Design and Implementation of Hardware to Perform Testing the Matching Packet Header Based on FPGA

Assist. Prof. Mehmet Efe Özbek¹ and Anwer Sabah.AL-Obaidi²

¹Aitlim University Ankara/Turkey
²Electrical Engineering Technical College, Middle Technical University, Baghdad, Iraq

Email: ¹meozbek@atilim.edr.tr , ²ENG.anwer89@gmail.com

Abstract. The hardware architecture of the parallel process multiple RAM that emulates the behaviors of content addressable memory for packet classification is presented in this paper. With the increase in Internet networks' speed, the speed of detection of intruders has become a basic requirement. In this work, a packet header field is used in a fast and efficient way to detect intruders to prevent them from accessing the data. The application test results were fast and compatible when used FPGA board technique from xilinx. Finally, the design, synthesis of this parallel process multiple RAM packet header detector has been achieved using Vivado 2018.2 simulator, and coding is written in Verilog HDL language and implemented on Virtex – 7 FPGA (Field Programmable Gate Array) kit.

Keywords -: packet classification; pipeline; FPGA. CAM RAMs

1. Introduction
Packet classification is a forwarding engine that uses the incoming packet header to generate the action as the output, which is usually one of the three decisions (Accept, Deny, or Drop). Therefore, the speed of the forwarding engine must be high enough to process data within the speed of guided media. Many studies have been done in classifying packet headers in the past, and programs have been used to implement various algorithms for classifying packets. Most of them depend on the hierarchical tree decision, which is very efficient in terms of usage memory, easy to create a stratified tree from multiple header fields for rules, and rule updates to enhance the search performance. However, these low-speed algorithms methods become useless due to its delay to the high speed of data and increase the number of rules[1][2][3][4]. Another method to test the packet header is to use packet forwarding using multidimensional effective range matching[5]. The traditional router architectures rely on buffer flow architectures to classify packets. The basic idea is that the arrival of packets determines the flows, meaning that if a packet belongs to a new stream arrives, more packets of that stream are expected to arrive soon. Using these expected behaviors, the first beam of streaming is processed by a slow pathway that analyzes the full head. Then the package header is inserted into a cache or hash table with the action to be applied to the first package and all other streaming packages. When subsequent packets arrive in this flow, the corresponding action can be determined from a cache or hash table[6]. Newly work has been moved to use:
- SRAM- solution on FPGA
- built Ternary content addressable memory (TCAM) in FPGA
  - In 2018[7][8] INAYAT ULLAH, and others, used multi-ported SRAM memory to design and implements a TCAM on FPGA using a multi-pumping to achieve efficient utilization. The depth limit for BRAMs on FPGAs limits the storage efficiency of TCAM bits, so they mapped sections of the TCAM table into sub-blocks of BRAM components and achieved a memory-efficient TCAM. They implemented their design on the Virtex-6 xc6vlx760 FPGA. Moreover, compared to TCAM designs based on current FPGAs, it offers 2.85 times better performance per memory.
  - In 2019,[6] Weiwen Yu, and Others proposed pseudo-TCAM architecture, they used the SRAM-based hardware architecture that simulates the behaviors of TCAM for classifying packets. They used a prefix inclusion coding method to encode the packet header, and encoding rules were set for SRAM-based matches using a bit selection policy. The specified bits from the input key were used to access a base in SRAM for comparison. The results obtained from the header classifier can reach the Throughput of 426 million packets per second.

The main focus of this paper is on building hardware for FPGA-based matching packet header testing. Subsets of RAM components equal to the number of bytes in the IP address were chosen in the packet header to achieve FPGA memory efficiency. Matching packet header testing was designed and implemented with VHDL based on the Virtex - 7 FPGA kit. This paper is organized as follows: Section 2 provides an overview of the FPGA memory packet address test with a summary of the procedure. The third section contains an explanation of the corresponding packet header design. The fourth section discusses the implementation of the proposed PH-Matching in the FPGA, and the results of the design. Finally, the conclusion in the final section.

2. An overview of header beam matching designs using FPGA
As the Internet has evolved towards multi-media applications and quality of service, packet processing has become a prerequisite for the next generation of Internet routers. Significant advances in internet speed have resulted in the rapid transition to higher speed FPGA components, and research in this area indicates a shift in use in packet processing design.

FPGAs put in force high-velocity block-RAM (BRAM) in a silicon substrate. For example, the Xilinx Virtex-6 FPGA carries 720 BRAMs; the Virtex-5[9] carries 288 at 36Kb and affords working frequencies above 500MHz [10]. Several researchers have used BRAM to implement different algorithms for classification or matching packet headers, each with their advantages and disadvantages. In addition to many papers being done to address the disadvantages of not effectively investing in size of memory, memory has been sub-divided into several small sizes and used with better investment. In addition to high speed and investment of BRAM and to avoid the circuit complexity, it has been made in simulating Triple Content Addressable Memory (TCAM)[11], design TCAM based on SRAM on FPGA. The TCAM compares the input word with all words stored parallel and outputs the address of the stored matched word. TCAM stores words in three states:0, 1, and X (don’t care 0 or 1)[12]. Commercial TCAMs are built into an application-specific integrated circuit (ASIC), which features high-speed searches. TCAM has been widely used for designing high-speed search engines and has applications in networks, artificial intelligence, lookup engine, image processing, and many other real-time applications.

The most important packet header fields are the source and destination address, source and destination port range, and protocol field. Typically, packet classification or matching works with three types of input data. It can be a prefix to specify an address, range to specify port numbers, or specify protocols with precision. So Ternary content addressable memories (TCAMs) are the best memory for address matching, while port and protocol matching can be implemented using block RAM (BRAMs).[13]
Despite the advantages provided by the TCAM memory, its implementation based on BRAM memory on FPGA, results in disadvantages that the increase in the word size of TCAM bits results in an exponential growth in BRAM memory usage. And due to the minimum depth limitation of BRAM of 512 × 72 bits, it is required several BRAM to implement a large TCAM word size. For example, the design proposed in[14], requires 56 BRAMs of size 36Kb to implement an 18Kb TCAM.

This disadvantage impact can be reduced by sequencing multiple small SRAM blocks in designing TCAM on FPGA. However, this also leads to decrease operational frequencies. This is specifically due to the complicated-extensive indicators routing among BRAMs and good judgment on account of immoderate utilization of BRAMs and complicated precedence encoding devices synthesized in good judgment slices.

3. The proposed IP matching design based on FPGA RAMs

To take advantage of BRAMs on FPGAs and avoid the circuit complexity and flaws that accompanied others’ designs. In this paper, we proposed an IP matching design based on parts of IPv4. They are four parts represented in decimal notation, and each fraction represents a decimal number ranging from 0 to 255 independently of the others. The packet header consists of five fixed fields: source IP address using 4 bytes, destination IP address using 4 bytes, source port number using 2 bytes, destination port number using 2 bytes, and protocols field using 1 byte. Since each byte is independent, it has allocated separate RAM memory to store its contents, also the 16-bits port number has been divided into two bytes to reduce the size of RAM memory as shown in figure 1.

![Figure 1: The shows 10 Distributed RAM to store all range of IP address and destination numbers.](image)

The specific prefixes addresses have been converted to a number of direct addresses and previously-stored into related addressable content RAMs (CAM RAMs) shown in figure 1.

The contents of the packet header have been used as the address to read the content of the distributed CAM RAMs, also used as the input to the comparators that compare these inputs with the RAMs outputs, all the RAMs, and comparators circuits run in parallel. The output of the comparators circuits arrives as input to the AND gate, the input header does not match unless all the comparators’ results are matches.

4. Implementation of the proposed packet Header (PH) Matching

In this work, the Vrtix-7[15] FPGA has been used to implement packet header matching, and according to the use of this new type of FPGA, a new Xilinx tool has been also used, I mean Vivado 2018.2. Xilinx Vivado, is a High-Level Synthesis Kit (HLS), a suite for capturing C ++ based design,
simulation, and synthesis to HDL format. The advantage of using this toolkit also reduces effort in HDL design capture and debugging while allowing flexibility in the final implementation of hardware to meet design limitations. The project summary& Details are shown in figure 2-part a & b and c.

**Figure 2(a):** The schematic diagram of the packet header matching

**Figure 2(b):** Details of CAM RAMs input and output lines
In the running simulation, 80 bits of input data have been used to define malicious clients that should be blocked. These data stored into the 10 CAM RAMs of size $2^8 \times 8$ bits, the “PH_BIT_enb” bit line has been set to write enable to write into the RAMs memory as the initial setting, then the write enables bit, disabled to start the reading the content of memory for the matching process with the new input data. Figure 3, shows the waveforms for, write-enabled, the input data, and the red color waveform of matching signal that change from mismatch to match when new data identical to that previously stored has been entered.

After the success of the design and implementation of the packet header matching which was used in a fast and efficient way to detect intruders, it prevented them from accessing the data. The two Vivado reports and the project summary can be viewed. The utilization Design Information that contains the slice Logic, a summary of Registers, Memory, Clocking, and other information. Table 1 shows the Slice Logic information. The design uses a small number of LUT and memories.
Table 1: the Slice logic information

| Site Type                  | Used | Fixed | Available | Util% |
|---------------------------|------|-------|-----------|------|
| Slice LUTs*               | 355  | 0     | 63400     | 0.56 |
| LUT as Logic              | 35   | 0     | 63400     | 0.06 |
| LUT as Memory             | 320  | 0     | 19000     | 1.68 |
| LUT as Distributed RAM    | 320  | 0     |           |      |
| LUT as Shift Register     | 0    | 0     |           |      |
| Slice Registers           | 11   | 0     | 126800    | <0.01|
| Register as Flip Flop     | 0    | 0     | 126800    | 0.00 |
| Register as Latch         | 1    | 0     | 126800    | <0.01|
| F7 Mixes                  | 160  | 0     | 31700     | 0.50 |
| F8 Mixes                  | 80   | 0     | 15850     | 0.50 |

The parts of report power analysis and report timing summary show the design used low power and the is no timing delay at the inputs or outputs also, there is zero checking multiple clocks register / latch pins. As a result, this report proved there are 81 input ports with no input delay specified; When checking the output delay, there is one port with no output delay. As shown in table 2 parts a and b. As expected, since all CAM RAMs designed to work in parallel.

Table 2(a): part of the power analysis report.

| On-Chip                  | Power (W) | Used | Available | Utilization (%) |
|--------------------------|-----------|------|-----------|-----------------|
| Slice Logic              | 0.813     | 600  | ---       | ---             |
| LUT as Distributed RAM   | 0.505     | 320  | 19000     | 1.68            |
| LUT as Logic             | 0.183     | 35   | 63400     | 0.06            |
| F7/F8 Mixes              | 0.108     | 210  | 63400     | 0.38            |
| BUF                      | 0.006     | 2    | 32        | 6.25            |
| Others                   | 0.000     | 2    | ---       | ---             |
| Register                 | 0.000     | 1    | 126800    | <0.01           |
| Signals                  | 1.697     | 200  | ---       | ---             |
| I/O                      | 0.324     | 83   | 210       | 39.52           |
| Static Power             | 0.103     |      |           |                 |
| Total                    | 2.937     |      |           |                 |

Table 2(b): part of the timing delay report.

6. checking no_output_delay

---

There is 1 port with no output delay specified. (EL28)

There are 0 ports with no output delay but user has a false path constraint

There are 0 ports with no output delay but with a timing clock defined on it or propagating through it

7. checking multiple_clock

---

There are 0 register/latch pins with multiple clocks.
5. Conclusion
In this study, was built TCAM based on RAM. We designed hardware architecture of the parallel process multiple RAM that emulates the behaviors of content addressable memory for packet classification to increases speed of detection of intruders on networks and we achieved low power & no Delay in inputs and output with no circuit complexity design. the reports in table 2 (a)&(b) above showing our result. by using Virtex – 7 FPGA kit.

References
[1] Y. K. Chang and H. C. Chen, “Fast packet classification using recursive endpoint-cutting and bucket compression on FPGA,” Comput. J., vol. 62, no. 2, pp. 198–204, 2019, doi: 10.1093/comjnl/bxy052.
[2] Y. Qi, L. Xu, B. Yang, Y. Xue, and J. Li, “Packet classification algorithms: From theory to practice,” Proc. IEEE INFOCOM, pp. 648–656, 2009, doi: 10.1109/INFCOM.2009.5061972.
[3] W. Eatherton, G. Varghese, and Z. Dittia, “Tree bitmap: hardware/software IP lookups with incremental updates,” ACM SIGCOMM Comput. Commun. Rev., vol. 34, no. 2, pp. 97–122, 2004.
[4] F. Baboescu and G. Varghese, “Scalable packet classification,” ACM SIGCOMM Comput. Commun. Rev., vol. 31, no. 4, pp. 199–210, 2001.
[5] B. Yang, Y. Qi, F. He, Y. Xue, and J. Li, “Discrete bit selection: Towards a bit-level heuristic framework for multi-dimensional packet classification,” Proc. IEEE INFOCOM, pp. 3–4, 2009, doi: 10.1109/INFCOMW.2009.5072130.
[6] W. Yu, S. Sivakumar, and D. Pao, “Pseudo-TCAM: SRAM-based architecture for packet classification in one memory access,” IEEE Netw. Lett., vol. 1, no. 2, pp. 89–92, 2019.
[7] I. Ullah, Z. Ullah, and J.-A. Lee, “Efficient TCAM design based on multipumping-enabled multiported SRAM on FPGA,” IEEE Access, vol. 6, pp. 19940–19947, 2018.
[8] Y. K. Li, and P. Zhou, “Efficient packet classification using TCAMs,” Comput. Networks, vol. 50, no. 18, pp. 3523–3535, 2006.
[9] P. A. Milder, F. Franchetti, J. C. Hoe, and M. Puschel, “Formal datapath representation and manipulation for implementing DSP transforms,” in 2008 45th ACM/IEEE Design Automation Conference, 2008, pp. 385–390.
[10] D. Pao, Y. K. Li, and P. Zhou, “Efficient packet classification using TCAMs,” Comput. Networks, vol. 50, no. 18, pp. 3523–3535, 2006.
[11] Z. Ullah, K. Ilgon, and S. Baeg, “Hybrid partitioned SRAM-based ternary content addressable memory,” IEEE Trans. Circuits Syst. I Regul. Pap., vol. 59, no. 12, pp. 2969–2979, 2012.
[12] T. Virtex and X. T. Fpgas, “Virtex-7 T and XT FPGAs Data Sheet : DC and AC Switching Characteristics,” vol. 183, pp. 1–68, 2013.