This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

Approximate Computing using Multiple-Access Single-Charge Associative Memory

Mohsen Imani, *Student Member, IEEE*, Shruti Patil, *Member, IEEE,*
and Tajana Šimunić Rosing, *Senior Member, IEEE*

Computer Science and Engineering Department, University of California San Diego,
La Jolla, CA 92093, USA
Email: {moimani, patil, tajana}@ucsd.edu

Abstract—Memory-based computing using associative memory is a promising way to reduce the energy consumption of important classes of streaming applications by avoiding redundant computations. A set of frequent patterns that represent basic functions are pre-stored in Ternary Content Addressable Memory (TCAM) and reused. The primary limitation to using associative memory in modern parallel processors is the large search energy required by TCAMs. In TCAMs, all rows that match, except hit rows, precharge and discharge for every search operation, resulting in high energy consumption. In this paper, we propose a new Multiple-Access Single-Charge (MASC) TCAM architecture which is capable of searching TCAM contents multiple times with only a single precharge cycle. In contrast to previous designs, the MASC TCAM keeps the match-line voltage of all miss-rows high and uses their charge for the next search operation, while only the hit rows discharge. We use periodic refresh to control the accuracy of the search. We also implement a new type of approximate associative memory by setting longer refresh times for MASC TCAMs, which yields search results within 1-2 bit Hamming distances of the exact value. To further decrease the energy consumption of MASC TCAM and reduce the area, we implement MASC with crossbar TCAMs. Our evaluation on AMD Southern Island GPU shows that using MASC (crossbar MASC) associative memory can improve the average floating point units energy efficiency by 33.4%, 38.1%, and 36.7% (37.7%, 42.6%, and 43.1%) for exact matching, selective 1-HD and 2-HD approximations respectively, providing an acceptable quality of service (PSNR>30dB and average relative error<10%). This shows that MASC (crossbar MASC) can achieve 1.77X (1.93X) higher energy savings as compared to the state of the art implementation of GPGPU that uses voltage overscaling on TCAM.

Index Terms—Approximate computing, Ternary content addressable memory, Associative memory, Non-volatile memory, GPUs

1 Introduction

The massive computation needs of big data requires efficient parallel processors. There is a significant amount of redundant data when processing streaming applications [1], [2]. Associative memory was introduced to exploit this observation and decrease the number of redundant computations [3], [4], [5], [6], [7], [8], [9]. In hardware, associative memories are implemented as look up tables using ternary content addressable memories (TCAMs) [4], [5]. However, TCAMs based on CMOS technology have low density and high energy consumption compared to SRAM [10]. This energy limits the application of TCAMs to classification [11] and IP look-up [12]. Voltage overscaling (VOS) has been used on CMOS-based TCAMs to reduce the energy consumption [13], [14]. However, this increases the system error-rate due to process variations and timing errors.

Non-volatile memories (NVMs) present a new opportunity for efficient memory-based computation using low energy TCAMs [15], [16], [17]. Resistive RAM (ReRAM) and Spin Torque Transfer RAM (STT-RAM) are two types of fast and dense non-volatile memories based on memristor and magnetic tunneling junction (MTJ) devices [18], [19]. Previous work has used NVMs to design fast and energy efficient TCAMs. However, the energy consumption of NVM-based TCAMs is still large because of the high number of charges and discharges in TCAM lines for each search operation. To further reduce NVM-based energy consumption, VOS has been applied on memristive TCAM while accepting results within 1-2 bits Hamming Distance (HD) between input and pre-stored TCAM patterns [6]. This aggressive voltage relaxation on the full TCAM bitline limits the TCAM size and degrades the computation quality of service below the acceptable range.

In this paper we propose an ultra-low energy multi-access single-cycle TCAM (MASC TCAM) which improves the energy consumption of the TCAM by performing multiple search operations after only a single precharging cycle. In conventional TCAMs, the match-lines (MLs) of all TCAM rows precharge to VDD voltage. During the search operation, all MLs, except the hit rows (if any) discharge. As a result, TCAM consumes a lot of energy independently of a hit or a miss. In contrast, MASC decreases the ML switching activity by limiting discharging only to the hit row. Consequently, majority of the miss rows of the TCAM retain their charge after the search operation for more search cycles

xxxx-xxxx/0x/$xx.00 © 200x IEEE Published by the IEEE Computer Society

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

without the need to precharge each time. The proposed MASC design is further split into several short word-size blocks so that the search can be performed in parallel. Each partial MASC TCAM uses an encoding scheme to limit the number of low resistance memristors to one in each block. When an input arrives, it discharges ML (match) just in case that it activates a low resistance cell. In all other cases, the ML stay charge and we can use that for multiple search operations.

Instead of using voltage overscaling to decrease TCAM energy consumption, the MASC TCAM architecture varies the period of precharging cycles on selective TCAM blocks to decrease the energy and control the computational error. The level of approximation is defined by the size of the hamming distance in each partial block. It is controlled by changing when precharging occurs. In addition, our design applys approximation starting from the least significant blocks of MASC which allows the system to balance TCAM energy savings and computational quality of service as a function of the running application. We reduce area overhead by using crossbar TCAM with no access transistors. Our evaluations on AMD Southern Island GPU architecture using eight OpenCL applications shows that using MASC (crossbar MASC) improve the average floating point units energy efficiency by 33.4%, 38.1%, and 36.7% (37.7%, 42.6%, and 43.1%) for exact matching, 1-HD and 2-HD approximation with acceptable quality of service. GPGPU using MASC (crossbar MASC), achieves 1.77X (1.93X) higher energy saving as compared to state of the art design of the GPGPU with voltage overscaled approximate TCAM. Considering both integers and floating point units indicates that the MASC (corssbar MASC) can improve the overall GPGPU computation energy by 34.7% (39.1%) on average delivering acceptable quality of service.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 describes the architecture and challenges of resistive associative memory. The design of the proposed MASC TCAM is described in Section 4. Section 5 discusses experimental results and Section 6 concludes the paper.

2 Related Work

Associative memory in the form of a look-up table has been used with parallel streaming processors to avoid doing redundant computations [3], [4], [5], [6]. In hardware, associative memories are implemented using TCAM blocks. CMOS-based TCAMs consist of two SRAM cells but their cost per bit is 8X more than SRAM [10]. High density and low energy consumption of NVMs such as ReRAM and STT-RAM improve the energy efficiency of memory based computation. ReRAMs have comparable read operation to SRAMs, but have limited endurance (10^6-10^7 write operation), which degrades their lifetime [18]. On the other hand, STT-RAMs have fast reads as well as high endurance (>10^{15}). However, the bidirectional write current and low ON/OFF ratio (~2) increases the area of the MTJ-based TCAM with respect to ReRAM-based TCAM cells [20], [21]. High endurance is necessary for TCAMs since they must be periodically updated. Our design addresses the endurance issue by limiting the write stress to only the start of kernel execution.

Several previous works have used NVMs to design stable and efficient TCAMs [15], [22], [21], [23]. Li et al. [15] designed a 1Mb energy efficient 2T-2R TCAM which is 10X smaller than SRAM-based TCAM. Another 3T-1R TCAM structure has been introduced in [22], which can search the entire CAM array in less than 1ns with very low energy consumption. An efficient 2Kb 4T-2MTJ based TCAM cell is proposed in [23]. This cell is for standby-power-free TCAM and has 86% area reduction respect to SRAM-based TCAM. Hanyu, et al. in [21] introduced 5T-4MTJ TCAM cell which searches input data on cell complementary with very high sense margin. However, the energy consumption of NVM-based TCAMs is still high because of high number of charge and discharge cycles for each search operation [14], [6]. Work in [24] showed that using large temporal memory for computation reuse is not efficient in CMOS, hence they combined both temporal and spatial reuse to get a high hit rate. Approximate TCAM using voltage overscaling is one way to decrease the search energy consumption of associative memories [6]. To improve the computation accuracy due to aggressive voltage overscaling, Imani *et al.* applied approximation selectively on associative memory to limit hamming distances to the least significant bits [25]. To increase the hit rate, wok in [26] proposed an approximate associative memory which can adaptively update CAM values using learning algorithm.

In contrast to previous efforts, we design a new MASC which can perform multiple search operations within a single precharging cycle. In MASC during a search only the match lines discharge while all the missed rows stay charged. The proposed design introduces selective hit line precharging and long precharging refresh to improve the search energy consumption of error-free MASC. Selective MASC block approximation balances processor energy consumption and quality of service using multiple precharging periods as a function of the running application.

3 Background

3.1. Memristive devices

Resistive memory has shown a great potential to be used as high performance NVMs [27]. To enable fast switching, ReRAMs use CMOS-based access transistors. General structure of memristor is based on metal/oxide/metal. Two metal layer (e.g. Pt) sandwich an oxide layer based on Ta, Ti and HF [28] , [29] , [30]. The metal/diode connection usually shows the Ohmic behavior (shown in *Fig. 1*). The data is prestored based on the memristor resistance state.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

The device turns ON by applying negative bias and is turned OFF by applying positive voltage across the device [28]. Read applys a small voltage across the BL and SL nodes and reads the data with a sense amplifier.

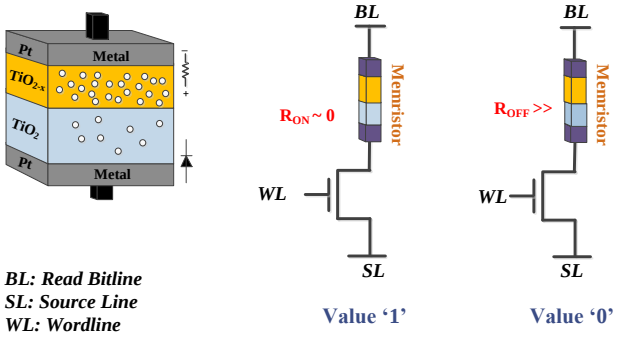


Fig. 1. Working mechanism of ReRAM structure

Crossbar memory is an access-free transistor memory architecture, which can be implemented purely by memristive devices. This memory achieves significantly low energy and high scalability, while occupying negligible area [31], [32]. The area of crossbar resistive memory is $4F^2/n$, where F is the minimum feature size and the n is the number of resistive layers in 3 dimensional space. However, since the cell is an access-free transistor, it can be implemented at the top of the chip. Sneak current is one of the main problem of crossbar resistive memories. This current injects through the ON devices. Various diode-based memristive devices have been proposed to address this undesirable current in crossbar memories [33]. Complementary resistive switching (CRS) is an effective technique to address the sneak current problem. Using $Pd/Ta_2O_{5-x}/TaO_y/Pd$ structure for complementary switching provides high endurance, high OFF resistance (~100GΩ) and switching speed of sub 1-ns [32]. In this memory device, switching is the result of changing the number of oxygen connections in tantalum layers by applying voltage across the device.

3.2 Resistive Associative Memory

Resistive associative memory consists of TCAM and resistive memory (1T-1R memory). In GPUs, associative memories are implemented alongside each FPU execution unit. The frequent input patterns and the related outputs are pre-stored in TCAM and resistive memory respectively (See Fig. 2). Any hit in the TCAM block stops the FPU execution by clock-gating the processor computation. This signal also activates the corresponding line of resistive memory to retrieve the precomputed result. The multiplexer (MUX block) places this result on the output bus. Associative memory is designed to perform the search operation in a single cycle, same as the execution time of each FPU stage. A hit on the TCAM block stops the computation of the rest of FPU stages.

Low energy consumption of the NVM-based associative memory enables application of these memories to query processing [34], search engine [35], text processing [36], image processing [37], pattern recognition [37], data mining [38] and image coding [39]. Several of these applications need large TCAMs with respect to word-size and number of rows to cover a variety of input operands. Designing large TCAMs has following challenges:

- Due to finite ON/OFF resistance ratio of NVMs in TCAM structure, a reliable search operation can occur on TCAMs with short word sizes. Large word sizes increase the leakage current of TCAM lines so that a hit may be incorrectly considered a miss. The effect of process variations makes the TCAM more sensitive to word sizes.
- A TCAM with many rows requires a large input buffer block to distribute the input signals among all the rows. Larger buffer sizes worsen search delay and energy consumption of TCAM.
- The primary factor that limits the number of TCAM rows and word size is the TCAM search energy. A large TCAM consumes a lot of energy which reduces the energy efficiency of the computation. Higher hit rate of a larger TCAM increases the percentage of the time that the processor clock gated, thus improving the energy efficiency.

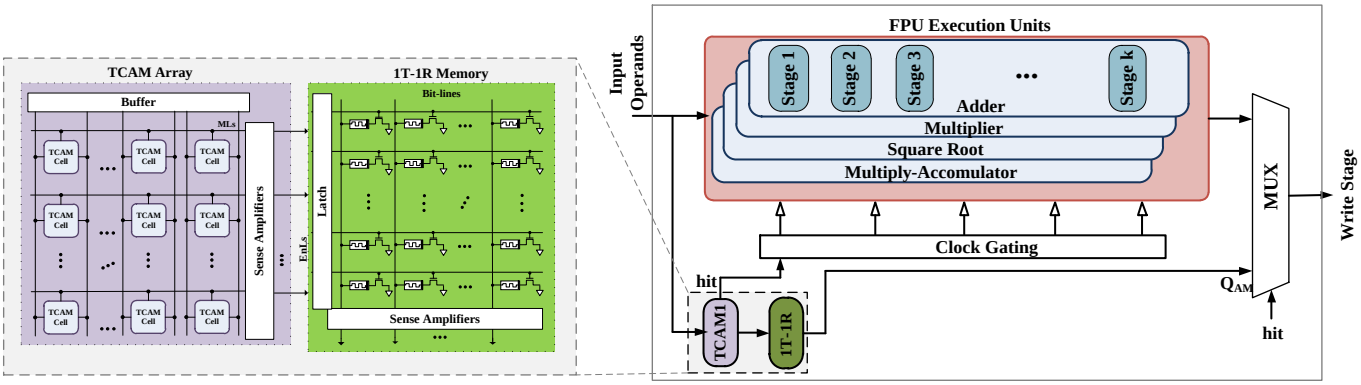


Fig. 2. Integration of associative memory with FPU execution unit

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

Techniques that reduce the energy consumption of associative memories use larger TCAM sizes to improve the TCAM hit rate and FPU energy efficiency. Approximations can be used to decrease the energy consumption of associative memory [6, 40]. These techniques apply VOS on associative memory to reduce the search energy consumption by accepting 1-2 bits hamming distance between input operand and prestored values in TCAM. In approximate associative memory mismatches can occur in any TCAM rows. All rows do not have the same impact on the result of computation. For example, a mismatch in the first set of TCAM rows usually has a larger impact on the computation accuracy as compared to a mismatch in the last rows because the first set of TCAM rows prestore operands with higher probability of hit. Therefore, aggressive voltage relaxation on entire TCAM block can result in high output error rates. For example, our preliminary results show that for *Sobel* and *Shift* applications running on approximate TCAM larger than 16-rows have an unacceptable quality of service. This limits the application of the associative memory to error-tolerant applications. Approximate TCAM with a larger number of rows increase the number of inexact matches and degrade the computational accuracy. This limits the size of a TCAM block fewer rows which reduces the intended benefits.

We explore design techniques that can enable large search energy savings with controllable quality of service. Our proposed design allows multiple search operations on a TCAM block with a single precharge. It implements selective approximation on TCAM blocks using a novel technique of long precharge refresh times to adaptively balance the computational energy and quality of service.

4 MASC TCAM ARCHITECTURE

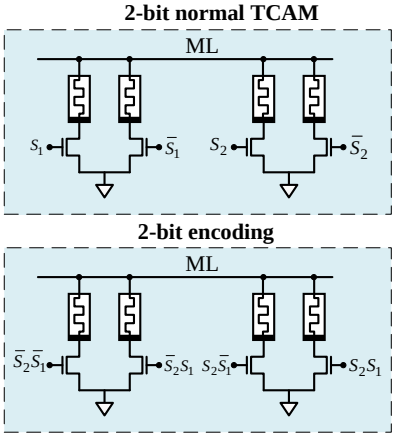
4.1 Motivation

In this paper we propose a Multiple-Access Single-Charge (MASC) TCAM which can perform multiple search operations with a single precharge of MLs. In conventional TCAM, all rows (MLs) precharge to VDD voltage. In search mode, if the input pattern has a mismatch with any prestored rows, the ML starts discharging. In the case of a hit in the TCAM, the voltage on hit ML(s) stays high, while all other rows discharge to zero. For the next search operation we must precharge all rows again. This high energy consumption of precharging and discharging of MLs is the primary cause of the high TCAM search energy consumption.

To decrease the charge and discharge energy, we propose MASC which can perform search operations with extremely low energy consumption. The functionality of the proposed cell is similar to conventional TCAM with the difference that during the search operations, MLs discharge only when there is a match with the input patterns. In case of a mismatch, TCAM rows retain their precharge voltage. This allows us to use ML voltage of missed rows to perform more search operations without another precharge cycle. After each search, we selectively precharge the hit MLs using a simple circuit. A complete refresh of MLs is performed after a specific number of cycles. This significantly reduces the energy consumption of the TCAM by decreasing the number of charges and discharges.

4.2 MASC TCAM Cell and Encoding

2T-2R TCAM: We use an encoding technique [15] to design the MASC TCAM, as shown in *Fig. 3*. Each TCAM cell has two memristor devices and two access transistors. The values of these memristors are prestored in the TCAM so that each block has only one low resistance (L). The tables of store and search operations of 2-bit TCAM with and without encoding schemes are shown in *Fig. 3*. For the search operation, the input patterns are first transformed by the encoding block. This block activates one of the four possible combinations of the input signals ($\bar{S}_1\bar{S}_2$, $\bar{S}_1S_2$, $S_1\bar{S}_2$, S_1S_2) based on *Fig. 3*. These signals activate only one access transistor per 2-bit encoding block. For a hit, the access transistor activates the memristor with low resistance and ML starts to discharge. In case of a mismatch, the access transistor related to just one of the high resistance (H) devices will be connected. This limits ML leakage currents to one transistor in each encoding block.



Store Data	Without Encoding	Two-bit Encoding
1 1	L H L H	H H H L
1 0	L H H L	H H L H
0 1	H L H L	H L H H
0 0	H L H L	L H H H
1 ×	L H H H	H H L L
0 ×	H L H H	L L H H
× 1	H H L H	H L H L
× 0	H H H L	L H L H
× ×	H H H H	L L L L

Search Data	Without Encoding	Two-bit encoding
1 1	0 1 0 1	0 0 0 1
1 0	0 1 1 0	0 0 1 0
0 1	1 0 0 1	0 1 0 0
0 0	1 0 1 0	1 0 0 0

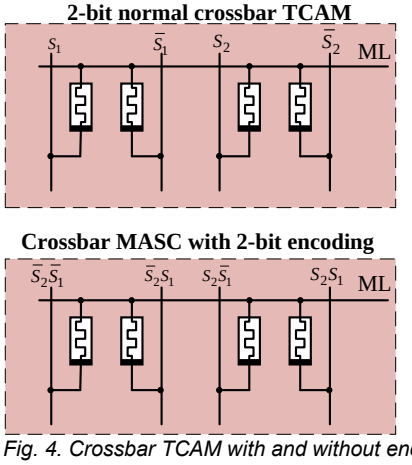
Fig. 3. 2T-2R TCAM cell with and without encoding scheme

In conventional TCAM, the number of leaky cells depends on the length of bitline. For large word sizes, the access transistors of many cells may be activated depending on the input pattern. This can discharge the ML unintentionally and yield incorrect search results. For efficient and reliable search in TCAM, sufficient margin must be present between the match and mismatch currents. The worst-case difference between the currents occurs when all cells are matched (no discharge), and when just one of the cells is unmatched. In contrast, in the proposed MASC TCAM, increasing the size of encoding blocks improves the TCAM sense margin and makes the search operation easier since in any encoding block size, the number of leaky cells from

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

ML on a miss is limited to a single cell.

Crossbar TCAM: We next propose a new MASC TCAM based on crossbar memristor memories. The structure of proposed crossbar MASC is shown in Fig. 4. This cell consists of two memristor devices with no access transistor. Data prestores in memristors based on their resistance states. During the search operation, the ML precharges to V_{Read} and the input (search data) activates the select lines. The ML starts discharging in the case of a mismatch between prestore and input data. In MASC, we use the encoding scheme to prestore and search the data on crossbar TCAM. In crossbar MASC with 2-bit encoding (see Fig. 5), one of the memristive devices is in low resistance (L) and others are in high resistance mode (H). To enable MASC functionality on the crossbar cell, during each search operation, just one of the select lines connects to ground (GND) and all others are connected to V_{dd}.



Store Data	Without Encoding	Two-bit Encoding
11	LHLH	HHHL
10	LHHL	HHLH
01	HLHL	HLHH
00	HLHL	LHHH
1×	LHHH	HHLL
0×	HLHH	LLHH
×1	HHLH	HLHL
×0	HHHL	LHLH
××	HHHH	LLLL

Search Data	Without Encoding	Two-bit encoding
11	0101	1110
10	0110	1101
01	1001	1011
00	1010	0111

Fig. 4. Crossbar TCAM with and without encoding scheme

Our evaluation shows that the lack of access transistor increases the leakage power of the crossbar MASC relative to 2T-2R structure. This results in lower noise margin and higher precharging cycles of crossbar MASC compared to 2T-2R TCAM, to obtain the same level of accuracy. Crossbar MASC can be implemented on the top of FPU with no area penalty. To control the leakage current of resistive based TCAM, we use W/SiGe/SiAg structure with high OFF resistance [31]. Our evaluation shows that the proposed small size crossbar TCAM has acceptable delay so that it can be placed alongside FPU pipelined stages.

4.3 MASC TCAM architecture

The architecture of proposed MASC TCAM is shown in *Fig. 5*. The MASC search operation is done in two stages. To avoid long search delay and energy consumption of large word size TCAMs, we split the search into several short word size TCAM searches. Each partial TCAM searches a part of the input data. In GPGPU with 32-bit search operations, we separated the bitline at 2:16 (sixteen 2-bit TCAMs), 4:8 TCAM (eight 4-bit TCAMs), and 8:4 TCAM (four 8-bit TCAMs) searches. This split sends 16, 8 and 4 output signals to the second TCAM stage (see *Fig. 5*). The second stage logically *OR*s the *EnL* signals of the first stage TCAM using another TCAM stage. If the input pattern matches in the same row of all partial TCAMs, the input pattern is considered as a hit on that row. This requires a single TCAM cell and small encoder block that produces complement signals (e.g. $\overline{EnL1}\_1$, $\overline{EnL1}\_2$, …, $\overline{EnL1}\_m$ for m-bit encoding) to activate the first TCAM cell. In other words, the data in the second TCAM stage is fixed, and if the data matches in the same row of all partial TCAMs, a single cell is activated.

4.4 Refresh and MASC Approximation

We define the refresh period as the number of search cycles that can be performed without precharging. This number depends on the word-size of the partial TCAM blocks. This precharging cycle is determined by the amount of the cell leakage through the ML in every search operation. In the proposed MASC, the best and the worst case leakage scenarios are the same since there is always

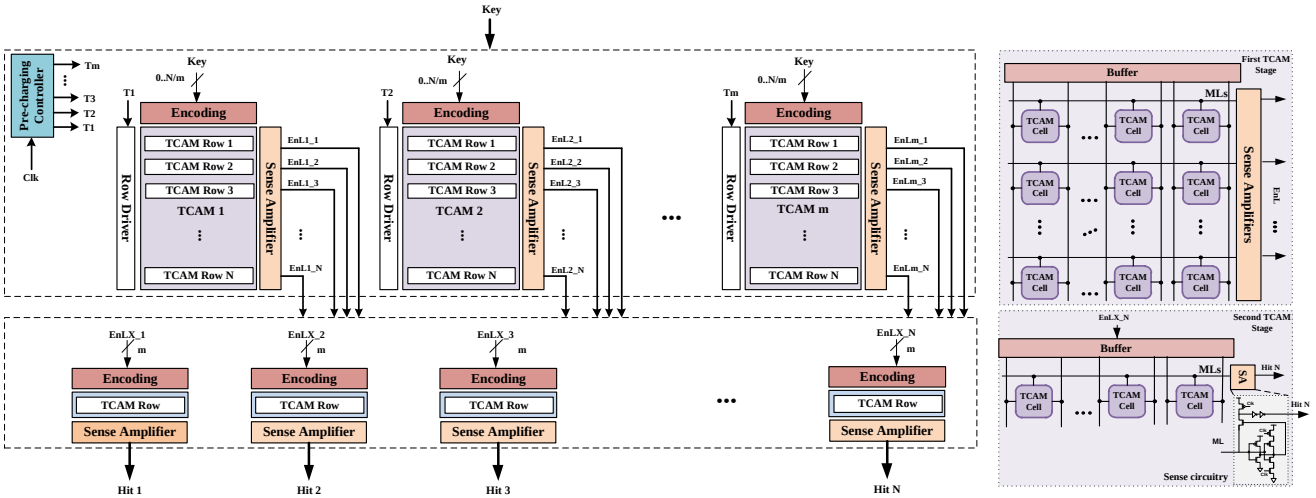


Fig. 5. Proposed multiple-access single-cycle TCAM architecture

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

only one leaky cell for a miss independent to the block size. The ML voltages and average TCAM energy consumption (per cycle) for 8-bit encoding TCAM block is shown in *Fig. 6*.

Considering process variations is essential to approximate mode, because TCAMs with voltage overscaling are sensitive to variations. For example, if we design a TCAM to work with 1-bit hamming distance (HD), variations on TCAM parameters can instead make the same TCAM have worse accuracy, such as giving back a 2-bit hamming distance result as if it is only 1-bit HD. Therefore, to ensure a more predictable design which can guarantee quality of service, we design our TCAM based on the worst-case scenario. To calculate this refresh cycle, we considered a 10% process variation on the resistance value, the size, and threshold voltage of access transistors [41]. Based on this variation, we calculate the maximum discharging current of the ML (low resistance, low threshold value and large transistor size) and then set the precharging cycle corresponding to 1-bit hamming distance. Such worst-case design guarantees that the ML will not accept more than desired mismatches under process variations.

We define a refresh period to be acceptable if it results in correct matches in 1000 Monte Carlo simulations. The 8-bit MASC can perform four consecutive search cycles without a complete precharge. With 2-bit and 4-bit MASC, the refresh periods are 7 and 5 cycles. The proposed technique reduces the search energy consumption of the TCAM significantly by reducing precharge requirement.

Performing more search operations than these refresh periods results in TCAM search error. *Fig. 7* shows the normalized ML voltage on TCAM with different precharging cycles at the last cycle of search. We set the search clock period as the maximum delay the last period. We consider having mismatch in every search cycle. Our circuit level simulation on 8-bit TCAM shows that in order to have an error-free search operation we need the ML voltage higher than 850mV (see *Fig. 7*). This means that for exact matching we can limit the period of precharging to 4 cycles. Measuring the average energy of a search operation after 4 cycles shows that the proposed design can achieve up to 3.2X lower search energy with respect to conventional TCAM design. Using longer precharging period makes the search operation unstable and increases the probability of error. Our evaluations also show that ML voltages of 775mV and 650mV correspond to one and two bits Hamming distance in a TCAM search. For MASC (crossbar MASC) 1-HD/2-HD are defined by 6/8-cycle (5/7-cycle) precharging cycle periods respectively.

Another advantage of the proposed TCAM is its ability to control approximation by implementing long precharging cycles on selective blocks. In the proposed architecture, all blocks do not have the same effect on the result of computation. Applying long precharge cycle on the least significant bits has lower impact on the results of computation compared to most significant bits. The MASC allows us to implement 1-HD and 2-HD approximation on selective TCAM blocks using multiple precharging cycles. A simple precharging controller shown in Fig. 5 sets the refresh time of each partial TCAM based on the running application and its quality of service requirements. We study the effect of approximation on the GPGPU energy savings and output quality in section 5.

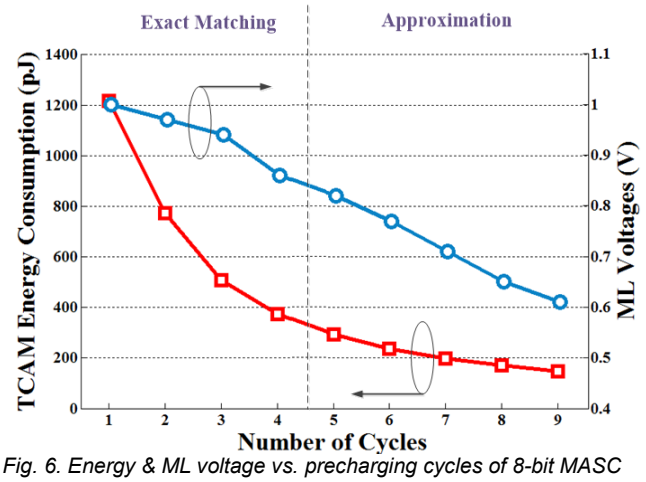


Fig. 6. Energy & ML voltage vs. precharging cycles of 8-bit MASC

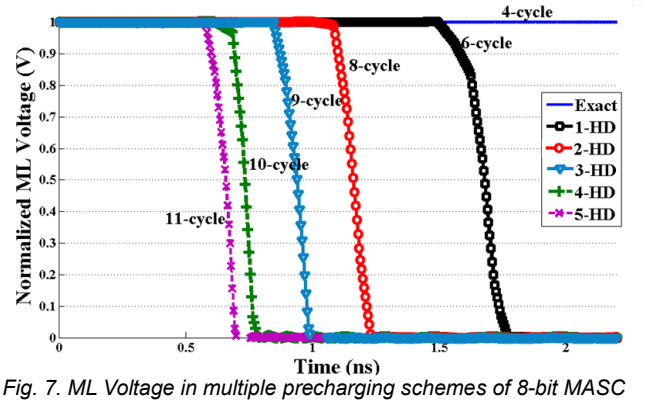


Fig. 7. ML Voltage in multiple precharging schemes of 8-bit MASC

4.5 MASC Framework

The execution flow of MASC has two main steps: design time profiling and runtime processing. During profiling we use OpenCL kernel and host code to train the associative memory. This allows us to find the high frequency patterns (HFP) to put in the MASC. In this work we used both vision and general OpenCL apps. Caltech 101 computer vision [42] dataset has testing and training data for our image processing applications. For general OpenCL applications we include Matrix Multiplication, Bionomial Option, QuasiRandom, DwHaar1D, with random streamed data. The training is done on 10% of the input dataset. The host code saves and ranks the input patterns for each FPU operation based on their frequency. The AMD compute abstraction layer (CAL) provides a runtime device driver library and allows the host program to work with the stream cores at the lowest level. The programming of the TCAMs is done in software by using host code (AM Updater block shown in *Fig. 8*). A number of MASC rows are initialized based on the energy efficiency and accuracy requirements.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

TCAM in all compute units of GPU is programmed concurrently with the same data based on the FPU type.

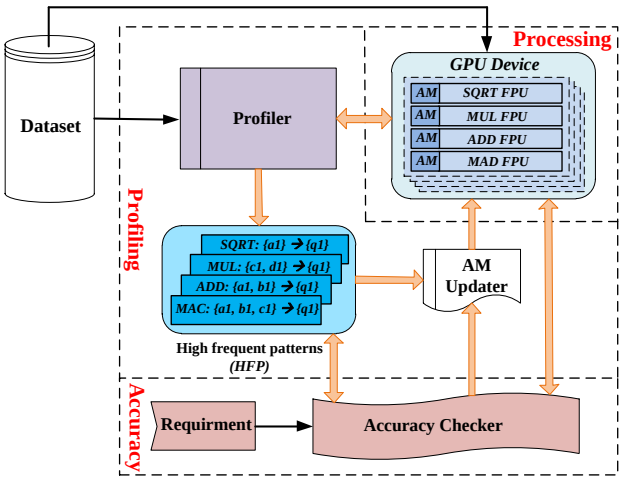


Fig. 8. Approximate GPU framework with MASC TCAM

To find the approximate configuration of MASC for each application, the accuracy checker block compares (see Fig. 8) the GPU output file using approximate and exact MASC. The framework continues putting more partial TCAM blocks in the approximate mode as long as the applications meet the quality requirements. In this work we define the requirement as 30dB PSNR for image processing applications [43] and 10% average relative error for other applications [44].

At run-time, MASC values are updated by preprofilied data as the application starts. The system checks the input patterns in FPUs and associative memory simultaneously. Any hit in TCAM stops the FPU computation and activates the corresponding row of resistive memory to retrieve precomputed values. Low energy consumption of proposed MASC allows us to use large size TCAM beside each processing unit to provide high hit rates.

Our proposed design does not sacrifice performance relative to the original GPU execution since the proposed associative memory is designed to perform the search operation with the same clock frequency of the FPUs. We minimize area cost by implementing crossbar MASC on top of the processing (FPUs). The only disadvantage of crossbar TCAM is its low scalability (<1024 rows). However, we show that this is not important in our case since the minimum GPGPU energy consumption occurs with TCAM that has less than 64 rows. MASC associative memory adds energy overheard to the computation which we evaluate in the next section.

5 Experimental Results

5.1 Experimental Setup

We evaluate the MASC-based associative memory on AMD Southern Island GPU Radeon HD 7970 device, a recent GPU architecture. We use eight OpenCL applications including four image processing: Sobel, Robert, Sharpen, Shift [45] and four general applications: Matrix Multiplication, Bionomial Option, QuasiRandom, DwHaar1D to show the efficiency of proposed MASC desing. We use Multi2sim, a cycle accurate CPU-GPU simulator [46] and modified the kernel code to do profiling and run-time simulation. The 6-stage balanced FPUs are designed using Synopsys Design Compiler [47] in 45-nm ASIC flow. The FPUs are optimized for power based on different TCAM delay.

We extracted the frequent patterns for adder (ADD), multipliers (MUL), SQRT, multiply-accumulator (MAC) FPUs. In GPGPU computation, the FPU operations have a variety number of inputs. The ADD and MUL accept two 32-bit, SQRT one 32-bit and MAC three 32-bit input operands. Thus, the corresponding TCAMs require 64-bit, 32-bit and 96-bit word sizes respectively. The circuit level simulation of TCAM design is done using HSPICE simulator on 45-nm technology. We use the memristive memory model from paper [31] to design the crossbar MASC architecture. The ON and OFF resistances are set to 50KΩ and 50GΩ respectively. The details of 2T-2R TCAM simulation (sizing, resistors and capacitors, etc.) are designed based on [15].

5.2 MASC Configurations

The MASC structure consists of two TCAM search stages. The second level TCAM is fast, area and energy efficient, such that it does not have a major impact on system efficiency. In the first stage of MASC, the search operation can be done in 2-bit, 4-bit or 8-bit encoding granularities. The size of encoding block shows a tradeoff between area and energy consumption; (i) MASC with small encoding blocks (e.g. MASC 2:16) improves the total area efficiency of the TCAM. In contrast, large blocks such as 4-bit and 8-bit encoding need more cells to design a TCAM. (ii) Large encoding blocks improve sense margin, since in the MASC structure the number of leaky cells from ML is always one independent of the block size. (iii) Additionally, MASC with large encoding blocks reduce the number of partial matches, further improving the energy efficiency. This is because hits in partial blocks increase the number of undesired charging and precharging.

We define *effective TCAM states (ETS)* as the ratio of number of TCAM rows to the number of available states at each MASC granularity, which indicates the probability of hit. For 32-row 2:16 MASC, *ETS* is $32/2^2 = 8$. This number decreases to 2 and 1/8 for 4-bit and 8-bit encoding blocks respectively. Lower *ETS* indicates that the system has lower probability of undesired hits. Our evaluation on four applications shows that MASC with *ETS* > 8 degrades search energy consumption to below that of the conventional TCAM. This limits the number of stages in 32-row and 64-row block, to 15-stages and 8-stages respectively. Thus, stability and energy efficiency of large encoding blocks come at the expense of area. However, the area overhead of resistive associative memories is negligible compared to the FPU area in GPU architecture.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

TABLE 1. CROSSBAR AND 2T-2R ENERGY (FJ) COMPARISON IN DIFFERENT MASC CONFIGURATIONS AND SIZES FOR *ADD/MUL (A/M)*, *SQRT (SQ)* AND *MAC (MC)* OPERATIONS.

MASC type & Configuration		2-row			4-row			8-row			16-row			32-row			64-row		
		SQ	*A/M*	*MC*	*SQ*	*A/M*	*MC*	*SQ*	*A/M*	*MC*	*SQ*	*A/M*	*MC*	*SQ*	*A/M*	*MC*	*SQ*	*A/M*	*MC*
2T-2R	***MASC 8:4***	40	55	66	61	88	106	99	139	162	99	206	233	240	304	340	404	507	558
	MASC 4:8	75	103	124	114	166	198	186	261	303	296	385	437	450	570	638	756	950	1045
	MASC 2:16	118	161	194	178	259	310	291	408	474	462	602	683	703	89	996	1181	1483	1632
Cross-bar	***MASC 8:4***	28	39	47	42	61	74	67	94	110	66	138	156	149	188	211	239	300	330
	MASC 4:8	54	75	90	80	117	140	129	181	210	206	269	305	290	367	411	432	542	597
	MASC 2:16	89	121	146	128	187	224	197	277	322	298	388	424	436	55	618	694	872	960

Further, if the associative memory area becomes an important design parameter, the MASC can be designed with smaller blocks. For energy efficient associative memory, employing MASC 8:4 results in the best energy and sense margin.

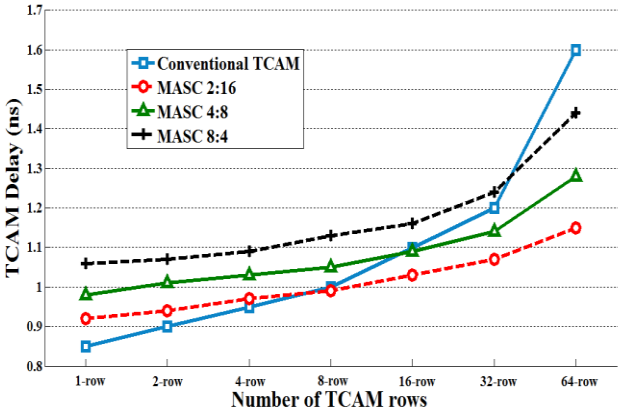


Fig. 9. MASC and conventional 2T-2R TCAM delay in different TCAM size and different MASC configuration.

Fig. 9 shows the delay of a 96-bit TCAM at different sizes. In the proposed design, the search operation is performed using several parallel partial searches. At small sizes, the conventional TCAM has lower delay than MASC because of its single-stage search operation. However, at large sizes, the search operation of the conventional TCAM slows down due to the precharging delay. A highly partitioned MASC uses short and parallel precharging cycles, which offsets the overhead of multiple stages. This results in lower delay especially in highly partitioned TCAMs at large sizes.

The energy comparison shows that crossbar MASC consumes much lower energy compared to 2T-2R MASC for each search operation. Table 1 compares the energy consumption of crossbar and 2T-2T MASC in different sizes for ADD/MUL, SQRT, and MAC operations. This energy difference of each operation is the result of having TCAMs with different word sizes. Since TCAM for MAC operation uses 96-bit to search and prestore three input operands, the corresponding TCAM consumes more energy than other operations in each search. Similarly, the TCAM corresponding to ADD/MUL consumes more energy than SQRT since its word size is 2X wider. The energy difference of crossbar and 2T-2R MASC becomes more obvious with large TCAM sizes such that the crossbar consumes ~1.7X lower search energy than 2T-2R MASC. Low energy consumption of crossbar MASC along with its capability to be placed at the top of the chip thus having low area overhead, make crossbar MASC an efficient associative memory for memory-based computation.

5.3 MASC Size and Energy Efficiency

To integrate the proposed design as an associative memory, we use 1T-1R memory structure along with MASC to pre-store the computational results for various operands. The normalized energy consumption of the GPGPU at different sizes is shown in Figure 6. The FPU energy is calculated based on the measured TCAM delay at each size. There are two factors that affect the total GPGPU energy consumption:

- **FPU energy:** In large TCAMs, higher hit-rate increases the percentage of the time that the processor is in clock-gate mode. Small TCAMs clearly benefit from doubling size where new rows can pre-store a pattern with a high percentage of hit. But large TCAM already covers most of the frequent patterns so that adding new lines does not have a major impact on hit-rate improvement and energy efficiency. In addition, the delay of large TCAM allows the design compiler to optimize the FPU energy consumption.
- **TCAM energy:** High search energy consumption of a large TCAM limits the energy efficiency of the GPGPU processing. Therefore, in conventional associative memory, increasing the size to larger than 8-rows does not improve the hit-rate (and hence the FPU energy) enough to compensate for the large energy. Considering these factors, the minimum energy point of GPGPU using conventional associative memory occurs at 8-row (See *Fig. 10* and Fig. *11*).

GPGPU using MASC associative memory achieves better energy efficiency at all TCAM sizes. The low energy consumption of MASC TCAM shifts the minimum energy point of the GPGPU to larger TCAM sizes with 16 and 32 rows. As Fig. *10* and Fig. *11* show, at these sizes, both 2T-2R and crossbar MASC have better or comparable energy savings compared to conventional TCAM. Our results show that GPGPU with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

2T-2R MASC 8:4, MASC 4:8 and MASC 2:16 can achieve 33.4%, 29.9% and 23.2% energy savings on average running eight different applications. Using crossbar MASC architecture improves the energy savings to 37.7%, 35.4% and 30.1% for MASC 8:4, MASC 4:8 and MASC 2:16 respectively. In MASC, using small TCAMs or TCAMs with large encoding blocks (small *ETS*) decrease the number of undesired hit of partial TCAMs. Therefore, the energy efficiency of MASC 8:4 is the result of better MASC partial hit controllability as compared to MASC 2:16 and MASC 4:8.

5.4 MASC Approximation

Approximation in MASC is defined by the period of the precharging. With a period of 4-cycles, an 8-bit MASC TCAM performs error-free searches. Increasing the refresh period of 8-bit TCAM to 6-cycle and 8-cycle creates one and two bits Hamming distance respectively. Our framework implements approximation in TCAM starting from the lower level TCAM blocks, since the mismatch on these blocks has lower effect on the computation result compared to error on most significant bits. Table 2 lists the maximum number of MASC blocks in 1-HD and 2-HD approximation, and hit-rate improvement compared to exact matching for different applications such that the output PSNR does not drop below 30dB. The system is able to apply the approximation on *m* lower blocks of each MASC TCAM based on the running applications. The small controller block in *Fig. 5* sends appropriate signals to row driver of each TCAM block to set the MLs refresh periods based on the running applications.

There is a trade-off between the quality of service and GPGPU energy consumption. As Table 2 shows, approximations of TCAM increases TCAM hit-rate based on the number and size of blocks in approximate mode, and the depth of approximation (1-HD or 2-HD). Implementing refresh time relaxation on a large number of bitlines improves energy efficiency by increasing the system hit-rate and reducing search energy, however this improvement is achieved at the expense of quality of service. In 2T-2R MASC 8:4, using long refresh periods on one block relaxes 25% of the entire bitline, while approximating a MASC 2:16 block relaxes 6.2% of the entire bitline. This wide range of relaxation significantly improves the system hit-rate and GPGPU energy saving. In addition, the system energy savings increase with TCAM sizes because large block with approximation benefit more from hit-rate improvement compared to small sizes.

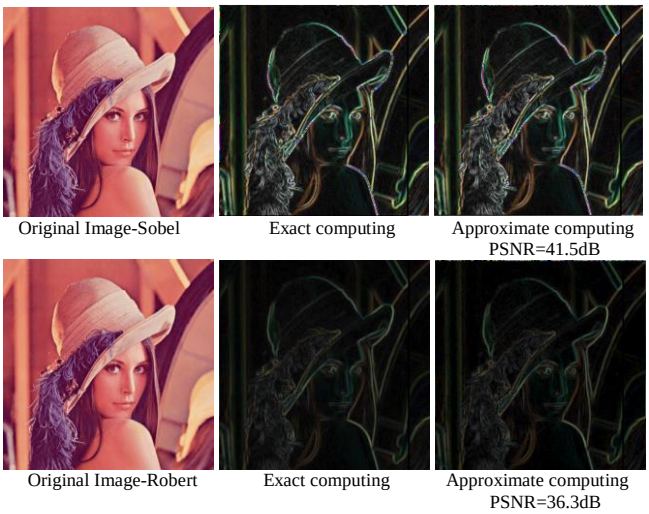


Fig. 12. Output quality for Sobel and Robert applications

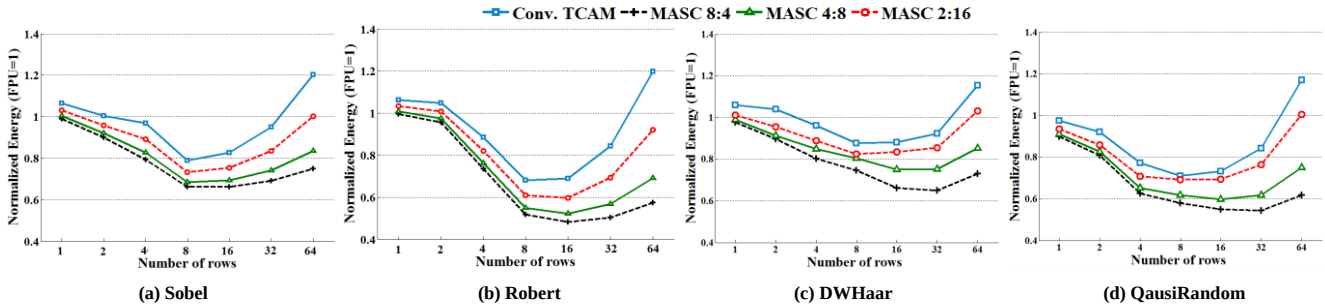


Fig. 10. Normalized GPGPU energy consumption for different 2T-2R MASC sizes.

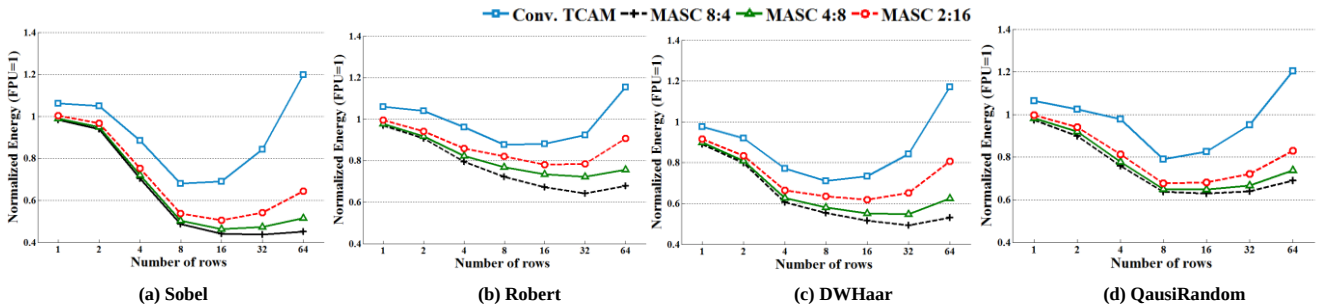


Fig. 11. Normalized GPGPU energy consumption for different crossbar MASC sizes.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2016.2565262, IEEE Transactions on Emerging Topics in Computing

TABLE 2. APPROXIMATION ON SELECTIVE 2T-2R MASC BLOCKS

Application Type		*Sobel*			*Robert*			*Sharpen*			*Shift*		
TCAM Configuration		*2:16*	*4:8*	*8:4*	*2:16*	*4:8*	*8:4*	*2:16*	*4:8*	*8:4*	*2:16*	*4:8*	*8:4*
1-HD	*# of blocks*	3 (18.7%)	2 (25%)	1 (25%)	3 (18.7%)	2 (25%)	2 (50%)	3 (18.7%)	3 (37.5%)	2 (50%)	2 (18.7%)	2 (25%)	1 (25%)
	PSNR	31.0dB	32.4dB	41.5dB	33.4dB	34.7dB	36.4dB	35.5dB	32.8dB	36.3dB	34.9dB	32.3dB	40.3dB
	Hit-rate improvement	9.6%	8.3%	5.1%	7.7%	9.3%	13.5%	7.8%	10.5%	12.2%	6.8%	7.6%	4.8%
2-HD	*# of blocks*	2 (12.5%)	1 (12.5%)	1 (25%)	2 (12.5%)	1 (12.5%)	1 (25%)	2 (12.5%)	2 (25%)	1 (25%)	1 (6.2%)	1 (12.5%)	0 (0%)
	PSNR	30.3dB	34.6dB	32.2dB	31.2dB	42.6dB	39.1dB	32.4dB	30.4dB	39.5dB	37.5dB	35.2dB	Origi-nal
	Hit-rate improvement	3.2%	4.5%	8.2%	7.9%	6.4%	9.1%	6.8%	8.1%	9.3%	2.8%	4.4%	0%

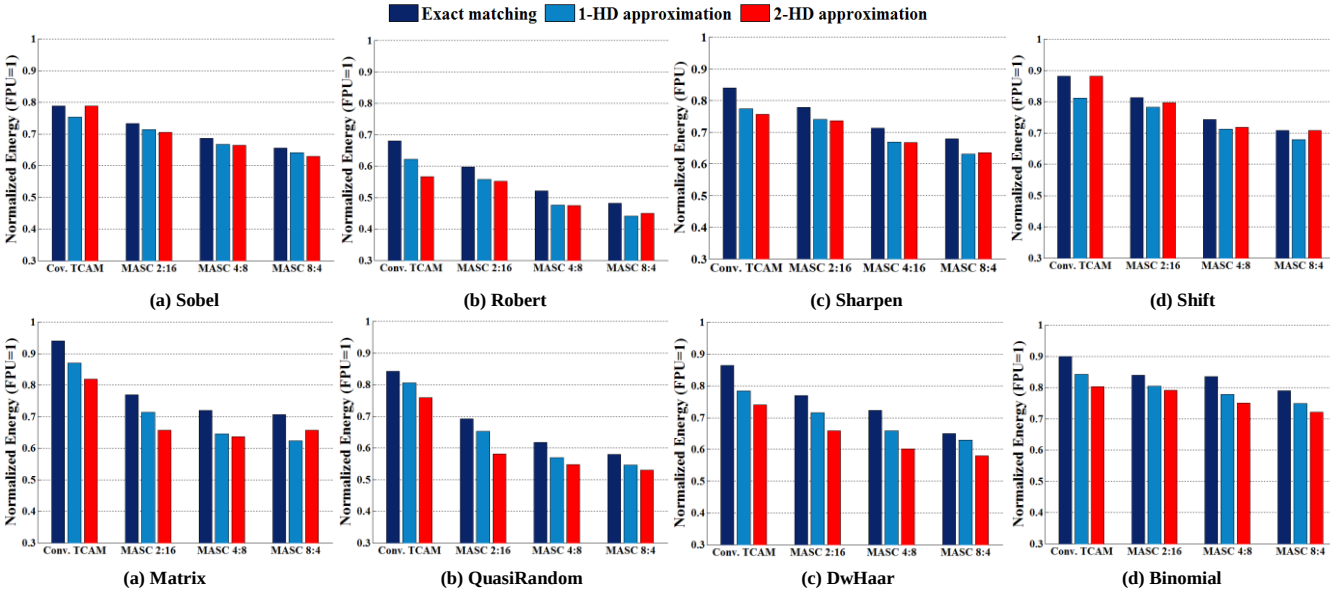


Fig. 13. Normalized Energy consumption of GPGPU in 1-HD and 2-HD approximation of in 2T-2T MASC

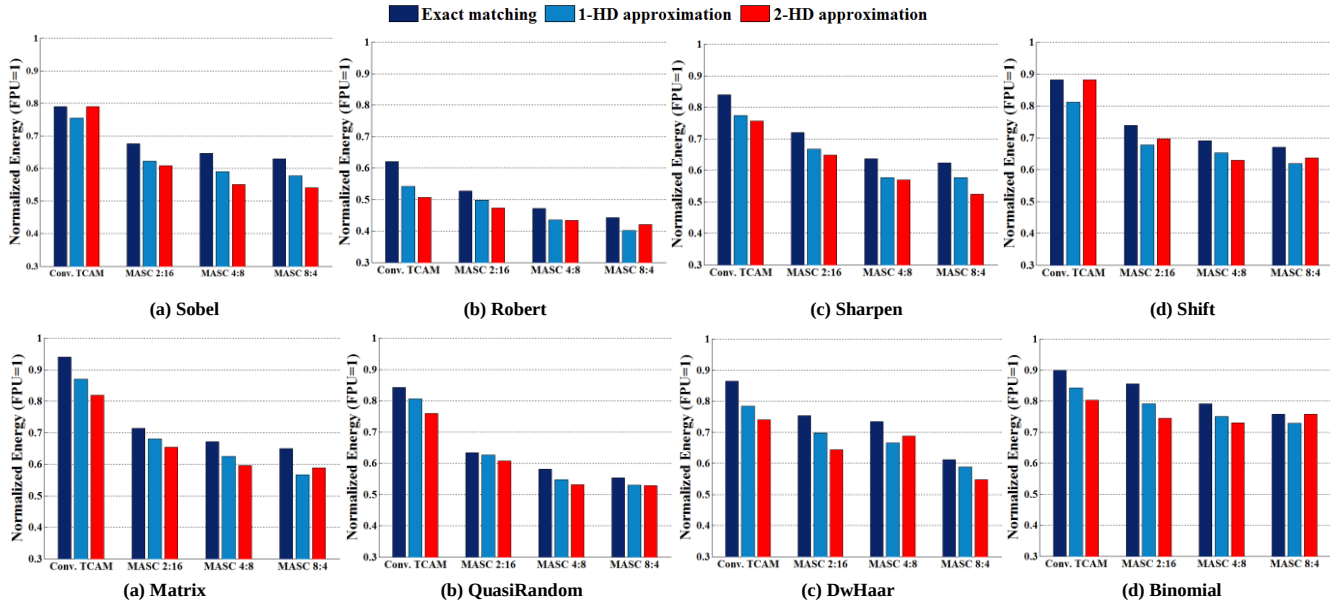


Figure 14. Normalized Energy consumption of GPGPU in 1-HD and 2-HD approximation of crossbar MASC

MASC can provide enough accuracy by selecting how many blocks are leveraging approximation. For example, Fig. *12* shows the visual results of *Sobel*, *Robert* and *Sharpen* application using the original computation (i.e., the golden image case) and approximate computation, resulting in no perceivable change.

FPUs consume most of the energy in GPGPU – in our case 89%. Fig. 13 and Fig. 14 show the energy improvement of the FPUs in the GPGPU architecture using 2T-2R and crossbar MASC with exact matching, 1-HD and 2-HD implementations. Our results show that the total computational energy of the FPUs using 2T-2R MASC 8:4, MASC 4:8 and MASC 2:16 decreases by 38.1%, 35.1% and 29.2% (36.7%, 36.1% and 31.4%) on average for 1-HD (2-HD) implementation respectively, ensuring acceptable quality of service (PSNR>30dB for image processing and average relative error <10% for other applications). This indicates that FPUs using proposed 2T-2R MASC achieves 1.8X higher energy savings for exact matching with respect to conventional TCAM. For crossbar MASC these energy savings are 42.6%, 39.4% and 34.2% (43.1%, 40.8% and 36.5%) on average for MASC 8:4, MASC 4:8 and MASC 2:16 in 1-HD (2-HD) approximation. In summary, our evaluation shows that using 2T-2R and crossbar MASC can improve FPUs computational energy by 1.77X and 1.93X compared to applying conventional voltage overscaling on TCAM.

Table 3 shows the impact of MASC on the overall GPGPU computation energy considering both floating point and integer units. The result shows that for all tested applications, the FPUs are the main source of GPGPU energy consumption, where they consume about 89% of overall energy. Therefore, for MAC 8:4 in 2-HD approximation, the 2T-2R and crossbar can improve overall GPGPU computation energy by 35% and 39% respectively, while providing good quality of service.

TABLE 3. OVERAL GPGPU COMPUTATION ENERGY SAVINGS USING 2T-2R AND CROSSBAR MASC

	Sobel	Robert	Sharpen	Shift	Matrix	Quasi	Dwh	Binom
2T-2R	33%	49%	33%	25%	34%	43%	39%	23%
Crossbr	40%	51%	43%	31%	39%	43%	42%	22%

6 CONCLUSION

In this paper we propose an ultra-low energy multiple-access single-charge TCAM which significantly decreases the energy consumption of the associative memory. A conventional TCAM discharges all missed rows when performing a search operation, consuming a large amount of energy. We propose the MASC TCAM design, which discharges only the hit row(s) while miss rows stay charged. This allows us to use the charge of the missed lines to perform multiple search operations. We explore the efficiency of the design on 2R-2R and crossbar MASC TCAMS. Our evaluation shows that the proposed 8-bit TCAM can achieve error-free search operations using a period of 4-cycles for precharging. Increasing the period of precharging improves the TCAM search energy efficiency at the expense of accuracy of the matching patterns. We also showed that crossbar MASC not only improves the search energy efficiency, but also addresses the area issue of MASC by implementing it at the top of the chip. The proposed approximate MASC (crossbar MASC) decreases the energy consumption of the overall GPGPU by 35% (39%) on average with acceptable quality of service. These savings of using MASC (crossbar MASC) are 1.77X (1.93X) higher than approximation using conventional voltage overscaling technique.

7 ACKNOWLEDGMENT

This work was sponsored by NSF grant #1527034 and UC San Diego Jacobs School Powell Fellowship.

8 REFERENCES

[1] A. Katal, M. Wazid, and R. Goudar, "Big data: Issues, challenges, tools and Good practices," in *Contemporary Computing (IC3), International Conference on*, pp. 404-409, 2013.

[2] C. Ji, Y. Li, W. Qiu, U. Awada, and K. Li, "Big data processing in cloud computing environments," in *Pervasive Systems, Algorithms and Networks (ISPAN), International Symposium on*, pp. 17-23, 2012.

[3] T. Kohonen, "Associative memory: A system-theoretical approach" *Springer Science & Business Media*, vol. 17, 2012.

[4] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," *Solid-State Circuits, IEEE Journal of*, vol. 41, pp. 712-727, 2006.

[5] L. Chisvin and R. J. Duckworth, "Content-addressable and associative memory: Alternatives to the ubiquitous RAM," *Computer*, IEEE, pp. 51-64, 1989.

[6] A. Rahimi, A. Ghofrani, K.-T. Cheng, L. Benini, and R. K. Gupta, "Approximate associative memristive memory for energy-efficient GPUs," in *Proceedings of Design, Automation & Test in Europe Conference (DATE*), pp. 1497-1502, 2015.

[7] M. Imani, S. Patil, and T. Rosing, "MASC: Ultra-Low Energy Multiple-Access Single-Charge TCAM for Approximate Computing," in *Proceedings of Design, Automation & Test in Europe Conference (DATE*), 2016.

[8] M. Breuer, "Multi-media applications and imprecise computation," in *Proceedings Conference on of Digital System Design*, pp. 2-7, 2005.

[9] V. Akhlaghi, A. Rahimi, and R. Gupta, "Resistive Bloom Filters: From Approximate Membership to Approximate Computing with Bounded Errors." in *Proceedings of Design, Automation & Test in Europe Conference (DATE*), 2016.

[10] A. Goel and P. Gupta, "Small subset queries and bloom filters using ternary associative memories, with applications," in *ACM SIGMETRICS Performance Evaluation Review*, pp. 143-154, 2010.

[11] K. Lakshminarayanan, A. Rangarajan, and S. Venkatachary, "Algorithms for advanced packet classification with ternary CAMs," in *ACM SIGCOMM Computer Communication Review*, pp. 193-204, 2005.

[12] S. Kaxiras and G. Keramidas, "IPStash: a set-associative memory approach for efficient IP-lookup," in *IEEE Computer and Communications Societies (INFOCOM)*, pp. 992-1001, 2005.

[13] H. Zhang, M. Putic, and J. Lach, "Low power gpgpu computation with imprecise hardware," in *Design Automation Conference (DAC)*, ACM/EDAC/IEEE, pp. 1-6, 2014.

[14] M. Imani, P. Mercati, T. Rosing, "ReMAM: Low Energy Resistive Multi-Stage Associative Memory for Energy Efficient Computing," in *International Symposium on Quality Electronic Design (ISQED)*, 2016.

[15] J. Li, R. K. Montoye, M. Ishii, and L.-Y. Chang, "1 Mb 0.41 μm² 2T-2R Cell Nonvolatile TCAM With Two-Bit Encoding and Clocked Self-Referenced Sensing," *IEEE Journal of Solid-State Circuits*, vol. 49, pp. 896-907, 2014.

[16] S. Paul, S. Chatterjee, S. Mukhopadhyay, and S. Bhunia, "Nanoscale reconfigurable computing using non-volatile 2-d sttram array," in *IEEE Conference on Nanotechnology*, pp. 880-883, 2009.

[17] J. Cong, M. Ercegovac, M. Huang, S. Li, and B. Xiao, "Energy-efficient computing using adaptive table lookup based on nonvolatile memories," in *IEEE International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 280-285, 2013.

[18] M. Kund, G. Beitel, C.-U. Pinnow, T. Rohr, J. Schumann, R. Symanczyk, et al., "Conductive bridging RAM (CBRAM): An emerging non-volatile memory technology scalable to sub 20nm," in *IEEE International Electron Devices Meeting*, 2005.

[19] Y. Kim, M. Imani, S. Patil, and T. S. Rosing, "CAUSE: Critical Application Usage-Aware Memory System using Non-volatile Memory for Mobile Devices," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 690-696, 2015.

[20] Q. Guo, X. Guo, R. Patel, E. Ipek, and E. G. Friedman, "AC-DIMM: associative computing with STT-MRAM," in *ACM SIGARCH Computer Architecture News*, pp. 189-200, 2013.

[21] T. Hanyu, D. Suzuki, N. Onizawa, S. Matsunaga, M. Natsui, and A. Mochizuki, "Spintronics-based nonvolatile logic-in-memory architecture towards an ultra-low-power and highly reliable VLSI computing paradigm," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE)*, pp. 1006-1011, 2015.

[22] M.-F. Chang, C.-C. Lin, A. Lee, C.-C. Kuo, G.-H. Yang, H.-J. Tsai, et al., "17.5 A 3T1R nonvolatile TCAM using MLC ReRAM with Sub-1ns search time," in *IEEE International Solid-stage circuits conference (ISSCC)*, 2015.

[24] A. Rahimi, L. Benini, and R. Gupta, "CIRCA-GPUs: Increasing Instruction Reuse through Inexact Computing in GP-GPUs," *IEEE Design & Test*, 2015.

[25] M. Imani, A. Rahimi, and T. Rosing, "Resistive Configurable Associative Memory for Approximate Computing." in *Proceedings of Design, Automation & Test in Europe Conference (DATE*), 2016.

[26] M. Imani, Y. Kim, A. Rahimi, and T. Rosing, " ACAM: Approximate Computing Based on Adaptive Associative Memory with Online Learning," *International Symposium on Low Power Electronics and Design (ISLPED)*, 2016.

[27] R. Waser and M. Aono, "Nanoionics-based resistive switching memories," *Nature materials*, vol. 6, pp. 833-840, 2007.

[28] Y. Xie, "Emerging Memory Technologies," ed: *Springer*, 2014.

[29] J. J. Yang, M. D. Pickett, X. Li, D. A. Ohlberg, D. R. Stewart, and R. S. Williams, "Memristive switching mechanism for metal/oxide/metal nanodevices," *Nature nanotechnology*, vol. 3, pp. 429-433, 2008.

[30] Y. Chen, H. Lee, P. Chen, P. Gu, C. Chen, W. Lin, et al., "Highly scalable hafnium oxide memory with improvements of resistive distribution and read disturb immunity," in *IEEE International Electron Devices Meeting (IEDM)*, pp. 1-4, 2009.

[31] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, et al., "A functional hybrid memristor crossbar-array/CMOS system for data storage and neuromorphic applications," *Nano letters*, vol. 12, pp. 389-395, 2011.

[32] Y. Yang, P. Sheridan, and W. Lu, "Complementary resistive switching in tantalum oxide-based resistive memory devices," *Applied Physics Letters*, vol. 100, 2012.

[33] E. Linn, R. Rosezin, C. Kügeler, and R. Waser, "Complementary resistive switches for passive nanocrossbar memories," *Nature materials*, vol. 9, pp. 403-406, 2010.

[34] N. Bandi, A. Metwally, D. Agrawal, and A. El Abbadi, "Fast data stream algorithms using associative memories," *in ACM international conference on Management of data (SIGMOD)*, pp. 247-256, 2007.

[35] K. Eshraghian, K.-R. Cho, O. Kavehei, S.-K. Kang, D. Abbott, and S.-M. S. Kang, "Memristor MOS content addressable memory (MCAM): Hybrid architecture for future high performance search engines," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, , vol. 19, pp. 1407-1417, 2011.

[36] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis, "Evaluating mapreduce for multi-core and multiprocessor systems," in *IEEE Symposium on High Performance Computer Architecture (HPCA)*, pp. 13-24, 2007.

[37] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *International conference on very large data bases(VLDB)*, pp. 487-499, 1994.

[38] T. Kohonen, "Content-addressable memories", *Springer Science & Business Media*, vol. 1, 2012.

[39] S. Panchanathan and M. Goldberg, "A content-addressable memory architecture for image coding using vector quantization," *IEEE Transactions on Signal Processing*, vol. 39, pp. 2066-2078, 1991.

[40] D. Mohapatra, V. K. Chippa, A. Raghunathan, and K. Roy, "Design of voltage-scalable meta-functions for approximate computing," in *Design, Automation & Test in Europe Conference (DATE)*, pp. 1-6, 2011.

[41] M.-F. Chang, A. Lee, P.-C. Chen, C. J. Lin, Y.-C. King, S.-S. Sheu, et al., "Challenges and Circuit Techniques for Energy-Efficient On-Chip Nonvolatile Memory Using Memristive Devices," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 5, pp. 183-193, 2015.

[42] "AMD APP SDK v2.5, Available: http://www.amd.com/stream."

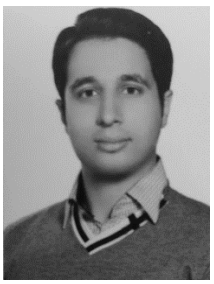
[43] R. Ubal, B. Jang, P. Mistry, D. Schaa, and D. Kaeli, "Multi2Sim: a simulation framework for CPU-GPU computing," in *International conference on Parallel architectures and compilation techniques (PACT)*, pp. 335-344, 2012.

[44] D. Compiler, "Synopsys Inc," ed, 2000.

[45] Caltech 101 Availble at: "http://www.vision.caltech.edu/Image_Datasets/Caltech101/".

[46] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electronics letters*, vol. 44, pp. 800-801, 2008.

[47] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *IEEE/ACM International Symposium on Microarchitecture*, pp. 449-460, 2012.



Mohsen Imani received his M.S. and BCs degrees from the School of Electrical and Computer Engineering at the University of Tehran in March 2014 and September 2011 respectively. From September 2014, he is a Ph.D. student in the Department of Computer Science and Engineering at the University of California San Diego, CA, USA. He is a member of the System Energy Efficient Laboratory (SeeLab), where he is searching for alternative computer architecture to address memory bottleneck and computation cost. Mr. Imani is a Powell Fellow student at UC San Diego. His research interests include approximate computing, neuromorphic computing and memory centric computing.

Shruti Patil received the Ph.D. degree from University of Minnesota Twin Cities in 2011. Her dissertation focused on computing circuits and systems with emerging technologies such as Spintronics and NEMS. She joined Intel Labs as a Research Scientist in 2012 and worked on microarchitectural simulations and performance modeling. She later joined Princeton University as a postdoctoral researcher to work on quantum architectures and compilers. She has also briefly explored emerging technologies for mobile platforms at University of California San Diego as a postdoc. Dr. Patil is currently working as a Software Engineer at Google. Her research interests include unconventional architectures and quantum computing.

Tajana Simunic Rosing is a Professor, a holder of the Fratamico Endowed Chair, and a director of System Energy Efficiency Lab at UCSD. She is currently heading the effort in SmartCities as a part of DARPA and industry funded TerraSwarm center. During 2009-2012 she led the energy efficient datacenters theme as a part of the MuSyC center. Her research interests are energy efficient computing, embedded and distributed systems. Prior to this she was a full time researcher at HP Labs while being leading research part-time at Stanford University. She finished her PhD in 2001 at Stanford University, concurrently with finishing her Masters in Engineering Management. Her PhD topic was Dynamic Management of Power Consumption. Prior to pursuing the PhD, she worked as a Senior Design Engineer at Altera Corporation.