GIRAF: General purpose In-storage Resistive Associative Framework

Leonid Yavits, Roman Kaplan and Ran Ginosar
Faculty of Electrical Engineering
Technion, Israel Institute of Technology
leonid.yavits@nububbles.com, romankap@gmail.com, ran@ee.technion.ac.il

I. INTRODUCTION

The premise of data centric, or near-data processing is reducing the memory access time by cutting the physical distance and increasing the bandwidth between processing units and memory. Since its inception, data centric processing mainly meant processing-in-memory (PIM). To process datasets larger than memory footprint, processing moves further down the memory hierarchy, achieving processing-in-storage.

In this paper, we propose a new General purpose In-data Resistive Associative Framework (GIRAF). We present the GIRAF architecture and its processing paradigm. GIRAF simultaneously functions as a data storage and a massively parallel fine-grain associative SIMD processor. GIRAF is based on Resistive Content Addressable Memory (RCAM). The processing is performed inside the storage arrays. There is no data transfer outside the storage arrays through a bandwidth limited interface to either a host CPU or a dedicated near-data processing unit. In GIRAF, every memory bit is directly connected to processing transistors, enabling ultra-high peak bandwidth and computation throughput while reducing the energy consumption, mainly attributed to reduction in data transfer.

II. GIRAF ARCHITECTURE

GIRAF employs resistive memory, organized in RCAM modules. Resistive memory stores information by modulating the resistance of nanoscale storage elements (memristors). Memristors are two-terminal devices, where the resistance of the device is changed by the electrical current or voltage. The resistance of the memristor is bounded by a minimum resistance R_{ON} (low resistive state, logic '1') and a maximum resistance R_{OFF} (high resistive state, logic '0').

RCAM is a scalable and highly dense alternative to CMOS CAM. Our GIRAF architecture uses a resistive crossbar and additional peripheral circuitry (Figure 1) to support associative storage and processing. RCAM module, presented in Figure 1, is the heart of GIRAF architecture. It comprises a RRAM crossbar, in which each memory line is also a baseline processing unit (PU), and peripheral circuitry. The latter includes key and mask registers, TAG logic, and optional daisy-chain interconnect. The basic RCAM cell is created by virtually pairing two RRAM cells (memristors), holding complementary values R and $\bar{R}$.

The *key* register (Figure 1a) contains a key data word to be written or compared against. The *mask* register defines the active fields for write, compare and read operations, enabling bit selectivity. The *TAG* marks the rows that are matched by the compare operation and are to be affected by the successive parallel write. A daisy-chain like bitwise interconnect allows PUs to intercommunicate, all PUs in parallel.

RCAM compare operation is performed as follows. The Match/Word line is precharged and the key is set on Bit (key-not on Bit-not) lines. In the columns that are ignored during comparison, the Bit and Bit-not lines are asserted '1'. If all unmasked bits in a row match the key (*i.e.*, when Bit line '1' is applied to an R_{ON} memristor and Bit-not line '0' is applied to an R_{OFF} memristor, or vice versa), the Match/Word line remains high, and '1' is sampled into the corresponding tag bit. If at least one bit mismatches, the Match/Word line discharges through an R_{ON} memristor and '0' is sampled into the tag.

Write operation is performed in two phases. First, the $V \geq V_{ON}$ voltage (where V_{ON} is a threshold voltage required to switch to the "on" state) is asserted to applicable Bit lines (to write '1's) and Bit-not lines (to write '0's). Second, the $V \leq V_{OFF}$ voltage (where V_{OFF} is a threshold voltage to switch to the "off" state) is asserted to Bit-not lines (to complement the '1's) and Bit lines (to complement '0's). The write affects only the tagged rows.

III. ASSOCIATIVE PROCESSING

Associative processor (AP) is a non-von-Neumann in-memory computer [2][3]. AP is based on CAM, which allows comparing the entire dataset to a search pattern (*key*), tagging

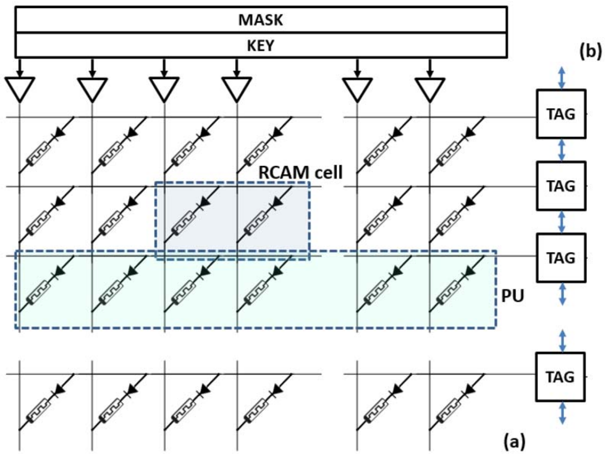

Figure 1: RCAM Module: (a) Resistive Crossbar and (b) Peripheral Circuitry.

the matching row, and writing another pattern to all tagged rows. AP performs no computations in conventional sense. Most arithmetic and logic operations can be structured as series of Boolean functions, which are implemented by the AP as follows.

The dataset is stored in CAM, typically one data element per CAM row (constituting a PU). AP controller sequentially matches all possible *input* combinations of a function arguments against the entire CAM content. The matching CAM rows are tagged, and the corresponding function values (precalculated and embedded in AP microcode), are written into the designated *output* fields of the tagged rows.

For an m-bit argument x ($x \in$ dataset), any Boolean function $b(x)$ has at most 2^m possible values ("at most" because of "don't cares"). Therefore, a brute-force approach would incur up to $O(2^m)$ cycles, regardless of the dataset size.

More efficiently, arithmetic operations can be performed in GIRAF in a word-parallel, bit-serial manner, reducing time complexity from $O(2^m)$ to $O(m)$. For instance, vector addition may be performed as follows [3]. Suppose that two m-bit RCAM columns hold vectors A and B; the sum S=A+B is written onto another m-bit column S. A one-bit column C holds the carry bit. The operation is carried out as m single-bit additions:

$$c[:] \mid s[:]_i = a[:]_i + b[:]_i + c[:] , \qquad i = 0, \ldots, m-1$$

where i is the bit index, ':' means all elements of the vector, and c and s are, respectively, the carry and sum bits. The single-bit addition is carried out in a series of steps. In each step, one entry of the full adder truth table is matched against the contents of the $a[:]_i, b[:]_i, c[:]$ bit columns and the matching rows (PUs) are tagged; the logic result (two-bit output of the truth table is written into the $c[:]$ and $s_i[:]$ bits of all tagged rows. During that operation, all but three input bit columns and two output bit columns of RCAM are masked out in each step. Overall, eight steps of one compare and one write operation are performed to complete a single-bit addition over all RCAM rows (i.e. over all vector elements), regardless of the vectors A and B lengths.

A fixed-point m bit addition and subtraction take $O(m)$ cycles. Fixed point multiplication and division in GIRAF require $O(m^2)$ cycles. Single precision floating point multiplication takes 4,400 cycles [4], regardless of the dataset size.

IV. EVALUATION

GIRAF is a processing-in-storage architecture, capable of internally maintaining the entire dataset. The alternative is a computer architecture (either data centric or CPU centric) where the dataset does not fit in internal memory, therefore requiring an external storage. Such external storage could either be a SSD, a NVDIMM based storage or a dedicated storage appliance. The bandwidth of such external storage is typically limited (10GB/s for a storage appliance to 24GB/s for a NVDIMM storage). The performance of such architecture is defined by the roofline model [1] as follows:

$$Attainable\ Perf = \min(Peak\ Perf,\ AI \times Peak\ Storage\ BW)$$

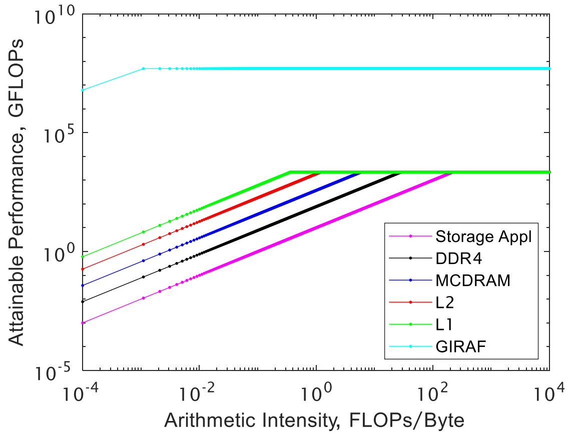

Figure 2: Roofline model based on [1], amended by BW chart of an external storage appliance and a model for 4TB GIRAF.

where $Peak\ Perf$ is the peak theoretical performance of the computer architecture, AI is arithmetic (or operational) intensity of a workload [5], and $Peak\ Storage\ BW$ is the peak external storage bandwidth (as demonstrated in Figure 2). In data intensive applications, characterized by low AI, the attainable performance of an architecture is typically limited by its peak storage bandwidth. We present GIRAF performance figure relative to the attainable performance of said baseline architecture, assuming more aggressive NVDIMM storage scenario.

The peak potential of GIRAF (with 4TB of storage in this example) is illustrated using the roofline model in Figure 2. It shows the roofline model of GIRAF against the backdrop of Intel's KNL [1], to which we add an external storage appliance access. Since GIRAF requires no external storage access, its attainable performance is defined by its peak internal bandwidth and its peak performance.

The peak internal bandwidth is achieved for instance during a transfer of an entire bit column to the TAG register. Another example is the broadcast of a single data item to the entire storage. The peak performance is calculated using a single precision floating point multiply-accumulate operation, performed in parallel on the entire dataset (assuming the dataset matches the GIRAF size, i.e. 1T 32bit data elements).

REFERENCES

[1] Doerfler, Douglas, et al. "Applying the roofline performance model to the intel xeon phi knights landing processor." International Conference on High Performance Computing. Springer International Publishing, 2016.

[2] Yavits, L., Kvatinsky, S., Morad, A., & Ginosar, R. (2015). Resistive associative processor. IEEE Computer Architecture Letters, 14(2), 148-151.

[3] Yavits, Leonid, Amir Morad, and Ran Ginosar. "Computer architecture with associative processor replacing last-level cache and SIMD accelerator." IEEE Transactions on Computers 64.2 (2015): 368-381.

[4] Yavits, Leonid, Amir Morad, and Ran Ginosar. "Sparse matrix multiplication on an associative processor." IEEE Transactions on Parallel and Distributed Systems 26.11 (2015): 3175-3183.

[5] Yavits L, Morad, A., & Ginosar, R. "The effect of communication and synchronization on Amdahl's law in multicore systems". Parallel Computing, vol. 40, no. 1, pp. 1-16, 2014.