C-GEP: Adaptive Network Management with Reconfigurable Hardware

Péter Orosz, Tamás Tóthfalusi
Faculty of Informatics
University of Debrecen
Debrecen, Hungary
email: oroszp@unideb.hu

Pál Varga
Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Budapest, Hungary
email: pvarga@tmit.bme.hu

***Abstract*—Carrying out network monitoring tasks remains a continuous challenge, partially because the line rate reaches and exceeds 100 Gbit/s. Besides the increasing data rate, the advent of programmable networks necessitates efficient solutions for supporting packet processing tasks in an adaptive way. Introducing a modification of a protocol or any new protocol in such a flexible infrastructure implies a novel management approach incorporating network monitoring equipment with reconfigurable architecture. The requirement for high throughput and high level of reconfiguration together put Field Programmable Gate Array (FPGA) technology into the focus of high performance networking.**

In this paper, we introduce a programmable, multi-purpose network platform called C-GEP that is based on a reconfigurable architecture. The system consists of two main building blocks: a high performance FPGA-based custom hardware platform and a firmware dedicated for network monitoring. We present the architecture focusing on the system-level integration of specific packet processors. The integration of processing building blocks into one high performance system has great challenges. These are primarily related to specific, limiting factors of system resources – which we discuss also in this paper.

Keywords—network management; network monitoring; 100 Gbit/s Ethernet; traffic analysis; reconfigurable hardware; Field Programmable Gate Array

I. INTRODUCTION

Core network infrastructures are one of the first adopters of high performance transmission technologies. The primary factor that drives the evolution of these critical infrastructures is the emergence of new services requiring increased bandwidth. These days, considering the global Internet traffic mix, real time media services (e.g., VoD, IPTV, OTT video, video conferencing, etc) are the top consumers of network capacity. Due to the real time requirement of these services, there is a raised expectation about network performance. It is not just the capacity, but other transmission properties that should be kept under tight control, i.e., loss, delay and delay variation. In order to provide an uninterrupted, high quality service, providers apply a wide scale of monitoring tools and metrics to measure the network performance. There are many critical network management tasks that require packet level network monitoring: detecting and localizing network faults and bottlenecks, measuring quality of service (QoS) metrics (packet delay, loss, jitter and reordering), performance analysis, misbehavior detection, etc. To assess quality of service (QoS) level for real time applications, a continuous packet-level flow monitoring is a common practice. In order to control the transmission properties throughout the network for an increasing scale of time sensitive applications, distributed monitoring of a predefined set of network links and nodes became an essential element of professional network management.

In this paper, we introduce a high performance, lossless network monitoring system called C-GEP that is based on a reconfigurable architecture. The system consists of two main building blocks: a high performance Field Programmable Gate Array (FPGA)-based custom hardware platform and a firmware dedicated for network monitoring. The reconfigurable property of the FPGA chip enables to turn the C-GEP hardware platform into a high performance networking device, e.g., network monitor, switch, router, firewall or intrusion detection system. Nevertheless, as a network monitoring system, it supports distributed and lossless packet level monitoring of Ethernet links up to 100 Gbit/s.

Distributed monitoring implies multiple synchronized instances of the C-GEP device that we call probes in the context of network measurement. Reconfiguration of any part of a hardware-based network monitor got more focus by the emergence of Software Defined Network (SDN) infrastructures with the primary design principle of network programmability. Introducing a modification of a protocol or any new protocol in such a flexible infrastructure is a vendor independent task. This raises the necessity of an adaptive network management approach that features network monitoring devices incorporating reconfigurable hardware elements. The requirement for high throughput and high level of reconfiguration together ended up in the design and implementation of the presented system.

The rest of the paper is organized as follows. Section II reviews the previous works related to high performance network monitoring, including research results and hardware accelerated implementations. In Section III, we introduce the monitoring system detailing its architecture, functions and performance parameters. Section IV gives an overview of the challenges related to system level integration of the processing stages, i.e., lossless packet capture, high precision clock synchronization, packet parsing and classification, and reduction of measurement data. Finally, we conclude the results and experiences in Section V.

II. Related Work

There is only a limited number of publications that involve system level presentation of high-performance network monitor architectures based on reconfigurable hardware. Typically these are about packet processing tasks (i.e., packet parsing, packet classification and payload inspection) that are part of some specific networking equipment (e.g., network monitor, firewall, IDS, etc.) form their own specific research fields, with very small system level integration effort. Nevertheless, the system level integration of specific packet processors introduces new challenges. These are primarily related to system resources that often set up constraints to the integration of processing building blocks into one high performance system. Handling the increasing data rate of the evolved transmission technologies emphasizes the requirement for an integrated approach of hardware-based packet processing. In this section, we review major contributions of the mentioned research fields and show some examples of the integration efforts related to reconfigurable architectures and network measurement.

Michael Attig *et al.* [1] introduced a high-level parsing description language. Source codes can be compiled to a Virtex-7 FPGA device to perform packet parsing at 400 Gbit/s data rate. Viktor Pus *et al.* [2] proposed a parser that is manually optimized for latency and chip area, operating at more than 100 Gbit/s data rate. Their introduced parsing engine provides the classic 5-tuple protocol metadata for each processed packet. Gordon Brebner *et al.* [3] proposed an object-oriented language for the compilation of high throughput packet processing engines. The compiler generates a pipeline architecture, which supports operation at 100 Gbit/s data rate. Thilan Ganegedara *et al.* [4] presented a 400 Gbit/s throughput capable architecture for a 5-tuple based classification module. The work uses four instances of a 100 Gbit/s engine, where each engine is also based on pipeline architecture. Weirong Jiang *et al.* [5] introduced a 12-tuple based classifier solution. The FPGA implementation is able to operate at 40 Gbit/s line rate combined with a 1k-element rule set. The proposed system is based on a multi-pipeline architecture. Jeffrey Fong *et al.* [6] proposed a one-chip solution for a complex classifier system, namely ParaSplit. The implementation can operate even at 1 Tbit/s throughput using multiple module instances. NetFPGA [7] is an open platform for prototyping line-rate packet processing systems. The NetFPGA-10G board is based on a Virtex-5 device, which features four 10 Gbit/s interfaces. Since the platform is primarily dedicated for research purposes, it provides base for many publications. The main NetFPGA-10G projects implement switch [8] functions operating at 10 Gbit/s data rate. These solutions are based on Microblaze to perform header field extraction and packet modification tasks. Microblaze [9] is an FPGA-based, 32-bit RISC architecture soft-processor. NetFPGA is an efficient hardware platform for monitoring systems. However, the presented works operate at 1 or 10 Gbit/s. Buffer monitoring system [10] is designed to monitor the utilization of packet buffers in switches and routers. The open source work monitors the packet arrival/departure/drop events on the incoming queues, and sends it to a software module. Alfio Lambardo *et al.* [11] developed a traffic monitor system to measure the amount and type of packets carried over a network. The incoming traffic is routed to the CPU, and the type of packets is determined based on the IPv4 header protocol field. Gianni Antichi *et al.* [12] designed a passive monitoring system, providing accurate timestamping (nanosecond accuracy). The system contains a 5-tuple based classification engine, where the fitted packets are routed to the CPU for further processing. Beyond the presented publications, major vendors of network measurement equipment also involve the FPGA technology to implement hardware accelerated packet-level monitoring. For 100 Gbit/s networks, Endace designed a system called Network Visibility Headend [13] that de-multiplexes 100 Gbit/s ingress traffic to multiple 10 Gbit/s egress interfaces. However, their solution requires multiple 10 Gbit/s monitor probes to accept distributed packets and perform packet parsing and classification – meaning that the 100 Gbit/s device merely serves as a 100/10 de-multiplexer.

III. Architecture of the C-GEP Monitoring System

C-GEP is a multi-purpose, programmable, FPGA-based Gigabit Ethernet Platform – the successor of C-Board [14]. Its main purpose is to provide means for handling 1, 10, 40 and 100 Gbit/s Ethernet traffic in a flexible manner. The architecture allows various networking applications for implementation – from SDN devices to media gateways; from traffic generators to DPI (Deep Packet Inspection) support. The first prototypes were programmed to work as monitoring probes for traffic measurement. The C-GEP functions well in this environment, due to its capabilities for lossless packet capture (even at 100 Gbit/s); ns-precise timestamping (when equipped with a built-in atomic clock); and packet filtering / forwarding based on complex rules (see Fig. 1).

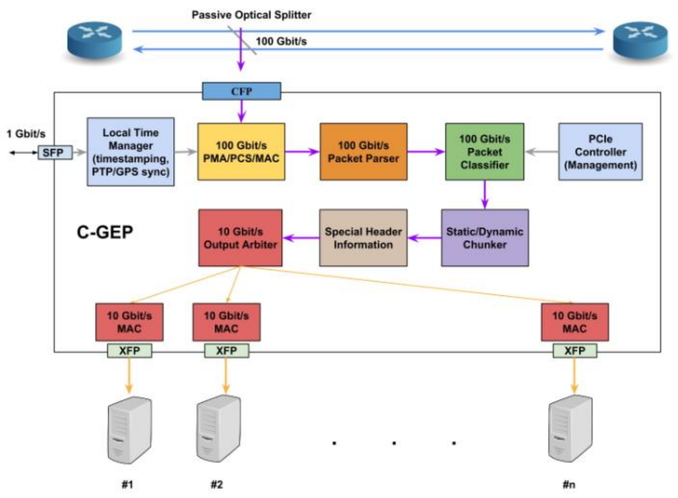


Fig. 1. Logical layout of the C-GEP monitoring system

A. *Implementing 100 Gbit/s Physical Layer and Media Access Control in a reconfigurable architecture*

Based on a reconfigurable architecture such as an FPGA, we have the possibility to design and implement even the lowest level packet processing tasks. In such a way, we can provide custom functions and I/Os to retrieve low-level, packet-related information to determine arrival time on wire, for example. Accordingly, we implemented the Physical Coding Sublayer (PCS) and the Media Access Control (MAC) inside the FPGA chip, which are the first stages within the internal data path. Nevertheless, there are many FPGA-based Intellectual Property (IP) cores on the market implementing 100 Gbit/s Ethernet MAC. Such IP-cores (considering today's price level), are very expensive and therefore contribute significantly to the price of the monitoring device.

The C-GEP board connects to the 100 Gbit/s core network using a CFP (C Form-factor Pluggable) transceiver. The CFP module is directly wired to the pins of the FPGA chip on the board. Within the Virtex-6 chip the high-throughput capable lines are handled with GTH I/O interfaces. The MAC module operates in non-segmented mode, using 512-bit wide data path combined with 312.5 MHz core frequency. In this operation mode, each start byte of the outgoing data is synchronized to the first byte of the 512-bit word. The segmented operation mode, as alternative for a lower frequency design, enables a packet to start in every 8-byte boundary of the 512-bit word. Since the subsequent processing phases operate on predefined packet header fields, segmented mode requires complex synchronization logic during metadata propagation. On the other hand,, the internal design of a non-segmented MAC module occupies more buffer logic than the segmented mode.

B. *Distributed Monitoring: Synchronizing Probes' Clocks*

In traffic analysis, high resolution and high precision timestamping is a basic requirement to evaluate QoS metrics (such as one way and round trip delays and delay variation in a network path) and to precisely detect or reconstruct network events (i.e., request-response pairs).

All of these management tasks get extreme importance when evaluating network performance against Service Level Agreements (SLAs). Another example of high synchronization accuracy is monitoring multiple links of a router. This is the case of the performance evaluation of a routing device. In this arrangement, each link is measured by a dedicated monitoring probe. A packet seen by one probe may be forwarded to an outgoing interface of the router that is monitored by another probe. Typical order of the forwarding delay in today's core routers is in the 0.5-5 microseconds range. This low level of delay demands for highly accurate clock synchronization between monitoring probes.

Monitoring systems should be prepared for the worst case scenario, and therefore internal clocks have to be synchronized so that the order of message pairs can be clearly determined. There are several solutions for this purpose (e.g., Network Time Protocol - NTP, Precision Time Protocol - PTP and Global Positioning System - GPS), providing different levels of accuracy. Since NTP is typically implemented as a software solution, it is not capable to provide synchronization accuracy better that 1 ms in LAN environment, and 100-1000 ms in longer distances. Accordingly, the software- and network-related constraints do not enable its application in high performance measurement systems. In contrast to NTP, PTP has significantly higher accuracy, but requires hardware support in each device throughout the network path. Monitoring systems involve both GPS and PTP as an enabling technology to support high accuracy clock synchronization throughout the distributed system.

Considering network monitoring, the properties and the performance of a timestamping engine is determined by the maximum rate of packet arrivals and the type of measurements. In the presented monitoring system we implemented GPS- and PTP-based clock synchronization supported by a high precision on-board atomic clock device. The C-GEP system is able to synchronize its high precision local clock with a PTPv1 (IEEE 1588-2002 compliant [15]) master device. The monitor probe operates as slave in the PTPv1 communication and uses one of its Gigabit Ethernet interfaces for this purpose. Clock synchronization mechanism was tested through software as well as hardware solutions. The software solution was a free PTPv1 Master implementation, namely *ptpd* [16]. In addition, our research group implemented a hardware solution for the PTPv1 tests, using a NetFPGA-1G board [7] and a chip scale atomic clock [17].

C. *Parsing Packet Headers*

The second stage within FPGA's internal data path is the packet parser module, which maintains the 512-bit wide path in each of its sub-modules. To achieve the target throughput, the parser engine has to operate at a core frequency of 312.5 MHz. Since the architecture of the FPGA devices is designed to support a wide range of processing tasks, its physical resources may set up limitations for certain type of applications. One of these drawbacks is the maximum operating frequency for the connection path, which is typically not higher than 400 MHz, even in a high-end device. This limit is against a high-throughput design goal, and requires target specific design perspective.

The operation of the parser engine is based on a parse graph, which is a combination of predefined header structures. The parse graph defines the possible state transitions during the parsing process (see Fig. 3). According to the scalability of the parse graph, the engine can be reconfigured at compilation or run time. Publications related to high performance packet parsing [1][2][3] commonly propose solutions based on reconfigurable parse graphs, in which the identifiable protocol-structure is reconfigurable through offline algorithms using a specified object oriented language. These header definition languages are designed for handling protocol structures with high-level programming tools. Reconfiguring the parse graph during online operation requires a synchronized update of the filter rules inside the classification engine. As an example, let's assume that IPv4 header extraction is replaced with IPv6 header recognition.

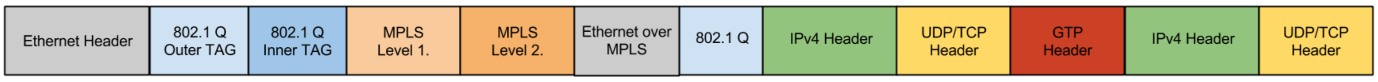


Fig. 2. Example for the header structure of a supported multi-encapsulated packet

After the end of the upgrade process, the parser engine provides IPv6 address fields instead of IPv4 ones. This operation turns the parser and classifier engines into an inconsistent state relative to each other, since the classification engine is still prepared for IPv4 filtering. To avoid packet loss due to the inconsistent processing state, the monitored links have to be redirected to a bypass route during the reconfiguration process. This solution may result in a false analysis result since unrelated traffic can be processed as part of a monitored flow, for example. In addition, the suddenly increased amount of data has to be stored and processed, otherwise packet loss occurs. To avoid the inconsistency, our packet parser graph is reconfigurable in compilation time only.

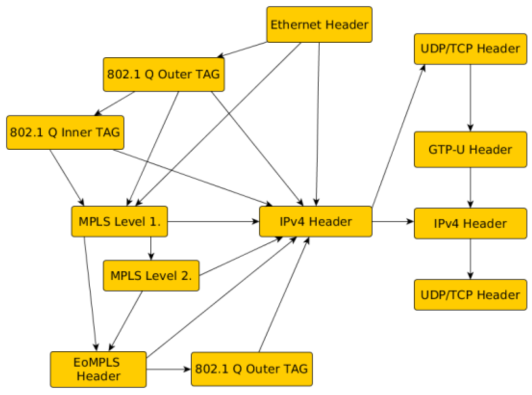


Fig. 3. Packet parser graph

Considering today's networking trends, evolving network services result in the appearance of new protocols or protocol fields at an increasing frequency. To cover the spectrum of network protocols present in core IP networks, our parser engine enables to identify multi-encapsulated packets beyond decoding the classical 5-tuple, i.e., IP address pair, port pair and transport protocol. Therefore, the engine extracts 14-tuple (protocol fields), which are: outer VLAN tag, inner VLAN tag (QinQ), MPLS tags (two levels), EoMPLS, VLAN tag in EoMPLS, IPv4 header, UDP or TCP header (see Fig. 2). The dynamic length of the IPv4 header using optional fields is also handled during the parsing process. The decoded protocol metadata are synchronized to the original packet, and buffered for further processing.

For assessing QoS of a network service, lossless packet processing is an essential requirement. Since the 100 Gbit/s MAC module operates in non-segmented mode, every 512-bit word (and therefore every clock cycle) can contain a new, minimum-sized packet. This property dictates a very strict timing constraint for the parser engine, which has to decode the header structure within one clock cycle. Using buffers or storage memory is not a viable option at this core frequency, because the intra-chip routing to the memory elements is often a timing critical point. In addition, memory modules are just temporal solutions, since transient traffic – such as a burst of minimum-sized packets – can cause overflow and packet drop. According to the chip limitations and the target tasks, the processing steps of the parsing engine are designed and integrated into a pipelined architecture. Each pipeline stage has a well-defined task during packet parsing to handle the identification and extraction of an embedded header. Since the core frequency is critical, one stage must contain simple operations on a predefined header. The pipeline structure in the C-GEP is also fixed at compilation time. To manage and follow the start of the headers, simple indices are calculated and propagated through the processing phases. At the end of the pipeline, a read stage protects the parse engine against overflow.

The output (i.e., the pre-processed traffic) of a monitoring system is always stored in a database for post-processing and statistics calculation. To decrease the amount of stored data, cutting the packet payload is a common solution. Since the protocol embedding is a complex and unknown property of the incoming packets, static cutting (static snap length) is widely used in practice. This method ignores private user data, which is out of the static window. However, it can happen, that the private data remains visible, or the important header information are not stored because of the static cutting. An advantage of a complex parse engine is the knowledge of the header structure, which is an important and required factor for dynamic snap length calculation. This property of the parser process ensures that all header information can be stored without private user data.

D. *Packet Filtering and Classification*

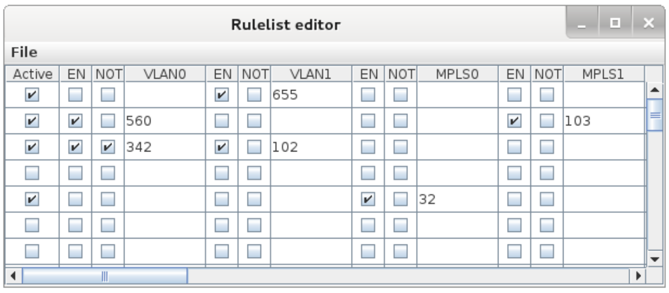
Packet classification is one of the main processes in a monitoring system, which is the third phase of the internal processing path. It expects input metadata about the incoming packet and about the base of the classification. The former is the extracted header field-set, coming from the packet parsing stage. The number and size of these header fields determine the complexity of the lookup engine. The latter is a field-set, namely rule, in which the highest priority match gives the result of the classification process. Rules are based on the extracted header information, namely tuples. The classical solutions are 5-tuple methods.

Convergence of mobile networks to IP, and the emergence of complex, more than 5-tuple classification get importance and became active research areas lately. To perform fine grained filtering, C-GEP operates with a 14-tuple classification engine. The classifier module of the C-GEP system continues the packet parser engine's data flow, including the data path width and frequency requirements.

E. *Transparent Reconfiguration of Filtering Rules*

One of the main design drives of a packet classification engine is on-the-fly, transparent reconfiguration without temporal redirection of the packet flow to a bypass path. Acquired packets are filtered based on the old rule until the update mechanism overwrites it.. The architecture of the filter module relies on the same principle as the parser engine, namely the pipelining. This feature implies a stage-by-stage update mechanism and a dedicated path for the configuration data. The filter process is synchronized with the update algorithm to perform real-time processing.

To assist the update mechanism, the system includes a graphical rule-generation software. Figure 4 shows this program that generates a configuration file, which contains the binary form of the rules in a special header format.


Rulelist editor

File

Active	EN	NOT	VLAN0	EN	NOT	VLAN1	EN	NOT	MPLS0	EN	NOT	MPLS1
☑	☐	☐		☑	☐	655	☐	☐		☐	☐	
☑	☑	☐	560	☐	☐		☐	☐		☑	☐	103
☑	☑	☑	342	☑	☐	102	☐	☐		☐	☐	
☐	☐	☐		☐	☐		☐	☐		☐	☐	
☑	☐	☐		☐	☐		☑	☐	32	☐	☐	
☐	☐	☐		☐	☐		☐	☐		☐	☐	
☐	☐	☐		☐	☐		☐	☐		☐	☐	

Fig. 4. Configuration file generator

The binary files can be reopened or exported to block RAM initialization files. This mechanism was useful during the validation of the prototypes. After generating the configuration file, new rules can be uploaded to the FPGA chip. The embedded Linux OS of the C-GEP platform contains functions for the rule update mechanism, which are written in C. The update functions use PCIe interface to reconfigure the classifier engine. The PCIe module at the FPGA side operates at 125 MHz core frequency, in contrast with the filter module. The clock domain crossing is solved by FIFOs to collect and synchronize new rules.

The classifier engine operates on the protocol metadata extracted by the parser module. Each rule contains 14 elementary conditions, which conditions can be concatenated using AND or NOT AND operations.

With the presented structure of the classifier engine (see Fig. 5), the C-GEP monitor platform is able to operate with 16 hardware-based filter rules. The number of rules in this architecture is limited because of the FPGA chip resource constraints. To increase the number of filtering rules, we designed and implemented a new block RAM based filtering solution (see Fig. 6). As a new approach, we choose a previous work from the decomposition-based classification engines, as the base of our filtering function [18]. Field-split bit vector (FSBV) is a parallel, lookup-based filtering method, using subfields from the original header information. Since it operates only at 167 MHz core frequency, it does not meet our frequency constraints. To enhance the FSBV algorithm, we optimized it to support 312.5 MHz operational frequency. Our solution is verified in simulation tests; the integration and firmware generation phase was not yet performed. Nevertheless, the optimized classifier engine is able to operate on n x 72 rules, similar to the FSBV solution. The extracted header information, coming from the parser engine, is split to 9-bit addresses for the block RAMs. Each address represents a 72-bit value, in which each bit belongs to one rule. If the 9-bit part of the actual rule fits on the 9-bit part of the extracted header information, it is represented by 1 at the given index otherwise the bit is set to 0. Each block RAM is addressed in parallel, where partial results are AND-ed for the final result. The drawback of this architecture is the lack of real-time reconfiguration. The rule fitting is an invalid operation during a writing clock cycle.

F. System Integration: One Chip, One System

The integration phase of the full system required hard cooperation between the presented processing engines, because of several implementation features and FPGA resource limitations.

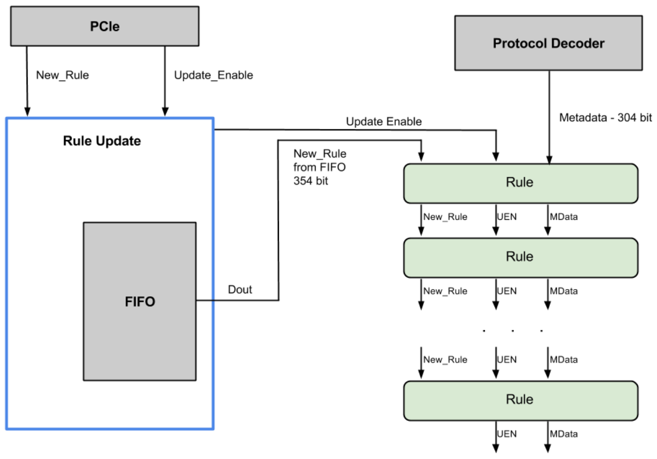


Fig. 5. Pipeline-based packet classifier

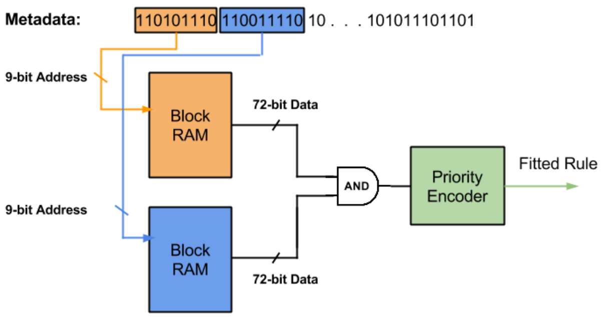


Fig. 6. Block RAM-based packet classifier

While packet parsing and packet classification are active research fields, both with a large number of contributions, there are only a few publications discussing the challenges of integrating these functionalities in a high performance monitoring system.

The size of the MAC module was a constraining factor of the system integration. The prototype board of C-GEP designed with an XC6VHX255T FPGA chip, in which the 100 Gbit/s Ethernet MAC occupies 30% of the logical routes. Adding the parser engine to the data path, the design occupies about 45% of the chip. Adding the basic Gigabit Media Independent (GMI) interface and PCIe modules, but excluding the classification engine, the half of the chip was allocated.

Considering physical resources, there is a trade-off between the size of the filter rule set and the complexity of the rules. In the C-GEP platform, we tried to balance between these two properties to cover common requirements. Besides line-rate classification, real-time rule update was another challenge of the implementation. To omit the bypass route for the captured packets during the reconfiguring phase, the classification engine was extended with an additional update path. This path occupies extra logic and results in a lower number of filter rules.

Beyond the integration of the basic processing modules (i.e., 100 Gbits/s Ethernet PCS/MAC, packet parser, packet classifier), another engineering task was to properly handle different clock domains within the system. There are four clock domains (e.g., 1 Gbit/s Ethernet, PCIe, clock synchronization and 100 Gbit/s packet processing) in the monitor architecture, requiring synchronization and clock domain crossing mechanisms.

A monitoring probe provides outgoing traffic flows based on specified criteria. Typically, the packets are routed to predefined output interfaces, where the routing method can be based on (i) matched filter rule, (ii) flow (conversation based routing), or (iii) per-packet round robin.

If the operator would like to monitor a sub-network, a filter rule based monitoring can be an effective solution. The flow-based monitoring is suitable for QoS measurement or DPI. Round robin – or balanced packet distribution – is the third routing method, which is appropriate for balancing the captured traffic between agents on per-packet basis. Lossless operation is not just a requirement for incoming packets, but also for outgoing traffic. TCP is a well-known solution for connection-oriented operation, which provides lossless packet transfer between two network nodes. However, TCP requires a significantly big chip area in a hardware implementation, and each interface instance requires a dedicated instance of the TCP module. To handle the lossless packet transmission with low hardware resources, our research group designed an UDP-based transport protocol, namely Rate Control Transport Protocol (RCTP) [19]. The advantage of this protocol lies in a flow control mechanism, which works at the recipient side, and the sender is based on simple logic to care about the FPGA resource limitation.

IV. Conclusion

While the evolution of network infrastructures is driven by two factors: the increasing amount of user data and the emergence of new services, there is a novel paradigm, namely the concept of programmable networks that shapes and redefines the architecture of IP-based network infrastructures. All of the mentioned changes together necessitate a new adaptive way of network management. To give performance as well as flexibility, reconfigurable hardware may appear as the central building block of a high performance network management system. In this paper, we introduced a multi-purpose, programmable, FPGA-based hardware platform that supports the mentioned new concept in many ways. Its main purpose is to provide means for handling 1, 10, 40 and 100 Gbit/s Ethernet traffic in a flexible manner. As Proof-of-Concept, we showed an FPGA firmware operating at 100 Gbit/s line rate and involving hardware-accelerated packet capturing, parsing, classification and clock synchronization engines. In this paper, we investigate the challenges of the integration of packet processing engines into one reconfigurable integrated circuit, and focus on major design and implementation trade-offs related to chip-level physical resources.

Acknowledgment

We would like to thank to all of our colleagues and students who contributed to the success of our project: Ákos Máté, Péter Mércse, Zoltán Nagy, István Pógár, Tamás Skopkó, Máté Varga, Péter Varga, János Végh from the University of Debrecen, György Horváth, Gábor Kródi, László Kovács and the rest of the AITIA team. This research was partially funded by the national project "C-GEP", GOP-1.1.1-11-2012-0031, in Hungary.

References

[1] M. Attig, G. Brebner, "400 Gb/s Programmable Packet Parsing on a Single FPGA", in Proc. ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems, 2011, pp. 12-23.

[2] V. Pus, L. Kekely, J. Korenek, "Low-Latency Modular Packet Header Parser for FPGA", in Proc. ACM/IEEE Symposium on Architectures for Networking and Communications Systems, 2012, Austin, Texas, USA

[3] G. Brebner, W. Jiang, "High-Speed Packet Processing using Reconfigurable Computing", IEEE Micro vol. 34, 2014, pp. 8-18.

[4] T. Ganegedara, V. K. Prasanna, "StrideBV: Single Chip 400G+ Packet Classification", in Proc. IEEE 13th International Conference on High Performance Switching and Routing, 2012, Belgrade, pp. 1-6.

[5] W. Jiang, V. K. Prasanna, "Scalable Packet Classification on FPGA", in Proc. IEEE Transactions on Very Large Scale Integration Systems, 2012, pp. 1668-1680.

[6] J. Fong, X. Wang, Y. Qi, J. Li, W. Jiang, "ParaSplit: A Scalable Architecture on FPGA for Terabit Packet Classification", in Proc. IEEE 20th Annual Symposium on High-Performance Interconnects, 2012, Santa Clara, CA, pp. 1-8.

[7] NetFPGA project, [Online]. Available: http://www.netfpga.org

[8] OpenFlow-Switch project, [Online]. Available: https://github.com/NetFPGA/NetFPGA-public/wiki/NetFPGA-10G-OpenFlow-Switch

[9] Xilinx Microblaze, [Online]. Available: http://www.xilinx.com/tools/microblaze.htm

[10] Buffer Monitoring System project, [Online]. Available: https://github.com/NetFPGA/netfpga/wiki/BufferMonitoringSystem

[11] Traffic Monitor project, [Online]. Available: https://github.com/NetFPGA/netfpga/wiki/TrafficMonitor

[12] Monitoring System project, [Online]. Available: https://github.com/NetFPGA/netfpga/wiki/MonitoringSystem

[13] Emulex EndaceAccess Network Visibility Headend, [Online]. Available: http://www.emulex.com/products/network-visibility-products-and-services/10040gb-network-visibility-headends/features/

[14] I. Moldovan, P. Varga, "A Flexible Switch-Router with Reconfigurable Forwarding and Linux-based Control Element", in Proc. IEEE 10th International Symposium on Electronics and Telecommunications (ISETC), 2012, Timisoara, pp. 217-220.

[15] IEEE 1588-2002, [Online]. Available: http://standards.ieee.org/findstds/standard/1588-2002.html

[16] PTP daemon, [Online]. Available: http://ptpd.sourceforge.net

[17] Chip Scale Atomic Clock, [Online]. Available: http://www.microsemi.com/products/timing-synchronization-systems/embedded-timing-solutions/components/sa-45s-chip-scale-atomic-clock

[18] W. Jiang, V. K. Prasanna, "Field-Split Parallel Architecture for High Performance Multi-Match Packet Classification Using FPGAs", in Proc. 21th Annual Symposium on Paralellism in Algorithms and Architectures, 2009, New York, USA, pp. 188-196.

[19] P. Orosz, T. Skopko, M. Varga, "RCTP: A Low Complexity Transport Protocol for Collecting Measurement Data", Infocommunications Journal, 2014 Vol. 6 No. 3, pp 28-36.