ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

Algorithm and Architecture for a Low-Power Content Addressable Memory Based On Sparse Compression Technique

S.Muthukrishnan[1],R.Karthikeyan[2], T.Janani[3]
HOD-ECE, Sri Eshwar College of Engineering, Kinathukadavu, Coimbatore[1]
Chief Executive, Vasantha Advanced Systems, Coimbatore[2]
PG Scholar, Sri Eshwar College of Engineering, Kinathukadavu, Coimbatore[3]

Abstract: We propose an extended versions are presented that elaborates the effect of the design's degrees of freedom, and the effect on non-uniformity of input patterns on energy consumption and the performance. The proposed architecture is based on a recently refined sparse clustered networks using binary connections that on-average eliminates most of the parallel comparisons performed during a search. Given an input tag, the proposed architecture computes a few possibilities for the location of the matched tag and performs the comparisons on them to locate a single valid match. And also by using a reordered overlapped search mechanism, most mismatches can be found by searching a few bits of a search word. Following a selection of design parameters, such as the number of CAM entries, the energy consumption and the search delay of the proposed design are 8%, and 26% of that of the conventional NAND architecture, respectively, with a 10% area overhead.

Keywords: Associative memory, content-addressable memory (CAM), low-power computing, sparse clustered networks (SCNs).

I. INTRODUCTION

A content addressable memory (CAM) is a type of memory that can be accessed using its contents rather than an explicit address. In order to access a particular entry in such memories, a search data word is compared against previously stored entries in parallel to find a match. Each stored entry is associated with a tag that is used in the comparison process. Once a search data word is applied to the input of a CAM, the matching data word is retrieved within a single clock cycle if it exists. This prominent feature makes CAM a promising candidate for applications where frequent and fast look-up operations are required, such as in translation look-aside buffers (TLBs) network routers database accelerators, image processing, parametric curve extraction, Hough transformation, Huffman coding/decoding virus detection, Lempel–Ziv compression and image coding. Due to the frequent and parallel search operations, CAMs consume a significant amount of energy. CAM architectures typically use highly capacitive search lines (SLs) causing them not to be energy efficient when scaled. For example, this power inefficiency has constrained TLBs to be limited to no more than 512 entries in current processors. In Hitachi SH-3 and Strong ARM embedded processors, the fully associative TLBs consume about 15% and 17% of the total chip power.

Consequently, the main research objective has been focused on reducing the energy consumption without compromising the throughput. Energy saving opportunities have been discovered by employing either circuit-level techniques, architectural-level techniques, or the code sign of the two some of which have been selected. Although dynamic CMOS circuit techniques can result in low-power and low-cost CAMs, these designs can suffer from low noise margins, charge sharing, and other problems.

A new family of associative memories based on sparse clustered networks (SCNs) has been recently introduced and implemented using field-programmable gate arrays (FPGAs) . Such memories make it possible to store many short messages instead of few long ones as in the conventional Hopfield networks with significantly lower level of computational complexity. Furthermore, a significant improvement is achieved in terms of the number of information bits stored per memory bit (efficiency). In this paper, a variation of this approach and a corresponding

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

architecture are introduced to construct a classifier that can be trained with the association between a small portion of the input tags and the corresponding addresses of the output data. The term CAM refers to binary CAM (BCAM) throughout this paper. Originally included in preliminary results were introduced for an architecture with particular parameters conditioned on uniform distribution of the input patterns.

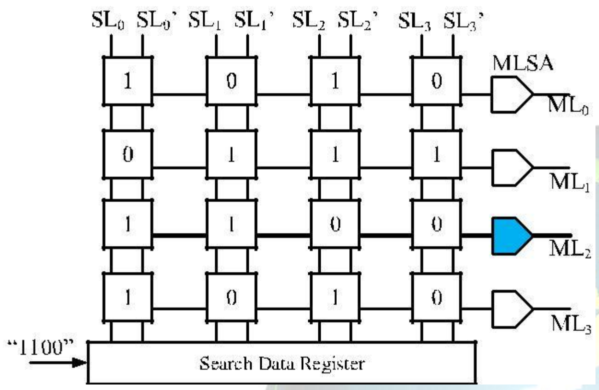


Fig. 1. Simple example of a 4 × 4 CAM array consisting of the CAM cells, MLs, sense amplifiers, and differential SLs.

The rest of this section is organized as follows. Section II describes basic operation of the CAM. In Section III, some of the recent research works related to this area are summarized. In Section IV, the proposed associativity algorithm is introduced. Section V describes the hardware architecture followed by Section VI with the simulation results.

II. CAM REVIEW

In a conventional CAM array, each entry consists of a tag that, if matched with the input, points to the location of a data word in a static random access memory (SRAM) block. The actual data of interest are stored in the SRAM and a tag is simply a reference to it. Therefore, when it is required to search for the data in the SRAM, it suffices to search for its corresponding tag. Consequently, the tag may be shorter than the SRAM-data and would require fewer bit comparisons.

An example of a typical CAM array, consisting of four entries having 4 bits each, is shown in Fig. 1. A search data register is used to store the input bits. The register applies the search data on the differential SLs, which are shared among the entries. Then, the search data are compared against all of the CAM entries. Each CAM-word is attached to a common match line (ML) among its constituent bits, which indicates, whether or not, they match with the input bits. Since the MLs are highly capacitive, a sense amplifier is typically considered for each ML to increase the performance of the search operation.

As an example, in TLBs, the tag is the virtual page number (VPN), and the data are the corresponding physical page number (PPN). A virtual address generated by the CPU consists of the VPN, and a page offset. The page offset is later used along with PPN to form the physical address. Since most TLBs are fully associative, in order to find the corresponding PPN, a fully parallel search among VPNs is conducted for every generated virtual address.

A BCAM cell is typically the integration of a 6-transistor (6T) SRAM cell and comparator circuitry. The comparator circuitry is made out of either an XNOR or an XOR structure, leading to a NAND-type or a NOR-type operation, respectively. The selection of the comparing structure depends on the performance and the power requirements, as a NAND-type operation is slower and consumes less energy as opposed to that of a NOR type.

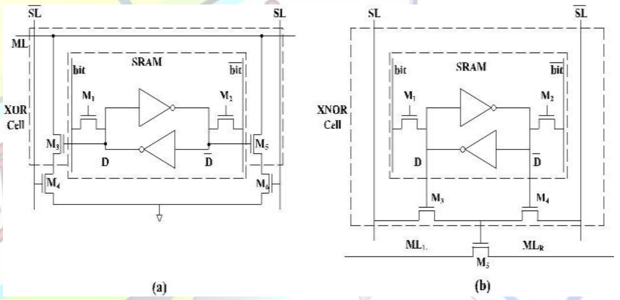


Fig. 2.Classical BCAM cell type(a)10T NOR(b)9T NAND

The schematic of two types of typical BCAM cells are shown in Fig. 2. In a NAND-type CAM, the MLs are precharged high during the precharge phase. During the evaluation phase, in the case of a match, the corresponding ML is pulled down through a series of transistors [M_5 in Fig. 2(b)] performing a login NAND in the comparison process. In a NOR-type CAM [Fig. 2(a)], the MLs are also precharged high during the precharge phase. However, during the evaluation phase, all of the MLs are pulled down unless there is a matched entry such that the pull-down paths $M_3 - M_4$ and $M_5 - M_6$ are disabled. Therefore, a NOR-type CAM has a higher switching activity compared with that of a NAND type since there are typically more mismatched entries than the matched ones.

III. Related Work

Energy reduction of CAMs employing circuit-level techniques are mostly based on the following strategies: 1) reducing the SL energy consumption by disabling the precharge process of SLs when not necessary and 2) reducing the ML precharging, for example, by segmenting the ML, selectively precharging the first few segments and then propagating the precharge process if and only if those first segments match. This segmentation strategy increases the delay as the number of segments is increased. A hybrid-type CAM integrates the low-power feature of NAND type with the high-performance NOR is type while similar to selective precharging method, the ML segmented into two portions. The high-speed CAM designed in 32-nm CMOS achieves the cycle time of 290ps using a swapped CAM cell that reduces the search delay while requiring a larger CAM cell (11-transistors) than a conventional CAM cell [9-transistors (9T)] used in SCN-CAM. A high-performance AND-type match-line scheme is proposed in, where multiple fan-in AND gates are used for low switching activity along with segmented-style match-line evaluation to reduce the energy consumption.

In the bank-selection architecture the CAM array is divided into B equally partitioned banks that are activated based on the value of added bits of length $\log 2(B)$ to the search data word. These extra bits are decoded to determine, which banks must be selected. This architecture was considered at first to reduce the silicon area by sharing the comparison circuitry between the blocks but was later considered for power reduction as well. The drawback of this architecture is that the banks can overflow since the length of the words remains the same for all the banks. For example, let us consider a 128k-entry CAM that incorporates 60-bit words and one additional bank-select bit such that two banks result with 64k entries each.

Therefore, each bank can have 2^{60} possibilities causing an overflow probability that is higher compared with when not banked. This overflow would require extra circuitry that reduces the power saving opportunity since as a result multiple banks are activated concurrently.

The precomputation-based CAM (PB-CAM) architecture divides the comparison process and the circuitry into two stages. First, it counts the number of ones in an input and then compares the result with that of the entries using an additional CAM circuit that has the number of ones in the CAM-data previously stored. This activates a few MLs and deactivates the others. In the second stage, a modified CAM hierarchy is used, which has reduced complexity, and has only one pull-down path instead of two compared with the conventional design. The modified architecture only considers 0 mismatches instead of full comparison since the 1s have already been compared. The number of comparisons can be reduced to $M \times \_\log (N + 2) \_ + (M \times N)/ (N + 1)$ bits, where M is the number of entries in the CAM and N is the number of bits per entry. In the proposed design, we demonstrate how it is possible to reduce the number of comparisons to only N bits. Furthermore, in PB-CAM, the increase of the tag length affects the energy consumption, the delay, and also complicates the precomputation stage.

IV. SCN-Cam Algorithm

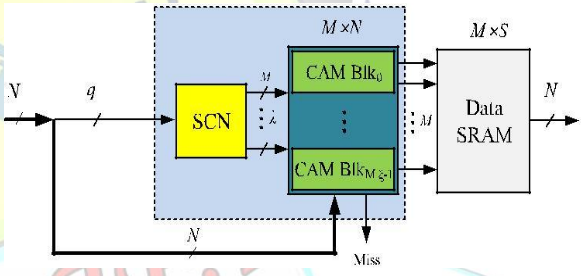


Fig. 3. Top level block diagram of SCN-CAM.

The CAM array is divided into $M/\zeta - 1$ sub-blocks that can be independently activated for comparison. The compare-enable signals are generated by the SCN-based classifier.

As shown in Fig. 3, the proposed architecture (SCN-CAM) consists of an SCN-based classifier, which is connected to a special-purpose CAM array. The SCN-based classifier is at first trained with the association between the tags and the address of the data to be later retrieved. The proposed CAM array is based on a typical architecture, but is divided into several sub-blocks that can be compare-enabled independently. Therefore, it is also possible to train the network with the association between the tag and each CAM sub-block if the number of desired sub-blocks is known. However, in this paper, we focus on a generic architecture that can be easily optimized for any number of CAM sub-blocks. Once an input tag is presented to the SCN-based classifier, it predicts which CAM sub-block(s) need to be compare-enabled and thus saves the dynamic power by disabling the rest. Disabling a CAM sub-block avoids

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

charging its highly capacitive SLs, while applying the search data, and also turns the precharge path off for the MLs.

A. SCN-Based Classifier

SCN-Based Classifier is used for either training or decoding purposes, the input tag is reduced in length to *q* bits, and then divided into *c* equally.

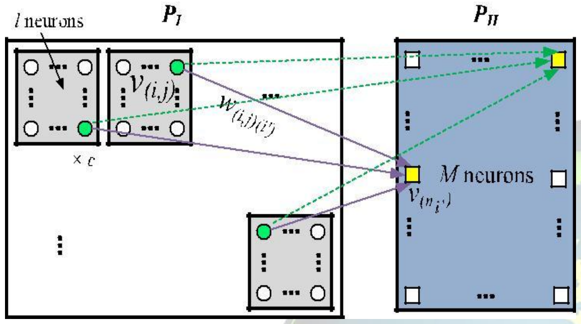


Fig. 4. Representation of the proposed SCN-CAM

As shown in Fig. 4, an SCN-based classifier consists of two parts: 1) P_I and 2) P_{II}. The neurons in P_I are binary, correspond to the input tags, and are grouped into *c* equally sized clusters with *l* neurons in each. Processing of an input tag in the SCN-based classifier is for either of the two situations: training or decoding. In this classifier, either for training or decoding purposes, the input tag is reduced in length to *q* bits, and then divided into *c* equally sized partitions of length κ bits each. Each partition is then mapped to the index of a neuron in its corresponding cluster in P_I, using a direct binary-to-integer mapping from the tag portion to the index of the neuron to be activated. Thus, $l = 2^\kappa$. If *l* is a given parameter, the number of clusters is calculated to be $c = q/\log_2(l)$.

Network Training: The binary values of the connections in the SCN-based classier indicate associations of the input tags and the corresponding outputs. The connection values are set during the training process, and are stored in a memory module such that they can later be used to retrieve the address of the target data. A connection has a value 1 when there exists an association between the corresponding neuron.

Network Update: When an update is requested in SCN-CAM, retraining the entire SCN-based classifier with the entries is not required. The new entry can therefore be added by adding new connections while keeping the previous connections for other entries in the network.

Tag Decoding: Once the SCN-based classifier has been trained, the ultimate goal after receiving the tag is to determine which neuron(s) in should be activated based on the given *q* bits of the tag. This process is called decoding in which the connection values are recalled from the memory. The decoding process is divided into two steps.

1) An input tag is reduced in length to *q* bits and divided into *c* equally sized partitions. The *q* bits can be selected within the tag bits in such a way to reduce the correlation.
2) *Local Decoding (LD):* A single neuron per cluster in P_I is activated using a direct binary-to-integer mapping from the tag portion to the index of the neuron to be activated.

B. Tag-Length Reduction

Given the input tags, the number of bits in the reduced-length tag, *q*, determines the number of possible ambiguities in P_{II}. The generated ambiguities can be corrected with additional comparisons to find the exact match in the CAM. Therefore, no errors are produced in determining the matched result(s). On the other hand, no length reduction leads to the generation of no ambiguities, but a higher level of hardware complexity in the SCN-based classifier, since more neurons are required.

C. Data Distribution

The number of ambiguities, generated in P_{II} is dependent on the correlation factor of the tag pattern that is the number of similar repeating bits in the subset of tags.

V. Circuit Implementation

A top-level block diagram of the implementation of SCN-CAM is shown in Fig. 3. It shows how the SCN-based classifier is connected to a custom-designed CAM array, where an example pertaining to the operation of a 4-bit CAM is demonstrated. A 10- transistor (10T) NOR-type CAM with NOR-type ML architecture was used. The conventional NAND and NOR-type CAM architectures were also implemented for comparison purposes.

In order to implement a circuit that can elaborate the benefit of the proposed algorithm, a set of design points were selected among 15 different parameter sets with the common goal of discovering the minimum energy consumption per search, while keeping the silicon-area overhead and the cycle time reasonable.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

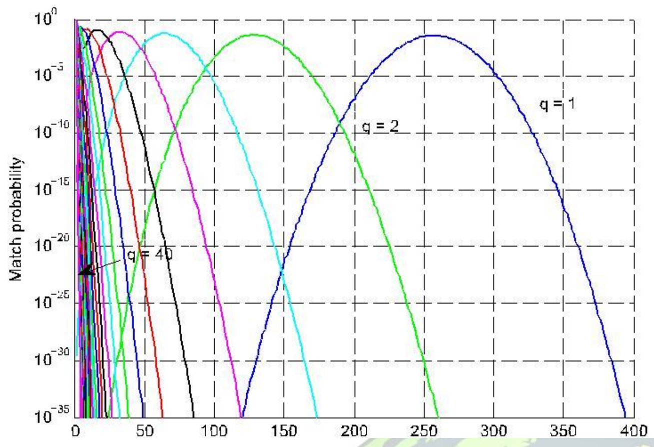


Fig. 5. Relationship between the length of the truncated tag (*q*), the number of matched entries in SCN-CAM (*λ*), and the estimated matching probability (*P(λ)*) for *M* = 512.

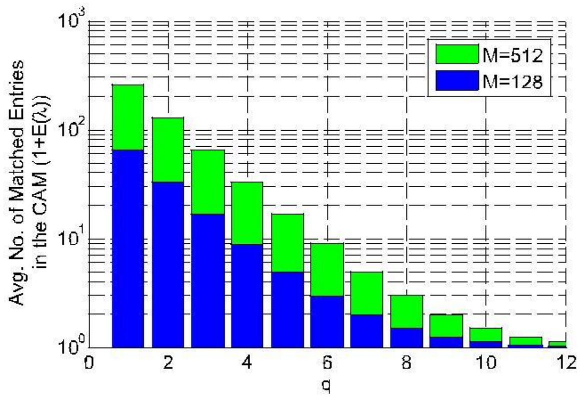


Fig. 6. Comparisons in SCN-CAM versus the number of bits in the reduced-length tag.

Fig. 5 shows simulations results on how it is possible to reduce the estimated number of required comparisons by increasing *q*. It is interesting to note that the number of clusters in P_I does not affect the number of neurons.

A drawback of such methods, unlike SCN-CAM, is that as the length of the tags is increased, the cycle time and the circuit complexity of the precomputation stage are dramatically increased. For design selections in Table I, this overhead is only 3.4% compared with that of the conventional CAM. The silicon area of the SCN-based classifier can be estimated by the area of the decoders, SRAM arrays, the precharging devices, interconnections and the standard cells.

TABLE I REFERENCE DESIGN PARAMETERS

	Parameter	Value
SCN	M	512
	N	128
	n	10
	ζ	8
	β	64
	$E(\lambda)$	1
	q	9
	c	3
	l	8
CAM	CAM type	XOR
	ML Arch.	NOR
	Supply Voltage	1.0V
	Technology	65 nm

Fig. 6 shows simulation results on expected value of the number of SCN-CAM entries and number of bits in the reduced length code.

A. SCN-CAM: Architecture of SCN-Based Classifier

The SCN-based classifier in SCN-CAM architecture generates the compare-enable signal(s) for the CAM sub-blocks attached to it. The architecture of the SCN-based classifier is shown in Fig. 7. It consists of c κ -to-l one-hot decoders, c SRAM modules of size $l \times M$ each, M $c-$input AND gates, M/ζ ζ $-$input OR gates, and M/ζ 2-input NAND gates. Each row of an SRAM module stores the connections from one tag to its corresponding output neuron. Each reduced-length tag of length q is thus divided into c subtags of κ bits each, where each subtag creates the row address of each SRAM module.

1) Training: During the training process, the SRAM mod-ules store the connection values between the input tags and their corresponding outputs, which are later used in the decod-ing process.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

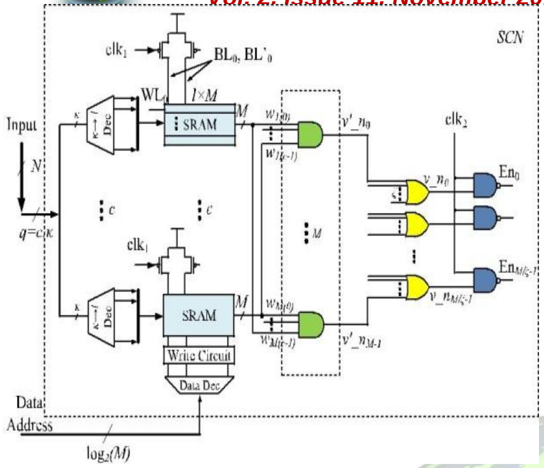


Fig. 7.Simplified SCN-CAM architecture

B. SCN-CAM: CAM Architecture

In order to exploit the prominent feature of the SCN-based associative memory, a conventional CAM array is divided into sufficient number of compare-enabled sub-blocks such that: 1) the number of sub-blocks are not too many to expand the layout and to complicate the interconnections and 2) the number of sub-blocks should not be too few to be able to exploit to energy-saving opportunity with the SCN-based classifier.

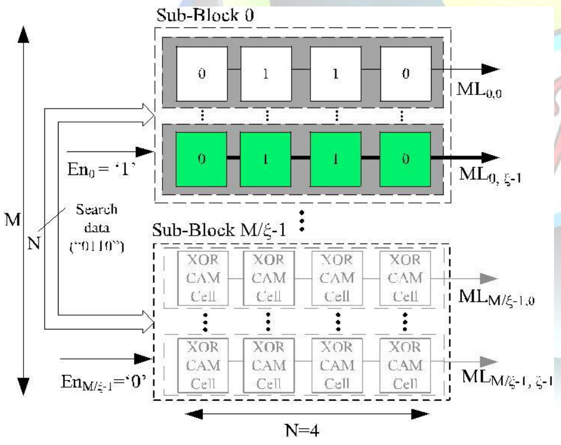


Fig .8.Simplified array organization of the proposed CAM architecture.

Fig .8 shows the simplified array organization for CAM architecture. For example when N = 4, search data word is 0110 and En_0 = 1. The sub-block compare-enable signals are generated by the SCN-based classifier.

VI. CIRCUIT EVALUATION

A complete circuit for SCN-CAM was implemented and simulated using HSPICE and TSMC 65-nm CMOS technology according to design parameters, including full dimensions of CAM arrays, SRAM arrays, logical gates, and extracted parasitics from the wires in the physical layout.

Fig. 9 shows the cycle time is measured by the maximum reliable frequency of operation in the worst-case cycle time (SS) scenario. The required silicon area of SCN-CAM is estimated to be 10.1% larger than that of the conventional NAND-type counterpart mainly due to the existence of the gaps between the SRAM blocks of the SCN-based classifier.

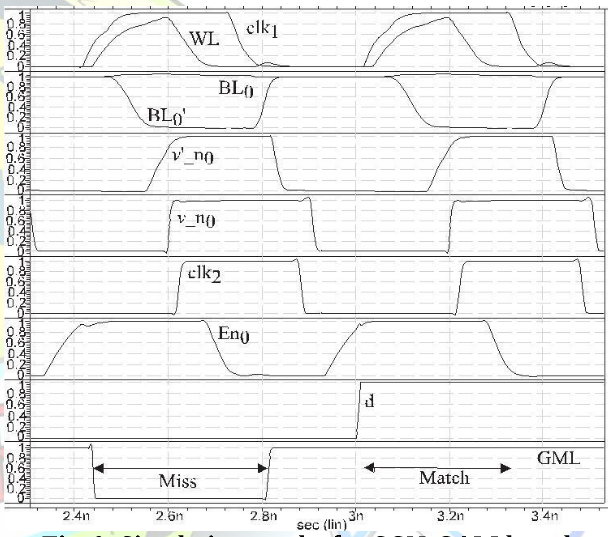
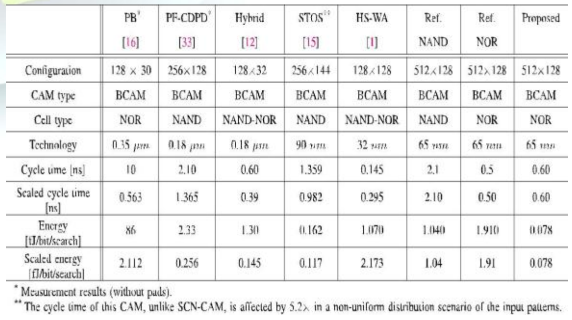


Fig. 9. Simulation results for SCN-CAM based on reference design parameters table

TABLE II Simulation Results



	PB* [16]	PF-CDPD* [33]	Hybrid [12]	STOS** [15]	HS-WA [1]	Ref. NAND	Ref. NOR	Proposed
Configuration	128 × 30	256×128	128×32	256×144	128×128	512×128	512×128	512×128
CAM type	BCAM	BCAM	BCAM	BCAM	BCAM	BCAM	BCAM	BCAM
Cell type	NOR	NAND	NAND-NOR	NAND	NAND-NOR	NAND	NOR	NOR
Technology	0.35 μm	0.18 μm	0.18 μm	90 nm	32 nm	65 nm	65 nm	65 nm
Cycle time [ns]	10	2.10	0.60	1.359	0.145	2.1	0.5	0.60
Scaled cycle time [ns]	0.563	1.365	0.39	0.982	0.295	2.10	0.50	0.60
Energy [fJ/bit/search]	86	2.33	1.30	0.162	1.070	1.040	1.910	0.078
Scaled energy [fJ/bit/search]	2.112	0.256	0.145	0.117	2.173	1.04	1.91	0.078

* Measurement results (without pads).
** The cycle time of this CAM, unlike SCN-CAM, is affected by 5.2× in a non-uniform distribution scenario of the input patterns.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 2, Issue 11, November 2015

VII. Conclusion

The proposed architecture (SCN-CAM) employs a novel associativity mechanism based on a recently developed family of associative memories based on SCNs.

SCN-CAM is suitable for low-power applications, where frequent and parallel look-up operations are required. SCN-CAM employs an SCN-based classifier, which is connected to several independently compare-enabled CAM sub-blocks, some of which are enabled once a tag is pre-sent to the SCN-based classifier. By using independent nodes in the output part of SCN-CAM's training network, simple and fast updates can be achieved without retraining the network entirely. With optimized lengths of the reduced-length tags, SCN-CAM eliminates most of the comparison operations given a uniform distribution of the reduced-length inputs. Depending on the application, non-uniform inputs may result in higher power consumptions, but does not affect the accuracy of the final result. In other words, a few false-positives may be generated by the SCN-based classifier, which are then filtered by the enabled CAM sub-blocks. Therefore, no false-negatives are ever generated.

Conventional NAND-type and NOR-type architectures were also implemented in the same process technology to com-pare SCN-CAM against, along with other recently developed CAM architectures. It has been estimated that for a case study design parameter, the energy consumption and the cycle time of SCN-CAM are 8.02%, and 28.6% of that of the conventional NAND-type architecture, respectively, with a 10.1% area overhead. Future work includes investigating sparse compression techniques for the matrix storing the connections in order to further reduce the area overhead.

References

[1]. A. Agarwal *et al.*, "A 128×128 b high-speed wide-and match-line content addressable memory in 32 nm CMOS," in *Proc. ESSCIRC*, Sep. 2011, pp. 83–86.

[2]. Y.-J. Chang and M.-F. Lan, "Two new techniques integrated for energy-efficient TLB design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 1, pp. 13–23, Jan. 2007.

[3]. H. Chao, "Next generation routers," *Proc. IEEE*, vol. 90, no. 9, pp. 1518–1558, Sep. 2002.

[4]. N.-F. Huang, W.-E. Chen, J.-Y. Luo, and J.-M. Chen, "Design of multi-field IPv6 packet classifiers using ternary CAMs," in *Proc. IEEE Global Telecommun. Conf.*, vol. 3. 2001, pp. 1877–1881.

[5]. M. Meribout, T. Ogura, and M. Nakanishi, "On using the CAM concept for parametric curve extraction," *IEEE Trans. Image Process.*, vol. 9, no. 12, pp. 2126–2130, Dec. 2000.

[6]. M. Nakanishi and T. Ogura, "A real-time CAM-based Hough transform algorithm and its performance evaluation," in *Proc. 13th Int. Conf. Pattern Recognit.*, vol. 2. Aug. 1996, pp. 516–521.

[7]. L.-Y. Liu, J.-F. Wang, R.-J. Wang, and J.-Y. Lee, "CAM-based VLSI architectures for dynamic Huffman coding," *IEEE Trans. Consum. Electron.*, vol. 40, no. 3, pp. 282–289, Aug. 1994.

[8]. C.-C. Wang, C.-J. Cheng, T.-F. Chen, and J.-S. Wang, "An adaptively dividable dual-port BiTCAM for virus-detection processors in mobile devices," *IEEE J. Solid-State Circuits*, vol. 44, no. 5, pp. 1571–1581, May 2009.

[9]. B. Wei, R. Tarver, J.-S. Kim, and K. Ng, "A single chip Lempel–Ziv data compressor," in *Proc. IEEE ISCAS*, May 1993, pp. 1953–1955.

[10]. S. Panchanathan and M. Goldberg, "A content-addressable memory architecture for image coding using vector quantization," *IEEE Trans. Signal Process.*, vol. 39, no. 9, pp. 2066–2078, Sep. 1991.

[11]. T. Juan, T. Lang, and J. Navarro, "Reducing TLB power requirements," in *Proc. Int. Symp. Low Power Electron. Des.*, Aug. 1997, pp. 196–201.

[12]. Y.-J. Chang and Y.-H. Liao, "Hybrid-type CAM design for both power and performance efficiency," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 8, pp. 965–974, Aug. 2008.

[13]. Z. Lei, H. Xu, D. Ikebuchi, H. Amano, T. Sunata, and M. Namiki, "Reducing instruction TLB's leakage power consumption for embedded processors," in *Proc. Int. Green Comput. Conf.*, Aug. 2010, pp. 477–484.

[14]. S.-H. Yang, Y.-J. Huang, and J.-F. Li, "A low-power ternary content addressable memory with Pai-Sigma matchlines," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 10, pp. 1909–1913, Oct. 2012.

[15]. N. Onizawa, S. Matsunaga, V. C. Gaudet, and T. Hanyu, "High-throughput low-energy content-addressable memory based on self-timed overlapped search mechanism," in *Proc. Int. Symp. Asynchron. Circuits Syst.*, May 2012, pp. 41–48.