DESIGN SPACE EXPLORATIONS OF HYBRID-PARTITIONED TCAM (HP-TCAM)

Zahid Ullah, Manish K. Jaiswal, and Ray C.C. Cheung

Department of Electronic Engineering
City University of Hong Kong
83 Tat Chee Avenue, Kowloon Tong, Hong Kong
Emails: {zullah2-c, mkjaiswal2-c}@my.cityu.edu.hk, r.cheung@cityu.edu.hk

ABSTRACT

Even though, TCAM provides search operation in a constant time, 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 expensive costs. Hence, the need for a TCAM architecture arises that can use SRAM (with additional logic) to behave like TCAM. This paper presents the idea of Hybrid-Partitioned, SRAM-based architecture (HP−TCAM), which provides the same functionality as TCAM. We implemented and analyzed an example design of 512×36 HP−TCAM on Xilinx FPGAs with its different design parameters. Energy/bit/search, as an important metric, for the design is 47.13 fJ on Virtex-7 FPGA. Furthermore, we have provided in detail, all the implementation results and power consumption for our designs.

1. INTRODUCTION

Ternary Content Addressable Memory (TCAM) is a special memory that contains large amounts of stored data for simultaneous comparison with input data. The search result of TCAM is the address, among matched addresses, at which input word is matched. TCAM can store 0, 1, and x states, where x is called don't care state, which is always in match case irrespective of the input bit.

The search operation in typical TCAMs has several severe disadvantages. TCAMs cost is about 30 times more (per bit of storage) than SRAM [1]. TCAM is not subject to the very intense commercial competition found in the RAM market [2]. Since TCAM needs comparison circuitry in each cell, it has lower bit density than RAM. Furthermore, comparison circuitry also adds complexity to TCAM architecture. The extra logic and capacitive loading due to the massive parallelism lengthens the access time of TCAMs, which is over 3.3 times longer than SRAM [3]. Furthermore, inborn architectural barriers also limit its total chip capacity. Complex integration of memory and logic also makes TCAM testing very time consuming [4]. RAM is available in a wider variety of sizes and flavours, and is more generic and widely available [5]. CAMs have a very limited pattern capacity and the cost per bit of RAM continuously decreases, as opposed to CAM technology [6].

FPGAs are used in many applications because of several reasons that include its reconfigure-ability, massive hardware parallelism, and rapid prototyping capability. FPGA devices such as Xilinx Virtex-6 and Virtex-7 provide high clock rate and large amounts of on-chip dual-port memory with configurable word width. Xilinx Virtex-7 2000T FPGA is ideally suited for the ASIC prototyping and emulation marketplace. Therefore, FPGA is a natural choice for implementing CAMs.

As shown in the literature, the development of CAM strongly demands to build up TCAM from conventional memory and use FPGA as an implementation technology. Recently, in [7] an SRAM-based TCAM (HP−TCAM) has been presented. The current work proposes and demonstrates the HP−TCAM by implementing it on Xilinx FPGAs, for its different design parameters (N: # of vertical partitions and L: # of horizontal partitions). We have thoroughly reported all the implementation details of HP−TCAM.

2. PREVIOUS WORK

Memory architectures in [6] and [8] emulate CAM functionality but there is an exponential increase in the memory size with the increase in number of bits in CAM word, thus making them prohibitive. Furthermore, the method in [8] only works on ascended data but in typical CAM applications data is totally random. Alternative in [9] integrates CAM and RAM to get overall CAM functionality, thus inheriting the instinctive disadvantages of CAM. The method makes groups by finding some distinguishing bits in CAM entries. In typical TCAM applications, data is entirely random, thus making groups are very time consuming.

An SRAM-based work has been presented in [10], which demonstrates the vertical partitioning emulation of TCAM with SRAM. RAM-based CAMs in [2] and [11] are based on hashing technique, thus having drawbacks of collisions and bucket overflow and therefore, cannot provide deterministic

Table 1. TCAM Table and its Hybrid Partitions (HP).

Address	Hybrid Partitions		Layer
0	$xx01$	$11xx$	
1	1101 HP_{11}	1100 HP_{12}	1
2	0111	0111	
3	1101 HP_{21}	$111x$ HP_{22}	2

search performance. In [2], the performance of the method becomes gracefully degradable as the number of stored elements increases. Traditional algorithmic search solutions need throughput of multiple clock cycles and also result in inefficient memory utilization [12].

3. ARCHITECTURE OF HP-TCAM

HP−TCAM exploits hybrid partitioning (HP) to divide conventional TCAM table into vertical (columns) and horizontal (rows) partitions. Hybrid partitioning is visualized in Table 1. Vertical partitioning (VP) divides W-bits TCAM word into N sub-words; each of which is of w-bits. Horizontal partitioning (HrP) divides each vertical partition using the original address range of conventional TCAM table into L horizontal partitions. Thus, HP results in total of L×N hybrid partitions. The dimensions of each hybrid partition are K × w where K is a sub-set from original addresses and w is the number of bits in a sub-word. Every sub-word in all hybrid partitions is mapped to its corresponding bit in its corresponding BPT and original address(s) of the sub-word are mapped to its/their corresponding bit(s) in the corresponding APT, respectively. Hybrid partitions spanning the same address range are in the same layer.

3.1. Layer Architecture

The layer architecture of HP−TCAM is shown in Fig. 1, the main components of which are N bit position tables (BPTs), N address position tables (APTs), N APT address generators (APTAGs), a local priority encoder (LPE), and ANDing operations (1−bit AND and K−bit AND).

In BPT, 2^w of memory bits are arranged into 2^{w-b} rows. Each row consists of 2^b data bits (which can be two or more), and last index (LI) (of length w+1 bits). The w−b most significant bits (MSBs) of sub-word act as an address, BPT address (BPTA), to BPT. The b less significant bits (LSBs), bit position indicator (BPI), in the sub-word are used to indicate a particular bit position in the row selected by BPTA. If the indicated bit position is high, sub-word is present otherwise absent. LI of a row is set to the total number of bits set in all previous rows reduced by 1.

1−bit AND operation ANDs the output of all BPTs. If the result of 1−bit AND operation is high, then it permits the search operation, otherwise mismatch occurs in the

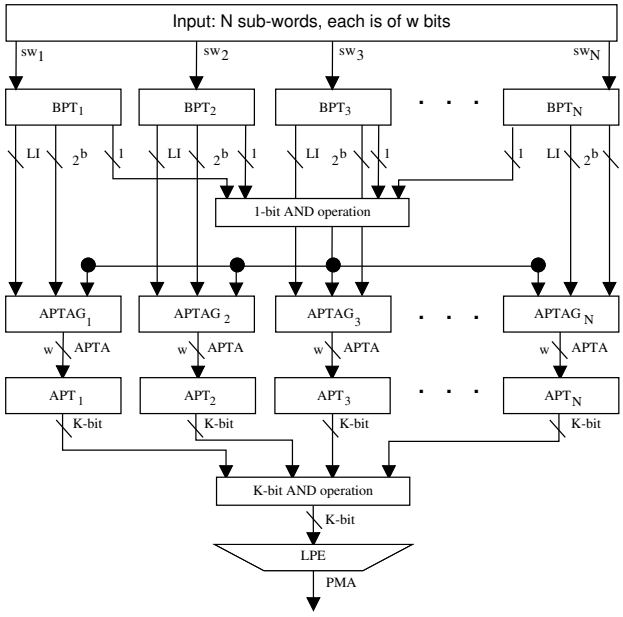

Fig. 1. Architecture of a layer of HP−TCAM.

corresponding layer. APTAG receives two values from its corresponding BPT, LI and 2^b. APTAG has $1's$ counter and adder. The $1's$ counter counts the number of $1's$ in the selected row of BPT up to the indicated bit position inclusive and then forwards the result to adder, which then adds the output of the $1's$ counter and LI of the selected row. The summation results in APTA.

Dimensions of each APT are $2^w \times$ K where w is the number of bits in a sub-word, 2^w represents number of rows, and K is the number of bits in each row where each bit represents an original address. K-bit AND operation ANDs bit-wise the output of all APTs. The result of K-bit AND operation has the potential match address (PMA). Since we emulate TCAM, and as in TCAM multiple matches may occur in a layer, LPE is used to select PMA among the outputs of K-bit AND operation.

3.2. Overall Architecture

The architecture of HP−TCAM is shown in Fig. 2, which is composed of L layers and a global priority encoder (GPE). GPE is used to select match address (MA) among multiple matches. GPE receives PMAs from all layers and outputs MA among PMAs.

3.3. Searching an entry in HP-TCAM

3.3.1. Layer Searching

Algorithm 1 illustrates the layer search operation. The layer receives N sub-words simultaneously, which are then applied to their respective BPTs in parallel. Each sub-word

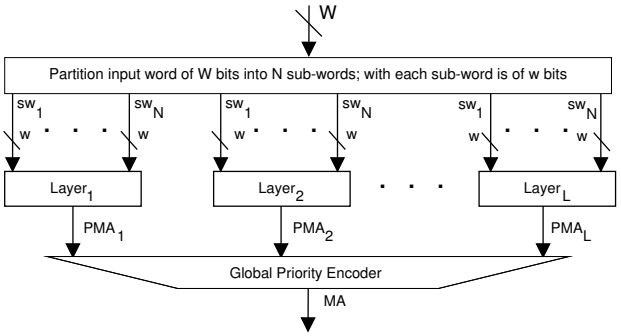

Fig. 2. Architecture of HP−TCAM.

is then divided into its BPTA and BPI, which occurs for all sub-words in parallel. Then, BPTA reads out a row and BPI accesses a particular bit in it. If the accessed bit is high, then sub-word is present, otherwise absent. Still we don't know at which address the sub-word is present. All BPTAs and BPIs perform their respective functions concurrently.

Algorithm 1 Pseudo-code for search operation in a layer of HP−TCAM

Input: N sub-words
Output: Potential match address (PMA)

1: → Apply N sub-words
2: → Apply all sub-words simultaneously to their BPTs
3: → Divide each sub-word into its BPTA and BPI
→ in its respective BPT: occurs in parallel in all BPTs
4: → Read bit position in BPT with BPI in a row selected
→ by BPTA: occurs in parallel in all BPTs
5: **if** (the accessed bits by all BPIs in their BPTs are high) **then**
6: →Continue search operation
7: →**a.** APTAGs generate their APTAs in parallel
8: →**b.** Read all APTs simultaneously
9: →**c.** AND bit-wise all K-bits rows
10: →**d.** Select PMA/mismatch occurs
11: **else**
12: → Mismatch occurs
13: **end if**

Search will continue, if the accessed bits by all BPIs in their BPTs are high. If any of the accessed bits is low, searching will stop in that layer. This explanation is equivalent to 1-bit AND operation in Fig. 1. All APTAGs generate their respective APTAs simultaneously. All APTAs read out K-bit rows from their corresponding APTs simultaneously, which are then bit-wise ANDed. LPE receives the K-bit AND operation result and outputs PMA. Mismatch of the input word can occur at two places in a layer search. 1:) if the result of 1-bit AND operation is 0, and 2:) when none of the bit is high after K-bit AND operation.

3.3.2. Overall Searching

Algorithm 2 shows the overall searching in HP−TCAM. An entry (search key) is applied to HP−TCAM, which is then divided into N sub-words. The sub-words are then searched in all layers in parallel. Algorithm 2 uses algorithm 1 at step 3. After searching, PMAs are available from all layers. GPE selects MA among PMAs; otherwise mismatch occurs.

Algorithm 2 Pseudo-code for search operation in HP−TCAM

Input: Search Key
Output: Match address (MA)

1: → Apply input word
2: → Partition search key (input word) into N sub-words
3: → Search sub-words simultaneously in all layers
4: → Select MA among PMAs/mismatch occurs

4. IMPLEMENTATION OF HP-TCAM

We implemented a real example design of HP−TCAM with different design parameters (L: # of layers and N: # of vertical partitions) for the size of 512×36 on Xilinx Virtex-5, Virtex-6 , and Virtex-7 FPGAs. We have taken four samples of the example design termed as cases. These are Case 1(L = 2, N = 4), Case 2(L = 4, N = 4), Case 3(L = 2, N = 3), and Case 4(L = 4, N = 3).

FPGA implementation results and power consumption for the design are tabulated in Table 2. Total dynamic power consumption for a look-up operation has been measured with $1.0v$ core voltage and 100 MHz clock speed using Xilinx Xpower Analyzer. Then, we computed energy/bit/search (E_{bs}) of the example design using equation 1.

$$E_{bs} = \frac{Power(W)}{Frequency(Hz) \times Total\ Number\ of\ bits} \quad (1)$$

We can see from Table 2 that energy/bit/search has an increase with the increase in memory size and further shows that for the same design case energy/bit/search decreases as the design case moves from Virtex-5 to Virtex-7 because the latest Xilinx FPGAs are power efficient.

We implemented the design cases with different stages of pipeline. Latency of all the design cases is five clock cycles. Each memory read takes one clock cycle, APTAG takes two clock cycles, K−bit AND operation takes one clock cycle, and priority encoder (PE) takes one clock cycle.

With increase in TCAM size, pipeline stages may increase that may result in more cycles for APTAG and K−bit AND operation. Pipeline stages may also increase if higher operating frequency is needed. For larger TCAM structure, there may be greater values for L and N but it is expected that the throughput is not affected by larger size of TCAM.

Table 2. FPGA Implementation Results and Power Consumption of HP−TCAM for Different Design Cases

Cases	FPGAs	BRAMs		SRs	LUTs	Power	Energy	Speed
		36K	18K			(mW)	(fJ)	(MHz)
Size: 512×36								
Case 1 (L = 2, N = 4)	Virtex-5	40	8	1745	4563	154.48	83.81	117.05
	Virtex-6	40	8	1289	3482	101.08	54.83	126.61
	Virtex-7	40	8	1289	3436	86.88	47.13	107.21
Case 2 (L = 4, N = 4)	Virtex-5	48	16	1801	7343	251.49	136.44	100.26
	Virtex-6	48	16	2057	5326	188.33	102.17	118.10
	Virtex-7	48	16	2057	5289	146.69	79.58	136.01
Case 3 (L = 2, N = 3)	Virtex-5	174	12	1019	3782	330.91	179.95	100.23
	Virtex-6	174	12	1115	3053	249.76	135.5	119.50
	Virtex-7	174	12	1115	3038	208.81	113.28	124.92
Case 4 (L = 4, N = 3)	Virtex-5	180	24	2429	6333	422.46	229.19	100.00
	Virtex-6	180	24	1709	4495	280.11	151.96	126.50
	Virtex-7	180	24	1709	4501	248.90	135.03	110.85

5. CONCLUSIONS AND FUTURE WORK

In this paper, we reviewed all the architectural details of HP−TCAM and provided implementation of an example design of size 512×36 for its design parameters (L and N) on Xilinx Virtex-5, Virtex-6, and Virtex-7 FPGAs. The development speed for SRAM and FPGA is faster than classical CAM, thus the proposed HP−TCAM design can enjoy the further improvements from SRAM and FPGAs. Classical TCAM cannot be directly implemented on FPGA, with our proposed approach, FPGA-based HP−TCAM provides a wider application usage. Other advantageous points include its independence of the type of data, support for an arbitrarily large input pattern, and its simpler design. More importantly, the throughput of HP−TCAM is one word comparison per clock cycle.

As shown in this paper, SRAM-based TCAM design is a rich research field and further investigations are necessary to find out various SRAM-based TCAM structures. Our future work includes further investigation on the parameter selection for the SRAM-based TCAM.

6. REFERENCES

[1] D. E. Taylor, "Survey & taxonomy of packet classification techniques," ACM computing surveys, Tech. Rep., 2004.

[2] P. Mahoney, Y. Savaria, G. Bois, and P. Plante, "Parallel hashing memories: an alternative to content addressable memories," in *IEEE-NEWCAS Conference, 2005. The 3rd International*, 2005, pp. 223–226.

[3] S. Dharmapurikar, P. Krishnamurthy, and D. Taylor, "Longest prefix matching using bloom filters," *Networking, IEEE/ACM Transactions on*, vol. 14, no. 2, pp. 397–409, 2006.

[4] N. Mohan, W. Fung, D. Wright, and M. Sachdev, "Design techniques and test methodology for low-power tcams," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 14, no. 6, pp. 573–586, 2006.

[5] P. Mahoney, Y. Savaria, G. Bois, and P. Plante, "Transactions on high-performance embedded architectures and compilers ii," P. Stenström, Ed., 2009, ch. Performance Characterization for the Implementation of Content Addressable Memories Based on Parallel Hashing Memories, pp. 307–325.

[6] S. V. Kartalopoulos, "RAM-based associative content-addressable memory device, method of operation thereof and atm communication switching system employing the same," Patent 6 097 724, August, 2000.

[7] Z. Ullah, K. Ilgon, and S. Baeg, "Hybrid partitioned sram-based ternary content addressable memory," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 59, no. 12, pp. 2969–2979, 2012.

[8] M. Somasundaram, "Circuits to generate a sequential index for an input number in a pre-defined list of numbers," Patent 7 155 563, December, 2006.

[9] ——, "Memory and power efficient mechanism for fast table lookup," Patent 20 060 253 648, November, 2006.

[10] Z. Ullah, M. Kumar Jaiswal, Y. Chan, and R. C. C. Cheung, "FPGA implementation of SRAM-based ternary content addressable memory," in *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, 2012, pp. 383–389.

[11] S. Cho, J. Martin, R. Xu, M. Hammoud, and R. Melhem, "CA-RAM: A high-performance memory substrate for search-intensive applications," in *Performance Analysis of Systems Software, 2007. ISPASS 2007. IEEE International Symposium on*, 2007, pp. 230–241.

[12] W. Jiang and V. K. Prasanna, "Large-scale wire-speed packet classification on fpgas," in *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, ser. FPGA '09, 2009, pp. 219–228.