatch, the match line is not yet discharged by the time of sampling, manifesting itself as a match. Thus, by reducing the supply voltage to 775mV, the TCAM reports a hit, even if the input pattern has a Hamming distance of 1 with any of the stored patterns (1-HD inexact matching). This way, an inexact matching operation is realized with reduced power consumption due to the lowered V_{DD}. VOS down to 725mV matches the input patterns with up to 2 bit-mismatches (2-HD inexact matching) at even lower energy costs. Further lowering the supply voltages results in an abrupt increase in the number of bit-mismatches that cannot be tolerated in approximate computing.

Inexact matching reduces the power consumption by VOS at the cost of relaxing the matching criteria. There are two downsides to this approach: (1) possibility of a false match, and reporting a wrong output as the result of the computation, and (2) having several matches, which would enable several word-lines in the ReRAM, resulting in the logical OR of the corresponding outputs being reported as the output of the AMM module, Q_{AMM}. Possibility of several matches can be avoided by ensuring a minimum Hamming distance among the stored patterns in the TCAM (e.g., 3 and 5 for 1-HD and 2-HD inexact matching respectively); this is practical given the typical TCAM word-size (i.e., 32, 64, or 96), and the small number of TCAM rows. As for the case of a false match, its likelihood is reduced by a proper sizing of AMM module described in Section VI-F. The significance of the error caused by a false match can also be decreased, by utilizing a *hybrid* fashion in the design of the AMM module. Such a hybrid module performs exact matching on few critical bits (e.g., the sign and exponent bits), and limits the inexact matching search to less critical bits. In Section VI-F, we apply the proposed inexact matching to different image processing kernels that can tolerate the incurred errors and display a high PSNR while benefiting from the lower energy consumption.

V. Access-Transistor-Free Memristive TCAMs

Utilizing a 2T-2R bit-cell structure to implement memristive TCAM significantly improves the area and power cost compared to the conventional CMOS TCAMs. However, this structure does not fully exploit the ultra-low-power and extremely-small characteristics of the memristive memory: the energy consumption of the access-transistors dominates the total energy consumption of the TCAM module, and the size of the TCAM cell is still limited by that of the transistors.

Here we implement an access-transistor-free memristive TCAM to address such issues that utilizes purely resistive bit-cells (i.e., 0T-2R) to store the data patterns and perform the match operation. The 0T-2R TCAM has the same interface and operates based on a similar "match-line discharge" mechanism: a match-line is precharged in the precharge phase, which will be discharged in the evaluation phase in case of a mismatch. However, this structure solely relies on memristors as the ON/OFF switches to disconnect the discharge paths. Fig. 8(a) shows the implementation of the access-transistor-free TCAM module.

The access-transistors are eliminated by exploiting the diode-like rectifying I-V characteristics, shown in Fig. 2(b), observed in W/SiGe/a-Si/Ag memristive devices [33]. Such memristive devices demonstrate an inherent rectifying behavior; while in case of positive biases, memristors in the ON (or OFF) state exhibit an electrical resistance of R_{ON} (or R_{OFF}), applying a negative bias yields the device to demonstrate a fairly high resistance, $R_{Rectify}$, in orders of giga-ohms for both ON and OFF states.

A. Access-Transistor-Free Memristive TCAM Operation

Each bit-cell is composed of two rectifying memristors, M_1 and M_2, which are placed in between the match line (ML) and the search lines (SL or $\overline{\text{SL}}$). The rectifying memristors' device stack is fabricated such that it exhibits a large $R_{Rectify}$ when the ML's voltage is larger than the voltage of the search line(s). The access-transistor-free bit-cell provides two paths from the ML to the search lines (SL and $\overline{\text{SL}}$). Hence, during the evaluation phase that the search pattern and its complement are applied on SL and $\overline{\text{SL}}$, one path discharges the ML, while the other tends to charge it: Either SL or $\overline{\text{SL}}$ is at V_{DD} while the other one is grounded. However, the "charge-path" is always

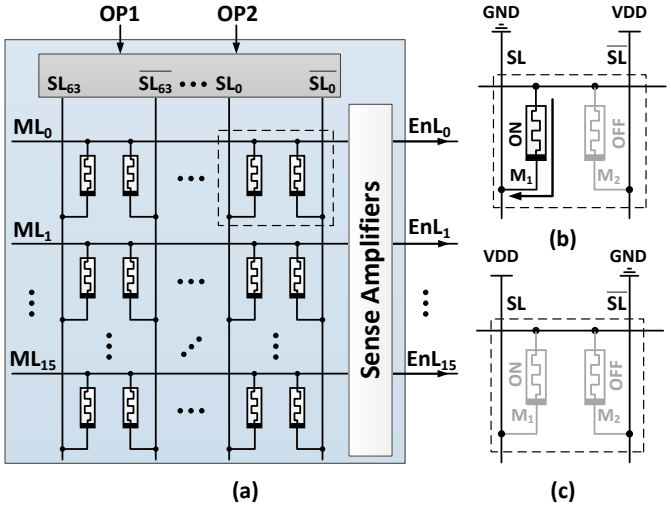


Fig. 8. A 16-row 0T-2R TCAM. (a) The access-transistor-free TCAM follows the same structure and interface as the 2T-2R TCAM, while 0T-2R bit-cells, highlighted in the dashed box, replace the 2T-2R bit-cells in Fig. 4(a). (b) Mismatch for {search pattern, stored pattern} = {0,1}. M_1 is ON, thus provides a low-resistance path from ML to the grounded SL that discharges the match line. The arrow shows the discharge path. M_2 is reverse-biased, and thus disconnects ML and $\overline{\text{SL}}$ (i.e., at V_{DD}). (c) Match for {search pattern, stored pattern} = {1,1}. M_1 is ON, but reverse-biased, thus exhibits a high rectifying resistance, disconnecting SL from ML. M_2 is also OFF, thus disconnects $\overline{\text{SL}}$ and ML, preventing the discharge.

reverse-biased, as the ML voltage level is always below V_{DD}. Hence, the charge-path is always disconnected according to the rectifying behavior of the W/SiGe/a-Si/Ag devices. This leaves the bit-cell with one possible "discharge path" which is connected (or disconnected) if the data stored in the bit-cell mismatches (or matches) the search pattern. Table II shows the resistance patterns stored in the memristive devices of a 0T-2R bit-cell for different logic values. The following elaborates on the match and mismatch events in the 0T-2R TCAM in details.

Consider the 0T-2R bit-cell in Fig. 8(b) that stores a logic '1'; M_1 and M_2 are in the ON and OFF states respectively. When searching for a logic '0', SL and $\overline{\text{SL}}$ are driven to GND and V_{DD} respectively. Hence, the low-resistance M_1 connects the match line to the grounded SL, and thus discharges the ML indicating a mismatch. Note that while $\overline{\text{SL}}$ is at V_{DD}, it does not "charge" the ML, as M_2 is reverse-biased and exhibit a large resistance.

Fig. 8(c) shows the case in which a search pattern '1' is applied to a bit-cell also storing a logic '1': SL and $\overline{\text{SL}}$ are driven to V_{DD} and GND respectively, while $\{M_1,M_2\}$ are in the {ON,OFF} states. In this case, both memristors exhibit a high-resistance as M_1 is reversed-biased and M_2 is in the high-resistance OFF state. As a result, a bit-match, does not affect the ML's voltage.

Note that the rectifying behavior of the W/SiGe/a-Si/Ag memristive devices [33] is crucial to the functionality of the 0T-2R TCAM cell. Without this inherent rectifying behavior, the "charge-paths" could keep charging the ML during the evaluation period, resisting the discharge of the ML even in case of bit-mismatches.

B. 0T-2R TCAM Design Considerations

The 0T-2R structure utilizes the search lines to discharge the ML, rather than the global GND lines. As a result, the line drivers for the search lines should be properly sized to enable effective discharge of several match-lines: large buffers are needed for large TCAMs which could dominate the energy consumption of the TCAM module and incur energy penalty. However, in Section VI-F we show that for the purpose of approximate computing-with-memory, the optimum size of a TCAM is sufficiently small, e.g., ≤ 32 rows. The buffers necessary for such a small TCAM have a negligible effect on the performance or the energy consumption of the TCAM.

TABLE II
RESISTANCE PATTERN STORED IN 0T-2R TCAM CELLS.

Logic Value	M_1	M_2
'1'	ON	OFF
'0'	OFF	ON
'X' (Don't care)	OFF	OFF

The 0T-2R structure generally has smaller noise margins compared to the 2T-2R TCAM. This is is due to the elimination of the transistors from the TCAM and relying only on memristors to disconnect the discharge paths: An OFF transistor typically have higher resistance values compared to an OFF memristor, and hence, it is a better "switch" to disconnect the discharge paths. The reduced noise margin limits the feasible amount of voltage overscaling to achieve 1-HD and 2-HD inexact matching, compared to the 2T-2R structure. Nevertheless, we have observed a consistent inexact matching behavior for the 0T-2R TCAM similar to the digital match signals shown in Fig 7, but with a shift in the VOS values. Moreover, we show that the energy saving due to the elimination of the transistors in a 0T-2R TCAM, is generally more significant than the possible energy saving due to further VOS in the 2T-2R structure. The 0T-2R structure also reduces the area overhead of the TCAMs. With the elimination of the transistors, the memristive TCAM can be monolithically integrated on top of the CMOS FPUs to increase the area efficiency, as shown in Fig. 9.

VI. EXPERIMENTAL SETUP AND RESULTS

In this section, we briefly describe our experimental setup, the AMM programming and execution flow, and evaluation of the AMM effectiveness in improving energy efficiency of FPUs in GPUs.

A. Experimental Setup

We focus on the AMD Southern Islands GPU, Radeon HD 7970. However, our technique is similarly applicable to other GPUs, as it leverages the application-level-tolerance to errors

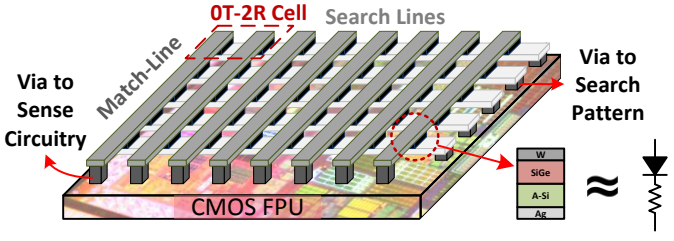


Fig. 9. Monolithical integration of a 0T-2R TCAM on top of a CMOS FPU.

which is an application property and is independent of the underlying architecture. Note that the proposed method is mainly suitable for data-intensive kernels that are amenable to approximate computing, such as applications in multimedia and vision domains. Such applications often exhibit inherent data-level parallelism that makes them ideal for GPU execution [48], [8]. Image processing applications are adopted from AMD APP SDK v2.5 [15] which is a software ecosystem written in OpenCL, suitable for stream applications. Multi2Sim [45], a cycle-accurate CPU-GPU simulation framework, is used for profiling and simulations. Four image processing filters are examined in this study: Roberts, Sobel, Sharpen, and Shift. These kernels typically apply a 2D convolution; we examine frequently activated FPUs during the kernel executions: adder (ADD), multiplier (MUL), multiply-accumulator (MAC), and SQRT. Accordingly, the 6-stage balanced FPUs are generated and optimized using FloPoCo [49]. These FPUs are synthesized and mapped using a 45-nm ASIC flow. The front-end flow has been performed using *Synopsys Design Compiler*, while *Synopsys IC Compiler* has been used for the back-end. The FPUs have been optimized for power and a signoff clock period of 1.5ns. Finally, *Synopsys PrimeTime* is used to report the power consumption at the nominal operating voltage of 1.0V.

AMM modules are designed with different word-sizes based on the type of FPU: the TCAM has a word-size of 32-bit for SQRT, 64-bit for ADD, MUL, and 96-bit for MAC, considering the single precision FPU operands. The resistive memory, or ReRAM, module has a fixed word-size of 32-bit for any FPU to maintain the outputs. Transistor-level SPICE simulations are performed in *Cadence Virtuoso* to estimate power and delay of the AMM module at the worst corner with regard to the data patterns. Regular memristive devices with an exemplar 50KΩ R_{ON} and 50MΩ R_{OFF} are employed to implement the 2T-2R TCAM cells, while rectifying memristive devices reported in [33] are utilized for 0T-2R TCAM implementation. Measurements on fabricated rectifying memristors exhibit R_{ON}, R_{OFF}, and $R_{Rectify}$ values of 50KΩ, 50GΩ, and 8GΩ respectively. Line resistance and capacitance values of $0.02\Omega/nm$ and $1.2aF/nm$ are derived from [50]. The 2T-2R AMM implementation exhibits 1-HD and 2-HD inexact matching behaviors at 775mV and 725mV respectively. The 0T-2R, however, performs inexact matching at slightly higher voltages, i.e., 830mV and 800mV for 1-HD and 2-HD inexact matching respectively. This is due to the reduced noise margins as discussed in Section V-B. Note that these voltages are selected conservatively to ensure correct 1-HD and 2-HD inexact matching for different types and sizes of AMM modules. A worst-case precharge time of 0.9ns is considered for all cases to ensure successful precharge operation even at the lowest voltage level (i.e. 725mV). We integrate a functional model of the AMM module into Multi2Sim for every FP operation to quantify the hit rates and PSNR drops.

B. AMM *Programming and Execution Flow*

The execution flow of an AMM-integrated GPU has two main stages: (1) design time profiling, and (2) run-time computing-with-memory. Fig. 10 illustrates this execution flow. The profiling stage identifies the computations with a high frequency of occurrence. In this stage, we have an OpenCL kernel and a host code with a training input dataset. To expose highly frequent set of operands at the finest granularity, we independently profile each type of FPU. The output of this stage is the list of highly frequent computations (HFC) for each FP operation: a sorted list of the input operand(s) and the corresponding result. The list is sorted based on the frequency of occurrence of input operand(s) and does not consider the commutative property of the operations. That is, $A+B$ and $B+A$ are considered as two different entries. This is to match the search behavior of the AMM modules.

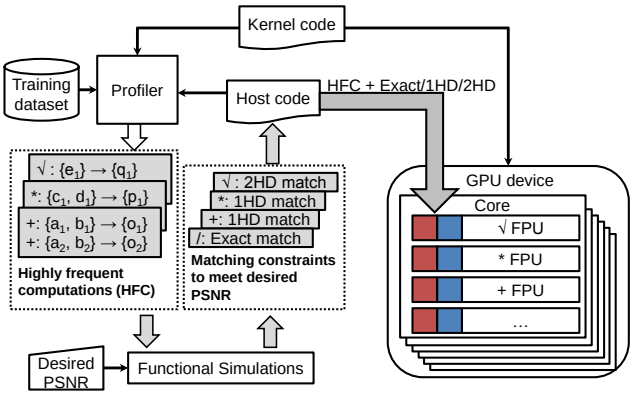
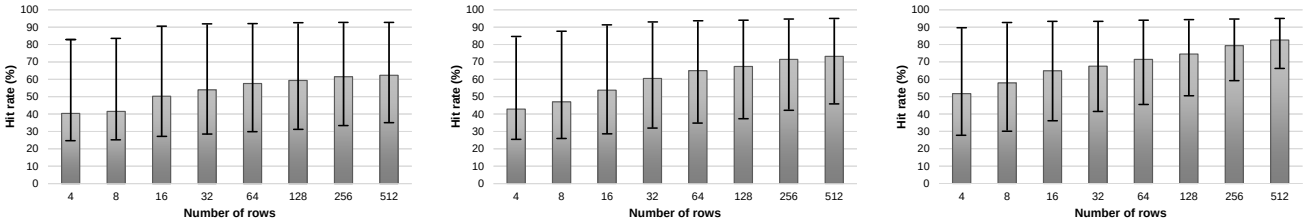


Fig. 10. Programming and execution flow for AMM modules.

Next, we need to determine the tolerable level of inexact matching in the TCAMs for each kernel. To this end, we leverage the Southern Islands functional simulator. We adjusted the simulator accordingly to incorporate the proposed AMM modules. The simulator starts with the exact matching and then increases the degree of approximation step-by-step by applying 1-HD and 2-HD inexact matching. The exact matching output is then considered as the *golden* reference and the outputs of the 1-HD and 2-HD inexact matching is compared with that to measure the PSNR. The maximum degree of inexact matching is determined for each AMM module such that the output maintains a desired PSNR $\geq$ 30dB in spite of the incurred errors. It is worth mentioning that the profiling stage is a *one-off* activity whose cost is amortized across all future usages of the kernel.

In the next step, the host code can transfer the output of the profiling stage to the AMM modules for run-time reuse. The AMD compute abstraction layer (CAL) provides a run-time device driver library that supports code generation and kernel loading. furthermore, it allows the host program to interact with the stream cores at the lowest-level. The AMM module is designed to be addressable by software therefore the host code can program it using the CAL. Right before launching the kernel execution, the host code programs the AMM modules: for each type of FP operation activated during the kernel, a subset of HFCs in conjunction with the degree of applicable inexact matching is set for the corresponding AMM modules. This could be up to few hundred bytes of data depending on the size of the AMM.

Note that the proposed method updates the values in the



(a) Exact matching (b) 1-HD inexact matching (c) 2-HD inexact matching

Fig. 11. Hit rate versus size of AMM for MAC operation during Roberts filter executions.

AMM module only once per kernel, and thus: (1) enhances the lifetime of the AMM module by putting a minimum write stress on the memristive devices, and (2) minimizes the performance overhead of the required write operations to a negligible level. This is in contrast to methods such as [51] that use memristors in place of regular registers. Such methods are critical with respect to the current limitations in number of write cycles for memristor devices. Typical memristive devices exhibit an endurance of 10^8 write operations [52].

C. Design Space for AMM

We explain the design space for utilizing the AMM modules as a case study for Roberts filter, one of our edge detection kernels. We evaluate the trade-off between the size of AMM module, i.e., the number of rows that store different patterns, with its hit rate. A higher hit rate means a greater number of operands are matched with the stored patterns in AMM module. Hence, there is no need for re-computing the results for those input patterns which leads to higher energy saving. We quantify the hit rate of an AMM module for multiply-accumulator (MAC) operator for 100 test input images. Fig. 11 summarizes the minimum, the maximum, and the average (shown in bars) hit rates of the AMM module for a wide range of sizes. The experiment is repeated for all three matching constraints.

Fig. 11(a) shows the hit rates for the exact matching. A 4-row AMM module displays the hit rates in the range of 25%–83%. Increasing the size of the AMM from 4-row to 8-row, and to 16-row improves the average hit rate from 40% to 42%, and to 50%. Overall, the average hit rate increases about 12% when the number of rows increases from 16 to 512. Such high hit-rates are achieved due to the value locality and similarity in GPU applications. For example, the Roberts filter typically applies a 2-D convolution with a matrix of fixed weights on input pixels, while the values of adjacent input pixels are mostly in a similar range [48]. A similar trend of the hit rate versus the AMM size is observed for the inexact matching, as shown in Figs. 11(b)-(c). When the number of rows increases from 16 to 512, the average hit rate improves about 19% and 18% for 1-HD and 2-HD inexact matching, respectively. Fig. 11 also illustrates that an AMM with a given size experiences higher hit rates by switching from the exact matching to any of the inexact matching modes. For instance, the hit rate of a 4-row AMM increases 12% on average (from 40% to 52%) by using 2-HD inexact matching instead of the exact matching. This increased hit rate is due to the relaxed matching constraint in the AMM modules: more of input patterns are approximately matched with the stored patterns.

In a nutshell, choosing a large AMM size has two disadvantages. (1) It diminishes the gain of energy saving, because after a certain size the average hit rate almost saturates, while the energy consumption of the AMM increases for larger sizes. For example, increasing the AMM size from 8-row by 64× only brings a 25% higher hit rate with 2-HD inexact matching. This significantly lowers the hit rate per unit of power consumed by the AMM. In Section VI-F, we show that enlarging the AMM beyond a certain size will not bring any energy saving. (2) It increases the likelihood of false matches that might quickly drop PSNR below the desired threshold. Our profiling results indicate that Roberts filter is able to tolerate the errors in computations (an average PSNR of 34dB) with the AMM modules with up to 512 rows using 2-HD inexact matching. Increasing the AMM size above 512 rows drops the PSNR below 30dB. Visual depiction and the corresponding PSNR of different matching constraints for one of the test images is shown in Fig. 12.

D. Energy Consumption of AMM Modules versus FPUs

Table III summarizes the energy consumption per operation for individual FPUs, and for the corresponding AMM modules with different number of entries. Furthermore, energy numbers are reported for AMM modules with different matching criteria and both 2T-2R and 0T-2R TCAM implementations. Reported energy numbers demonstrate the considerable energy saving potential of the AMM modules. For example, in case of an 8-row AMM module for a SQRT operation, a 2T-2R AMM can perform the SQRT operation at ∼8X lower energy costs compared to the CMOS FPU, at the nominal voltage of 1.0V. A 0T-2R implementation of the same size AMM further improves the energy efficiency, with ∼13X less energy compared to the FPU.

By allowing the inexact matching, the energy saving of an 8-row SQRT AMM can be further improved by factors of 17X and 23X, for 1-HD and 2-HD inexact matching respectively.

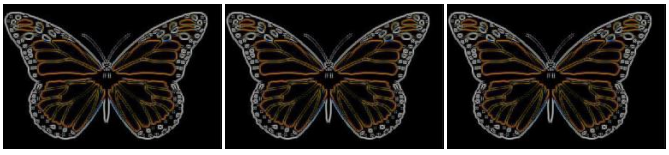


Fig. 12. Visual depiction of the output quality degradation with exact, 1-HD, 2-HD inexact matching for the Roberts filter, based on an 8-row AMM implementation.

TABLE III
ENERGY CONSUMPTION (FJ) PER OPERATION IN 45NM TECHNOLOGY FOR FPUS AND 0T-2R AND 2T-2R AMM MODULES.

Module	FPU (1.0V)	Matching	2T-2R AMM				0T-2R AMM			
			4-row	8-row	16-row	32-row	4-row	8-row	16-row	32-row
ADD	4742	Exact	1176	1403	1858	2740	749	848	1056	1511
		1-HD	644	732	906	1262	476	539	677	1007
		2-HD	505	555	709	999	468	464	649	913
MUL	9891	Exact	1176	1403	1858	2740	749	848	1056	1511
		1-HD	644	732	906	1262	476	539	677	1007
		2-HD	505	555	709	999	468	464	649	913
SQRT	9983	Exact	934	1137	1528	2322	629	727	932	1380
		1-HD	514	594	756	1084	402	464	599	961
		2-HD	397	441	593	864	394	444	575	956
MAC	12051	Exact	1410	1653	2122	3096	867	967	1178	1639
		1-HD	774	867	1052	1422	550	612	753	1037
		2-HD	612	667	832	1124	533	584	718	986

The reduction in power consumption is realized by lowering the operating voltage of the modules by up to 27%. Such saving trend is consistent for different types of FPUs, and different sizes of AMM modules. Note that while the AMM modules work reliably under the designated VOS points (see Section IV-D), these VOS points are well below the critical operating voltage of CMOS FPUs. Hence, such aggressive VOS cannot be applied to the FPUs, as it would cause an unacceptably huge number of timing errors.

Fig. 13 highlights the energy consumption of different implementations of an ADD operation: FPU, 2T-2R AMM and 0T-2R AMM. The energy consumption of the AMM module is directly proportional to its size. Thus increasing the size of the AMM beyond a limit will hurt the energy efficiency, as the AMM module would consume higher energy than the CMOS FPU. Hence, larger AMM modules, e.g., greater than 64, incur energy penalty, even in case of an ideal hit-rate.

E. Energy Consumption of 0T-2R AMM *versus 2T-2R* AMM

Table III also shows the power saving merits of the 0T-2R implementation compared to the 2T-2R implementation. At the nominal voltage of 1.0V, the 0T-2R module consumes ∼2X less energy, for all TCAM sizes. It can be observed that the energy saving potential of the 0T-2R implementation tends to be lower for: (1) smaller TCAMs; and (2) relaxed matching criteria, e.g., 2-HD inexact matching.

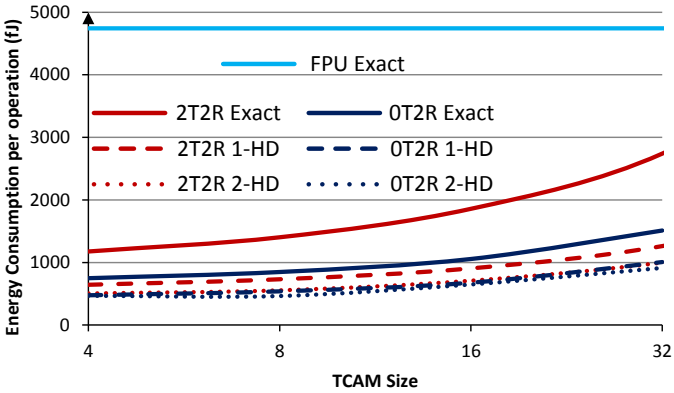


Fig. 13. Energy consumed during an ADD operation. Approximate computing-with-memory using the proposed AMM module offers a significant potential reduction in the energy consumption over the CMOS FPU.

The 0T-2R implementation improves the energy consumption by removing the transistors from the bit-cell structure of the TCAM. However, the remaining parts of AMM module, i.e., line drivers, sensing circuitry, latches, and the second stage 1T-1R MRAM, are not affected. Hence, better energy savings are achieved with larger TCAMs that have more entries (rows) and a larger number of bits per entry. An AMM module to perform a MAC operation has 96 bit-cells per entry. Hence, using 0T-2R implementation saves 96×2 transistors per row compared to a 2T-2R implementation. In contrast, the reduction in the number of transistors per row is 32×2 for a SQRT AMM. As a result, for an exemplar 32-row AMM with the exact matching operation, the energy saving advantage of the 0T-2R implementation over the 2T-2R implementation reduces from 47% for a MAC AMM to 40% for a SQRT AMM.

The smaller energy saving advantage of the 0T-2R implementation in case of inexact matching is mainly due to the higher supply voltages at which a 0T-2R AMM performs inexact matching compared to the 2T-2R AMM, i.e., 830mV and 800mV versus 775mV and 725mV for 1-HD and 2-HD inexact matching respectively. For example, while operating a 16-row 0T-2R ADD AMM at the exact matching mode offers 43% energy saving compared to the 2T-2R implementation, relaxing the matching criterion to 1-HD and 2-HD matching decreases the advantage of 0T-2R over 2T-2R to 25% and 8% respectively. To summarize, the 0T-2R structure offers better energy saving for all exact and 1-HD inexact matching cases, as well as 2-HD inexact matching for operations with 2 or 3 operands, i.e. ADD, MUL, and MAC. However, in the worst case of 2-HD inexact matching for one operand SQRT operation, the 0T-2R implementation has up to 10% higher energy consumption.

F. Energy Saving with Corresponding PSNR

In this section, we take the hit-rates into the account, and present the actual energy saving offered by the AMM modules during the execution of the image processing kernels. Moreover, we consider the effect of the 1-HD and 2-HD inexact matching on the output quality. For the four image processing kernels, our profiler utilizes 10% of the Caltech 101 computer vision dataset [16] as the training set in order

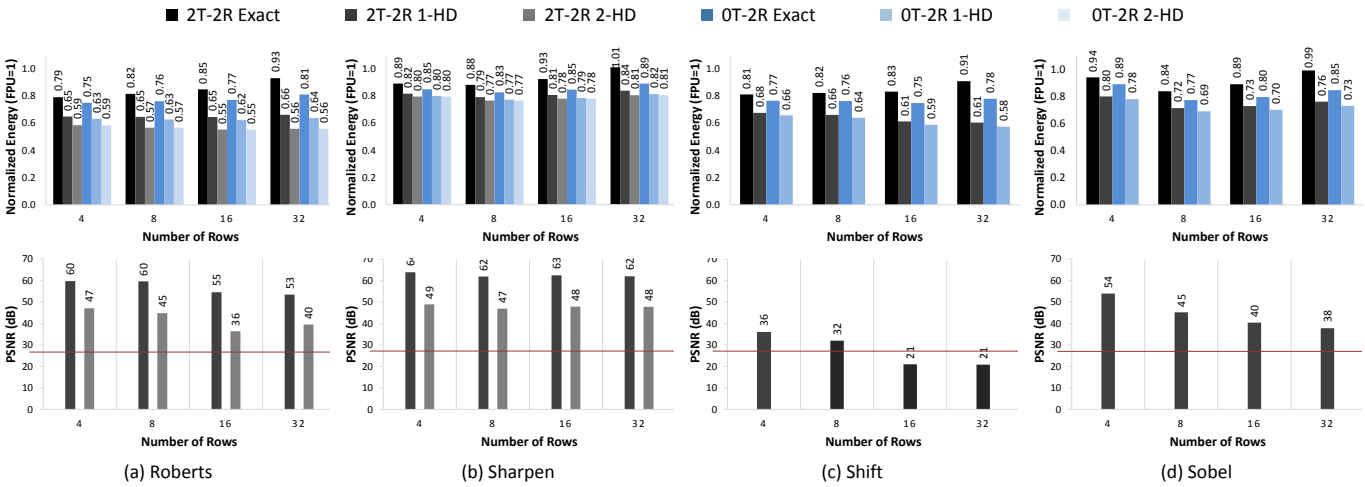


Fig. 14. AMM normalized energy and PSNR: for different TCAM implementations, sizes, matching criteria, and kernels – values are averaged over the full dataset [16]. It should be noted that same-sized 0T-2R and 2T-2R implementations yield the same PSNR.

to extract the HFC as explained in Section VI-B. Next, the host code loads the top X pairs of the HFC to the X-entry AMM modules, before the kernel execution, where $X \in \{4, 8, 16, 32\}$. The full Caltech dataset [16] is then considered to quantify the average energy saving and the corresponding average PSNR degradation of employing inexact AMM modules.

Fig. 14 shows the energy consumption, normalized to that of FPUs for each kernel, as well the degraded PSNR in case of inexact matching. Different bars represent different cases of 0T-2R or 2T-2R implementation, as well as various matching criteria. AMM modules with sizes smaller than 32 rows provide a significant range of energy efficiency (23%–45%) for different application kernels. Relaxing the matching criteria improves the energy efficiency of the AMM module, while an acceptable PSNR above 30dB is maintained for most of the applications. For example, the energy saving advantage of utilizing 2T-2R AMM modules for Roberts filter, increases from 21% in case of exact matching, to 45% by allowing 2-HD inexact matching, while maintaining a PSNR of 36dB. The increased energy efficiency is due to the aggressive voltage overscaling of the AMM module.

The size of the AMM also affects the amount of energy saving. As shown in Fig. 14(d), 2T-2R AMM modules with 4 rows improve the average energy of Sobel by 20% using 1-HD inexact matching. Increasing the AMM size to 8 rows leads to a higher average energy saving of 28% due to the higher hit rate. However, increasing the size beyond 8 rows is not optimum because the potential energy saving offered by the extra hit events is less than the energy overhead due to the increased AMM sizes.

Fig. 14 also demonstrates the energy-saving merits of the 0T-2R implementation. In Shift, for example, an exact-matcher 0T-2R implementation offers 4% to 13% lower energy consumption than a 2T-2R implementation, depending on the AMM size. Relaxing the matching criterion to allow 1-HD inexact matching reduces the energy saving advantage of the 0T-2R implementation to about 2%. It is shown that in case of 2-HD inexact matching, both 0T-2R and 2T-2R implementations offer about the same energy efficiency.

The average PSNR degrades with larger AMM sizes, which is due to the higher possibility of false matches. While Sharpen and Roberts filters exhibit acceptable PSNR even in case of 2-HD approximate matching, Sobel and Shift kernels can only tolerate errors introduced in the 1-HD approximate matching. Increasing the number of stored patterns beyond 32 (or 8) for Sobel (or Shift) abruptly increases the likelihood of a false match, which introduces more computational errors and drops the PSNR below 30dB. Considering an acceptable PSNR being $\geq$ 30dB, AMM modules with 8 rows provide the best average energy saving for Sobel (31%), Sharpen (23%), and Shift (36%); Roberts exhibits the best energy saving of 45% with 16-rows AMM modules. Choosing 8-row as the size of the AMM modules brings an average energy saving of 33% across all four kernels, while guaranteeing an acceptable PSNR.

VII. Conclusion

This work aims to address the following challenge: *how to exploit both memristive technology and approximate computing to increase energy efficiency in GPUs?* Our approach is to integrate associative memristive memory (AMM) modules with the FPUs in GPUs to save energy in threefold. (1) AMM modules pre-store highly frequent input patterns to avoid actual computations on the FPUs by recalling the output at lower energy costs. (2) We reduce this energy cost by applying VOS to AMM modules that unleashes an untapped energy efficiency through approximate computing. (3) We further reduce the energy consumption of the TCAM part by using an access-transistor-free crossbar that realizes a purely memristive TCAM, i.e., 0T-2R bit-cell structure. We demonstrated that the proposed inexact matching due to the VOS and the 0T-2R AMM structure results in significant energy saving, and the incurred errors due to the inexact matching can be tolerated by image processing kernels, displaying an acceptable PSNR larger than 30dB.

Acknowledgments

This work was supported by the NSF's Variability Expedition (1029783), ERC-AdG MultiTherman (291125), FP7 Virtical (288574), and AFOSR-MURI (FA9550-12-1-0038).

REFERENCES

[1] NVIDIA's GeForce GTX TITAN X. http://developer.amd.com/tools-and-sdks/opencl-zone/amd-accelerated-parallel-processing-app-sdk/.

[2] A. Andrei, M.T. Schmitz, P. Eles, Zebo Peng, and B.M. Al Hashimi. Quasi-static Voltage Scaling for Energy Minimization with Time Constraints. In *Design, Automation and Test in Europe, 2005. Proceedings*, pages 514–519 Vol. 1, March 2005.

[3] S.G. Ramasubramanian, S. Venkataramani, A. Parandhaman, and A. Raghunathan. Relax-and-Retime: A Methodology for Energy-efficient Recovery based Design. In *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, pages 1–6, May 2013.

[4] J. Patel. CMOS Process Variations: A Critical Operation Point Hypothesis. Online Presentation, 2008.

[5] Liangzhen Lai and Puneet Gupta. A Case Study of Logic Delay Fault Behaviors on General-Purpose Embedded Processor Under Voltage Overscaling. Technical report, Dept. of Electrical Engineering, University of California Los Angeles, Los Angeles, CA 90095, August 2014.

[6] Mark Gottscho, Abbas BanaiyanMofrad, Nikil Dutt, Alex Nicolau, and Puneet Gupta. Power / Capacity Scaling: Energy Savings With Simple Fault-Tolerant Caches. In *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference*, DAC '14, pages 100:1–100:6, New York, NY, USA, 2014. ACM.

[7] V. Gupta, D. Mohapatra, Sang Phill Park, A. Raghunathan, and K. Roy. IMPACT: IMPrecise Adders for Low-power Approximate Computing. In *Low Power Electronics and Design (ISLPED) 2011 International Symposium on*, pages 409–414, Aug 2011.

[8] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger. Neural Acceleration for General-Purpose Approximate Programs. In *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pages 449–460, Dec 2012.

[9] Abbas Rahimi, Amirali Ghofrani, Kwang-Ting Cheng, Luca Benini, and Rajesh K. Gupta. Approximate Associative Memristive Memory for Energy-efficient GPUs. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, DATE '15, pages 1497–1502, San Jose, CA, USA, 2015. EDA Consortium.

[10] 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. *Solid-State Circuits, IEEE Journal of*, 38(1):155–158, Jan 2003.

[11] V. C. Ravikumar, Rabi N. Mahapatra, and Laxmi Narayan Bhuyan. EaseCAM: An Energy and Storage Efficient TCAM-Based Router Architecture for IP Lookup. *IEEE Trans. Comput.*, 54(5):521–533, May 2005.

[12] Jing Li, R.K. Montoye, M. Ishii, and L. Chang. 1 Mb 0.41 μm^2 2T-2R Cell Nonvolatile TCAM With Two-Bit Encoding and Clocked Self-Referenced Sensing. *Solid-State Circuits, IEEE Journal of*, 49(4):896–907, April 2014.

[13] Qing Guo, Xiaochen Guo, Yuxin Bai, and Engin İpek. A Resistive TCAM Accelerator for Data-intensive Computing. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-44, pages 339–350, New York, NY, USA, 2011. ACM.

[14] Meng-Fan Chang, Jui-Jen Wu, Tun-Fei Chien, Yen-Chen Liu, Ting-Chin Yang, Wen-Chao Shen, et al. Embedded 1Mb ReRAM in 28nm CMOS with 0.27-to-1V read using swing-sample-and-couple sense amplifier and self-boost-write-termination scheme. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, pages 332–333, Feb 2014.

[15] Amd app sdk v2.5. http://www.amd.com/stream.

[16] Caltech 101. http://www.vision.caltech.edu/Image_Datasets/Caltech101/.

[17] S. Paul, S. Chatterjee, S. Mukhopadhyay, and S. Bhunia. Nanoscale Reconfigurable Computing using Non-volatile 2-D STTRAM Array. In *Nanotechnology, 2009. IEEE-NANO 2009. 9th IEEE Conference on*, pages 880–883, July 2009.

[18] S. Paul, S. Chatterjee, S. Mukhopadhyay, and S. Bhunia. Energy-Efficient Reconfigurable Computing Using a Circuit-Architecture-Software Co-Design Approach. *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, 1(3):369–380, Sept 2011.

[19] J. Cong, M. Ercegovac, Muhuan Huang, Sen Li, and Bingjun Xiao. Energy-efficient Computing using Adaptive Table Lookup based on Nonvolatile Memories. In *Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on*, pages 280–285, Sept 2013.

[20] H. Jarollahi, N. Onizawa, V. Gripon, N. Sakimura, T. Sugibayashi, T. Endoh, H. Ohno, T. Hanyu, and W.J. Gross. A Nonvolatile Associative Memory-Based Context-Driven Search Engine Using 90 nm CMOS/MTJ-Hybrid Logic-in-Memory Architecture. *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, 4(4):460–474, Dec 2014.

[21] Abbas Rahimi, Amirali Ghofrani, Miguel Angel Lastras-Montano, Kwang-Ting Cheng, Luca Benini, and Rajesh K. Gupta. Energy-Efficient GPGPU Architectures via Collaborative Compilation and Memristive Memory-Based Computing. In *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference*, DAC '14, pages 195:1–195:6, New York, NY, USA, 2014. ACM.

[22] F. Alibart, T. Sherwood, and D.B. Strukov. Hybrid CMOS/Nanodevice Circuits for High Throughput Pattern Matching Applications. In *Adaptive Hardware and Systems (AHS), 2011 NASA/ESA Conference on*, pages 279–286, June 2011.

[23] Hang Zhang, Mateja Putic, and John Lach. Low Power GPGPU Computation with Imprecise Hardware. In *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference*, DAC '14, pages 99:1–99:6, New York, NY, USA, 2014. ACM.

[24] Abbas Rahimi, Luca Benini, and Rajesh K. Gupta. Spatial Memoization: Concurrent Instruction Reuse to Correct Timing Errors in SIMD Architectures. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 60(12):847–851, Dec 2013.

[25] Abbas Rahimi, Luca Benini, and Rajesh K. Gupta. Temporal Memoization for Energy-efficient Timing Error Recovery in GPGPUs. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6, March 2014.

[26] R. Waser and M. Aono. Nanoionics-based Resistive Switching Memories. *Nature Materials*, 6:833–840, 2007.

[27] L. Chua. Resistance Switching Memories are Memristors. *Applied Physics A: Materials Science & Processing*, 102(4):765–783, 2011.

[28] K.-T. Cheng and D. B. Strukov. 3D CMOS-Memristor Hybrid Circuits: Devices, Integration, Architecture, and Applications. In *International Symposium on Physical Design (ISPD)*. IEEE, March 2012.

[29] J. J. Yang, M. D. Pickett, X. Li, D. A. A. Ohlberg, D. R. Stewart, and R. S. Williams. Memristive Switching Mechanism for Metal/Oxide/Metal Nanodevices. *Nature nanotechnology*, 3(7):429–433, 2008.

[30] Dmitri B Strukov and R Stanley Williams. Exponential Ionic Drift: Fast Switching and Low Volatility of Thin-film Memristors. *Applied Physics A*, 94(3):515–519, 2009.

[31] E. Linn, R. Rosezin, C. Kugeler, and R. Waser. Complementary Resistive Switches for Passive Nanocrossbar Memories. *Nature materials*, 9(5):403–406, 2010.

[32] Y. Yang, P. Sheridan, and W. Lu. Complementary Resistive Switching in Tantalum Oxide-based Resistive Memory Devices. *Applied Physics Letters*, 100(20):203112, 2012.

[33] K.-H. Kim et al. A Functional Hybrid Memristor Crossbar-array/CMOS System for Data Storage and Neuromorphic Applications. *Nano Letters*, pages 389–395, 2012.

[34] A. Ghofrani, M. A. Lastras-Montao, and K.-T. Cheng. Toward Large-Scale Access-Transistor-Free Memristive Crossbars. In *Asia and South Pacific Design Automation Conference*. IEEE, 2015.

[35] A Ghofrani, M.A Lastras-Montano, and Kwang-Ting Cheng. Towards Data Reliable Crossbar-based Memristive Memories. In *Test Conference (ITC), 2013 IEEE International*, pages 1–10, Sept 2013.

[36] C. Ho, C.-L. Hsu, C.-C. Chen, J.-T. Liu, C.-S. Wu, C.-C. Huang, C. Hu, and F.-L. Yang. 9nm Half-pitch Functional Resistive Memory Cell with 1 uA Programming Current using Thermally Oxidized Sub-stoichiometric WOx Film. In *International Electron Devices Meeting (IEDM)*, pages 19.1.1–19.1.4. IEEE, 2010.

[37] S. Pi, P. Lin, and Q. Xia. Cross Point Arrays of 8nm x 8nm Memristive Devices Fabricated with Nanoimprint Lithography. *Journal of Vacuum Science Technology B: Microelectronics and Nanometer Structures*, 31(6):06FA02–06FA02–6, Nov 2013.

[38] J. P. Strachan, A. C. Torrezan, M. Feng Miao, M. D. Pickett, J. J. Yang, W. Yi, G. Medeiros-Ribeiro, and R. S. Williams. Measuring the Switching Dynamics and Energy Efficiency of Tantalum Oxide Memristors. *Nanotechnology*, 22(50):505402, November 2011.

[39] International Technology Roadmap for Semiconductors (ITRS), Emerging Research Devices, 2011.

[40] Q. Xia, W. M. Tong, W. Wu, J. J. Yang, X. Li, W. Robinett, T. Cardinali, et al. On the Integration of Memristors with CMOS Using Nanoimprint Lithography. In *SPIE Advanced Lithography*, pages 727106–727106. International Society for Optics and Photonics, 2009.

[41] A. Kawahara, R. Azuma, Y. Ikeda, K. Kawai, Y. Katoh, et al. An 8 Mb Multi-layered Cross-point ReRAM Macro with 443 MB/s Write Throughput. *IEEE Journal of Solid-State Circuits*, 48(1):178–185, 2013.

[42] J. J. Yang, M.-X. Zhang, M. D. Pickett, M. Feng, J. P. Strachan, W.-D. Li, W. Yi, D. A. A. Ohlberg, B. J. Choi, W. Wu, J. H. Nickel, G. Medeiros-Ribeiro, and R. S. Williams. Engineering Nonlinearity into Memristors for Passive Crossbar Applications. *Applied Physics Letters*, 100(11):113501, 2012.
[43] S. H. Jo, T. Kumar, M. Asnaashari, W. D. Lu, and H. Nazarian. 3D ReRAM with Field Assisted Super-Linear Threshold (FASTTM) Selector Technology for Super-dense, Low Power, Low Latency Data Storage Systems. In *Design Automation Conference (ASP-DAC), 2015 20th Asia and South Pacific*, pages 575–575, Jan 2015.
[44] L.A. Lastras-Montano, M.M. Franceschini, B. Rajendran, and C. Lam. Coding for Sensing in Content Addressable Memories. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 1923–1927, June 2010.
[45] Multi2Sim: A Heterogeneous System Simulator. https://www.multi2sim.org/.
[46] Kostas Pagiamtzis and Ali Sheikholeslami. Content-Addressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey. *Solid-State Circuits, IEEE Journal of*, 41(3):712–727, 2006.
[47] Seungjun Kim, Hu Young Jeong, Sung Kyu Kim, Sung-Yool Choi, and Keon Jae Lee. Flexible Memristive Memory Array on Plastic Substrates. *Nano letters*, 11(12):5438–5442, 2011.
[48] Mehrzad Samadi, Davoud Anoushe Jamshidi, Janghaeng Lee, and Scott Mahlke. Paraprox: Pattern-based Approximation for Data Parallel Applications. In *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, pages 35–50, New York, NY, USA, 2014. ACM.
[49] FloPoCo: Floating-Point Cores Generator. http://flopoco.gforge.inria.fr/.
[50] M. A. Lastras-Montaño, A. Ghofrani, and K.-T. Cheng. Architecting Energy Efficient Crossbar-based Memristive Random Access Memories. In *ACM/IEEE Internation Symposium on Nano-scale Architectures (NANOARCH)*, July 2015.
[51] S. Kvatinsky, Y. Nacson, Y. Etsion, E. Friedman, A Kolodny, and U. Weiser. Memristor-based Multithreading. *Computer Architecture Letters*, 2013.
[52] Hong-Yu Chen, Shimeng Yu, Bin Gao, Peng Huang, Jinfeng Kang, and H.-S.P. Wong. Hfox based vertical resistive random access memory for cost-effective 3d cross-point architecture without cell selector. In *Electron Devices Meeting (IEDM), 2012 IEEE International*, pages 20.7.1–20.7.4, Dec 2012.

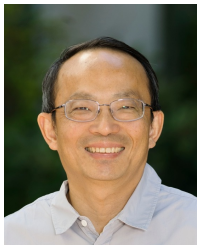
Amirali Ghofrani (S'09) is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering at the University of California Santa Barbara. He has received his B.Sc. in computer engineering from the University of Tehran, Iran, in 2007, and two M.Sc. degrees in computer engineering from the University of Tehran, Iran, and University of California Santa Barbara in 2010 and 2013. His main research focus is on addressing variation and reliability issues of memristive memories and finding novel applications for them. His other research interests include reliability and testing of Network on Chip interconnections, Transaction Level Modeling (TLM), high level synthesis, and assertion based verification. He has more than 20 publications in these research areas. He was the recipient of the 2011 International Test conference Best Paper Award, 2013 IEEE Philadelphia Section & Test Technology Technical Council Gerald W. Gordon Award, and several leadership awards in UCSB. Other than research, he enjoys playing soccer, and has won three championship titles in the UCSB intramural competitions.

Abbas Rahimi (S'10) is currently a postdoctoral scholar in the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley. He is a member of the Berkeley Wireless Research Center and collaborating with the Berkeley Redwood Center for Theoretical Neuroscience. He received the M.S. and Ph.D. degrees in computer science and engineering from the University of California, San Diego in September 2015; and the B.S. degree in computer engineering from the University of Tehran, in March 2010. His research interests are in brain-inspired hyperdimensional computing, variability tolerance, massively parallel memory-centric architectures and interconnections. In these areas, he has published more than 30 papers in top tier conferences and journals. He received the Best Paper Candidate at 50th IEEE/ACM Design Automation Conference.

Miguel Angel Lastras-Montaño is a PhD Candidate in the Electrical and Computer Engineering Department at the University of California, Santa Barbara. He received his B.S. degree in Engineering Physics and M.S. degree in Applied Sciences from the Universidad Autonoma de San Luis Potosi, Mexico, in 2008 and 2010, respectively; and a M.S. in Computer Engineering from University of California, Santa Barbara in 2012. His research interests include the architectural aspects of low-power crossbar-based memristive memories and its 3D monolithic integration with standard CMOS processes. He is additionally interested in the general-purpose computing on graphics processing units (GPGPU) for scientific applications.

Luca Benini (F'07) is a Professor of Digital Circuits and Systems at ETH Zurich, Switzerland, and is also a Professor at University of Bologna, Italy. His research interests are in energy-efficient system design and multicore SoC design. He is also active in the area of energy-efficient smart sensors and sensor networks for biomedical and ambient intelligence applications. He has published more than 700 papers in peer reviewed international journals and conferences, four books and several book chapters. He is member of the Academia Europea.

Rajesh K. Gupta (F'04) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 1984; the M.S. degree in electrical engineering and computer science from the University of California, Berkeley, in 1986; and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1994. He is a Professor of computer science and engineering at the University of California, San Diego (UCSD), and holds the Qualcomm endowed chair.



Kwang-Ting Cheng (S'88-M'88-SM'98-F'00) received his Ph.D. in EECS from the University of California, Berkeley in 1988. He worked at Bell Laboratories from 1988 to 1993 and joined the faculty at the University of California, Santa Barbara in 1993 where he is now Associate Vice Chancellor for Research and Professor in Electrical and Computer Engineering. He was the founding director of UCSB's Computer Engineering Program (1999-2002) and Chair of the ECE Department (2005-2008). His current research interests include mobile embedded systems, mobile computer vision, and SoC design validation and test. He has published more than 400 technical papers, co-authored five books, supervised 32 PhD dissertations, and holds 12 U.S. Patents. He currently serves as Director for the Department of Defense MURI Center for 3D hybrid circuits which aims at integrating CMOS with high-density memristors. Cheng, an IEEE fellow, received 10+ Best Paper Awards from various IEEE conferences and journals. He has also received the 2004-2005 UCSB College of Engineering Outstanding Teaching Faculty Award. He served as Editor-in-Chief of IEEE Design and Test of Computers (2006-2009) and was a board member of IEEE Council of Electronic Design Automation's Board of Governors, and IEEE Computer Society's Publication Board, and working groups of International Technology Roadmap for Semiconductors (ITRS).