LOOK-UP PROCESSOR FOR IPV4/IPV6 PACKET CLASSIFICATION

Tomáš MÁLEK, Bachelor Degree Programme (3)
Dept. of Computer Systems, FIT, BUT
E-mail: xmalek07@stud.fit.vutbr.cz

Supervised by: Ing. Jan Kořenek

ABSTRACT

This paper deals with design and implementation of Look-up processor which performs IPv4 and IPv6 packet classification on the basis of configurable rules. They can be specific e.g. for routing, filtering or attack detection so the target usage of the processor results from their selection. The origin intention is to use it in PC-based router which is being developed under the Liberouter project [1]. The proposed architecture enables achieving multigigabit speed. Its design was implemented in VHDL language and the target technology is FPGA.

1 INTRODUCTION

Internet is a global network connecting millions of independent computers from all over the world and it is not centrally controlled. Many specialized hardware devices, such as routers, switches, monitoring adapters, intrusion detection systems and so on, are needed to integrate into network infrastructure.

In most of them it is necessary to realize packet classification. Information from the header of a packet, which represents the fundamental data unit, is analyzed and on the basis of it an appropriate action, e.g. sending to an interface, filtering, marking as suspicious, exporting to software, is performed. In another case, this information is used only for statistical purposes.

There are many various approaches to packet classification in hardware. They utilize content addressable and static memories [2], hash functions [3] or alternatively general purpose processors [4]. The proposed architecture combines sequential processing by specialized processor with classification rule matching in content addressable memory. It enables dividing a task between two independent but cooperating units. While the first one controls the processing and performs simple classification steps, the second one looks up in the rule table and covers most of comparing operations. Using this technique gives good results for the application of elementary short rules as well as complex long ones which require multiple matching in the table.

2 HARDWARE REALIZATION

The main task of Look-up processor is to find the record describing how to process a packet. In fact, it does not work with neither whole packet nor its whole header but only with some important selected information from packet header, which creates the structure called *unified header*. They are stored in four buffers representing the input of processor.

2.1 COMPUTATION MODEL

The computation model of Look-up machine is logically composed of two main units, *Processing unit* and *CAM block*. The Processing unit analyzes unified headers according to program stored in static memory SSRAM. The program consists of single instructions performing especially various types of comparison tests.

In case that special `CAM` instruction is detected, the control flow of processing of actual unified header is handed over to CAM block, which performs matching in classification rule table represented by content addressable memory TCAM. The match result is the address of next instruction where a program continues in processing. The lookup run is finished by `EXE` instruction, which contains the classification result.

2.2 INSTRUCTION SET

The instruction set is specific for the purposes of matching the classification rules. There are three types of implemented instructions:

- *jump instructions* are especially conditional relative jumps. They compare registers of unified header with specified constants. A mask can be used to select a part of the register that should be compared. Hard jumps are also supported.
- *`CAM` instruction* invokes look-up in classification rule table. It loads a subset of registers from a unified header and finds them in TCAM. Its parameter denotes the subset to be matched, there are 256 possible variants.
- *`EXE` instruction* terminates the classification and sends packet identification along with its parameter to the output queue.

2.3 ARCHITECTURE

The architecture of Look-up processor (Fig. 1) consists of two main units, Processing unit and CAM block. They work quite independently so while CAM block is performing the rule matching request which has been just invoked, the Processing unit is analysing next unified header. Both units need data from shared unified header buffers. By reason of simple control, the time slots are allocated to ensure exclusive access.

Processing unit (Fig. 2) performs program instructions to process unified headers. Its architecture is represented by 8-phase cyclic pipeline, which is essential for high-speed processing. The throughput is increased by higher frequency and also by parallel processing because every pipeline stage can be allocated for different unified header. It enables to analyze up to 8 headers in every moment.

CAM block (Fig. 3) controls TCAM memory and looks up in classification rule table on the basis of requests from Processing unit. Employed memory with its 272-bit width of a word enables fast matching IP addresses and other items from packet header. Moreover, it

assures the communication with unified header buffers and provides their data both for itself and for Processing unit.

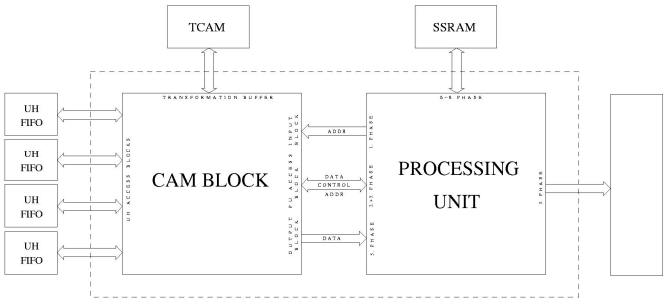

Fig 1: *Block structure*

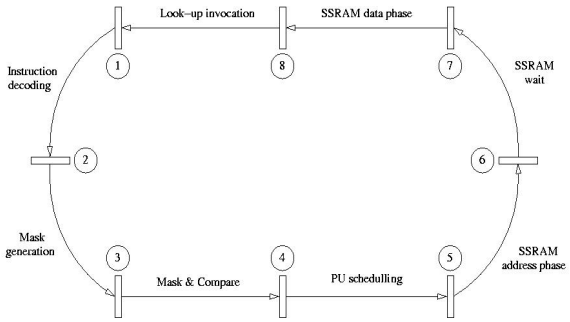

Fig 2: *Processing unit pipeline*

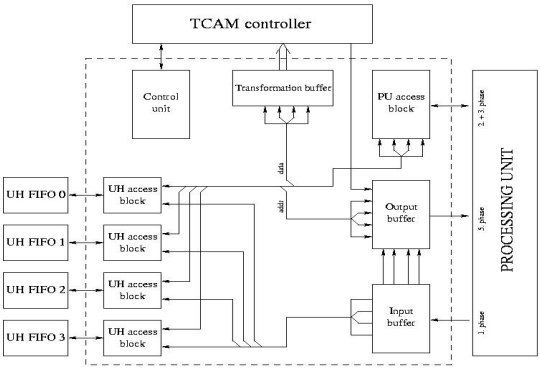

Fig 3: *CAM block*

3 CONCLUSION

This paper introduced the model of universal classification processor which consists of sequential processing unit and fast content addressable memory. Due to well designed architecture, multithreading processing and maximal utilization of shared unified header buffers on the basis of allocated time slots, it is possible to realize packet classification resulting from the requirements of present network infrastructures. The proposed architecture is despite high throughput universal and can be used in various network devices or in other spheres where e.g. classification or string matching is required.

REFERENCES

[1] Liberouter: Liberouter Project WWW page. http://www.liberouter.org, 2006

[2] Antoš, D., Kořenek, J.: String Matching for IPv6 Routers. In SOFSEM 2004: Theory and Practice of Computer Science. Praha, CZ, 2004.

[3] Song, H., Dharmapurikar, S., Turner, J., Lockwood, J.: Fast Hash Table Lookup Using Extended Bloom Filter: An Aid to Network Processing. Philadelphia, PA, 2005.

[4] Walton, S., Hutton, A., Touch, J.: High-Speed Data Paths in Host-Based Routers. Computer, p. 46-52, November 1998.