Indian Journal of Research in Applied Sciences Engineering (IJRASE)
Vol.3.No.1 2015 pp 1-4
available at: www.goniv.com
Paper Received :08-04-2015
Paper Accepted:02-05-2015
Paper Reviewed by: 1. R. Venkatakrishnan 2. R. Marimuthu
Editor : Prof. P.Muthukumar

Low Power and High Performance SRAM-based Architecture for TCAM

Gopinath, PG Scholar
Department of VLSI Design
Department of ECE
ShanmuganathanEngg College
Pudukkottai, India

A.Anand
Asst.Prof
Department of ECE
ShanmuganathanEngg College
Pudukkottai, India

R.Sornalatha
Asst.Prof
Department of ECE
ShanmuganathanEngg College
Pudukkottai, India

ABSTRACT

Ternary content addressable memories (TCAMs) perform high-speed lookup operation but when compared with static random access memories (SRAMs), TCAMs have certain limitations such as low storage density, relatively slow access time, low scalability, complex circuitry, and are very expensive. Thus, can we use the benefits of SRAM by configuring it (with additional logic) to enable it to behave like TCAM? This brief proposes a novel memory architecture, named Z-TCAM, which emulates the TCAM functionality with SRAM. Z-TCAM logically partitions the classical TCAM table along columns and rows into hybrid TCAM subtables, which are then processed to map on their corresponding memory blocks.

Keywords: TCAM, SRAM, High Speed Look-up

I. INTRODUCTION

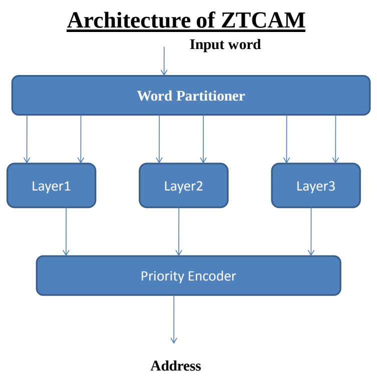
Ternary content addressable memory (TCAM) allows its memory to be searched by contents rather than by an address and a memory location among matches is sent to the output in a constant time. A typical TCAM cell has two static random access memory (SRAM) cells and a comparison circuitry and has the ability to store three states − 0, 1, and *x* where *x* is a don't care state. The *x* state is always regarded as matched irrespective of the input bit. The constant time search of TCAM makes it a suitable candidate in different applications such as network routers, data compression, real-time pattern matching in virus-detection, and image processing [1]. TCAM provides single clock lookup; however, it has several disadvantages compared with SRAM. TCAM is not subjected to the intense commercial competition found in the RAM market [2]. TCAM is less dense than SRAM. The comparator's circuitry in TCAM cell adds complexity to the TCAM architecture. The extra logic and capacitive loading due to the massive parallelism lengthen the access time of TCAM, which is 3.3 times longer than the SRAM access time [3]. Inborn architectural barriers also limit the total chip capacity of TCAM. Complex integration of memory and logic also makes TCAM testing very time consuming [1]. Furthermore, the cost of TCAM is about 30 times more per bit of storage than SRAM [4]. RAM is available in a wider variety of sizes and flavors, is more generic and widely available, and enables to avoid the heavy licensing and royalty costs charged by some CAM vendors [5]. CAM devices have very limited pattern capacity and also CAM technology does not evolve as fast as the RAM technology [6]. Field-programmable gate array (FPGA) is used in many applications, for example, in networking systems [7] and [8] owing to several reasons that include its reconfigure-ability, massive hardware parallelism, and rapid prototyping capability. Recent FPGA devices such as Xilinx Virtex-7 [9] provide high clock rate and a large amount of on-chip dual-port memory with configurable word width. Currently, TCAMs are used in networking systems but they are expensive and not scalable with respect to clock rate or circuit area compared with RAMs [10]. The throughput of classical TCAMs is also limited by the relatively low speed of TCAMs [11]. Thus, SRAM- and FPGA-based TCAMs can be used in applications such as in networking chips to achieve high speed and high throughput.

II.LITERATURE SURVEY

We summarize RAM-based solutions for CAM in this section. The methods proposed in [2] and [12] use hashing to build CAM from RAM but these methods suffer from collisions and bucket overflow. If many records have been placed in an overflow area, then a lookup may not finish until many buckets are searched. In [12], when stored keys contain don't care bits in the bit positions used for hashing, then such keys must be duplicated in multiple buckets, which need increased capacity. On the other hand, if the search key contains don't care bits which are taken by the hash function, multiple buckets must be accessed that results in performance degradation. In [2], the performance of the method becomes gracefully degradable as the number of stored elements increases. Furthermore, it emulates binary CAM, not TCAM. Thus, hashing cannot provide deterministic performance owing to potential collisions and is inefficient in handling wildcard. Traditional algorithmic search solutions take multiple clock cycles [11] and also result in inefficient memory utilization [10]. In contrast, Z-TCAM has a deterministic search performance that is independent of data, efficiently handles the wild-cards, and has better memory utilization. The method proposed in [13] combines RAM and CAM to develop the CAM functionality. This approach makes partitions of the conventional TCAM table using some distinguishing bits in CAM entries. But making partitions of totally random data is a very tedious and time consuming job. Because the method uses TCAM as a part of the overall architecture, it brings the intrinsic TCAM disadvantages in the overall architecture of [13] but Z-TCAM is generic and has an easy partitioning scheme. RAM-based CAMs presented in [6] and [14] have an exponential increase in memory size with the increase in number of bits in CAM word, thus making them prohibitive. For instance, if a CAM word has 36 bits, its size would be 236 = 64 GB in [6]. Furthermore, the method in [14] only works on ascended data but in typical CAM applications data are totally random. By arranging the data in ascending order, the original order of entries is disturbed. So, there must be a way to store the original addresses, which is lacked by [14]. If original addresses are considered, the memory and power requirements further increase.

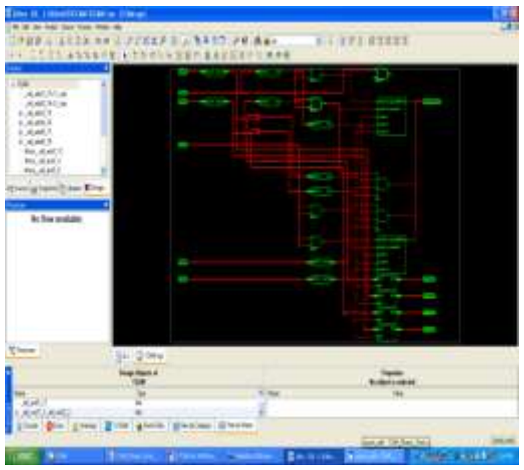
III.PROPOSED METHOD

Hybrid partitioning (HP) is a collective name given to vertical partitioning and horizontal partitioning of the conventional TCAM table. An example of HP is given in Table I. HP partitions conventional TCAM table vertically (columnwise) and horizontally (rowwise) into TCAM subtables, which are then processed to be stored in their corresponding memory units. This processing (data mapping) has been explained in Section IV-A with an example (Table II) to demonstrate the layer architecture of Z-TCAM. Vertical partitioning (VP) implies that a TCAM word of *C* bits is partitioned into *N* subwords; each subword is of *w* bits. VP is used in Z-TCAM to decrease memory size as much as possible. Horizontal partitioning (HrP) divides each vertical partition using the original address range of conventional TCAM table into *L* horizontal partitions. HrP cannot be used alone as it is area, power, and cost hungry but is used to create layers. HP results in a total of $L \times N$ hybrid partitions. The dimensions of each hybrid partition are $K \times w$ where *K* is a subset from original addresses and *w* is the number of bits in a subword. Hybrid partitions spanning the same addresses are in the same layer. For example, HP21 and HP22 span the same address range and are in layer 2.



Layer architecture is shown in Fig. 2. It contains *N* validation memories (VMs), 1-bit AND operation, *N* original address table address memories (OATAMs), *N* original address tables (OATs), *K*-bit AND operation, and a layer priority encoder (LPE). *1-bit* AND *Operation* ANDs the output of all VMs. The output of 1-bit AND operation decides the continuation of a search operation. If the result of 1-bit AND operation is high, then it permits the continuation of a search operation, otherwise mismatch occurs in the corresponding layer.

IV. EXPERIMENTAL RESULTS



Performance Analysis:

Performance Parameters	Experimental Results
Power Consumption	0.045 mW
Current Consumption	0.025 mA

REFERENCES

[1] N. Mohan, W. Fung, D. Wright, and M. Sachdev, "Design techniques and test methodology for low-power TCAMs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 6, pp. 573–586, Jun. 2006.
[2] P. Mahoney, Y. Savaria, G. Bois, and P. Plante, "Parallel hashing memories: An alternative to content addressable memories," in *Proc. 3rd Int. IEEE-NEWCAS Conf.*, Jun. 2005, pp. 223–226.
[3] 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.
[4] D. E. Taylor, "Survey and taxonomy of packet classification techniques," ACM Comput. Surveys, New York, NY, USA: Tech. Rep. WUCSE-2004-24, 2004.
[5] P. Mahoney, Y. Savaria, G. Bois, and P. Plante, "Transactions on highperformance embedded architectures and compilers II," in *Performance Characterization for the Implementation of Content Addressable Memories Based on Parallel Hashing Memories*, P. Stenström, Ed. Berlin, Germany: Springer-Verlag, 2009, pp. 307–325.
[6] S. V. Kartalopoulos, "RAM-based associative content-addressable memory device, method of operation thereof and ATM communication switching system e ploying the same," U.S. Patent 6 097 724, Aug. 1, 2000.
[7] W. Jiang and V. Prasanna, "Scalable packet classification on FPGA," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 9, pp. 1668–1680, Sep. 2012.
[8] M. Becchi and P. Crowley, "Efficient regular expression evaluation: Theory to practice," in *Proc. 4th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, Nov. 2008, pp. 50–59.
[9] Xilinx, San Jose, CA, USA. *Xilinx FPGAs* [Online]. Available: http://www.xilinx.com
[10] W. Jiang and V. K. Prasanna, "Large-scale wire-speed packet classification on FPGAs," in *Proc. ACM/SIGDA Int. Symp. Field Program. Gate Arrays*, 2009, pp. 219–228.
[11] W. Jiang and V. Prasanna, "Parallel IP lookup using multiple SRAMbased pipelines," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, Apr. 2008, pp. 1–14.
[12] S. Cho, J. Martin, R. Xu, M. Hammoud, and R. Melhem, "CA-RAM: A high-performance memory substrate for search-intensive applications," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw.*, Apr. 2007, pp. 230–241.