Design & Implementation of High Speed Network Devices Using SRL16 Reconfigurable Content Addressable Memory (RCAM)

Qutaiba Ibrahim
Computer Engineering Department, University of Mosul, Iraq

Abstract: *The Content Addressable Memory or CAM is a memory unit that uses content matching instead of addresses. CAMs are used in different networking, telecommunications and storage applications because of their parallel, fast search capabilities. In this paper the behavior of the SRL16 CAM design methodology was described using VHDL and implemented using FPGA technique. Then, the performance of the method was compared to other traditional CAM design methods. The proposed RCAM is configured and used as the main part of different network devices and units (Ethernet Address Filtering Unit, WLAN MAC Address Filtering Unit ,Firewall on Chip (FoC), QoS Packet Classification Unit, Routing Table Search Unit and Network Intrusion Detection System(NIDS) search unit). The successful implementations of this RCAM prove its suitability to be used in different high performance network devices.*

Keywords: *SRL16 Content Addressable Memory, Network Devices, FPGA, Throughput.*

Received May 3, 2009; Accepted May 20, 2010

1. Introduction

Content Addressable Memories or CAMs are a class of parallel pattern matching circuits. In one mode, these circuits operate like standard memory circuits and may be used to store binary data. Unlike standard memory circuits, however, a powerful match mode is also available. This match mode permits all of the data in the CAM device to be searched in parallel. While CAM hardware has been available for decades, its use has typically been in niche applications, embedded in custom designs. Perhaps the most popular application has been in cache controllers for central processing units. Here CAMs are often used to search cache tags in parallel to determine if a cache “hit" or “miss" has occurred. Clearly in this application performance is crucial and parallel search hardware such as a CAM can be used for a good effect [12].

A second and more recent use of CAM hardware is in the networking area [11]. As data packets arrive into a network router, processing of these packets typically depends on the network destination address of the packet. Because of the large number of potential addresses, and the increasing performance demands, CAMs are beginning to become popular in processing network address information. Examples of such applications are: Ethernet & WLAN address filtering, Internet protocol filtering, Data compression, Pattern recognition, Cache tags, Fast look-up for routing tables and Searches for data switches, firewalls, bridges, and routers

Usually , CAM size (in bits) is expressed as the multiplication result of word_width (bit) by the number of words (CAM depth).CAM chips were implemented using different techniques. However, in order to design a reconfigurable CAM, Field Programmable Gate Array (FPGA) technique (because of its flexibility) is used [10, 5].

Basically, designers and researchers in this field use one of three different ways to implement a CAM on FPGA devices [8].

1.1. SRL16 Design

It introduces a methodology for designing flexible, small to medium size CAMs. By using shift register primitives built into an FPGA slice, a reconfigurable Look Up Table (LUT) (two LUTs per slice) is used to implement a single clock cycle read CAM. A 4-bit CAM word fits into each LUT. The most important features of this method are [8]:

- One read clock cycle (or match access time).
- 16 write clock cycles.
- Generic “word width” from four bits up to any multiple by four bit value,
- The CAM depth is a multiple by 16 word value.
- Implemented using 10 complex VHDL code modules.

1.2. Distributed Selectram-Based Implementations

Distributed SelectRAM-based implementations offering large word width and depth adapted specifically for Asynchronous Transfer Mode (ATM) applications. The most important features of this method are [6]:

- Implementation = 10 bits per LUT
- Read operation requires 16 clock cycles
- Write operation requires one clock cycle
- Implemented using 4 complex VHDL code modules.

1.3. CAM using BlockRAM memory

This solution is optimal for applications requiring one or two clock cycles for both read and write and 8-bit width. This methodology is based upon the True Dual-Port feature of the FPGA's block memories. The most important features of this method are [3]:

- Implementation = Block RAM memory (CAM16x8 in each memory)
- Read operation requires one clock cycle
- Write operation requires one clock cycle after one erase cycle.
- Implemented using 8 complex VHDL code modules.
- The CAM design methods mentioned earlier can be evaluated using the following metrics [8]:
    - Maximum frequency: It can be defined as the inverse of the maximum electronic propagation delay through the designed CAM circuits.
    - Number of *'Match'* clock cycles: It reflects the speed at which the CAM gives the match result between the input and stored data.
    - Number of *'Write'* cycles: It reflects the speed at which the 'CAM' stores (or write) a new data word into the CAM.
    - Area Utilization: It can be defined as the amount of FPGA chip resources allocated for the CAM parts.

In this paper, SRL16 CAM design method is adopted. The design was implemented on different Xilinx platforms and used efficiently to build various network devices.

2. Description of the Suggested Array Method

The SRL16 CAMs is built according to VHDL Design Dataflow Style Architecture [4], which specifies the circuit as a concurrent representation of the flow of data through the circuit. In the dataflow approach, circuits are described by showing the input and output relationships between the various built-in components of the VHDL language. The dataflow style works fine for small and primitive circuits. But as circuits become more complicated, it is usually advantageous to switch to behavioral style models (this could explain the use of many complex VHDL modules in the previous design methods).

The reference design described in this method has an adjustable word width and depth. This reference design also proposes an encoding module to generate the output address. Because of the hierarchical VHDL code structure, it is easy to use this module for various CAM sizes according to the designers need. Usually the encoder requires only one additional clock cycle to generate both the output address and a match flag. A wide OR gate of all the decoded addresses creates this match flag.

Write Operation: The Shift Register mode is used to store new data in a location. 16 clock cycles shift the result of the comparison between the input data to be written and a 4-bit down counter (16 states). If the counter value equals the input data value then a "1" is shifted into the SRL16E primitive. Otherwise, a "0" is shifted in. The result is a 4-bit decoder in each LUT after the 16 clock cycles. Figure 1 is an 8-bit CAM word write operation.

Read Operation: The input data to be compared is used as an address of the Shift Register. Only one out of the 16 locations in the SRL16E has a "1" corresponding to the data stored previously.
If the input data addresses this location, a match is found. The carry-chain will propagate this "1" (wide AND configuration) and if all the SRL16E in a particular slice column output a "1", a match is found. The global output of the carry-chain is one line of the CAM decoded address. Figure 1 illustrates the algorithm of the VHDL design.

3. VHDL Description of the Suggested CAM

The operation of the SRL16 CAM was described in VHDL and the timing diagram of the different cases is shown in Figure 2. The pin diagram of the suggested CAM is shown in Figure 3, while table 1 lists the action(s) expected by each pin(s).

The timing diagram shows that the suggested CAM gives the matching result in one clock cycle. However, Write operation could be finished in 16 clock cycle.

4. FPGA Implementation of SRL16 CAM

Experimentally, the adopted CAM design was implemented on Vertex2 (XC2V2000) platform. The performance of the SRL16 method was investigated by measuring the area utilization of the device and the maximum frequency against the CAM size. The size of the CAM could be changed by varying either the word length or the number of words (CAM depth). Figures 4 to 6 show the effect of such variations.

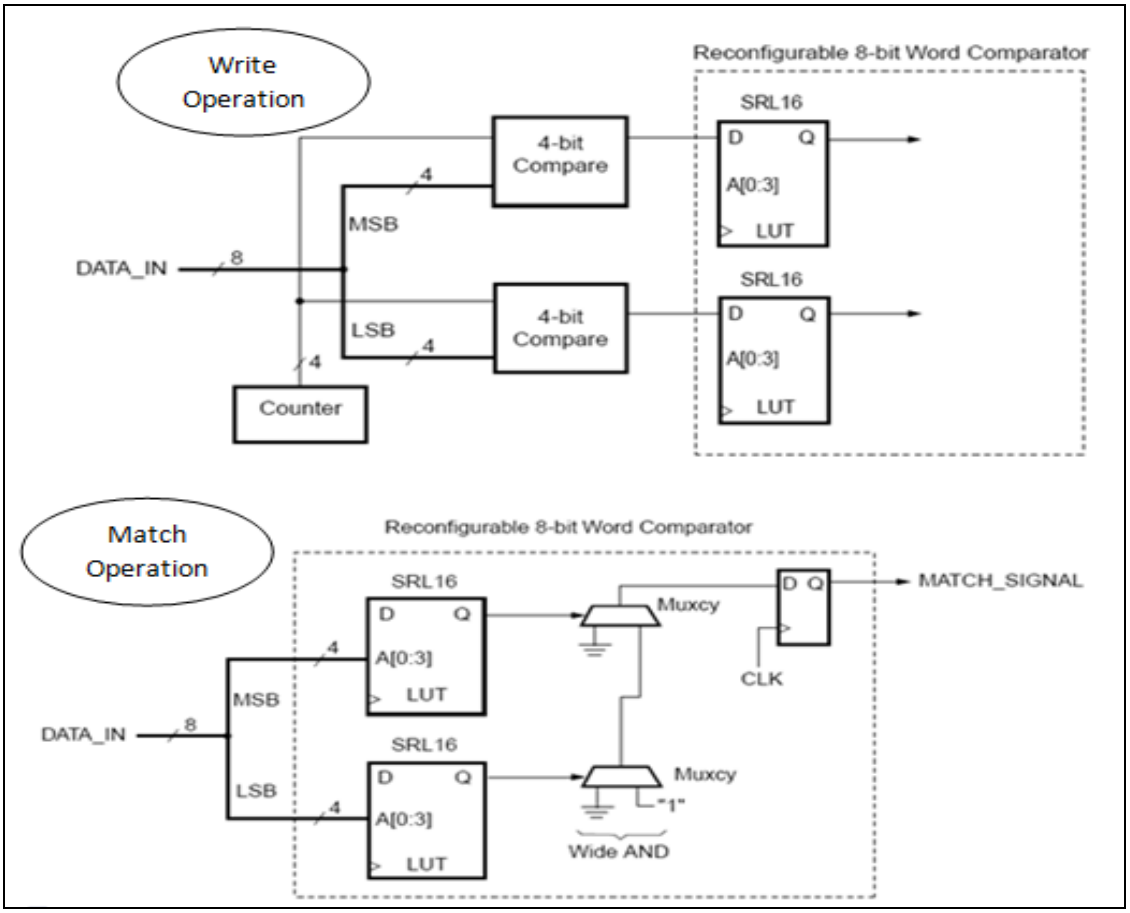

Figure 1. The basic structure of the SRL16 design method.

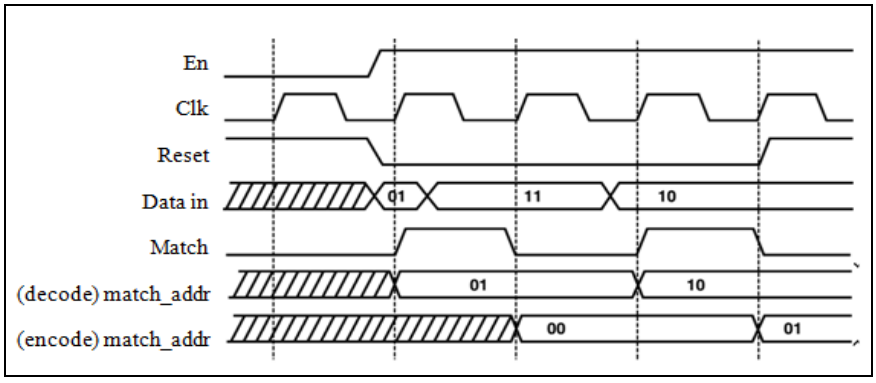

Figure 2. The timing diagram of SRL16 CAM.

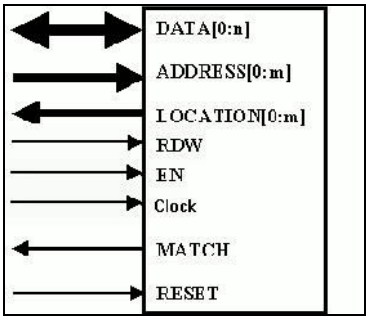

Figure 3. The pin diagram of SRL16 CAM.

Table 1. SRL16 CAM Pin's actions.

Pin	Description	Function
D[0-n]	Data lines of the CAM	Transfer data to/from the unit
A[0-m]	Address lines of the unit	Give the address of an array Location
EN	Operational mode of the unit	'1' Match mode '0' Read or Write mode
RDW	Read/Write signal	'1' Read form unit '0' Write to unit
Match	Comparison result	'1' Match found '0' Match not found
Location[0-m]	Match address	Gives the location of the matched word
Reset	Unit activity	'0' CAM is active '1' CAM is not active

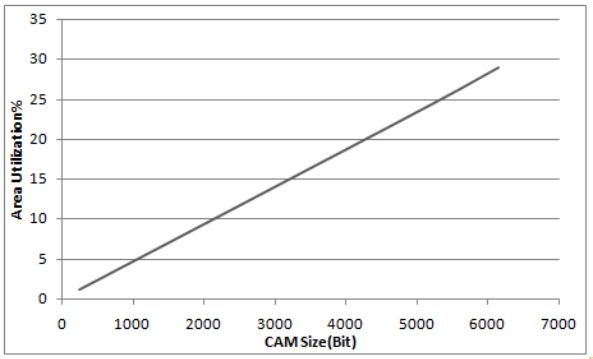

Figure 4. Effect of varying CAM size on Utilization.

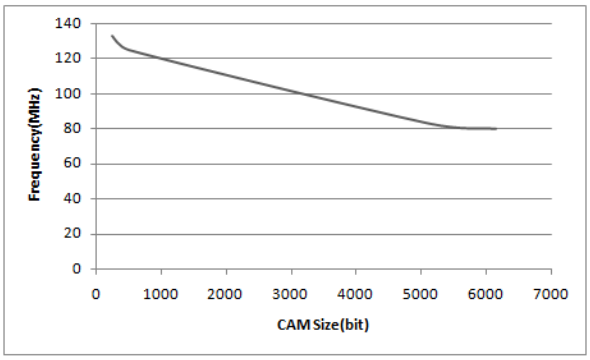

Figure 5. Effect of varying CAM size on match Frequency.

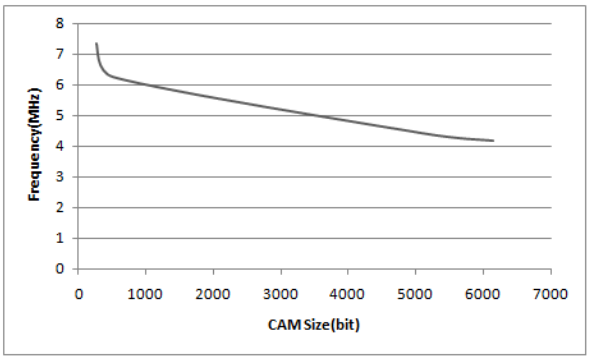

Figure 6. Effect of varying CAM size on write Frequency.

The following remarks could be extracted from the figures:

- Area utilization of the device increases with the increment in the CAM size. The calculations show that each 4 stored bits occupies a single slice of the chip.
- The maximum working frequency decreases with the increment in the CAM size. This could be caused by the increment in the electronic propagation delay results from the additional logic circuits.

5. Comparing the SRL16 Method Against Other CAM Design Methods

In this section, the performance of the adopted method is compared with the previously mentioned CAM design methods [3, 8]. Table 2 summarizes the comparison results among various methods to build a (32x8 bit) CAM, which was then implemented on a Vertex (XCV50) platform.

Table 2. Comparisons with other Design Methods.

SRL16 Method (CAM 32 x 8) Percentage of Slices: 13% Maximum Frequency : 125 MHz (for Read and Match) and 6.25 MHz for (Write)
Block RAM Method (CAM32x8) Two block RAMs and 16 slice Maximum Frequency :90 MHz for (read and write) and 222 MHz (Match)
Distributed SelectRAM(CAM256x16) Percentage of Slices: 5% Maximum Frequency : 83 MHz for (read and write) and 7 MHz for (Match)

It is obvious that the main disadvantage of the investigated method is the high area utilization as compared to the other methods. In order to give a better understanding of the benefits of the current method, the characteristics of the different CAM design methods was summarized in Table 3.

Table 3. The characteristics of the different CAM design methods.

Method	Advantages	Disadvantages
SRL16E	One clock cycle for Match or Read & Flexible CAM size	Require 16 clock cycle for Write
Distributed SelectRAM	One Clock Cycle for Read or Write & Used for large CAM size	Require 16 clock cycle for Match, Built for specific application (ATM)
Block RAM	One clock cycle for Match, Read and Two clock cycles for Write, High working frequency & Optimized area utilization	Word length is limited to 8 bit only.

6. Network Applications of SRL16 RCAM

In this section, SRL16 RCAM is used to build several network devices and units. The purpose is to investigate its ability to be a flexible solution in different situations. For better performance, all the suggested designs were implemented in FPGA using Vertex2 (XC2V2000) platform. Throughput values for all devices are calculated as the product of ((bus width × maximum frequency)/ number of clocks to finish a MATCH cycle).

6.1. Ethernet Address Filtering Unit

In order to enhance the security of the networks, modern Ethernet switches have the ability to achieve layer 2 packet filtering separately by each of its ports [1]. This is done by supplying these ports with a layer 2 packet filtering unit. The action of this unit is to compare the source MAC address portion of each

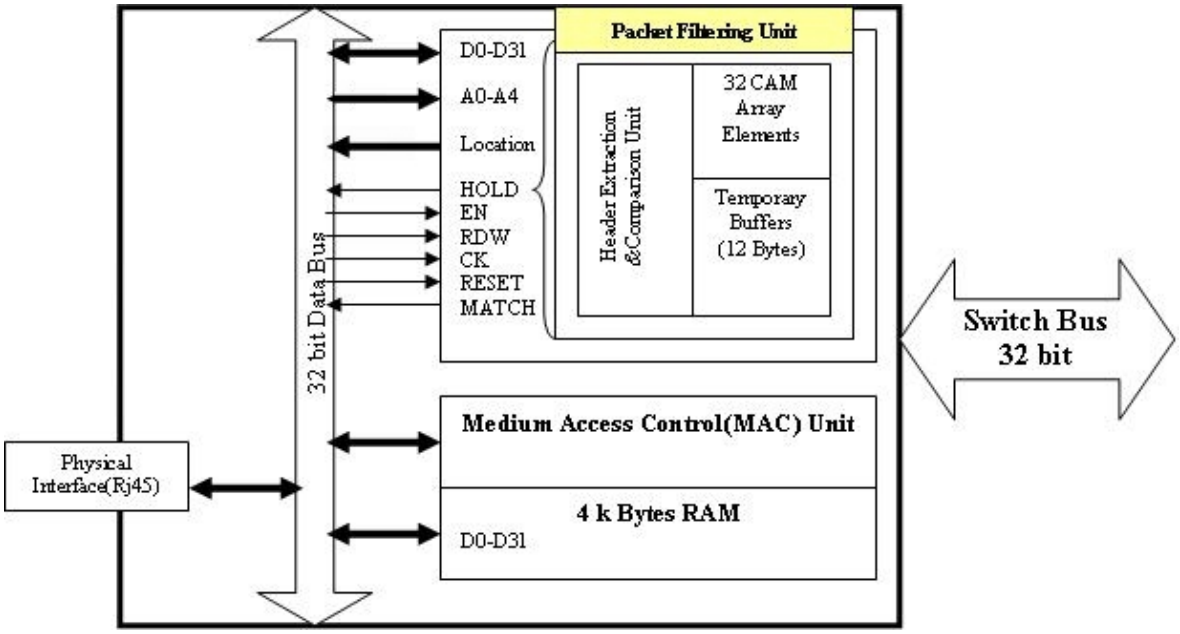

Figure 7. Ethernet Port Supplied with the suggested Address Filtering Unit.

incoming Ethernet packet with pre-stored addresses. If a match is found, then the packet is passed to the switch, otherwise, it is discarded. In this unit, RCAM is considered to be the heart of the design, see figure 7.

A VHDL module for RCAM size (48x32) is built to describe the operation of this unit.The statistics obtained from the FPGA implementation report using Vertex2 (XC2V2000) platform indicated that Ethernet packet filtering unit occupied (5%) of the chip and it may works at a maximum frequency value of (261 MHz). Throughput of this device is (1.15 Gbps).

6.2. WLAN MAC Address Filtering

Whereas one Access Point (AP) or a group of APs can be identified by a Service Set IDentifier (SSID), a client computer can be identified by the unique Medium Access Control (MAC) address of its 802.11 NIC. To increase the security of an 802.11 network, each AP can be programmed with a list of MAC addresses associated with the client computers that are allowed to access the AP. If a client's MAC address is not included in this list, the client is not allowed to associate with the AP. MAC address filtering (along with SSIDs) provides improved security, but it is best suited to small networks where the MAC address list can be efficiently managed. Each AP must be manually programmed with a list of MAC addresses, and the list must be kept up-to-date. In practice, the manageable number of MAC addresses filtered is likely to be less than 255[6].

In this application, RCAM is assumed to be a part of an access point and responsible for the MAC address filtering operation. RCAM is used to store the 48 bit MAC addresses of the authorized clients and it performs a comparison operation with the source address field of any arrived packet. In the VHDL module, the RCAM size is set to be (48(word width) × 128 (CAM depth)) and its data bus is assumed to has (32bit) width. The operation of the MAC address filtering unit begin by receiving the required *source address* (as extracted by the WLAN port) field from the incoming *Request to Send* (RTS) packet (as shown in figure 8).

Frame Control (2 Bytes)	Duration (2 Bytes)	Destination Address (6 Bytes)	Source Address (6 Bytes)	CRC (4 Bytes)

Figure 8. Frame Format of RTS Packet.

This step requires 2 clock cycles. Then, the extracted field is compared with all the pre-stored MAC values to discover its acceptance statues. Meanwhile, the packet transfer operation is paused by the 'HOLD' signal from the MAC address filtering unit. After finishing the comparison procedure (in *one* clock cycle), the filtering unit either generates an accept (MATCH signal is "1") or not-accept (MATCH signal is "0") signal to decide accepting or rejecting the join request packet. The unit may work in another mode, in which, permitted MAC address values is written to (or read from) the CAM's array. This mode is initiated when the 'EN' input signal is set to '0'. In this mode, two clock cycles is needed to finish a single *read/write* process. The block diagram of the proposed unit is shown in Figure 9.

The comparison and timing unit is responsible for the management of the source address reception procedure (into the temporary buffers), handling the Read/Write procedures to/from the CAM's array and performing the comparison operation.

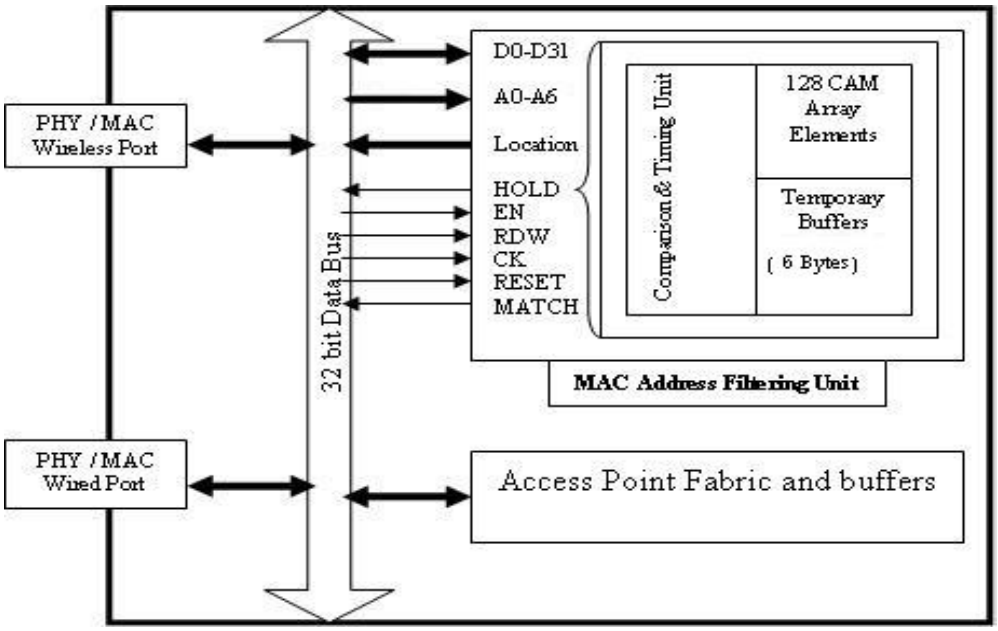

Figure 9. WLAN access point supplied with the suggested Address Filtering Unit.

Running the VHDL simulation shows that it needs an extra three clock cycles to handle packet checking procedure. The proposed MAC address filtering unit is implemented in FPGA using Vertex2 (XC2V2000) platform. The statistics obtained from the implementation report indicated that MAC address filtering unit occupied (20 %) of the chip and it may works at a maximum frequency value of (230 MHz) with a throughput value of (2.45 Gbps).

6.3. Firewall on Chip (FoC) Unit

A firewall is a set of related devices, deployed strategically between a company's private network and one or more unsecured networks (particularly including the Internet). A firewall functions to protect the resources of the private network from users on the outside (external) networks. The first line of defense in firewall protection, and the most basic, is the packet filter firewall. Packet filters examine incoming and outgoing packets and apply a fixed set of rules to the packets to determine whether they will be allowed to pass. The packet filter firewall is typically very fast because it does not examine the data in the packet. It simply examines the type of the packet along with the source and destination IP addresses (32 bit each) and port numbers (16 bit each), and then it applies the filtering rules[9].

Firewall on Chip (FoC) is a packet filtering unit consists of a single chip only. This unit could be a part of a larger network device such as switches or routers [1]. FoC could be designed to have a similar procedure to that mentioned earlier in this section.

A VHDL module RCAM size (96x64) is built to perform the actions of a FoC device. It was assumed to have a system data bus of 32bit width. The RCAM is configured to have 64 filtering rules (CAM depth), each rule has a 96 bit length (64 bit for both source and destination IPs and 32 bit for both source and destination ports). For design simplicity, it was assumed that *'ANDing'* is the logic relation between the various parts of the rule. This allows treating any rule as one block rather than separated fields. Also, it was assumed that the *Store & Forward* switch (or router) is a part of an *Ethernet* network which has the header shown in figure 10.

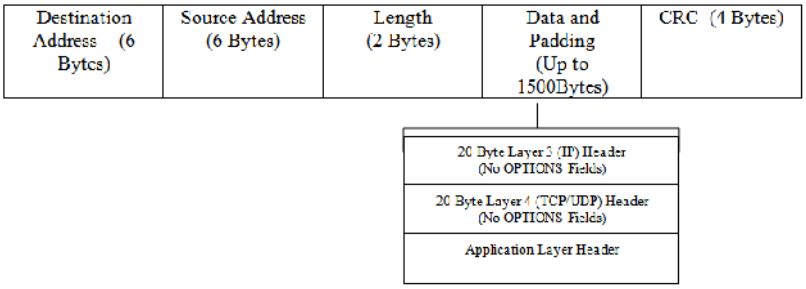

Figure 10. Ethernet Frame Format.

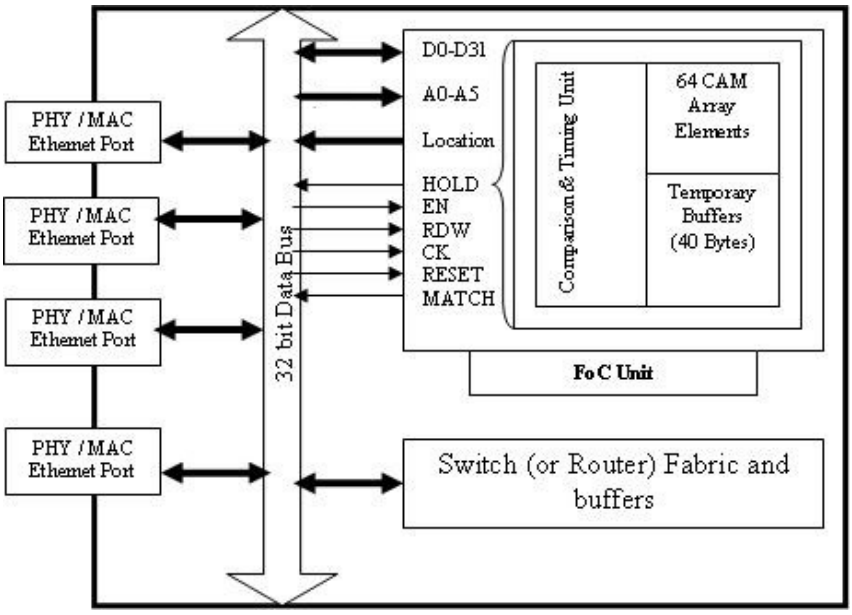

Figure 11. Block Diagram of the Suggested FoC.

The operation of the FoC unit begins on the reception of any packet, and prior to sending it to the switch's buffers. The required fields (Destination and Source IPs, Destination & Source ports) is extracted from the different headers and stored in a temporary buffer inside FoC unit. This step requires 10 clock cycles, after which a HOLD signal is generated to pause the packet transfer procedure. After completing the comparison process (in one clock cycle), FoC either generates a '1' MATCH signal (for access) or '0' MATCH signal (for deny). Also, FoC may work in the programming mode, in which rules are read/written to/from the FoC and needs 3 clock cycles to be finished. Figure 11 Shows the block diagram of the proposed FoC unit.

The proposed FoC unit is implemented in FPGA using Vertex2 (XC2V2000) platform. The statistics obtained from the implementation report indicated that FoC unit occupied (20 %) of the chip and it may works at a maximum frequency value of (237 MHz) with a throughput value of (361 Mbps).

6.4. QoS Packet Classification Unit

Packet classification is crucial for Quality of Service (QoS) because it enables the switch or router to differentiate the traffic streams and treat them differently depending on their individual requirements. All QoS is based on a classification scheme – the more thorough and flexible the classification capabilities, the more advanced the capability to support QoS. Any special treatment of the traffic (prioritization, scheduling, bandwidth distribution, filtering) may be based on the result from the classifier[12].

The higher the throughput of the switch, the more flows there are that require identification for this special treatment. The benefit of the classification scheme rapidly diminishes if the implementation creates its own bottleneck. Classification can be divided into two parts; data extraction, where the relevant fields are extracted from the packet header, and data comparison, where the extracted fields are compared to predefined data. Once a packet can be assigned to a flow, it can be forwarded and queued with an associated class of service that may be defined on a per flow basis. A flow can be uniquely identified by a 4-tuple consisting of source IP address (32 bits), source TCP or UDP port (16 bits), destination IP address (32 bits), destination TCP or UDP port (16 bits) and Type of Service (TOS) (8 bits). This represents a minimum of 104 bits to uniquely identify a flow in IPv4[12].

From the above discussion, it is obvious that RCAM is an essential part in the QoS Packet Classification Unit. It was assumed that the switch's (or router's) ports (in which the Qos units is localized) has 16 priority queues. Our VHDL module is modified to have a (104 × 16) CAM dimensions with a system bus has a 32bit width. Figure 12 shows the Block diagram of the proposed QoS Packet Classification Unit.

The QoS unit task begin when the switch transfer a packet to one of its ports. The unit receives the packet and stores it temporarily to find its destination queue. If a 'Match' is found , then the (4 to 16 bit) decoder enables one of the queues (depending on the 'Location' value) and start the packet transfer operation to that queue. Otherwise, the packet is

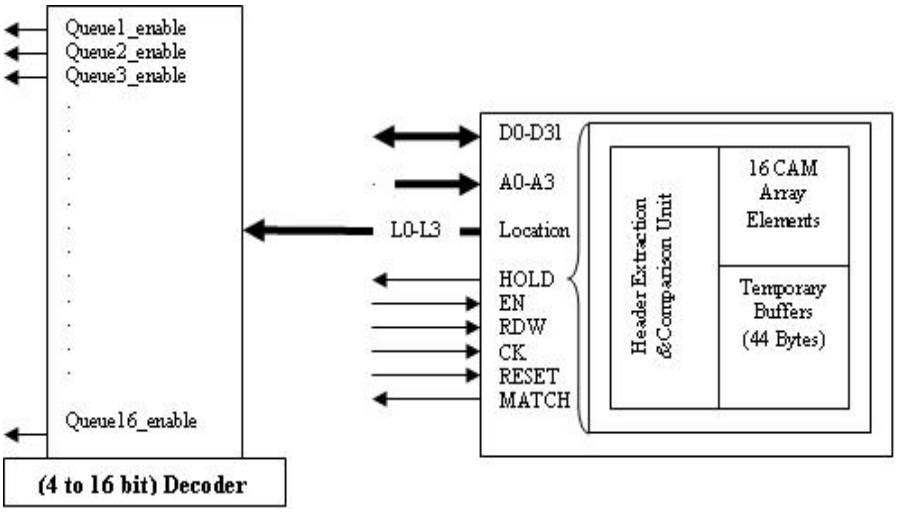

Figure 12. Block Diagram of the Suggested QoS Packet Classification Unit.

transferred to the lowest priority queue. Running the simulation indicates that 12 clocks is needed by the QoS unit (one for comparison and 11 for packet transfer operation to the temporary buffers) to finish packet directing operation. Also , the unit may works in the programming mode, in which rules isread/written to/from the unit's array and needs 4 clock cycles to be finished.

After implementing the proposed QoS unit in FPGA using Vertex2 (XC2V2000) platform, it was found that the unit occupies (5%) of the chip and it may works at a maximum frequency value of (267 MHz) with a throughput value of (370 Mbps).

6.5. Routing Table Search Unit

The routing table is used by the routing module to determine the next-hop address of the packet. Every router keeps a routing table that has one entry for each destination network. The entry consists of the destination network IP address, the shortest distance to reach the destination in hop count, and the next router (next hop) to which the packet should be delivered to reach its final destination, see figure13. The hop count is the number of networks a packet enters to reach its final destination. In an Routing Information Protocol(RIP) algorithm, this value does not exceed 255. A router should have a routing table to consult when a packet is ready to be forwarded[11].

Destination IP address (32 bit)	**Next Hop Address (32 bit)**	**Hop Count (8 bit)**	**Interface number (8 bit)**

Figure 13. Routing table fields.

In order to get a fast routing decision , content addressable memory must be used. Only the destination IP address is stored in the CAM, and it is known here as the search key. The search key is simultaneously compared with all CAM entries and the highest priority match is returned. The corresponding next hop and interface number can then looked up in an associated data RAM, see figure 14.

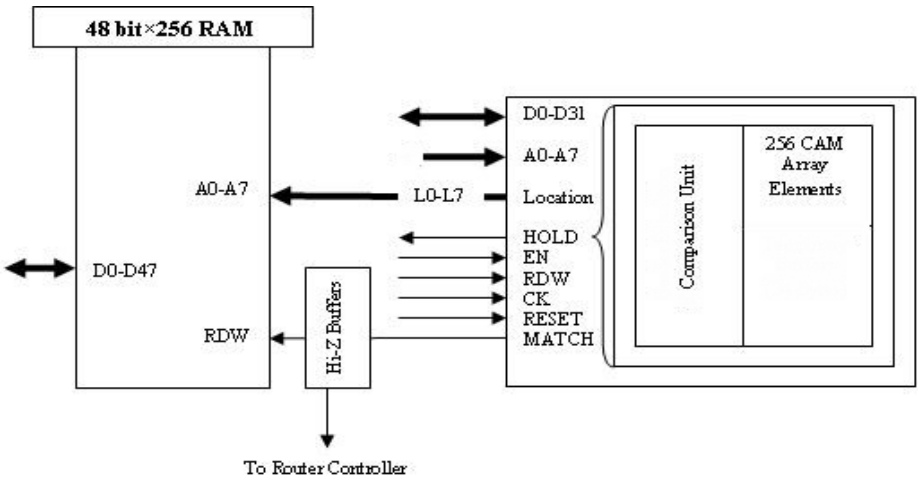

Figure 14. Block diagram of the suggested Routing Table Search Unit.

CAM VHDL module is modified to handle the above situation. The array dimensions was set to be (32 bit × 256) without temporary buffers. When a packet arrived, the header extraction process is done by the router's fabric and the corresponding destination IP address is compared with all CAM elements in one clock cycle. If a 'Match' is found, then the routing table contents is read from the RAM. Otherwise, the Hi-Z buffers become inactive and a signal 'Destination Not Found' is sent to the router's controller. Our simulation results shows that it needs three clock cycles to give the routing decision (one for CAM operation and two for RAM response). In the programming mode operation, both CAM and RAM are activated to store the routing table contents and it needs 3 clock cycles to finish this operation(one for storing IP value in the CAM and two for writing the rest of the table in the RAM).

After implementing the proposed searching unit in FPGA using Vertex2 (XC2V2000) platform, it was found that the unit occupies (26%) of the chip and it may works at a maximum frequency value of (219 MHz) with a throughput value of (2.3 Gbps).

6.6. Network Intrusion Detection System(NIDS)Search Unit

Network intrusion detection systems (NIDS) are an important tool to protect network systems from external attack. NIDS are used to identify and analyze packets that may signify an impending threat to organization's network [5]. This could be done by storing *special signatures* (which represent a threat indicator) then comparing the data of the incoming packets against these signatures. Inspecting incoming packets for tell-tale signatures can be time consuming especially if the number of possible signatures is large[5]. In this manner, RCAM could be used to accelerate the keyword match operation.

In our design, we reconfigure the RCAM to work as the search engine of an NIDS unit. It was configured to have the size of (80bit x 64 words), which represents a 47 *Snort Web Attacks rules* with an average of 10 character per word[15]. We assumed that RCAM operation is assisted by other units which receive packets, extract the data words and characters ,then applying each word to the RCAM to be compared with its contents, see figure15.

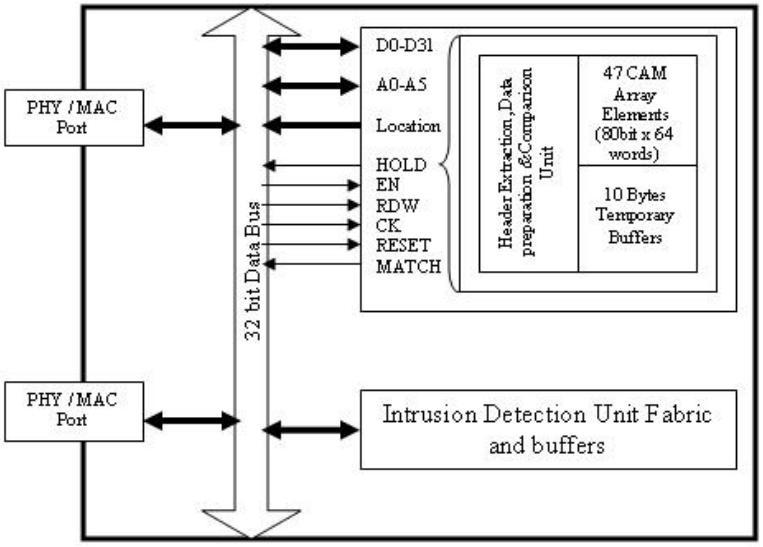

Figure 15. Block diagram of the suggested NIDS.

The data comparison begins after storing the intended word inside the (10 Bytes) temporary buffers, and needs 4 clock cycles to be finished (3 clocks for transferring the 80 bit word through 32 bit data bus and one clock for *Match* operation). In the same manner, it takes 3 clock cycles to finish Read/Write operation of the rules.

The statistics obtained from the FPGA implementation report indicate that RCAM unit occupied (16 %) of the chip area and it may works at a maximum frequency of (271 MHz) with a throughput value of (2.17 Gbps).

7. Conclusions

In this paper, a SRL16 CAM design method was presented. The operation of the design was described in VHDL and implemented using FPGA technique. The successful implementations of various CAM sizes to meet the needs of different network devices, candidate the current method to take its place as a flexible and easy to adopt CAM design method. Also, its high throughput values allows to build high response devices suitable for working in high speed networks.

References

[1] Bandi N., Schneider S., Agrawal D. and El Abbadi A., "Hardware Acceleration of Database Operations Using Content Addressable Memories", *Proceedings of the First International Workshop on Data Management on New Hardware (DaMoN 2005);* June 12, 2005, Baltimore, Maryland, USA.

[2] Brelet J., "Designing Flexible, Fast CAMs with Virtex Family FPGAs", Xilinx Inc., 1999.

[3] Brelet J., "Using Block RAM for High Performance Read/Write CAMs", Xilinx Inc., 2000.

[4] Brown S. and Vranesik Z., "Fundamentals of Digital Logic with VHDL Design", McGraw Hill Inc., 2000.

[5] Cole E., R. Krutz and J. Conley , "Network Security Bible", 1'st Edition, Wiley Publishing Inc., 2005.

[6] Defossez M., "Content Addressable Memory (CAM) in ATM Applications", Xilinx Inc., 2001.

[7] Khan J., and Khwaja A., "Building Secure Wireless Networks with 802.11", Wiley Publishing, Inc, 2003.

[8] Lakshminarayanan K., Rangarajan A. and Venkatachary S.," Algorithms for Advanced Packet Classification with Ternary CAMs", Proceedings of *SIGCOMM'05 Conference* , August 22–26, 2005, Philadelphia, Pennsylvania, USA.

[9] Lockwood J., Naufel T., Turner J. and Taylor D. , "Reprogrammable Network Packet Processing on the Field Programmable Port Extender (FPX)," ACM International Symposium on Field Programmable Gate Arrays (FPGA), pages 87–93, Monterey, CA, USA, Feb. 2001.

[10] Moss B.G., "An Egress Queue Manager For 8x100 Mbps and 1x1Gbps Ethernet Switch Port Controllers", Msc Thesis, Simon Fraser University, 2002.

[11] Pagiamtzis K. and Sheikholeslami A., "Content-Addressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey", IEEE Journal of Solid-State Circuits, Vol. 41, No. 3, March 2006.

[12] Paul S. and Bhunia S., "Reconfigurable Computing Using Content Addressable Memory for Improved Performance and Resource Usage", Proceedings of *DAC 2008Conference* , June 8–13, 2008, Anaheim, California, USA.

[13] Sanguinetti J. and Pursley D., " High-Level Modeling and Hardware Implementation with General Purpose Languages and High level Synthesis", White Paper, Forte Design Systems, 2002.

[14] Taylor D., J. Turner , J. Lockwood , T. Sproull , D. Parlour , "Scalable IP Lookup for Internet Routers", IEEE Journal on Selected Areas in Communications (JSAC), Vol. 21, No. 4, pp. 522-534,2003.

[15] Yi S., Kim B., Oh J., Jang J., Kesidis G. and Chita R., "Memory Efficient Content Filtering Hardware for High Speed Intrusion Detection Systems, Proceedings of *SAC'07 conference*, March 11- 15, 2007, Seoul, Korea

Qutaiba Ali was born in Mosul, Iraq, on October, 1974. He received the B.S. and M.S. degrees from the Department of Electrical Engineering, University of Mosul, Iraq, in 1996 and 1999, respectively. He received his Ph.D. degree from the Computer Engineering Department, University of Mosul, Iraq, in 2006. Since 2000, he has been with the Department of Computer Engineering, Mosul University, Mosul, Iraq, where he is currently an assistance professor. His research interests include computer networks analysis and design, embedded network devices and network security. Dr. Ali instructed many topics (for Post and Undergraduate stage) in computer engineering field during the last ten years and has many publications (more than 33) in numerous journals and conferences. He acquires many awards and appreciations form different parties for excellent teaching and extra scientific research efforts. Also, he was invited to join many respectable scientific organizations such as IEEE, IENG ASTF, WASET and many others. He was participate (as technical committee member) in eleven IEEE conferences in USA, Malaysia, South Korea, China and Egypt and joined the editorial board of five scientific international journals.