ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

Design and Implementation of a High-Performance Look-Up Table (LUT) Accelerator using CAM-RAM Architecture in Verilog HDL

B.Sai Poojitha[1], R.L.B.R. Prasad Reddy[2]

[1]PG Scholar, Dept.of ECE, Srinivasa Institute of Technology & Science, Kadapa.
[2]Associate Professor, Dept.of ECE, Srinivasa Institute of Technology & Science, Kadapa.
Email: poojithabandameeda9@gmail.com, rajendra.409@gmail.com

Abstract –

This paper presents the design and implementation of a hardware accelerator based on Content Addressable Memory (CAM) and Random Access Memory (RAM) for high-speed function approximation and data transformation. Traditional Arithmetic Logic Units (ALUs) often struggle with non-linear functions (such as sin(x), log(x), or complex scaling) due to the high clock-cycle latency of iterative algorithms. This design addresses these bottlenecks by utilizing a parallel search-and-retrieve architecture. The core of the accelerator is a Binary CAM implemented in Verilog HDL, which acts as a "hardware search engine." It takes a search key as input and identifies its location in a single clock cycle. This location is then used as an address for a synchronized Result RAM, which outputs the pre-computed value. The design is optimized for Xilinx FPGA architectures by utilizing distributed RAM and Look-Up Table (LUT) primitives to minimize resource consumption and maximize frequency. Key features of the implementation include: Single-Cycle Latency: Achieving near-instantaneous results for complex mathematical inputs. Resource Efficiency: Leveraging Xilinx-specific synthesis attributes to infer efficient hardware structures. Scalability: A parameterized design allowing for variable data widths and memory depths. Experimental results via Xilinx Vivado simulation and synthesis demonstrate a significant reduction in execution time compared to traditional software-based or iterative hardware approaches, making it an ideal candidate for high-speed networking and real-time digital signal processing (DSP) application

Keywords – Look-Up Table, CAM-RAM, Verilog HDL, ALU.

1. INTRODUCTION

Hardware based IP lookup solutions are generally categorized into two classes; SRAM- based, and CAM-based solutions. SRAM-based solutions are basically the hardware counterpart of the software-based solutions. The trie/ tree based data structures, which are constructed from the routing table prefixes and their corresponding next hop information, are directly implemented on the SRAM hardware. Since tree traversal algorithms require several memory accesses until the leaf node of the tree is reached, SRAM-based solutions may suffer from high latencies, which are not desired for high-speed IP lookup. Therefore, the studies targeting SRAM platform for the IP lookup solution, generally tend to decrease the depth of the trees to be constructed, by applying some optimization algorithms on the routing table prefix set, or to increase throughput by employing parallel and/or pipelined operation. However, parallel or pipelined architectures require several SRAM memories and the memory sizes of the employed SRAMs may not be of equal size. Especially for pipelined architectures, each level of the tree is assigned to a separate SRAM, and the required size for the SRAMs doubles at each level, which makes this approach infeasible for commercial SRAM utilization. There-fore, parallel and/or pipelined SRAM-based architectures are mostly implemented on FPGA platform, which allows users to design several different-sized SRAM blocks on the same chip. CAMs (Ternary Content Addressable Memories) are special hardware that allows the search of the whole content simultaneously for a given key data. For IP lookup, the prefixes are sorted in ascending order with respect to their length, and the longest matching prefix is selected as the matching entry with the highest address via a priority encoder. Conventional CAMs are larger in size compared their SRAM counterpart of the equal storage capacity, since they contain two SRAM cells for each entry, one for prefix and one for the mask, and additional comparator circuitry. Thus, CAMs are more expensive and consume more power compared to SRAMs. Therefore researchers, who favour CAMs for IP lookup, either try to decrease the power consumption, or to increase the storage efficiency. Power consumption techniques are similar to each other, partitioning the match line into segments, a pipelined search is carried, and search is continued only if there is a match in

the previous stage. This approach results in the loss of the single cycle search capability of the CAMs. Storage efficiency is increased by algorithmic transformations of the prefixes, and these types of approaches make the table management hard, especially for the updates. As in the case of SRAM-based solutions, there are also studies utilizing FPGAs for CAM implementations as well.

2. RECONFIGURABLE CAM ARCHITECTURE FOR HIGHSPEED IP LOOKUP

Ternary Content Addressable Memories (CAMs) are the most often used components to build hardware IP lookup engines for backbone routers. CAMs are special type of memories which search the entire content of the memory in a single cycle for a given input and then return the address of the matching entry if there is a match in the stored content of the CAM. A priority encoder (PE) has to accompany the CAM, when there is more than one match for the searched input data. Therefore, the design of the PE logic is very significant to enable LPM when CAMs are used to build the IP lookup tables in routers. One of the main contributions of this thesis is the is the S-DIRECT-Scalable and Dynamically Reconfigurable CAM architecture that is custom designed for IP address lookup. The novel features of this architecture are:

1. An S-DIRECT CAM of any size is constructed from Basic CAM Blocks (BTB) each which has an individual PE logic. Hierarchically arranging the PE logic of the BTBs results in inherent PE for the resulting CAM. The single cycle search of the CAM includes priority encoding process. Hence, we achieve design scalability eliminating the requirement of custom built PE circuitry that depends on the size of the CAM.
2. S-DIRECT has non-blocking update capability. Writing an S-DIRECT entry does not interrupt the search on the remaining entries, simultaneous update and search operations
can be performed.
3. The update requests are processed and subsequently the updates are carried out entirely in hardware without additional software support. The update requests are served in constant time.
4. S-DIRECT is not limited to a specific hardware platform or a specific CAM cell implementation provided that the hardware requirements that we define are satisfied.

We demonstrate the viability of S-DIRECT by implementing it on FPGA as the selected hardware platform. FPGA fulfills the hardware requirements that we define for the update capabilities. The conventional way to implement a CAM architecture is using a prefix register and a mask register for tri-state marking that we call prefix/mask register (PMR)-based implementation. A comparator circuitry is used for each entry. To this end, we implement SDIRECT CAM entries with both this legacy PMR approach and with a different approach which utilizes the Look Up Table (LUT) resources to validate that it is a general architecture. Combined with our integrated priority encoder and address decoder circuitry, we implemented a 16Kx32 CAM with our LUT-based approach and tested it. We achieved a 153 MHz operating frequency that corresponds to a line rate close to 50 Gbps. S-DIRECT is a custom CAM architecture that is designed to perform two main functions for IP lookup. The first one is finding the LPM for a given IP address with the
corresponding next hop index value for the matching prefix. The second one is updating the entries in the IP lookup table.

The overview of the S-DIRECT architecture is depicted in Fig. 1. The main components of S-DIRECT are the Augmented CAM Block (ATB), the Preprocessing Logic Block (PLB), and the Next Hop Index Memory (NHIM). Each CAM entry in ATB stores an IP prefix together with its length. Accordingly, we partition an ATB of D entries into up to 32 blocks where D is the desired size of the IP look up table. Each block bk has mk entries of prefix length k where $1 \leqslant k \leqslant 32$. The locations of the blocks in ATB are sorted in the increasing order of the prefix lengths and there is no presumed order within a block. We assume that, mk are known for all k at the design stage similar to the assumptions in. PLB is responsible for appropriately formatting the inputs of S-DIRECT to be applied to ATB and NHIM and for controlling the write cycles during the CAM update operations. NHIM is an auxiliary memory of D lines and stores the IP router outgoing port index (next hop index) corresponding to the matched IP prefix which is represented by M bits. Here, M is determined according to the number of outgoing ports of the IP router. The next hop index for the IP prefix that is stored in address a in ATB is also stored in address a in NHIM. The Default Next Hop Index Register (DNHIR) holds the default next hop index that is assigned for IP addresses that are not matched in ATB. The output of S DIRECT is next_hop_index which is the next hop index for the searched IP address if there is a match or the default next hop value otherwise.

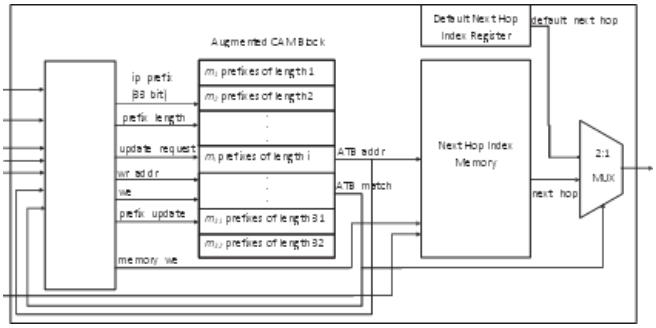

Fig.1: S-DIRECT Complete Architecture.

Each CAM entry in the ATB consists of a prefix cell augmented with an additional length cell. The prefix cell implements a logic function where ip_prefix and prefix_match are the input and output of that logic function, respectively. Furthermore, prefix_update is the input that changes the truth table of the prefix cell logic function when we=1. S-DIRECT is custom designed for LPM IP lookup. Hence, the prefix cell logic function is defined such that prefix_match=1 if the p most significant bits of ip_prefix input are the same as a given p bit prefix. For the rest of this paper we call this given p bit prefix as the stored content of the prefix cell. Similar to the prefix cell, the length cell implements a logic function with the respective input and output ports of prefix_length and length_match. The length cell logic function is defined such that, for a CAM entry that resides in block bk in the ATB, length_match=1 if prefix_length=k − 1. Similar to the prefix cell, we call this k − 1 value as the stored content of the length cell. Here, k is fixed once before the operation and does not change during the operation. Any hardware platform that satisfies the requirements can be used for S-DIRECT implementation. We choose Xilinx XC7VX1140T (Virtex-7) FPGA as our implementation platform in this thesis. The S-DIRECT architecture is implemented on ISE 14.1 software and the total on-chip power consumption is estimated by Xilinx's Power Estimator tool. This total power includes both the device dependent static power which is dissipated when there is no signal activity and the additional dynamic power that is dissipated during the operation. The conventional way to implement a CAM architecture is using a prefix register and
a mask register for tri-state marking that we call prefix/mask register (PMR)-based implementation. A comparator circuitry is used for each entry. PMR-based implementation depletes the FPGA resources very quickly as also indicated in. We implement S-DIRECT with the ATBs of different sizes with PMR-based CAM entries on Virtex-7 and present the resource and power consumption results in Table 1 to demonstrate this problem.

Table 1: Resource Utilization for Different Size PMR-Based CAM Implementations on Xilinx XC7VX1140T FPGA.

Size (Number of ATB entries)	Number of Slice LUTos (Available:712000)		Number of Slice Registers (Available:1424000)		On-chip Power (mW)		Max. Freq. (MHz)	Corr. line rate for IP lookup (Gbps)
	Used	Utilization (%)	Used	Utilization (%)	Dynamic	Total		
16	378	0.05	1100	0.07	18	463	284	90.88
64	2195	0.31	4295	0.30	43	488	224	71.68
256	9182	1.29	17008	1.19	143	589	195	62.40
1024	36108	5.07	67735	4.76	497	947	173	55.36
4096	140573	19.74	270828	19.02	1955	2423	160	51.20
16384	553679	77.76	1102287	77.41	8003	8554	153	48.96

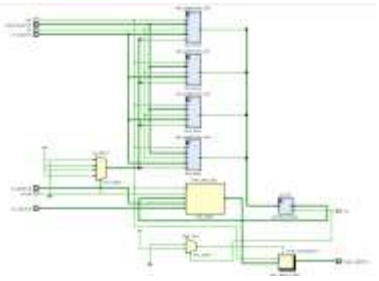

Fig.2: Technology mapped (physical) synthesis of the design

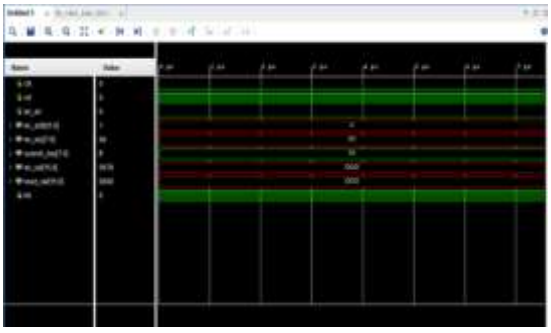

Fig.3: Placed and routed design Simulation

Fig.4: Placed and routed design Simulation (zoomed view)

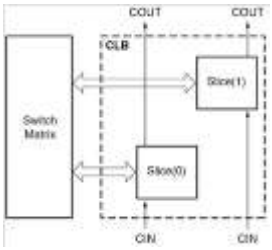

Fig.5: Xilinx CLB

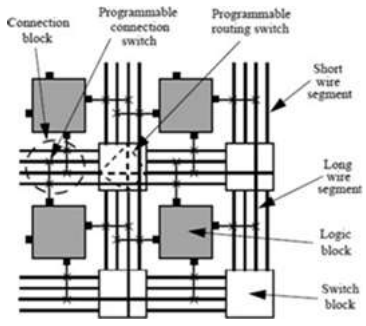

Fig.6: Island-style FPGA architecture

3. SYSTEM IMPLEMENTATION

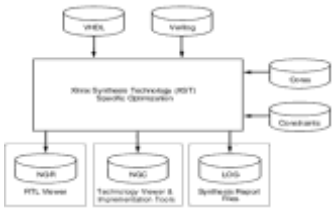

Fig.7: General XST Flow

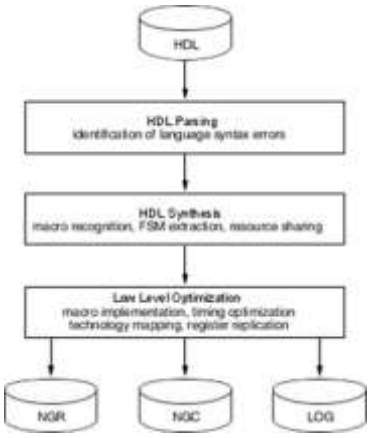

Fig.8: XST Synthesis Steps

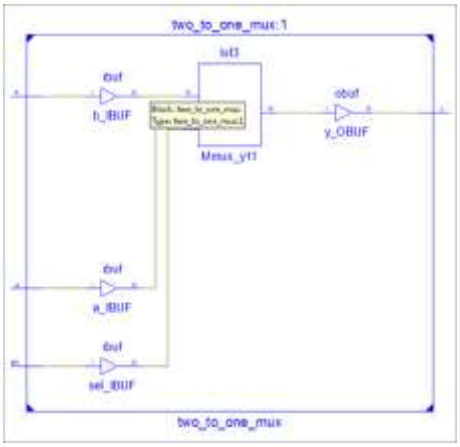

Fig.9: Technology mapping of a 2-to-1 MUX on Xilinx FPGA

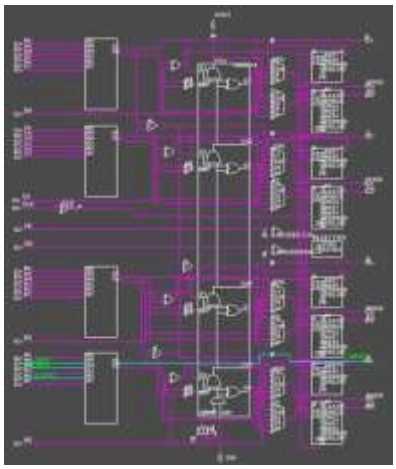

Fig.10: The inside view of the CLB slice where the 2-to-1 MUX mapped

Fig.11: Power Supply of the Design

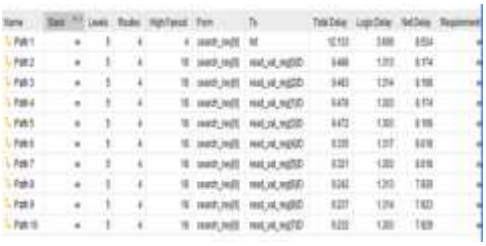

Fig.12: Delay Report of the Implemented Design

CONCLUSION

The objective of this thesis is the design and implementation of IP look up architectures with design scalability and non-blocking update capability where the updates are carried out entirely on hardware. SRAM-based and CAM-based architectures are the two main IP lookup approaches that are implemented on hardware. The researchers, who utilize SRAM-based architectures in their solutions, basically criticize the high power consumption of CAMs to justify their solutions. SRAM-based algorithms utilize tree-based data structures and algorithms for routing table management. A tree search requires several memory accesses to get LPM, hence, pipelined tree structures are used to increase the throughput. Pipelined architectures require several SRAMs of different sizes, and therefore implementing pipelined structures is only feasible on FPGA platform. There are studies promoting tree-based pipelined IP lookup algorithms claiming low power designs on FPGA. These studies state that their motivation for the pipelined tree-based SRAM architectures is that the CAMs are power hungry. However we should note that the FPGA's power consumption is not mentioned in these studies. In stated that an FPGA consumes 14 times more power than its ASIC counterpart for logicand-memory only implementations. CAMs have single cycle search capability. Moreover, updating the forwarding table in

CAM-based schemes is generally simpler than that in trie based algorithms. We further improve the update capability on the CAM by proposing a novel architecture for CAMs. Our architecture adds the non-blocking update capability to the standard CAMs, and with the added length cell field to the basic CAM entry, determining the address to update can

directly be performed on the hardware without requiring a software support. We propose a CAM architecture, which differs from the CAM architectures in the literature in the following ways:

1. Embedded priority encoder: Single cycle search also includes priority encoding process
2. Non-blocking update: Write process on any entry do not block the search on the remaining entries, simultaneous update and search operations can be performed
3. Update on the hardware: No additional software support is required for processing the update request to determine the address of the CAM entry where the update will occur

We feature S-DIRECT together with an SRAM block in order to generate a hybrid IP lookup solution for a real-life routing table that we call SHIP. SRAM is used as a DMA engine for prefixes of length smaller than or equal to 24-bit, so that it gives also match result in single cycle, compliant with the CAM operation. We implemented both S-DIRECT and SRAM block on a high-end FPGA (Xilinx Virtex-7 1140T). SRAM implementation on the FPGA is performed by instantiating the dual-port BRAM components which are already embedded in the FPGA. Our main contribution with the proposed SHIP architecture is that routing table updates are fully processed and realized on hardware, which is not accomplished by any hardware based IP lookup solution previously. Furthermore, the updates are performed without blocking the ongoing IP lookup processes. Besides these contributions, our solution also satisfies major IP lookup performance requirements in the following ways:

• The proposed architecture is scalable to larger routing table sizes
• We can support more than 16M prefixes on a simple FPGA-memory combination
• IP lookup is performed in one clock cycle, independent of the number of prefixes in the table, hence it gives the lowest latency among the solutions
• A lookup throughput of 153 MLPS can be achieved on the FPGA, which is more than the requirement for line rate operation of OC-768, 40 Gbps link.

The proposed architectures can be implemented on any hardware platform provided that they satisfy the architectural requirements stated in the thesis. While performing FPGA implementations of our architectures, we not only demonstrated the viability of our proposals, but also provided the efficient way of implementations of our architectures on the FPGA platform. For the FPGA implementation phase, we investigated the inconsistencies of the FPGA architecture and synthesis tool for our design, and proposed some suggestions for both FPGA architecture and the synthesis tool for better FPGA utilization. We believe that, if the number of run-time reconfigurable logic blocks is increased and/or they are placed close to each other on the FPGA, the path delays would be smaller, resulting in a higher operating frequency, and hence in a higher lookup throughput. The operating frequency of our design is limited by the long combinational path delays in the PE design of the S-DIRECT. However, this is inevitable when the single cycle search capability is desired. As the number of match lines (CAM size) increases, the number of logic levels in the PE design also increases, and the achievable clock frequency decreases. The BRAM resources on the contemporary FPGAs did not allow us to build a single-chip solution with our proposed hybrid DMA and CAM-based architectures. However, we believe that in the near future this inefficiency will be eliminated, and our architecture can be implemented as a single-chip solution as well. S-DIRECT architecture can also be easily scaled to IPv6 addressing scheme by employing 128-bit for prefix cells, and 7 bits for length cells, keeping the remaining logic as it is. Actually, the global unicast IP address of the IPv6 assigns most significant 64 bits to network ID, and the remaining 64 bits to Interface ID. Therefore, for the practical application, the required number of bits for the prefix cell and length cell is 64 and 6 respectively. A current study also investigates the prefix distribution of the contemporary IPv6 routing tables, and observes that over 80% of the prefixes are either 32, 48 or 64 bit length, and propose a hash based algorithm for IPv6

Address lookup. Note that, the CAM architectures are the natural hash tables where no collision occurs, and hence compliant with the S-DIRECT architecture proposed in this thesis study can be used in IPv6 routers as well.

REFERENCES

[1] M. Ruiz-Sanchez, E. Biersack, and W. Dabbous, “Survey and taxonomy of ip address lookup algorithms,” Network, IEEE, vol. 15, no. 2, pp. 8 –23, mar/apr 2001.

[2] K. Pagiamtzis and A. Sheikholeslami, “Content-addressable memory (cam) circuits and architectures: a tutorial and survey,” Solid-State Circuits, IEEE Journal of, vol. 41, no. 3, pp. 712 – 727, march 2006.

[3] (2013) Xilinx synthesis and simulation design guide (ug626).

[4] (2013) Xilinx ug474 7 series fpgas configurable logic block user guide.

[5] D. Chen, J. Cong, Y. Fan, and L. Wan, “Lopass: A low-power architectural synthesis system for fpgas with interconnect estimation and optimization,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol. 18, no. 4, pp. 564–577, 2010.

[6] H. Lim and N. Lee, “Survey and proposal on binary search algorithms for longest prefix match,” Communications Surveys Tutorials, IEEE, vol. 14, no. 3, pp. 681–697, 2012.

[7] L. Luo, G. Xie, Y. Xie, L. Mathy, and K. Salamatian, “A hybrid ip lookup architecture with fast updates,” in

INFOCOM, 2012 Proceedings IEEE, 2012, pp. 2435–2443.