Circuits and Systems, **2013, 4, 299-303**
http://dx.doi.org/10.4236/cs.2013.43041 Published Online July 2013 (http://www.scirp.org/journal/cs)

Power and Time Efficient IP Lookup Table Design Using Partitioned TCAMs

Youngjung Ahn, Yongsuk Lee, Gyungho Lee
Department of Computer Science & Engineering, Korea University, Seoul, South Korea
Email: yjahn@formal.korea.ac.kr, duchi@korea.ac.kr, ghlee@korea.ac.kr

Received May 7, 2013; revised June 7, 2013; accepted June 14, 2013

ABSTRACT

This paper proposes a power and time efficient scheme for designing IP lookup tables. The proposed scheme uses partitioned Ternary Content Addressable Memories (TCAMs) that store IP lookup tables. The proposed scheme enables O(1) time penalty for updating an IP lookup table. The partitioned TCAMs allow an update done by a simple insertion without the need for routing table sorting. The organization of the routing table of the proposed scheme is based on a partition with respect to the output port for routing with a smaller priority encoder. The proposed scheme still preserves a similar storage requirement and clock rate to those of existing designs. Furthermore, this scheme reduces power consumption due to using a partitioned routing table.

Keywords: IP Lookup Device; Routing Table; TCAMs; Insertion

1. Introduction

IP routers have been growing more complex with each passing year. These routers must continually support new features such as Quality of Service and Service Level Agreement monitoring, and they must perform these services at an increasing pace to match new connection speeds. The central design issue for these devices is performing IP lookup and classification at wire speed. Many routers now include a co-processor to perform the crucial task of IP lookup. The most critical issues are related to performance degradation, which occurs when a new entry is inserted into the table and due to the limitations imposed by the large encoder logic. In the worst case, the insertion will take O(N) time to be completed, where N represents the number of entries in the routing table. Table size is expected to grow in the future at an exponential rate, and the update time will linearly grow with table size [1]. On an average day, this same router may spend up to 10% of the time updating the table. No lookups are possible while an insertion is occurring. The insertion time has a dramatic effect on the performance of routers with large tables [2,3].

TCAMs allow a third matching state of "X" or "Don't Care" for one or more bits in the stored data word, thus adding flexibility to the search [4]. TCAMs have a very high cost/density ratio, and they have to search the entire table for each lookup that causes high power consumption [5]. Therefore, TCAMs are used in specialized applications such as packet classification and routing in high performance routers, where the searching speed cannot be accomplished using a less costly method. TCAMs are currently used to perform the task of the Longest Prefix Match (LPM) in high-end routers because they are able to search a table in parallel in one cycle. In such high end routers, it is crucial to have a time and power efficient scheme for updating IP lookup tables with new entry insertions. This paper proposes a novel solution that eliminates the need for routing table sorting per prefix length. As a result, the time penalty for inserting a new entry in the IP routing table is O(1). Moreover, it significantly reduces power consumption because of its partitioned TCAMs. This paper describes the proposed design and implementation details along with the simulation results and design evaluation.

2. Related Works

There have been many attempts to address the insertion problem in IP routing tables [6]. A commonly used technique involves adding empty space after each prefix group. If there is an empty slot after the required prefix group, the TCAM may keep a few empty memory locations every x non-empty memory locations ($x < N$, where N is the table size). The average case update time improves to O(x), but it degenerates to O(N) if the interme-

diate empty spaces are filled up [7]. A better technique is called the L-Algorithm, which can create an empty space in a TCAM in no more than L memory shifts [7]. L-Algorithm reduces the worst-case to O(L) where L is the length of the IP address. This means the algorithm would have a worst-case insertion time of O(32) for IPv4 and O(128) for IPv6. This algorithm takes advantage of the fact that sorting within each length group is unnecessary, so only one entry from each group needs to be moved. This provides a great improvement over the O(N). However, even with the L-Algorithm, problems can still occur in an actual router. A routing table update due to a new entry insertion may still need 32 or 128 cycles; this is undesirable in high-end routers. The design may also require a large overhead to manage the free space in the table. Also, when a burst of updates is received by the router, the router will not be able to route packets while the insertion is taking place. This means that the incoming packets need to be buffered. If the buffer size is not sufficient to hold all the received packets, then some will be dropped. This hurts the major selling point of routers, level of the worst-case performance and reliability. If the insertion time is reduced to O(1), then the need for additional buffering would be unnecessary since the insertions can happen at lookup speed.

TCAMs are currently used to perform the task of the Longest Prefix Match (LPM) in high-end routers because they are able to search a table in parallel in one cycle. However, TCAMs are inefficient in terms of power consumption even though the search time is fast. The size of TCAMs in IP lookup devices is increasing. There are currently many studies on reducing the power consumed by IP lookup devices by dividing routing tables [8,9].

3. Design Approach and Implementation Details

In order to keep the table sorted, the table must be updated, and the result would take O(N) moves. In the proposed design, the need for sorting is removed, and thus insertion time is improved to O(1). **Figure 1** shows the proposed design. The output port divides the table, so it is partitioned into smaller tables. The number of tables generated equals the number of output ports of the router, and each table holds a collection of all the entries that map to the output port that it corresponds with. All entries in a partitioned table map to the same output port, making it unnecessary to sort the entries in the table.

When searching, each TCAM checks the IP address in parallel. Each table outputs the matched lengths to a selection logic. The selection logic chooses the longest length, and the packet is sent to the output port based on the table that had the longest prefix match. When insertion is need, the output port matching the entry is checked by analyzing the entry. After figuring out the matching output port, the entry is inserted into an open location in the corresponding table. Note that the entry goes to "any" open location in the corresponding table.

Figure 2 shows the modified design for the partitioned table. Typical TCAM cells storing the network address prefix are shown on the left, and memory cells storing the lengths with the priority encoder removed are shown on the right. When the logic selects the table containing the LPM, the information of the output port is known, and there no longer a need for SRAM memory to store such information. Since the TCAM cells can be connected to the corresponding SRAM memory cells because they are directly related, the priority encoder and address decoders are also not needed. In addition, more than one row can be asserted at once. This is possible because there can only be one match per length. If a lookup results in more than one match of the same length, then this would mean that duplicate entries have been stored. Therefore, the maximum number of matches is 32-bits, the length of an IP address. Lengths are stored in a one-cold encoding. Since lengths are being stored instead of output ports in the SRAM cells, the number of SRAM cells needed is more than that in a typical router.

Since the TCAM cells are directly connected to the SRAMS cells, the proposed design reduces the complexity of the traditional design. For each entry, 32 TCAM cells are used to store the prefix, and 32 SRAM cells are used to store the prefix length. If more than one entry match is found, then each asserts a corresponding output line to reveal the prefix length. The selection logic checks the length output to determine the table containing the longest prefix match. This can be used to choose the longest match since there can only be a maximum of

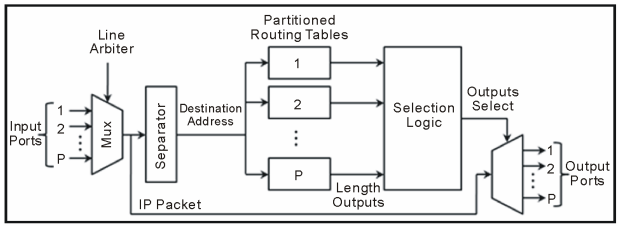

Figure 1. Proposed architecture.

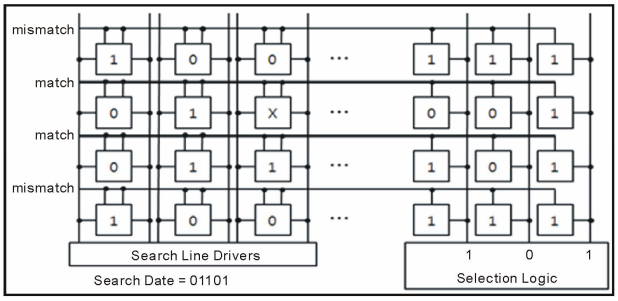

Figure 2. Modified routing table.

one matching entry for each prefix length between 1 and 32.

For the proposed design, the SRAM cell word lines are directly connected to the TCAM match lines. During normal SRAM operation, only one word line should be activated at once because if multiple word lines are activated, then the neighboring SRAM cells may become corrupt due to the shared bit line. The TCAM search may have multiple matches, and it can cause multiple SRAM word lines to be activated. Additional access transistors are added to each SRAM cell to activate the SRAM cell only if a "0" is stored in the cell. Only one SRAM cell will be activated for each column since each match can correspond to only one length, and the length is stored in a one-cold encoding. To enable access to the cell for writing, two additional transistors are needed and four transistors in total are added for each 6T SRAM cell. This is shown in **Figure 3**. The standard 6T SRAM cell is presented in the dotted line. The four extra transistors will increase the area and latency of the SRAM cell.

The size of each partition of the table should still be fairly large because the total table size is very large and the number of output ports is usually quite low. Long lines will significantly increase the latency [10]. This requires each partition to be further divided into smaller blocks, as is shown in **Figure 3**. The outputs from each block are combined in an OR scheme. Since there can only be one match for each length, there will be no conflicts.

Normally, the distribution of the prefixes will not be balanced across partitions. In some cases, more than 90% of the prefixes forward to the same port [11]. Many configurations are possible since this proposed design subdivides the partitions further to reduce the bit line length.

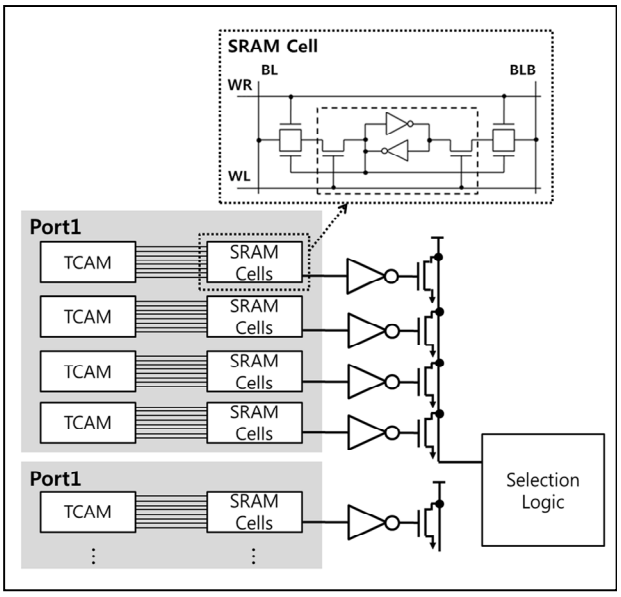

Figure 3. SRAM cell design and partitioned layout.

By combining the outputs of the blocks, *i.e.*, sub-tables, the OR scheme controls the boundaries of each partition. One end of the boundary connects to the selection logic, while the other end connects to a column pull-up. A basic switching scheme allows the partition sizes to be programmable. The number of possible configurations can be determined by the minimum grouping size. Due to hardware complexity limitations, switching connections cannot be available at all sub-table boundaries.

The selection logic takes inputs from the output lines of the partitions of the table. The logic uses 32 inputs for each output port, so if the router has four output ports, then 128 inputs are connected. The selection logic first determines the longest length that was found. Then, the output port is determined based on the table that the LPM is from. The logic finds the port with which the highest length is associated. Then, the packet can be forwarded to the correct port. This is a very simple logic and is similar in complexity to a small priority encoder.

4. Evaluation of the Design

Improving insertion time is only beneficial if the lookup latency is not slower than that of the original design. The proposed design is compared to a typical co-processor design. The typical design for this comparison contains a standard TCAM that is capable of holding up to 256 K entries. There are four and sixteen output ports in typical core routers. By analyzing this, four output ports have been used. This latency of the new and old designs could be broken down into several parts and could also be separately compared as shown in **Table 1**. As the table shows, both designs have similar latencies. Although the SRAM reading time is longer than that of the proposed design, the selection logic is much simpler than the logic of the priority encoder. The first portion of both designs begins with the TCAM search. The design of the TCAM cells remains unchanged so that the latency of the TCAM search could remain identical. In the proposed design, the next step is to access the SRAM cells that correspond to the matched prefixes from the TCAM search to determine the length of the prefix.

The overall size of the SRAM cell has increased due to the addition of access transistors. This increases the length of each bit line causing the capacitance to become larger, and this increases the necessity time to pull down

Table 1. The latency.

Original Design	TCAM Search	Priority Encoder	SRAM Read	Total
	~6.5 ns	~3.6 ns	2.7 ns	~12.8 ns
Proposed Design	TCAM Search	SRAM Read	Selection Logic	Total
	~6.5 ns	4.1 ns	~0.98 ns	~11.6 ns

the line. Since the bit lines of one large table are too long, the partition table is further broken into sub-tables. Each table outputs the bits corresponding to the lengths that were found in the sub-table. The results from each sub-table and each length are combined using an OR scheme. The optimal configuration was found by comparing the delay due to the size of each sub-table and the delay of combining the results from each sub-table. Cadence tools in a 0.25 um technology were used to simulate the access time of various SRAM configurations to compare the standard of SRAM cells. The capacitances of the bit lines were estimated based on the area increase of the additional transistors. These results show that the original SRAM performs approximately 33% faster than the new SRAM in all configurations. A full custom layout would provide more accurate results. Larger size is preferred in order to reduce the number of outputs that should be combined together. **Table 2** shows the simulation result.

In old designs, the delay of the major logic is from the priority encoder. This circuit becomes larger when it handles the input from all TCAM search lines. The proposed design directly connects the TCAM cells to the SRAM cells so that the priority encoder could be removed. The address decoder of the SRAM from the old design can be removed too. The major logic delay according to the proposed design is from the selection logic. First, the logic chooses the longest length, and it then determines the length of the partition table that was already found. The circuit is similar to a very small priority encoder.

The logic was designed by using Mentor Graphics ModelSim in Verilog. The design was synthesized using a 0.50 um technology and simulated by using Mentor Graphics LenardoSpecturm. The results show that the selection logic is much easier than the priority encoder and has a smaller delay. The results are shown in **Table 3**.

To evaluate the scalability of the proposed design, the differences between the two designs were compared. The TCAM cells have not been modified, so they will scale in the same way. When the SRAM cells were modified, they have shown about 33% difference in performance. Although the proposed design continuously has a difference, it will also scale in the same way. The largest difference between the two designs is found in the difference in logic. The selection logic scales logarithmically depend on three things. The length of the IP address, the number of output ports on the router and the size of the routing table. Since the length of the IP address is unexpected to increase after IPv6 and the number of output ports to change, the most important factor is the number of entries in the routing table. A priority encoder's delay will also scale logarithmically with respect to the number of entries in the table [12]. Its delay is not affected by the number of output ports or length of the IP address because there are no dependencies. The address decoder in SRAM will have a logN scaling factor. Based on the above analysis, both designs are expected to scale similarly as the routing table size grows. The power consumption in the partitioned routing table is shown as below. When the length of prefix is 32-bits while the number of entry of routing table is n, the power consumption of TCAMs is denoted as (1) [13]. P(n) consists of one TCAMs and describes the power consumption while the number of entry is n.

$$P(n) = \sqrt{n} \times (0.5 \times \log_2 n + 1) + 0.5 \times \log_2 n \quad (1)$$

In IP lookup, the behavior of TCAMs can be divided by IP searching and notification of matched Prefix. According to [14], the ratio of power consumption of two functions is shown as (2). The IP search in TCAMs is advanced in all partitioned routing tables. In addition, the notification of matched Prefix is progressed in one partitioned routing tables.

$$\text{search} : \text{match} = 17 : 83 \quad (2)$$

The proposed scheme divides the routing table corresponding to the output port. The number of entry of each partitioned table is n/x when the number of output port is x. By using this property and (2), the total power consumption of partitioned routing table is shown as (3).

$$P_{total} = (1 - 0.17) \times x \times P\left(\frac{n}{x}\right) + (1 - 0.83) \times p\left(\frac{n}{x}\right) \quad (3)$$

Table 4 describes power consumption and efficiency corresponding to the number of entry, when the number of output port is sixteen. When the routing table can be divided into sixteen tables, efficiency is increased by approximately 28% compared to the power consumption of existing TCAMs. When the total number of entries is increased, the efficiency of power consumption is decreased because the number of x entries of each partitioned routing table is increased.

Table 2. SRAM simulations.

Size	Original SRAM	Proposed SRAM
128	1.4 ns	2.0 ns
256	2.0 ns	3.0 ns
512	2.7 ns	4.1 ns

Table 3. Logic simulations.

	Priority Encoder	Selection Logic
Gates	27,668	3015
Delay	14.4 ns	3.8 ns

Table 4. Power consumption.

# of Entry	# of Output Port	Power Consumption		Efficiency
		TCAMs	Partitioned TCAMs	
128 K	16	3447.87	2432.89	29.44
256 K	16	5129.00	3660.05	28.64
512 K	16	7612.31	5488.88	27.89
1024 K	16	11274.00	8207.60	27.20

5. Conclusion

The purpose of IP router is to make a decision on a routing path to use and to forward a packet corresponding to the decided route. An existing router, which stores a prefix in the routing table, has a difficulty to meet QoS requirement from rapidly expanding internet environment—keep speeding up and adding new routes. The contribution of the proposed scheme is mainly to reduce routing table updating time and also the power consumption at the same time. The new design improves the routing table updating time by storing new prefix in routing table in unsorted manner: The worst case updating time in existing design O(N) reduces to O(1). In order to do this, the routing table is partitioned per output port, while the SRAM that stores a prefix length is directly connected to each partition of the routing table. This allows partitioned TCAMs to be employed in the design for shorter delay and lower power consumption. The logic of priority encoder and another logic related to existing SRAM are replaced to simple selection logic. This removes not only the needs for ordering the table by prefix length but also the lookup process for finding an output port.

6. Acknowledgements

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2004615).

REFERENCES

[1] "Latest Version of AS65000-BGP Routing Table Statistics Analysis Report." http://bgp.potaroo.net/as2.0/bgp-active.html

[2] V. Srinivasan, B. Nataraj and S. Khanna, "Methods for Longest Prefix Matching In a Content Addressable Memory," US Patent 6237061, 1999.

[3] R. Guo and J. G. Delgado-Frias, "IP Routing Table Compaction and Sampling Schemes to Enhance TCAM Cache Performance," *Journal of Systems Architecture*, Vol. 55, No. 1, 2009, pp. 61-69. doi:10.1016/j.sysarc.2008.08.001

[4] K. Pagiamtzis and A. Sheikholeslami, "Content-Addressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey," *IEEE Journal of Solid-State Circuits*, Vol. 41, No. 3, 2006, pp. 712-727. doi:10.1109/JSSC.2005.864128

[5] S. Kaxiras and G. Keramidas, "IPStash: A Power-Efficient Memory Architecture for IP-Lookup," 36*th International Proceedings of Symposium on Microarchitecture*, San Diego, 3-5 December 2003, pp. 361-372.

[6] M. J. Akhbarizadeh and M. Nourani, "An IP Packet Forwarding Technique Based on Partitioned Lookup Table," *IEEE International Conference on Communications*, Vol. 4, 2002, pp. 2263-2267.

[7] D. Shah and P. Gupta, "Fast Updating Algorithms for TCAMs," *IEEE Micro*, Vol. 21, No. 1, 2001, pp. 36-47. doi:10.1109/40.903060

[8] T. Kocak and F. Basci, "A Power-Efficient TCAM Architecture for Network Forwarding Tables," *Journal of Systems Architecture*, Vol. 52, No. 5, 2006, pp. 307-314. doi:10.1016/j.sysarc.2005.12.001

[9] V. C. Ravikumar, R. N. Mahapatra and L. N. Bhuyan, "EaseCAM: An Energy and Storage Efficient TCAM-Based Router Architecture for IP Lookup," *IEEE Transactions of Computer*, Vol. 54, No. 5, 2005, pp. 521-533. doi:10.1109/TC.2005.78

[10] B. S. Amrutur and M. A. Horowitz, "Speed and Power Scaling of SRAMs," *IEEE Transactions on Solid-State Circuits*, Vol. 35, No. 2, 2000, pp. 175-185. doi:10.1109/4.823443

[11] The Internet Performance Measurement and Analysis Project. http://ftp.chg.ru/pub/network/routing/ipma/Manual/

[12] C. H. Huang and J. S. Wang, "High-Performance and Power-Efficient CMOS Comparators," *IEEE Journal of Solid-State Circuits*, Vol. 38, No. 2, 2003, pp. 254-262. doi:10.1109/JSSC.2002.807409

[13] W. Lu and S. Sahni, "Low-Power TCAMs for Very Large Forwarding Tables," *IEEE/ACM Transactions on Networking*, Vol. 18, No. 3, 2010, pp. 948-959. doi:10.1109/TNET.2009.2034143

[14] B. Agrawal and T. Sherwood, "Ternary CAM Power and Delay Model: Extensions and Uses," *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, Vol. 16, No. 5, 2008, pp. 554-564.

