A Nonvolatile Associative Memory-Based Context-Driven Search Engine Using 90 nm CMOS/MTJ-Hybrid Logic-in-Memory Architecture

Hooman Jarollahi, *Student Member, IEEE*, Naoya Onizawa, *Member, IEEE*, Vincent Gripon, *Member, IEEE*, Noboru Sakimura, Tadahiko Sugibayashi, Tetsuo Endoh, *Member, IEEE*, Hideo Ohno, *Member, IEEE*, Takahiro Hanyu, *Senior Member, IEEE*, and Warren J. Gross, *Senior Member, IEEE*

***Abstract*—This paper presents algorithm, architecture, and fabrication results of a nonvolatile context-driven search engine that reduces energy consumption as well as computational delay compared to classical hardware and software-based approaches. The proposed architecture stores only associations between items from multiple search fields in the form of binary links, and merges repeated field items to reduce the memory requirements and accesses. The fabricated chip achieves $13.6\times$ memory reduction and 89% energy saving compared to a classical field-based approach in hardware, based on content-addressable memory (CAM). Furthermore, it achieves $8.6\times$ reduced number of clock cycles in performing search operations compared to the CAM, and five orders of magnitude reduced number of clock cycles compared to a fabricated and measured ultra low-power CPU-based counterpart running a classical search algorithm in software. The energy consumption of the proposed architecture is on average three orders of magnitude smaller than that of a software-based approach. A magnetic tunnel junction (MTJ)-based logic-in-memory architecture is presented that allows simple routing and eliminates leakage current in standby using 90 nm CMOS/MTJ-hybrid technologies.**

***Index Terms*—Associative memory, context-driven search, logic-in-memory, magnetic tunnel junction (MTJ), sparse clustered networks.**

I. Introduction

THERE is a significant need for energy-efficient context-driven search (CDS) engine [1] due to the drastic growth in the amount of stored information in complex digital databases such as DBLP, Twitter, YouTube, or LinkedIn. In such applications, *items* from multiple *search-fields* are often queried to refine the search results.

Conventional software-based multiple-field search engines such as [1] consume high energy since exhaustive back-and-forth memory-access operations through I/O drivers, and also high-latency comparison operations are performed. Content-addressable memory (CAM)-based [2]–[4], Tree-based [5], and Hash-based schemes [6], [7] are alternative search approaches that when used in hardware, are typically intended for high-speed longest-prefix matching (LPM) in network processing, for which fast IP-lookup operations are required against the stored contents.

CAMs employ a brute-force search scheme by matching an input search-word against all of the CAM entries in an attempt to find a matched word or words that point to the corresponding output in another memory module, that is typically a static random access memory (SRAM). Therefore, CAMs consume large amounts of dynamic, as well as standby energies in modern CMOS technologies due to the increase of leakage energy dissipation of SRAMs. Another reason for high energy consumption of CAMs, when used for multiple-field search applications, is due to their data-storage inefficiency. In order to be able to search for a set of outputs, associated with a single input item, the shared input item must be stored redundantly for every association. For example, if an input item "Blue" is associated with two different entries, "A," and "B," two copies of "Blue" are stored in the CAM array dedicated for the search field of "Color." On the other hand, if a single output is associated with multiple input items, redundant SRAM words are occupied with the same output. These redundancies, especially if full-text of the items and outputs are stored, result in increasing the memory usage, the number of search operations, and thus standby and dynamic energy consumptions.

The two-field structure of the CAM-SRAM array, i.e., tag output, requires dedicating an array for each search field. Therefore, independent search operations result in large number of outputs, whose intersections are the actual desired search results. Therefore, large delays in transferring and postprocessing of the independent search results are required.

Tree-based search structures perform the search operation in a tree-like process searching a few bits at a time. This approach results in multiple-levels of searching, and thus increases the latency and the memory usage to store pointers

Manuscript received March 28, 2014; revised July 10, 2014; accepted August 13, 2014. Date of publication October 21, 2014; date of current version December 09, 2014. This work was supported by the Japan Society for the Promotion of Science (JSPS) Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST) program and Fonds québécois de la recherche sur la nature et les technologies (FRQNT). This paper was recommended by Guest Editor S. Mukhopadhyay.

H. Jarollahi and W. J. Gross are with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC, H3A 0E9 Canada (e-mail: hooman.jarollahi@mail.mcgill.ca; warren.gross@mcgill.ca).

N. Onizawa, T. Endoh, H. Ohno, and T. Hanyu are with the Research Institute of Electrical Communication, Tohoku University, Sendai 980-8577, Japan (e-mail: nonizawa@m.tohoku.ac.jp; tetsuo.endoh@cies.tohoku.ac.jp; ohno@riec.tohoku.ac.jp; hanyu@ngc.riec.tohoku.ac.jp).

V. Gripon is with the Electronics Department, Télécom Bretagne, 29238 Brest Cedex 3, France (e-mail: vincent.gripon@telecom-bretagne.eu).

N. Sakimura and T. Sugibayashi are with the NEC Corporation, Tsukuba, Ibaraki 305-8501, Japan (e-mail: n-sakimura@ap.jp.nec.com; sugibayashi@da.jp.nec.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2014.2361061

2156-3357 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

from nodes to the children [6]. Hash-based search architectures store compressed versions of the information and often require large-memory usage, and result in output collisions which would require postprocessing operations that typically incorporate multiple hash-tables for each length of the stored entry. This is unattractive since the length of an item in a search engine can be arbitrary unlike that of the IP addresses.

Parts of this work were presented in [8]. In this paper, we present an extended system-level algorithm, an architecture, a fabricated chip, and measurement results for an associative memory-based multiple-field search engine. The algorithmic advantages and improvements of the proposed system, from a hardware perspective, were previously presented in [8].

The proposed system reduces the energy consumption of the search operation and can be tuned to confine the search scope to either one or a few possibilities. A false-negative result never occurs, i.e., if a matched entry exists, it is always found.

The proposed system, L-SCAN (link, sparse, context, associative, network), features the following concepts. 1) It exploits a unique feature of a recently-introduced family of associative memory, that is based on sparse clustered networks (SCNs) [9]–[12]. It eliminates the necessity for storing the text of the search items and the outputs. Consequently, it also eliminates the energy-hungry brute-force search operations. 2) Due to the use of low-complexity and parallel logical operations that frequently request access to memory cells, the concept of logic-in-memory (LIM) structure is incorporated into the proposed system, that combines a logical operation and a storage device, and places them in a special-purpose LIM-cell, which are arranged into arrays or LIM words. 3) And a nonvolatile magnetic-tunnel-junction (MTJ) device is used in each LIM-cell that features 3-D stack-ability for compactness, and is leakage-free during standby [13]–[15].

In contrast to CAMs, the proposed system permits sharing repeating items and outputs, as well as the associations between them for multiple database entries. Once it is trained with the associations, it retrieves associated outputs given partial input-items, without storing the data-bits directly, which otherwise leads to a large memory usage. Instead, only the associations between the related items from different search-fields are stored in the form of binary links. The location of such binary links are determined by the values of the items. What makes the data-representation different in the proposed system from hash-based counterpart is that in the proposed system, input items are segmented into multiple parts when required, which reduces the memory requirement. This segmentation is performed when the number of different items a search field is required to cover is large. Therefore, the length of each database entry is considered only to avoid collisions. Once the length is enforced by the data-distribution in the application, the diversity of the database entries is realized in selecting the hardware-parameters.

The organization of the paper is as follows. In Section II, a classical associative-memory and a new family of them are reviewed. In Section III, system-level algorithm of the proposed system is presented. Section IV presents discussions of design-space. In Section V, the hardware implementation of an L-SCAN test chip is presented along with architectures of its building blocks as well as a discussion of design flow. In Section VI measurement results of the fabricated chip are shown. In Section VII comparison results with state-of-the-art existing solutions is presented. Section VIII concludes the paper.

II. Background

Associative memories store and retrieve data patterns without the necessity of presenting an explicit input address to them in order to retrieve a previously-stored entry as in indexed memories. Associations between bits of data pattern are stored in the form of weighted links in the storage process, that is also known as *training*. Recovering the missing parts of a partial pattern is achieved with a tunable error probability, whose value depends on the memory parameters.

A. Hopfield Neural Networks

A classical associative memory is Hopfield neural network (HNN) [16] in which the associations are stored in the form of integer-weighted links. The data patterns are first mapped on to the network of nodes that later perform computations to recover the unknown bits. Then, the connections weights are updated between the mapped nodes, and the weight values are stored. A full pattern is quickly retrieved using the nodes and the stored links after a few iterations when a partial pattern is presented to the network, and provided that the network has been previously *trained* with that pattern.

HNNs have two main drawbacks. First, the ratio between the number of memory bits stored to that used (memory efficiency) approaches zero as the network is scaled up to increase the memory capacity. Second, the structure of the network is such that the number of different patterns the network can store (pattern diversity) is limited since the length of a pattern needs to be unnecessarily increased as the network is scaled. Therefore, due to a limited capacity, the network is capable of storing few long messages as opposed many few-bit messages, which is more desirous in search engines as there exists many short items to search through.

B. Sparse Clustered Networks

The drawbacks of the HNNs have recently been addressed in a new family of associative memories known as sparse clustered networks (SCNs) [9], where significantly higher pattern diversities and memory efficiencies were presented compared to those of HNNs. However, similar to HNNs, SCNs can be represented using a graph model consisting of nodes and connections. There are a few differences between the graph models of SCN and HNN. 1) Nodes are clustered where each cluster corresponds to a part of a pattern. 2) Unlike HNN, where all nodes are interconnected, a connection between two nodes in SCN can only exist between nodes from different clusters and not within the same cluster. 3) And a connection is binary-weighted as opposed to its integer-weighted counterpart, which indicates whether or not two nodes are related. If two nodes are connected through different data patterns, the resulting binary link is shared.

As shown in Fig. 1, an SCN is a network consisting of n nodes, which have binary states, and are arranged into c clusters. The number of nodes of the clusters are not necessarily equal,

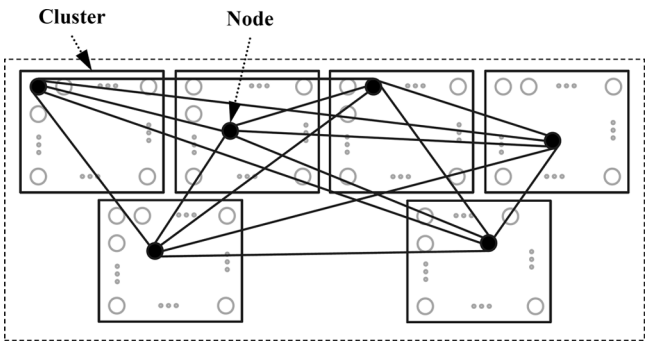

Fig. 1. Graphical representation of node, clusters, and a binary association.

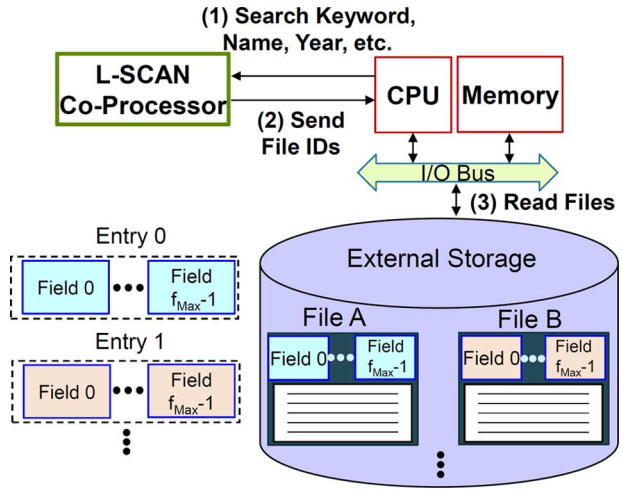

Fig. 2. System level block diagram of the proposed system (L-SCAN) showing how its co-processor communicates with the CPU during search.

although an equal number of nodes is considered in [9] due to its convenience for mathematical analysis. The original SCN in [9] realizes an equal number of nodes per cluster for all clusters. In this paper, different cluster sizes are employed and grouped into fields of clusters that correspond to a search field. A pattern is recognized as a fully-interconnected graph, also known as a *Clique*, including at most one node from each cluster.

1) SCN: Message Training: In order to recover partial input patterns, an SCN-based associative memory is first *trained* with the full data patterns. The training process means realization and storage of the binary associations between segments of data pattern.

The training process starts with *Local Decoding*. In local decoding, an input pattern (*message*), m, of length K bits, is segmented into portions of κ_i $(0 \leq i \leq c-1)$ bits each. This segmentation results in the creation of c sub-messages for which c is also equal to the number of clusters in SCN. Each cluster consists of $l_i = 2^{\kappa_i}$ nodes with binary states [9]. The index of each node is equal to the integer value of its corresponding sub-message. In this paper, a direct binary-to-integer mapping is performed, which maps the integer value of a sub-message to the index of the node of a cluster. Once the nodes corresponding to an input message are determined in all of the clusters, the corresponding binary connections (links) are added to the network. Only these links are stored in memory instead of the sub-messages. A data pattern is therefore represented as a fully-interconnected graph.

The density of a network is defined as the ratio between the number of used connections by the stored messages, to that of the total number of possible connections. The density is derived in [9] using total number of stored messages, M, the number of nodes per cluster $(l = l_i)$, and regardless of the number of clusters.

2) SCN: Message Retrieval: During the retrieval process, a partial input pattern is presented to the network. If the network has been previously trained with a pattern or patterns that partially match with the partial input, by using the stored connections, a *sparse* number of binary-valued nodes are activated (set to "1"), recovering the unknown parts after a few iterations. After each iteration, the number of activated nodes in each cluster is reduced until the value of no more nodes are affected by iterations. The number of iterations depends on the distribution of data patterns and the network parameters.

The decoding process [9] is performed in two stages: 1) local decoding, which is similar to that of the link storage process and 2) global decoding, where the corresponding links are retrieved from the memory cells, and the missing data bits are recovered using the retrieved link values and the local decoding results. Only the global decoding process is performed iteratively to retrieve the message.

III. An Associative Memory-Based Multiple-Field Search Engine

The proposed system is depicted in Fig. 2 showing the communication structure of the proposed co-processor chip with a processor running the interface software program, and the external storage device. A database entry consisting of fields, including inputs and outputs, is also shown.

The co-processor chip is an SCN-based associative memory performing multiple-field search operations by recovering the previously trained search results given a collection of input items from different search fields. The training process is performed by a database administrator prior to the search operation. For example, in an article-publishing database, field items such as specific keywords, list of authors, publication year, and the text of the paper are stored in a file with a unique file ID. The files are stored in the external storage device. The generated search results from the proposed system may include a few detectable false-positive results which can be filtered externally in software or directly displayed. However, a false-negative never occurs. In this paper, the ratio between the average number of retrieved results, including the desired ones, to that of the total possibilities in the search scope is referred to *search focus rate* (SFR). SFR is tunable using the design parameters, some of which influence its hardware complexity.

The role of the co-processor chip during search is thus to focus the search scope from a large number of possibilities to a few ones such that the number of generated results is much smaller than the total number entries a CPU-based alternative would search through by itself otherwise.

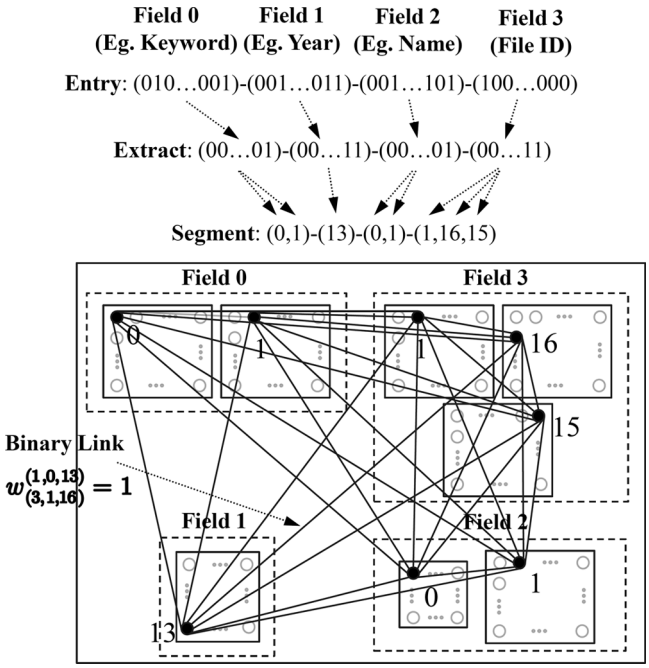

Fig. 3. Example of the proposed SCN-based multiple-field search engine consisting of four fields with various number of clusters in each field and various number of nodes in the clusters.

The average number of generated search results is tunable by optimizing the design parameters such as the number of nodes per cluster, the number of clusters, the length of the inputs, and the total number of database entries.

The SCN-based associative memory used in the co-processor is illustrated in Fig. 3 using an example showing a network consisting of four fields: three input fields (Field 0—Field 2), and an output field (Field 3). In general, it consists of $f_{\max}$ fields, where each field consists of c_f $(0 \leq f \leq f_{\max} - 1)$ clusters. The number of nodes in the ith cluster $(0 \leq i \leq c_f - 1)$ of the fth field, $l_{(f,i)}$, is not necessarily equal in all clusters. Each node in a cluster along with one node in the other clusters of the same field represent an item. The operation of the proposed system is realized in two stages. 1) Training the co-processor: where association between the related search items and the search results (File IDs) are trained to the chip. 2) Search operation: where the file contents are searched by presenting the known items in the search fields.

A. Training

Training process in the proposed system is performed by finding and storing the association of the field items in the co-processor as it is an SCN-based associative memory. During the training stage, a series of operations are performed:

1) Data Collection: A database entry consisting of the items from various search fields and the associated file ID(s) is obtained. Each search field covers a collection of search items that are shared among multiple database entries unlike the CAM-based or CPU-based counterparts, where two database entries do not share an item, and thus duplicate items need to be stored.

2) Input Preparation: Each input item in a field is first reduced in length by extracting K_f bits to be used as inputs in the proposed system. In this paper, a maximum length of 256 bits is dedicated for each search field before the extraction. K_f can be selected depending on the diversity of the items of field f, D_f, and the minimum required SFR. The extraction method depends on the data distribution of the field items and aims to reduce collisions between the extracted items.

If the items for each field have a uniform-random distribution, i.e., no particular similarity patterns can be discovered between the input patterns, the extracted bits can be selected randomly. The index (position) of the selected bits for extraction are consistent in all inputs. In databases, where a field receives inputs with high degrees of similarity, an SCN-based associative memory still generates the correct results, as shown in [17], [18]. However, higher number of pattern similarities result in the generation of a larger number of output search results that include the desired ones.

One simple method is to reduce the similarities by using random vector-projection of the inputs, where an H-bit input pattern is first multiplied (using logical XOR operations) by a $[H \times N]$ random matrix $(N > H)$ of binary elements to generate an alternative input pattern with fewer similar bits. The extracted items are then segmented into c_f parts.

3) SCN Mapping and Link Storage in L-SCAN Co-Processor: During mapping, nodes in clusters corresponding to the input segments are activated (set to "1") according to a specific rule describing the relationship between the segment value and the index of the node to be activated. Then the binary links are added between the activated nodes. To establish the binary links between the nodes, an input pattern with length of K_f-bits is divided into c_f segments such that

$$w_{(f',i',j')}^{(f,i,j)} = \begin{cases} 1, & \text{if } \begin{cases} i \neq i' \\ \text{and } \exists m_n \in \{m_0 \dots m_{M-1}\} \\ s_{(f,i,m_n)} = j \text{ and } s_{(f',i',m_n)} = j' \end{cases} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $w_{(f',i',j')}^{(f,i,j)}$ denotes the binary value of the connection from the jth node of the ith cluster in the fth field to the j'th node of the i'th cluster of the f'th field. The link values are stored in memory for later use during search. m_n is the nth database entry among M total entries combining the information from all the input/output fields. It is segmented into $f_{\max}$ fields and c_f segments in each field. The ith segment $(0 \leq i \leq c_f - 1)$ of the fth field of m_n is denoted by $s_{(f,i,m_n)}$, and its length in bits is denoted by $\kappa_{(f,i)}$ for all M database entries such that

$$\kappa_{(f,i)} = \log_2(l_{(f,i)}). \tag{2}$$

In (1), the mapping rule is that the value of segment $s_{(m_n,f,i)}$ is directly mapped to the index of the node to be activated. Therefore, there should exist $2^{\kappa_{(f,i)}}$ nodes to cover all possibilities.

The definition of density in [9] does not directly apply in the proposed system since the number of nodes in the clusters are not necessarily equal. Therefore, assuming that in M entries trained to the co-processor, all fields are used to define a clique, a new definition of density can be defined between two specific clusters as *Local Density*

$$d_{(f,i,f',i')} = 1 - \left(1 - \frac{1}{l_{(f,i)} l_{(f',i')}}\right)^M \tag{3}$$

where (3) is the ratio between the number of used binary links to that of the total possible connections between clusters i and i'. In [9], it is shown how increasing the value of density affects the message error rate in such an associative memory. In the proposed system, increasing the value of local densities can increase the number of false-positive search results and thus increases SFR. Therefore, in order to reduce SFR as a means to improve the search quality, one can reduce the value of the local densities in two ways: 1) by reducing the total number of stored entries, 2) and/or increasing the number of nodes in clusters with large diversities of the items. When designing the network, a degree of liberty is to adjust the number of clusters. A larger number of clusters results in a smaller silicon area consumption since it decreases the number of possible connections. Consequently, the density is increased together with the number of false positives.

B. Search Operation

Once the co-processor has been trained with the database entries, the search operation can be initiated. The search operation in the proposed system is partially illustrated in Fig. 2. The search process in more detail is a series of operations.

1) Data Collection: This operation is similar to that of the training process. The CPU obtains the search queries from the search fields with available information.

2) Input Preparation: In this step, the CPU generates required data for the input of the co-processor using a similar approach that was explained in the input preparation operation of the training process. The difference is in the status of the inputs as some parts of the inputs are missing such as the file IDs.

3) Search Operation in the Co-Processor: Once the search operation is initiated after the prepared data inputs are presented to the co-processor, the CPU waits for the co-processor to perform the search operation and transfers its outputs to the CPU. The search process in the co-processor is performed by the following.

Local Decoding: The K_f-bit input, prepared in the input preparation operation, is divided into $\kappa_{(f,i)}$ $(0 \leq i \leq c_f - 1)$ segments. Each segment is mapped to activate a binary node in the ith cluster of the fth field. The mapping method is the same as the SCN mapping operation in the training process. A field with missing input information, such as that of the File ID field, will have all of its nodes activated to permit all possible search results.

Global Decoding: Once the corresponding nodes to the input segments are activated in all the clusters, global decoding is performed using the stored links and the indexes of the activated nodes. This process can be iterative, for which a node in any cluster remains or becomes activated in each iteration, if and only if $v^*_{(f,i,j)} =$ "1"

$$v^*_{(f,i,j)} = \left(\bigwedge_{\substack{i'=0 \\ (f',i') \neq (f,i)}}^{\psi-1} \bigvee_{j'=0}^{\gamma-1} w^{(f,i,j)}_{(f',i',j')} v_{(f',i',j')} \right) \bigwedge v_{(f,i,j)} \tag{4}$$

where $v^*_{(f,i,j)}$ is the updated value of the jth node of the ith cluster in the fth field after an iteration. ψ is equal to $\sum_{i'=0}^{c_f-1} c_{i'}$, and γ is equal to $l_{(f,i')}$. The indexes of the activated nodes in the output field are transferred to the CPU after global decoding operation is completed in the chip. An iteration may thus reduce the number of falsely-activated nodes, and may thus reduce the delay to transfer the results.

4) External-Storage Content-Retrieval: In this operation, the CPU reads the chip outputs, and generates the required addresses for the external storage device using the chip outputs.

Formation of File IDs: The file IDs are formed from the chip outputs by concatenating the indexes of the activated nodes in the file ID field, and then realizing all possible combinations that can be created using the index of an activated node in a cluster to all those of other activated nodes in other clusters in that field. For instance, if nodes 3 and 4 are activated in cluster 0, and node 5 is activated in cluster 1 of the file ID field consisting of only two clusters, the possible file IDs are: "3,5" and "4,5." Therefore, iterations can affect both the number of activated nodes, and the formation delay.

File Access: In the next step, the content of the corresponding files to the formed file IDs are retrieved from the external storage device. The addresses are either realized directly by the file IDs, or by mapping them to another set of numbers. In either of the two situations, if a generated address is not valid (e.g., if the address is not in the scope of the addresses), the CPU drops it before accessing the external storage device.

5) Filtration and Display of Search Results: Since the formed file IDs may include false-positive results, the retrieved file contents may also include a few false-positive contents. The search results at this step can be further processed in either of the two ways. 1) The false positive contents are post-processed (filtered) in software by matching the search inputs with the information included in the file contents. Then the search results are displayed. 2) Or the search results are directly displayed without any filtration in form of a list to select from.

C. Updating

In order to update the proposed system with a new database entry, following scenarios are possible depending on the application:

1) Frequent Updates: If frequent updates are required in an application, only a single, but large cluster, can be dedicated for the output-field. This way, once an association is requested to removed, and updated with a new set of input items, then all the connections from the nodes of the output clusters can be removed. Large clusters increase the memory requirement to store the connections.

2) Moderately or Rarely-Frequent Updates: If less-frequent updates occur in an application, it is possible to temporarily mark *dirty* the associations that are no longer valid in software, and drop them if searched. This approach increases the network density, which results in increasing the number of false-positives that can be removed in a postprocessing software program. However, no false-negative results are ever produced. Eventually, the proposed system is retrained with freshly updated associations after several updates take place.

IV. Design Space Exploration

In this section, the behavior of the proposed system is investigated, by simulating its key characteristics in software, such as

the number of generated search results, and the delay to transfer data from the chip to the CPU, when varying various design parameters. The key algorithmic parameters in the design of the proposed system, from an associative-memory perspective, include: The number of nodes, clusters, fields, and database entries.

The specific configurations and experiments presented in this section are selected to create a guideline on how to decide the values for the essential design parameters given the design requirements. A general roadmap is to create an initial intelligent guess, using the understandings in this section, then simulate to learn about its characteristics, and finally tune the parameters knowing their effects of their variations. An intelligent guess requires knowledge about the database diversity, silicon-area limit, number of search-fields and typical search scenarios.

In [9], the authors showed that density, number of clusters, and number of erased clusters determine the error performance of an SCN, that translates into the number of falsely activated nodes. The total number of clusters and the number of erased clusters in the proposed system are determined by the search-engine requirements such as how many fields are more frequently used during search, or the desired diversity of search items in each field. For example, if 1024 keywords are required, then either a single cluster consisting of 1024 nodes must be considered, or if constrained by the silicon-area, that single cluster can be divided into multiple clusters to reduce the number of nodes per cluster, and thus reduce the total memory requirement. On the other hand, increasing the number of clusters exacerbates the number of false positives as the values of local densities are increased.

Selecting the value of local density between two fields that are frequently used during search (e.g., Keyword, and file ID), can be used as a guide to create the intelligent guess for the number of nodes in a cluster. First, a target density, e.g., 60% is selected that determines the error rate. Then, according to (3), given a specific number of SCN messages, (e.g., $M = 10\,000$), and assuming equal number of nodes between the two clusters, the number of nodes in each cluster is rounded to 105.

Based on the understandings and experimental results discussed in this section, a set of parameters has been selected for the fabricated test-chip of the co-processor, as shown in Table I.

Application-specific algorithmic parameters include number of fields, diversity of each field, number of entries, number of iterations, and maximum acceptable error probability (resulting false positives). Hardware-specific constraints are area, speed, and energy dissipation.

Optimizing all parameters for the best figures of merit in all aspects is a sophisticated multi-dimensional problem requiring exhaustive simulations, which depends on an application's parameters, requirements and hardware constraints. We present a heuristic for selecting a set of initial design parameters that after simulating the system behavior given these parameters, can either be adjusted to meet the requirements or suggest relaxation of the hardware constraints. The studies in this section can be used for this purpose.

Let us assume that the goal is to minimize silicon area, while achieving a desired maximum error probability and exploiting the energy reduction opportunity the algorithm demonstrates.

TABLE I
L-SCAN CO-PROCESSOR TEST-CHIP PARAMETERS

Parameter (No.)	Value
Input Search Fields	3: Field 0 (Eg. Keywords), Field 1 (Eg. Year), Field 2 (Eg. Name)
Output Fields	1: File ID (Field 3)
f_{Max}	4
Clusters/Field (c_f)	$c_0 = 2, c_1 = 1, c_2 = 2, c_3 = 3$
Clusters ($\sum_{f=0}^{f_{\text{Max}}-1} c_f$)	8
Nodes/Cluster	$l_{(0,0)} = l_{(0,1)} = 128$ $l_{(1,0)} = 32$ $l_{(2,0)} = 16, l_{(2,1)} = 128$ $l_{(3,0)} = l_{(3,1)} = l_{(3,2)} = 128$
Iterations	1
Unique file IDs	2,000
Field 0 items	16,384
Field 1 items	32
Field 2 items	2,048
Entries (M)	10,000
Field 0 items/File ID	5 (Avg.)
Search Item length (bits)	256
Segmented Item Length (bits)	$\kappa_{(0,0)} = \kappa_{(0,1)} = 7$ $\kappa_{(1,0)} = 5$ $\kappa_{(2,0)} = 4, \kappa_{(2,1)} = 7$ $\kappa_{(3,0)} = \kappa_{(3,1)} = \kappa_{(3,2)} = 7$

Minimizing the memory requirement (and thus reducing the number of LIM cells) is the first step to maximize operating frequency and optimize energy-efficiency. However, reducing the memory affects error probability and thus the transfer delay as more ambiguities are generated. It is important to note that the false positives are detectable and can later be filtered in software. In another application, minimizing the total search delay could be a key goal. In that case increasing the number of iterations helps reduce this delay.

First, given a fixed number of fields, enforced by the application, we choose a single cluster per field. The number of nodes per cluster is then set to be equal to the diversity of each field. This situation will result in the lowest possible error probability since a single cluster can produce fewer ambiguities compared to multiple clusters. Therefore, the transfer delay of the results is also minimized. A single cluster per field maximizes the number of required memory cells, thus the silicon area, and energy. Therefore scaling up the number of nodes in a single cluster reduces speed as it increases the delay for sensing the output of each LIM array, and also requires more complex encoders/decoders. Such system could still be more energy-efficient than the conventional approaches.

To reduce the required memory given a fixed number of entries, the number of nodes in each cluster can be divided into

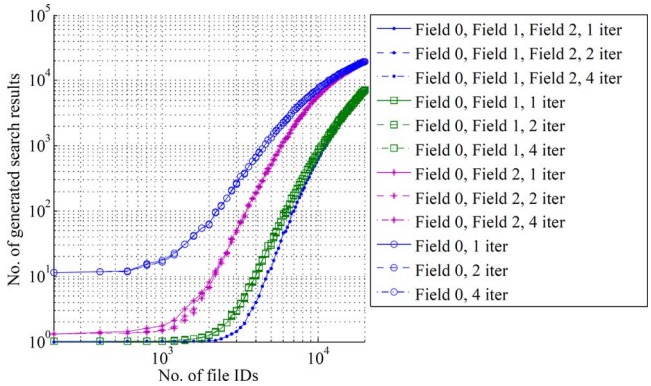

Fig. 4. Relationship between the number of search results, generated by the proposed system, and the total number of associated database entries in the form of file IDs for various search scenarios.

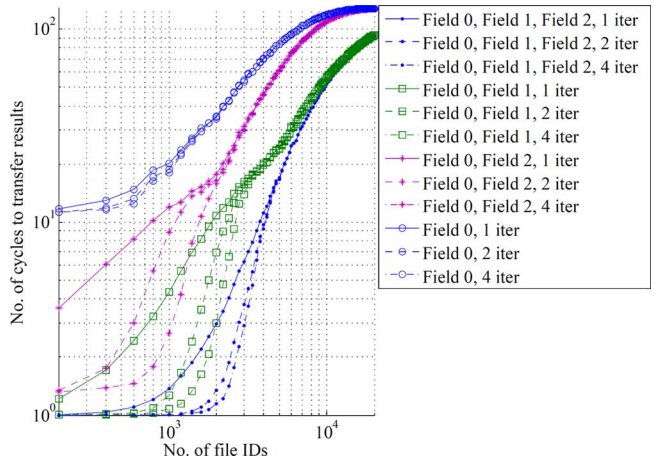

Fig. 5. Relationship between the number of search results, generated by the proposed system, and the number of required clock cycles to transfer the node indexes for various search scenarios.

multiple clusters, resulting in the coverage of the same diversity, a reduction in area, but a higher error probability. For example, let us consider a single cluster including 16 nodes which can map a 4-bit input to one of its nodes. If memory requirement is restricted, a 4-bit input can be divided into two clusters (e.g., two 2-bit clusters) resulting in two clusters with four nodes each. Therefore, eight nodes in total are required to represent the same diversity. This number is half the number of nodes in a single cluster counterpart, and thus resulting a reduced memory requirement according to (5).

If the achieved error probability is not acceptable given number of entries (affecting local densities), either the number of entries needs to be reduced or the number of clusters in each field. The density of interest is within the small-slope region of its error probability response (see Fig. 6) although a more aggressive slope may also be tolerated.

The simulation results in this section are presented prior to the optional filtration operation in software as discussed in step 5 in Section III-B.

A. Density Effect

As the proposed system stores the associations within the database entries (also referred as storing the entries in short), the number of the used connections is increased, causing the local density values to also increase. Fig. 4 depicts the relationship between the number of stored database entries (i.e., uniquely associated file IDs), and the number of generated search results in the proposed system for various search scenarios. The rate of change in the number of generated search results to that of the associated file IDs) is equal to SFR.

B. Search Scenarios

The number of generated search results in the proposed system depends on the search scenario such as how many search fields with valid search information have been presented. In Fig. 4, four search scenarios are presented.

The advantage of the proposed system in focusing the search scope is better realized, as shown in Fig. 4, when large number of search fields with presented inputs are required to identify desired search results. Receiving information from larger number of fields helps reduce the search scope since more useful nodes are activated to contribute in the global decoding.

C. Iteration Effect

It is also observed from Fig. 4 that the number of iterations in the global decoding has a negligible impact on the number of generated search results. The reason lies in the fact that when larger number of nodes are activated at the first iteration compared to second or more, the combination of the generated indexes often result in invalid file IDs that are detected in the CPU, and dropped for further processing. However, iterations may reduce the number of falsely-activated nodes in the output clusters, and thus reduce the delay to transfer the results to the CPU.

Fig. 5 depicts the effect of iterations on the number of clock cycles to transfer the search results from the co-processor to the CPU for various search scenarios. The number of clock cycles are simulated by evaluating the maximum number of activated nodes among all output clusters, and assuming that each activated node requires a clock cycle to transfer its index. It is assumed that the indexes of the activated nodes in a cluster are transferred serially for each cluster but in parallel with other clusters. Considering the observations from Figs. 4 and 5, it can be concluded that the search scope after using the co-processor can be significantly focused. Only a single matched result is determined if the total number of associated file IDs (stored database entries) falls into a region, where a small rate of changes in the number of search results can be observed (e.g., when the number of file IDs are < 2000). This rate of change is SFR, that is a parameter which can be defined in the design requirement of the proposed system for a specific database to control the number of false-positive results.

D. Number of Nodes and Clusters

Fig. 6 depicts the effect of varying the number of nodes and clusters, in fields with various diversities, on the total number of generated search results for the proposed system with configurations specified in Table II.

It can be observed that increasing the number of nodes and clusters (Config. III) in the output-field has a larger impact on limiting the scope of search than increasing them in other fields. The reason lies in the fact that the sweeping variable is the number of file IDs, which is associated with the output-field.

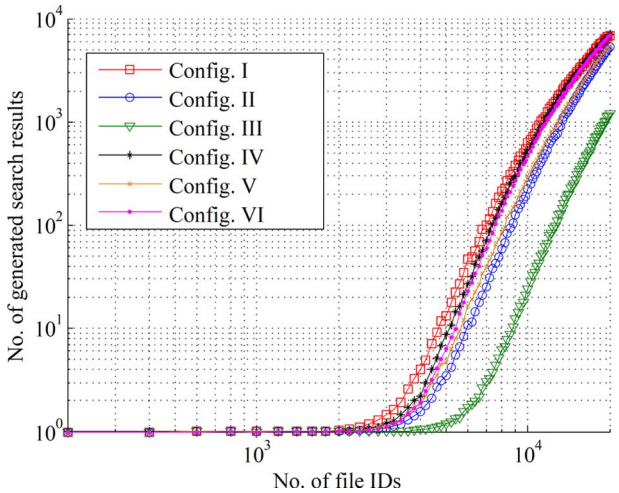

Fig. 6. Relationship between the number of search results, generated by the proposed system, and the total number of associated database entries in the form of file IDs for various configurations of the SCN-based association memory in the proposed system.

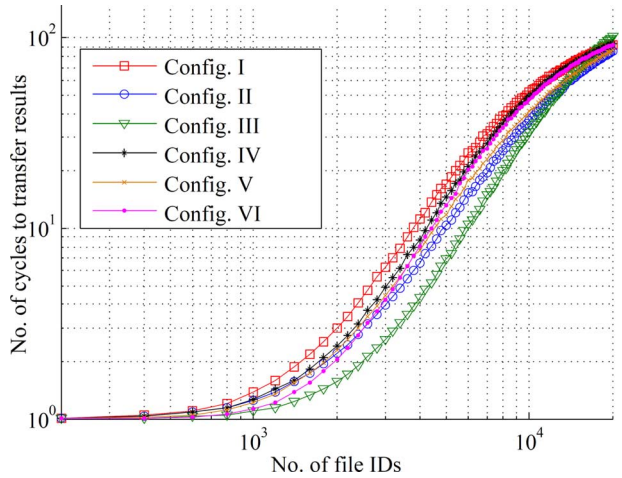

Fig. 7. Relationship between the number of search results, generated by the proposed system, and the number of required clock cycles to transfer the node indexes for various configurations of the SCN-based association memory in the proposed system.

TABLE II
DESCRIPTION OF L-SCAN CONFIGURATION PARAMETERS

	Configuration					
Parameter	I	II	III	IV	V	VI
$\kappa_{(0,0)}$	7	8	7	7	7	7
$\kappa_{(0,1)}$	7	8	7	7	7	7
$\kappa_{(0,2)}$	-	-	-	7	-	-
$\kappa_{(1,0)}$	5	5	5	5	5	6
$\kappa_{(1,1)}$	-	-	-	-	5	-
$\kappa_{(2,0)}$	7	7	7	7	7	7
$\kappa_{(2,1)}$	4	4	4	4	4	4
$\kappa_{(3,0)}$	7	7	8	7	7	7
$\kappa_{(3,1)}$	7	7	8	7	7	7
$\kappa_{(3,2)}$	7	7	8	7	7	7
c_0	2	2	2	3	2	2
c_1	1	1	1	1	2	1
c_2	2	2	2	2	2	2
c_3	3	3	3	3	3	3

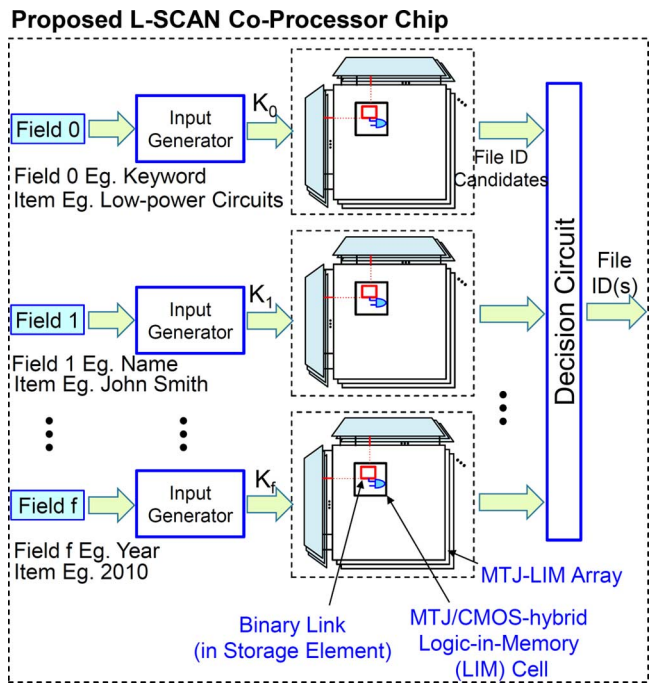

Fig. 8. System-level architecture of the proposed system performing a search operation.

Increasing the number of clusters in a field, while keeping the total number of nodes constant, however, has a negative impact on the number of falsely-activated nodes. The reason lies in the fact that as the number of nodes per cluster is decreased, the number of used connections from its nodes is increased since the nodes are shared with a larger number of items. Therefore, the values of local densities between the clusters are increased. An advantage though is that the total number of connections to be stored, and thus the memory requirement, is reduced.

On the other hand, increasing the number of clusters without reducing the number of nodes per cluster, results in the reduction of the number of shared nodes among multiple entries. Therefore, the number of false-positive results are reduced as the values of local densities are decreased.

Fig. 7 shows the relationship between the total number of stored database entries for various configurations in Table II on the number of cycles to transfer the results.

V. HARDWARE IMPLEMENTATION

The algorithm for training and search operations in the proposed system were introduced in Sections III-A, and III-B, respectively. Fig. 8 shows the proposed hardware implementation of the co-processor test-chip, that is implemented based on the example network shown in Fig. 3, and the selected parameters in Table I. The measured results are based on a single iteration. Increasing the number of iterations results in reducing the delay to transfer the results, but does not affect the search accuracy.

Energy consumption of a search operation is typically defined in CAMs by measuring the amount of energy/bit/search [2], [3], without including the energy consumption of I/O buffers and pads to transfer data outside of the chip. In the proposed system, energy consumption is reduced in comparison with classical hardware-based search-alternatives (e.g., CAM, and CPU), by exploiting the unique structure of the SCN-based associative memory to associate input and output data-patterns. In this application, search-inputs are associated with the search-results in a memory-efficient way that eliminates the necessity to directly store the text of search-data, or a compressed version of it, in the search engine. Furthermore, standby energy is also eliminated in the proposed system since the memory elements used in it are nonvolatile. The binary-links associating the data-patterns are stored in arrays of Logic-in-Memory (LIM) cells, where each cell integrates a logical operation and a memory device. In each array, there exists a shared output for each row, which is, along with other outputs, processed by standard-logic cells. In addition, there exists shared differential bit-lines for each column, that are used during training, in order to write the link values into the LIM cells. MTJ devices are stacked above 90 nm CMOS layers to provide a compact and nonvolatile architecture. The use of an MTJ-based architecture in this paper is interesting mainly due to the following reasons. 1) Nonvolatility of the memory structure in the proposed search engine is important for power efficiency particularly when the search engine is in a standby mode because it can be turned off, and turned back on when needed without requiring to retrain the entire database entries into the search engine. 2) Since it is nonvolatile, after an integrated system (including a CPU) is turned off for a while or restarted, the search engine does not need to be retrained. 3) MTJ devices are extremely compact memory structures, and are comparable to the size of a via between two layers permitting efficient routing in dense arrays compared to the SRAM-based counterpart. These features are attractive in our LIM-based architecture. 4) Previous studies have shown that MTJ devices are much more tolerant to radiations causing soft-errors [19].

In the co-processor, the total number of required MTJ-LIM cells, P, is given by

$$P = \sum_{f=0}^{f_{\mathrm{Max}}-1} \sum_{i=0}^{c_f-1} 2^{\kappa_{(f,i)}} \cdot \sum_{f'=0}^{f_{\mathrm{Max}}-1} \sum_{\substack{i'=0 \\ i' \neq i}}^{c_f-1} 2^{\kappa_{(f',i')}} \qquad (5)$$

whereas, for the proposed application, a CAM-based counterpart would require $f_{\mathrm{Max}} \times Q \times M$ CAM cells, where Q is the required length of a search item in the database. This memory requirement is not taking into account the number of memory bits in the SRAM array that is attached to each CAM array. In a CAM-based counterpart, the SRAM array stores the file IDs.

A. Architecture of MTJ-Based Logic-in-Memory Arrays

Fig. 9 shows the architecture of the MTJ-based LIM (MTJ-LIM) arrays according to the parameters defined in Table I.

1) Training: During the training process of the co-processor, according to Section III-A, K_f-bit reduced-length input-items, including the file IDs, are segmented into c_f segments of $\kappa_{(f,i)}$ bits. The value of each segment as a row address along with that

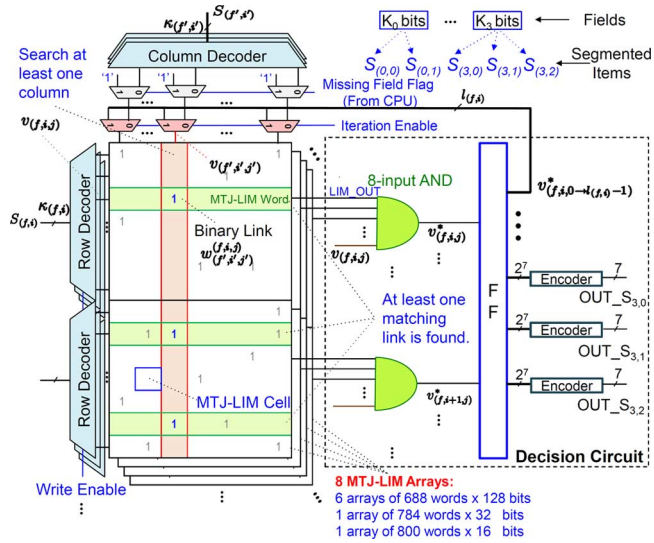

Fig. 9. MTJ/CMOS-hybrid architecture of the co-processor integrating MTJ-LIM arrays with standard logic architectures.

of a different segment as a column address identifies the location of a binary link to be stored in an MTJ-LIM cell following (1).

Segment $S_{(f,i)}$ corresponds to the ith cluster of the fth field, and is the row address-bus that along with a different segment, $S_{(f',i')}$, $(i' \neq i)$ used as a column address-bus, define the location of the link value to be stored. $S_{(f,i)}$ is the signal name whose value is $s_{(f,i,m_n)}$ in (1).

The number of segments and their lengths can be optimized to meet the required number of different possibilities for items in each field (diversity), and the available silicon area. In a field with small diversity, no segmentation may be required. For large diversity (e.g., Keywords and File IDs), the concatenation of the segments in the related field represents an item.

To store the link from a node, whose index is represented by the value of a segment, to another node represented by the value of another segment, an MTJ-LIM array is accessed using a $\kappa_{(f,i)} : 2^{\kappa_{(f,i)}}$ column decoder, where $(\mathrm{Max}(2^{\kappa_{(f,i)}}) = 128)$, and $\kappa_{(f',i')} : 2^{\kappa_{(f',i')}}$ $(i' \neq i)$ row decoders operating in parallel. Each MTJ-LIM array (out of eight) consists of r_i rows of $2^{\kappa_{(f,i)}}$ MTJ-LIM cells, storing one bit in each cell, where r_i is given by

$$r_i = \sum_{f'=0}^{f_{\mathrm{Max}}-1} \sum_{\substack{i'=0 \\ i' \neq i}}^{c_f-1} 2^{\kappa_{(f',i')}}. \qquad (6)$$

Therefore, in each MTJ-LIM array, there exists sufficient number of MTJ-LIM cells to store the links from any possible value from one segment to that of all the values of other segments, without explicitly storing the items.

2) Search: The architecture used during the search operation is the implementation of (4), using the MTJ-LIM arrays operating with standard logic cells, which were also used during the training operation. It is performed by presenting the known reduced-length segmented inputs to the column decoders while leaving the row decoders inactive. If the value of the input of a column-decoder is unknown, all of its outputs are set to "1" to

permit searching all possible links. In an algorithmic perspective, this is equivalent to activating all nodes in a cluster if no information is provided for that cluster. The MTJ-LIM arrays implement the first part of (4) that is

$$\bigvee_{j'=0}^{\gamma-1} w_{(f',i',j')}^{(f,i,j)} v_{(f',i',j')}. \tag{7}$$

A Decision Rule (DR) is then applied by the decision circuit using the output of each MTJ-LIM word following the last part of (4). The DR indicates that the jth value $(0 \leq j \leq 2^{\kappa_{(f,c_f)}} - 1)$, represented by the $S_{(f,i)}$, is the index of a node to be activated if two conditions are valid: 1) the jth column of the LIM-Array is selected by the column decoder whose input is $S_{(f,i)}$, and 2) there exists at least one link (output of "1" from an MTJ-LIM word) from a selected word (by the value of another segment) in all other MTJ-LIM arrays. Therefore, a $(\psi = 8)$-input AND operation is performed for each word. The output of the AND operations can result in multiple "1"s, indicating the possibility of multiple search results. The search results from the clusters in the output field are then serially transferred to the CPU using priority-encoders and shift-registers, while the MTJ-LIM arrays are in standby. The outputs of the decision circuit are connected through feedback to the inputs of the MTJ-LIM arrays to permit iterations, if necessary, depending on the application.

As discussed in Section IV-C, iterations may reduce the delay to transfer the search-results, but not the search accuracy.

B. Architecture of MTJ-Based Logic-in-Memory Cells

An MTJ-device is a compact two-terminal nonvolatile structure whose resistance value is changed from high to low by flowing enough current in it in one direction, and from low to high in the reverse direction [20]. Therefore, it can be used as a nonvolatile memory element whose stored value is maintained after removing the power source from it, and thus eliminates the leakage current in dense structures such as SRAMs. MTJ process is fabricated on the top of that of the CMOS, and is suitable for compact architectures such as logic-in-memory, where many parallel logical operations are performed requiring parallel accesses to memory.

Fig. 10 depicts the architecture of an MTJ-LIM word consisting of $l_{(f,i)}$ cells, integrating logic and memory in a cell. The LIM words are arranged into MTJ-LIM arrays with parameters shown in Table I. The logic-in-memory operation in each cell is equivalent to that of a logical NAND with two inputs: v1, and w1, and an output: LIM_OUT which is pre-charged before a read (search) operation. Each MTJ-LIM cell includes differential bit-lines (BL, BL') that are only used during training (write), a bias voltage $(V_{\rm bias})$, and an output (LIM_OUT), which are shared among other MTJ-cells in a row. Write and read operations are performed through N2-MTJ and N3-N1-MTJ paths, respectively.

P1 is a voltage-controlled current-source whose current value is controlled by $V_{\rm bias}$, and is used during read. When v1 is equal to "1," and while a read operation is being performed, P1 permits distinction between a P and an AP state in the MTJ-device, sensed at LIM_OUT by a sense amplifier. Without P1, the

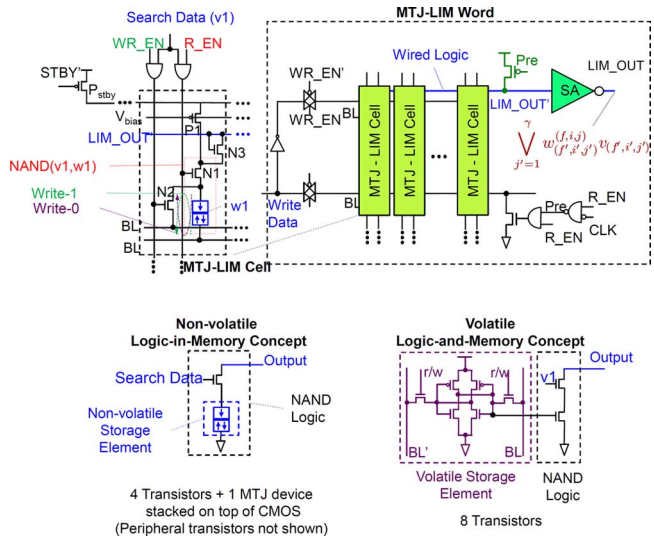

Fig. 10. Circuit diagram of an MTJ-LIM word.

pull-down path to ground quickly discharges the pre-charged LIM_OUT to ground in both states of an MTJ-device. The voltage of LIM_OUT, when the pull-down path is established while the MTJ-device is operating at a P or AP state, is referred to $V_{\rm LIM\_OUT,P}$, and $V_{\rm LIM\_OUT,AP}$ respectively. The difference is referred as $\Delta V_{\rm LIM\_OUT}$. A good cell architecture maximizes this difference to optimize the read-margin of the cell.

N3 is a diode-connected transistor, and is thus always in saturation. It is used to optimize $\Delta V_{\rm LIM\_OUT}$. When v1 = "1," N3's substrate-to-source voltage (V_{sb}) depends on the state the MTJ-device is operating at. The AP state of the MTJ-device results in a larger threshold voltage for N3 compared to the P state, and thus results in a slower pull-down of LIM_OUT. The difference in discharging rate of $V_{\rm LIM\_OUT}$ increases $\Delta V_{\rm LIM\_OUT}$.

Fig. 11 depicts HSPICE simulation waveforms plotting three curves when sweeping $V_{\rm LIM\_OUT}$ using a voltage source from 0 to 1.2 V versus 1) the amount of current passing through N1-MTJ which occurs during read ($V1 = R\_EN =$ "1") when the MTJ-device is in low-resistance state, i(mnp), 2) N1-MTJ current when MTJ device is in high-resistance state, i(mnap), and 3) the negated current through P1, i(mp). The two intersections of curve i(mp) with i(mnp) and i(mnap) determine the steady-state voltage of LIM_OUT for a read "1" and read "0." This simulation is used to optimize the aspect-ratios of the MTJ-LIM cell transistors as well as the bias voltage $(V_{\rm bias})$ such that $\Delta V_{\rm LIM\_OUT}$ is maximized. Throughout the simulations, it is determined that the average value for $\Delta V_{\rm LIM\_OUT} = 0.38$ V at 200 MHz clock frequency among different process corners. Therefore, the threshold of the sense-amplifier is designed to be equal to 0.79 V $(V_{DD} - 0.5 \times \Delta V_{\rm LIM\_OUT})$ as LIM_OUT is first pre-charged to V_{DD} before evaluation. During a read operation, which is controlled by R_EN, $l_{(f,i)}$ two-input logical NAND operations are performed according to (4) in parallel through a path constructed by N1, N3 and the MTJ device. The logical NAND operation in each MTJ-LIM cell has two inputs: $w_{(f',i',j')}^{(f,i,j)}$ and $v_{(f',i',j')}$. The output of each MTJ-LIM cell, LIM_OUT, performs a wired-NOR operation as it is first pre-charged to V_{DD} before R_EN enables a read operation. If

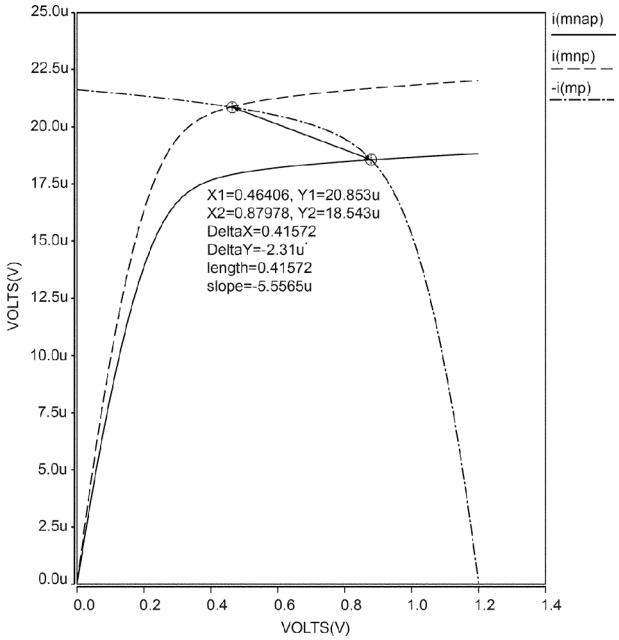

Fig. 11. HSPICE Simulation waveforms used to optimize the noise margin of the MTJ-LIM cell.

at least one of the NAND operations outputs a "0," condition 2 of the DR is partially met and a pre-charged LIM_OUT is pulled down to trigger an inverted sense amplifier.

The threshold voltage of the sense-amplifier is thus decided depending on the desired frequency of operation, the maximum effect of process variations, and the optimized value of $\Delta V_{\mathrm{LIM\_OUT}}$.

A write operation is performed (through N2) on an MTJ-device to change its state of resistance from high to low or the inverse by flowing current from BL to BL' or the reverse depending on the write value. To eliminate standby energy, the power supply is detached from the MTJ-LIM cells during standby (using P_{stby}), as the MTJ-LIM cells are nonvolatile.

C. Design Flow

The hardware architecture of the proposed system including behavioral description of the MTJ devices was implemented in VHDL, simulated, and verified using ModelSim. The RTL design was converted to standard cells using Cadence Encounter RTL Compiler. The converted result included black boxes for the MTJ arrays that were later filled with the custom layouts.

MTJ-LIM cells and arrays were designed, and simulated using HSPICE and NEC $90\ \mathrm{nm}$ CMOS technology in combination with MTJ device models to meet the design requirements. The custom-layout of an MTJ-LIM cell, and the arrays with different sizes were then drawn using Cadence Virtuoso and Cadence SKILL scripting language. The coordinates of the physical pins and array boundary of each MTJ-LIM array were then extracted using Cadence Abstract Generator. The layout of the entire chip was then prepared using Cadence SoC Encounter and the results from Cadence Abstract Generator. The resulting layout was then imported in Cadence Virtuoso for final verification.

VI. Measurement Results

Fig. 12 shows the fabricated chip for L-SCAN co-processor chip. The fabricated architecture of the co-processor is based on the design parameters shown in Table I. The entire $25\ \mathrm{mm}^2$ available die-area has been exploited to include the MTJ-based LIM arrays in combination with standard CMOS primitive cells.

Fig. 13 shows measurement results of an MTJ-device illustrating its behavior as a nonvolatile storage device toggling between the two states of resistance: a high-resistance state (AP), and a low-resistance state (P). According to the measurements, the achieved resistance in the AP, and P states are $R_{AP} \approx 5.4\ \mathrm{K}\Omega$ and $R_P \approx 3.0\ \mathrm{K}\Omega$.

Fig. 14 shows the measured waveforms of the fabricated MTJ-based co-processor chip. The fundamental operation of the co-processor is to determine which value of a segment is linked with an input segment. LIM_OUT1 is the sense-amplifier output of a word in a LIM-array for which at least one of the MTJ-LIM cells has created a pull-down on the pre-charged output. On the other hand, LIM_OUT0 corresponds to an MTJ-LIM cell for which none of MTJ-LIM cells have generated a pull-down path. The waveforms prove the correct function of an MTJ-LIM word by showing two possible scenarios in the evaluation phase: 1) a word is linked to the input and 2) no links have been stored corresponding to the input.

Furthermore, the co-processor chip has been verified for functionality and energy measurement at 200 MHz by first writing associations according to Table I, and then retrieving the expected file IDs (according to software simulations) using scan-chain registers that serially transferred the outputs. Measured data corresponding to the overall system is in the form of streams of binary patterns that were captured from a logic analyzer given partial inputs to the chip. We analyzed the outputs by verifying that the partial inputs have been recovered correctly by matching them against original complete inputs. The total energy consumption of a system employing L-SCAN co-processor chip, including I/O pads and buffers, can be modelled as

$$E_{\mathrm{total}} \approx E_1 + T \cdot E_2 + E_{\mathrm{CPU}} \tag{8}$$

$$E_{\mathrm{CPU}} \approx \prod_{i=0}^{c_{f_o}-1} n_i \cdot E_3 + E_4 + F \cdot E_5 + R \cdot E_6, \tag{9}$$

where E_1 is the total energy consumption of the co-processor chip for R iterations, T is the maximum number of clock-cycles required to transfer the indexes of the activated nodes in the output-clusters to the CPU. It is assumed that each output-cluster serially transfers the indexes, but in parallel with other output-clusters. E_2 is the energy consumption to transfer the index of an activated node per output-cluster, n_i is the number of activated nodes in the ith output-cluster out of c_{f_o} clusters in the output field, E_3 is the energy consumption to form a file ID from the obtained indexes, E_4 is the energy consumption to detect and drop the invalid file IDs in software, F is the number of valid file IDs, E_5 is the energy consumption to retrieve a file from the external storage-device to the CPU, and E_6 is the energy consumption of the CPU per clock cycle, while idle.

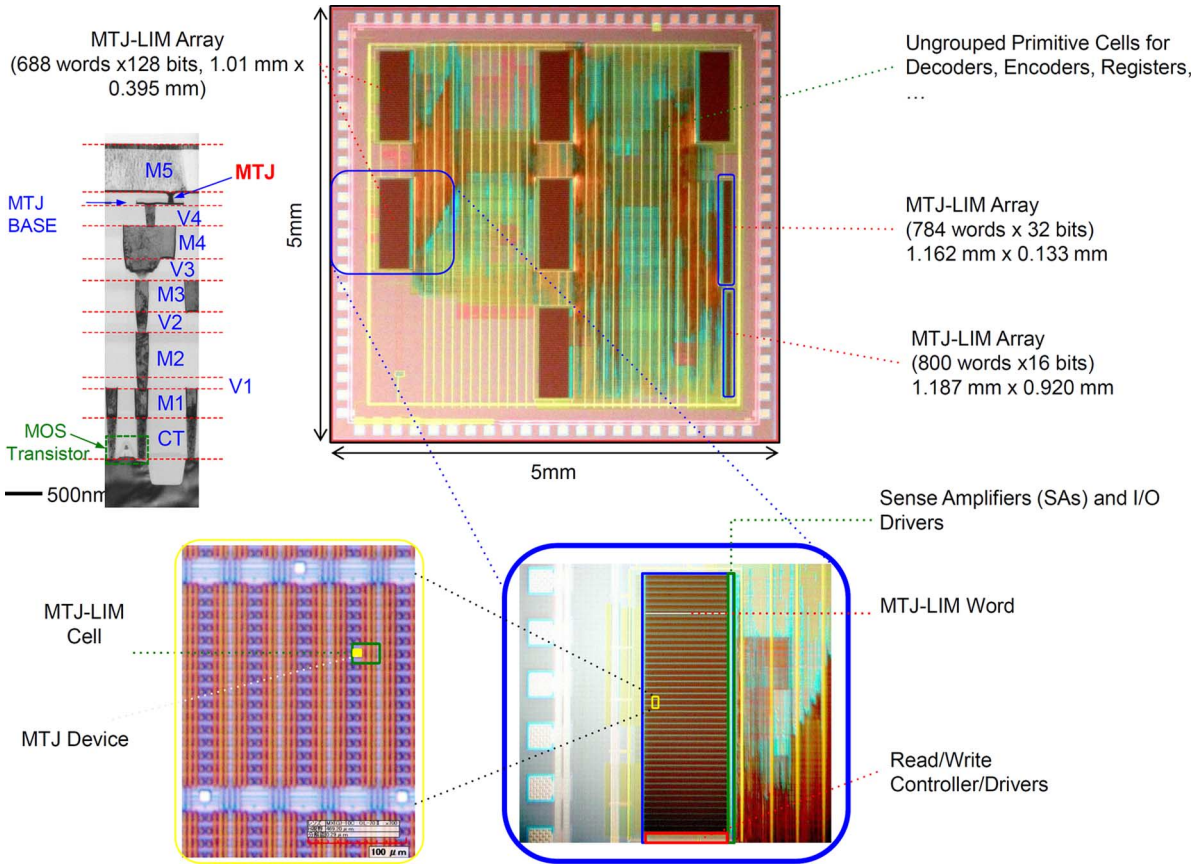

Fig. 12. Die micrograph of L-SCAN co-processor chip.

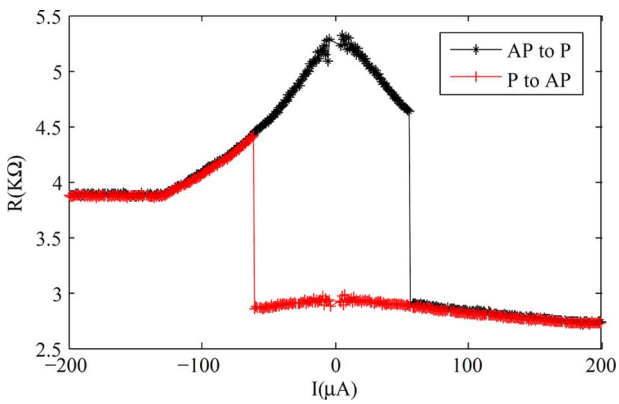

Fig. 13. Measured IR characteristic of an MTJ device, used in the co-processor, toggling between the two low-resistance and high-resistance states: Parallel (P) and Anti-Parallel (AP).

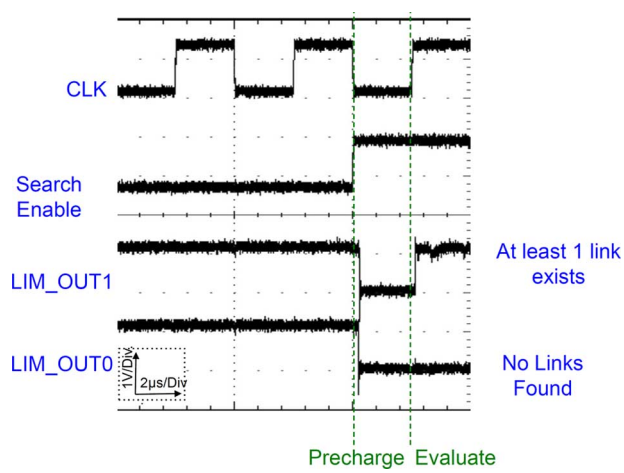

Fig. 14. Measured waveforms.

The total energy-consumption model of a CAM-based alternative is similar to that of the proposed system. The difference lies in the fact that in a CAM-based architecture, no iterations are required, and the CPU processing of the results is different. The search-fields are separated in CAMs, and thus the intersection of the generated results is processed in CPU.

VII. Comparison

Table III shows the comparison of the measured chip results of the fabricated co-processor chip, a fabricated ultra low-power processor, and CAM-based architectures. The fabricated processor is in the same CMOS technology as in the proposed system, and is based on Texas Instrument MSP430 with 16-bit instruction, 32-bit data, and 64-KB SRAM. The energy consumption in the processor was measured for performing an exhaustive context-driven search in software with related search parameters in Table I. The fabricated measurement results of the proposed system include the energy consumption of I/O pads and their drivers when the results are transferred to the CPU.

The CAM-based architectures are adapted to perform the function of the proposed system by dedicating one CAM array

TABLE III
FEATURE COMPARISON

Parameter	L-SCAN (This work)	CPU-CDS	CAM-CDS [2]	CAM-CDS [3]	CAM-CDS [4]
Process Technology	75 nm (ϕ) P-MTJ / 90 nm CMOS	90 nm CMOS	65 nm CMOS	32 nm CMOS	75 nm (ϕ) P-MTJ / 90 nm CMOS
Power Supply (V)	1.20	1.00	1.00	0.95	1.20
Average Energy/bit/search (fJ/bit/search)	[1]0.063	80.1 ([1,2]115.3)	1.98 ([2,3]3.95)	0.58 ([2,3]2.60)	[3]3.44
Frequency	200 MHz	200 MHz	250 MHz	1 GHz	200 MHz
Required Physical Memory (Mb)	0.54 ([4]Non-Volatile)	7.32	7.32	7.32	7.32 ([4]Non-Volatile)
Min./Average Transfer Cycles	3/16.69	-	5/158.75		
Min./Average Search Cycles	2/2	240 / 3.4×10^6	4/4	2/2	2/2
Min./Average Search + Transfer Cycles	5/18.69	240 / 3.4×10^6	9/162.75	7/160.75	7/160.75
Min./Average Search + Transfer Delay (ns)	25/93.45	1.2×10^3 / 1.7×10^7	36/651	7/160.75	35/803.75

[1] The energy consumption of the core, I/O buffers, and pads for a complete search and transfer operation.
[2] Scaled to 90 nm CMOS (1.2V) using $E^* = E \times (90/\text{Technology})(1.2/V_{DD})^2$.
[3] The energy consumption of the core only.
[4] MTJ-based LIM cells.

for each search field. However, the energy consumption in transferring their results is not included in the comparisons as these measurements are not generally included in literature.

The search delay values, correspond to the average delay in computing the indexes of activated nodes in the proposed system for an iteration, the delay in finding the matched entries in CAMs at the operating frequency of the proposed system, and the delay in running the software search program in the processor. In the proposed system, the delay is limited by the current generation of MTJ-devices, and the pads.

The average delay to transfer the generated search results in the proposed system can be calculated using Fig. 7, and the operating frequency of the chip. The total delay, that is average delay of search plus average delay of transfer, for an average search scenario is equal to 18.69 clock cycles, which is $8.6\times$ smaller than that of a CAM-based architecture transferring independent matched results. Since it is required to transfer the results serially after the search is completed, the co-processor, and CAM architectures cannot be easily pipelined.

The energy consumption per bit per search is not affected by varying the frequency of operation for the same processing technology as the load capacitances at each circuit node remains unchanged. Therefore, for comparison purposes, it is possible to normalize various frequencies to a reference one without changing the energy values.

The proposed system achieves 97%, 89%, and 98% energy reduction, on average, compared to adapted versions of [2] and [3], and [4] for use in the application in the proposed system, respectively. The energy consumption of the proposed system is three orders of magnitude smaller than that of its CPU-based counterpart running search in software.

The proposed system requires $13.6\times$ fewer memory bits compared to those of CAM-CDS, that along with the 3-D stacking of MTJ-devices and the LIM structure, achieves a compact low-energy search architecture.

It may be possible to reduce the supply voltage of a high-frequency but low-energy CAM (e.g., [3]) to reduce the energy consumption at the frequency of the proposed system, since their intended operating frequency is higher. However, even at low-voltages (e.g., 0.5 V), the estimated scaled energy of [3] (0.161 fJ/bit/search) is $2.6\times$ higher than that of the proposed system, due to its algorithmic reduction of hardware-complexity.

VIII. CONCLUSION

In this paper, the algorithm, architecture, and fabrication results of a nonvolatile multiple-field search engine were presented employing the concept of sparse clustered networks. A test chip for the proposed architecture (L-SCAN) was fabricated in a $90\ \text{nm}$ CMOS/MTJ-hybrid process. It uses a logic-in-memory (LIM) architecture to reduce the delay of memory accesses, and simplify routing between logical gates and memory units. Magnetic-tunnel-junction devices were used as memory elements of the LIM architecture to take advantage of a compact and leakage-free architecture in the proposed co-processor chip.

In the proposed system, the memory requirement to construct a search engine is reduced by $13.6\times$ compared to a classical hardware-based search architecture using CAMs. The algorithmic reduction in the memory requirement and access, as well as the architecture techniques used in the LIM-based design of the proposed system reduce the energy consumption

per search by 89% compared to a state-of-the-art CAM recently introduced in the literature. Furthermore, for comparison purposes, an ultra low-power CPU was also fabricated in the same technology, and its energy consumption was measured running a software-based search algorithm. The measured energy consumption of the proposed system is smaller by three orders of magnitude, while the search operation is accelerated by five orders of magnitude. The large energy consumption and delay of a CPU-based search engine is mostly due to the exhaustive search nature of the search algorithm, and frequent memory accesses using I/O buffers.

Furthermore, the search delay of the fabricated CPU running serial search algorithm was also simulated, and its number of clock cycles were measured on average to complete several search operations with different scenarios. Due to its parallel computation structure, the proposed system achieves $8.6\times$ reduced number of clock cycles on average in performing a search operation compared to CAMs, and five orders of magnitude reduced number of clock cycles compared to the CPU-based counterpart running search in software.

Future generation of MTJ-devices will improve speed of operation and larger ratio between high to low resistance in its states of operation, which will result in reduction of dynamic energy consumption.

Acknowledgment

The authors would like to thank S. Matsunaga, A. Mochizuki, A. Tamakoshi, Y. Takako, and R. Nebashi for helpful discussions.

References

[1] K. Taha and R. Elmasri, "XCDsearch: An XML context-driven search engine," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 12, pp. 1781–1796, Dec. 2010.

[2] I. Hayashi, T. Amano, N. Watanabe, Y. Yano, Y. Kuroda, M. Shirata, K. Dosaka, K. Nii, H. Noda, and H. Kawai, "A 250-mhz 18-Mb full ternary CAM with low-voltage matchline sensing scheme in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 48, no. 11, pp. 2671–2680, Nov. 2013.

[3] I. Arsovski, T. Hebig, D. Dobson, and R. Wistort, "A 32 nm 0.58-fJ/bit/search 1-GHz ternary content addressable memory compiler using silicon-aware early-predict late-correct sensing with embedded deep-trench capacitor noise mitigation," *IEEE J. Solid-State Circuits*, vol. 48, no. 4, pp. 932–939, Apr. 2013.

[4] S. Matsunaga, N. Sakimura, R. Nebashi, Y. Tsuji, A. Morioka, T. Sugibayashi, S. Miura, H. Honjo, K. Kinoshita, H. Sato, S. Fukami, M. Natsui, A. Mochizuki, S. Ikeda, T. Endoh, H. Ohno, and T. Hanyu, "Fabrication of a 99%-energy-less nonvolatile multi-functional CAM chip using hierarchical power gating for a massively-parallel full-text-search engine," in *Proc. Symp. VLSI Circuits (VLSIC)*, 2013, pp. C106–C107.

[5] Y. Chang, F. Kuo, H. Guo, and C. Su, "Layeredtrees: Most specific prefix based pipelined design for on-chip IP address lookups," *IEEE Trans. Computers*, 2013, to be published.

[6] J. Hasan, S. Cadambi, V. Jakkula, and S. Chakradhar, "Chisel: A storage-efficient, collision-free hash-based network processing architecture," in *Proc. 33rd Int. Symp. Comput. Archit.*, 2006, pp. 203–215.

[7] S. Dharmapurikar, P. Krishnamurthy, and D. Taylor, "Longest prefix matching using bloom filters," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 397–409, Apr. 2006.

[8] H. Jarollahi, N. Onizawa, V. Gripon, T. Hanyu, and W. J. Gross, "Algorithm and architecture for a multiple-field context-driven search engine using fully-parallel clustered associative memories," in *IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Belfast, U.K., Oct. 20–22, 2014.

[9] V. Gripon and C. Berrou, "Sparse neural networks with large learning diversity," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1087–1096, Jul. 2011.

[10] H. Jarollahi, N. Onizawa, V. Gripon, and W. J. Gross, "Architecture and implementation of an associative memory using sparse clustered networks," in *IEEE Int. Symp. Circuits Syst.*, Seoul, Korea, May 2012, pp. 2901–2904.

[11] H. Jarollahi, N. Onizawa, V. Gripon, and W. J. Gross, "Reduced-complexity binary-weight-coded associative memories," in *Proc. 2013 IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2013, pp. 2523–2527.

[12] H. Jarollahi, N. Onizawa, V. Gripon, and W. J. Gross, "Algorithm and architecture of fully-parallel associative memories based on sparse clustered networks," *J. Signal Process. Syst.* vol. 76, no. 3, pp. 235–247, Sep. 2014.

[13] X. Guo, E. Ipek, and T. Soyata, "Resistive computation: Avoiding the power wall with low-leakage, STT-MRAM based computing," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2010, pp. 371–382.

[14] M. Sharad, D. Fan, and K. Roy, "Ultra low power associative computing with spin neurons and resistive crossbar memory," in *Proc. 50th Annu. Design Automat. Conf.*, 2013, pp. 107:1–107:6.

[15] S. Chaudhuri, W. Zhao, J. O. Klein, C. Chappert, and P. Mazoyer, "High density asynchronous LUT based on non-volatile MRAM technology," in *Proc. Int. Conf. Field Program. Logic Appl.*, Aug. 2010, pp. 374–379.

[16] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," in *Proc. Nat. Acad. Sci.*, Apr. 1982, vol. 79, pp. 2554–2558.

[17] H. Jarollahi, V. Gripon, N. Onizawa, and W. J. Gross, "A low-power content-addressable memory based on clustered-sparse networks," in *Proc. 24th IEEE Int. Conf. Appl.-Specific Syst., Archit. Processors*, Jun. 2013, pp. 305–308.

[18] H. Jarollahi, V. Gripon, N. Onizawa, and W. J. Gross, "Algorithm and architecture for a low-power content-addressable memory based on sparse clustered networks," *IEEE Trans. Very Large Scale (VLSI) Syst.*, 2014, to be published.

[19] D. Kobayashi, Y. Kakehashi, K. Hirose, S. Onoda, T. Makino, T. Ohshima, S. Ikeda, M. Yamanouchi, H. Sato, E. Enobio, T. Endoh, and H. Ohno, "Influence of heavy ion irradiation on perpendicular-anisotropy CoFeB-MgO magnetic tunnel junctions," *IEEE Trans. Nucl. Sci.*, vol. 61, no. 4, pp. 1710–1716, Apr. 2014.

[20] S. Ikeda, K. Miura, H. Yamamoto, K. Mizunuma, H. D. Gan, M. Endo, S. Kanai, J. Hayakawa, F. Matsukura, and H. Ohno, "A perpendicular-anisotropy CoFeBMgO magnetic tunnel junction," *Nature Mater.*, vol. 9, no. 9, pp. 721–724, 2010.

Hooman Jarollahi (S'09) received the B.A.Sc. and M.A.Sc. degrees in electronics engineering from Simon Fraser University, Burnaby, BC, Canada, in 2008 and 2010, respectively. He is currently working toward the Ph.D. degree at the Department of Electrical and Computer Engineering of McGill University, Montreal, QC, Canada.

His research interests are design and hardware implementation of energy-efficient and application-specific VLSI systems, such as associative memories and content-addressable memories. During 2012 and 2013, he was a visiting scholar at the Research Institute of Electrical Communication (RIEC) of Tohoku University, Sendai, Japan.

Mr. Jarollahi was the recipient of Teledyne DALSA award in 2010, for which he presented a patented architecture of a power and area-efficient SRAM.

Naoya Onizawa (M'06) received the B.E., M.E., and D.E. degrees in electrical and communication engineering from Tohoku University, Sendai, Japan, in 2004, 2006, and 2009, respectively.

He is currently an Assistant Professor in Frontier Research Institute for Interdisciplinary Sciences at Tohoku University, Sendai, Japan. He was a postdoctoral fellow at Tohoku University from 2009 to 2011 and at University of Waterloo, Canada in 2011 and at McGill University, Canada from 2011 to 2013. His main interests and activities are in the energy-efficient VLSI design based on asynchronous circuits and multiple-valued circuits, and their applications, such as LDPC decoders, associative memories, and network-on-chips.

Dr. Onizawa received the Best Paper Award in IEEE Computer Society Annual Symposium on VLSI in 2010.

Vincent Gripon received the M.S. degree from École Normale Supérieure of Cachan, Cachan, France, and the Ph.D. degree from Télécom Bretagne, Brest, France.

He is a permanent researcher with Télécom Bretagne (Institut Mines-Télécom), Brest, France. His research interests include information theory, neuroscience, and theoretical and applied computer science. His intent is to propose models of neural networks inspired by information theory principles, what could be called informational neurosciences. He is also the co-creator and organizer of an online programming contest named TaupIC which targets French top undergraduate students.

Noboru Sakimura received the B.E. and M.E. degrees in electrical and electronic engineering from Hiroshima University, Hiroshima, Japan, in 1996 and 1998, respectively.

In 1998, he joined the Silicon System Laboratories, NEC Corporation, Kanagawa, Japan, where he engaged in research and development of wide bandwidth delta-sigma ADCs. Since 2000, he has been working on research and development of large-scale MRAMs at Device Platforms Research Laboratories, NEC Corporation, Kanagawa, Japan. His current interests are in design of high-speed MRAMs.

Tadahiko Sugibayashi received the B.S. and M.S. degrees in material science from Osaka University, Osaka, Japan, in 1984 and 1986, respectively.

He joined NEC Corporation in 1986, where he worked on memory LSI design. He is now engaged in the development of MRAM. He is a Department Manager of Device Platforms Laboratories, NEC Corporation, Kanagawa, Japan.

Mr. Sugibayashi is a member of the Institute of Electronics, Information and Communication Engineers of Japan.

Tetsuo Endoh (M'89) was born in Tokyo, Japan, in 1962. He received the B.S. degree in physics from the University of Tokyo, Tokyo, Japan, in 1987 and the Ph.D. degree in electronic engineering from Tohoku University, Sendai, Japan, in 1995.

He joined Toshiba Corporation in 1987 where he was engaged in research on NAND-type flash memory and advanced CMOS device design at ULSI Research Laboratories, the Research and Development Center, Toshiba Corporation, Kawasaki, Japan. He became a Lecturer at the Research Institute of Electrical Communication, Tohoku University, in 1995, an Associate Professor in 1997, and a Professor in April 2008. He became a Professor at the Center for Interdisciplinary Research, Tohoku University, in May 2008. Currently, he has been a Professor at the Graduate School of Engineering in Tohoku University, since May 2012. He has also been the Deputy Director of the Center for Spintronics Integrated Systems, Tohoku University, since March 2010, and the Director of the Center for Innovative Integrated Electronic Systems, Tohoku University, since October 2012. He has been engaged in research on advanced CMOS device design (e.g., 3DMOS devices, verticalMOS devices, etc.), clean room technology, low-power and high-speed circuit technology and advanced memory (e.g., 3-D memory and memory based on novel principles). He was a member of the "High-performance Low-power Consumption Spin Devices and Storage Systems" program as part of the Research and Development for Next-Generation Information Technology program of MEXT. He is currently engaged in the development of novel nano-LSI with hybrid technology between spin memory devices and extended CMOS technology. He has served as the research leader of the Research and Development of Vertical Body Channel MOSFET and Its Integration Process project as part of the Research of Innovative Material and Process for Creation of Next-generation Electronics Devices program of JST-CREST. He has also served as a research subleader of the Research and Development of Ultra-low Power Spintronics-based Logic VLSIs program as part of the Funding Program for World-Leading Innovative R&D on Science and Technology (FIRST Program).

Dr. Endoh received the LSI IP Design Award in 2001. He was a recipient of the Japanese Journal of Applied Physics (JJAP) Paper Award in 2009. He received the sixth Fellow Award of Japan Society of Applied Physics in 2012. He also received the 2012 International Conference on Solid State Devices and Materials (SSDM) Paper Award in 2012. He assumed an International Collaboration Fellow of Sendai city in 2012. He is a Fellow of the Japan Society of Applied Physics (JSAP). He is a member of the Institute of Electronics, Information, and Communication Engineers (IEICE).

Hideo Ohno (M'82) received the Ph.D. degree from the University of Tokyo, Tokyo, Japan, in 1982.

He is Director of Center for Spintronics Integrated Systems Tohoku University, Director and Professor of Laboratory for Nanoelectronics and Spintronics, Research Institute of Electrical Communication, Tohoku University, and Principal Investigator and Professor of WPI Advanced Institute for Materials Research, Tohoku University. His current research interests include physics and applications of spin-related phenomena in semiconductor and in metal-based nanostructures.

Prof. Ohno received the IBM Japan Science Award (1998), the IUPAP Magnetism Prize (2003), Japan Academy Prize (2005), Presidential Prize for Research Excellence, Tohoku University (2005) and the 2005 Agilent Technologies Europhysics Prize. He has been a Fellow of the Institute of Physics (IOP) since 2004, an Honorary Professor of Institute of Semiconductors, Chinese Academy of Sciences since 2006, a Fellow of the Japan Society of Applied Physics (JSAP) since 2007, and a Fellow of American Physical Society (APS) since 2012. Tohoku University appointed him as a Distinguished Professor. The IEEE Magnetics Society named him for the Distinguished Lecturer for 2009. He was recently awarded the Thomson Reuters Citation Laureate (2011), the JSAP Outstanding Achievement Award, and IEEE David Sarnoff Award (2012).

Takahiro Hanyu (S'87–M'89–SM'12) received the B.E., M.E., and D.E. degrees in electronic engineering from Tohoku University, Sendai, Japan, in 1984, 1986, and 1989, respectively.

He is currently a Professor in the Research Institute of Electrical Communication, Tohoku University, Sendai, Japan. His general research interests include nonvolatile logic circuits and their applications to ultra-low-power and/or PVT-variation-free VLSI processors, and multiple-valued current-mode circuit and its application to power-aware asynchronous network-on-chip systems.

Dr. Hanyu received the Sakai Memorial Award from the Information Processing Society of Japan in 2000, the Judge's Special Award at the ninth LSI Design of the Year from the Semiconductor Industry News of Japan in 2002, the APEX Paper Award of Japanese Society of Applied Physics in 2009, the Excellent Paper Award of IEICE, Japan, in 2010, Ichikawa Academic Award in 2010, and the Best Paper Award at IEEE Computer Society International Symposium on VLSI 2010.

Warren J. Gross (SM'10) received the B.A.Sc. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1996, and the M.A.Sc. and Ph.D. degrees from the University of Toronto, Toronto, ON, Canada, in 1999 and 2003, respectively.

Currently, he is an Associate Professor with the Department of Electrical and Computer Engineering, McGill University, Montréal, QC, Canada. His research interests are in the design and implementation of signal processing systems and custom computer architectures.

Dr. Gross is currently Chair of the IEEE Signal Processing Society Technical Committee on Design and Implementation of Signal Processing Systems. He has served as Technical Program Co-Chair of the IEEE Workshop on Signal Processing Systems (SiPS 2012) and as Chair of the IEEE ICC 2012 Workshop on Emerging Data Storage Technologies. Dr. Gross served as Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING. He has served on the Program Committees of the IEEE Workshop on Signal Processing Systems, the IEEE Symposium on Field-Programmable Custom Computing Machines, the International Conference on Field-Programmable Logic and Applications and as the General Chair of the 6th Annual Analog Decoding Workshop. He is a licensed Professional Engineer in the Province of Ontario.