Copyright Notice

Institute of Communication Networks and Computer Engineering
University of Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart, Germany
Phone: ++49-711-685-68026, Fax: ++49-711-685-67983
Email: mail@ikr.uni-stuttgart.de, http://www.ikr.uni-stuttgart.de

Designing High-Speed Packet Processing Tasks at Arbitrary Levels of Abstraction — Implementation and Evaluation of a MIXMAP System

Simon Hauger
Institute of Communication Networks and Computer Engineering
Universität Stuttgart, Pfaffenwaldring 47, 70569 Stuttgart, Germany
simon.hauger@ikr.uni-stuttgart.de

ABSTRACT

Packet processing systems of forthcoming high-speed network nodes demand extremely high processing rates, but also modularity and easy adaptability due to new or evolving protocols and services. As the fixed architecture and instruction set of current network processors sometimes hinders an efficient implementation of processing tasks, we introduced the MIXMAP architecture [4] that is designed to offer programmability at multiple levels of abstraction. Now we describe the prototypical realization of this architecture showing its feasibility. Our results indicate that up to 170 million packets per second can be processed with this architecture using current FPGAs. By implementing packet processing tasks at register-transfer level and at software level, we validate the architecture's applicability and the benefits of implementing at an appropriate level of abstraction.

Categories and Subject Descriptors

C.2.6 [**Computer-Communication Networks**]: Internetworking—*Routers*

General Terms

Design, Experimentation

1. INTRODUCTION AND MOTIVATION

Routers and switches in future high-speed networks have to be adaptable to new processing requirements due to new or changed protocols or services. Additionally, the exploration and evaluation of new networking ideas for a future Internet benefits tremendously by highly programmable and modular network nodes that can be used for testing environments [2, 1]. Secondly, with the introduction of ever higher line rates, packet processing systems have to process several hundred millions of packets per second per ingress port.

Field Programmable Gate Arrays (FPGAs) and network processors (NPs) feature both the required flexibility and performance. The former offers fine grain configurability at *register-transfer level*, while the latter is programmed at the more abstract *software level* hiding hardware details from the programmer. At first glance, NPs seem more suitable to design complex packet processing functions. However, sometimes tasks need operations that are not directly supported by the NP's instruction set. Examples are complex bit operations, atomic load-modify-store operations, or operations requiring specialized hardware, like small and fast associative memories. Moreover, the given system architecture of the NP (e. g. the organization of its processing engines or memories) may not be suitable to certain operations. Often, emulating such operations is time-consuming and inefficient, while implementing them at register-transfer level on an FPGA would be fast and resource-efficient (cf. [5]).

In [4] we presented a novel architecture for a high-performance network processing unit, called MIXMAP. It provides a framework for modules that are programmed at different levels of abstraction. Yet, its structure guarantees to comply with the high line rate requirements of future high-speed packet networks (e. g. a 100 Gbps Ethernet network).

In this paper (and on the poster) we give details on the design and the implementation of a prototypical packet processing system featuring the MIXMAP architecture, as well as new performance results. We built modules for the design at register-transfer level and at software level. Using these we implemented several current and future processing tasks. Thus, this work shows the feasibility and applicability of our previously proposed architecture.

2. THE MIXMAP ARCHITECTURE

The objectives of the MIXMAP architecture are (1) a very high processing rate, (2) a modular framework, and (3) programmability at arbitrary levels of abstraction.

A very high packet processing rate is guaranteed by a high-performance pipeline structure as the basis of the system, see Fig. 1 a. The pipeline forwards its data from one stage to the next each clock cycle. The amount of data processed in each stage is at least that of a minimum-sized packet.

Modularity is accomplished by defining functional units with fixed interfaces on top of the pipeline structure, as depicted in Fig. 1 b. The interface definition specifies what data and meta-data has to be transfered from one unit to the next and that each unit receives and transmits this data every clock cycle. Obeying these interface specifications a functional unit may have any arbitrary internal structure.

Finally, each functional unit can be implemented utilizing

ANCS'09 October 19–20, 2009, Princeton, New Jersey, USA.

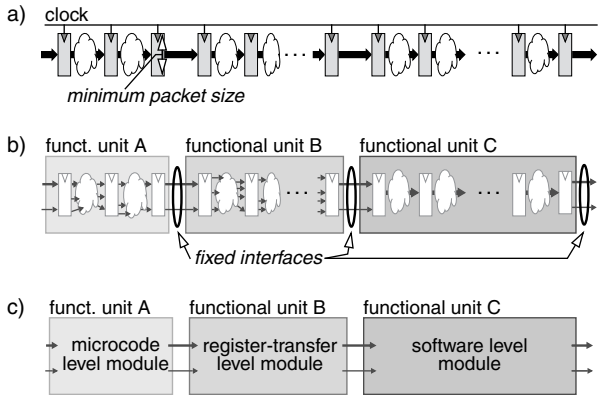

Figure 1: a) high - performance pipeline structure, b) functional units on top of pipeline, c) use of modules with different levels of abstraction

one of several possible modules, as shown in Fig. 1 c. The modules offer different hardware infrastructures. Thus, one can choose a module with the most suitable level of abstraction when implementing a new functional unit.

The architecture therefore allows e. g. to quickly add a new processing task to a router by programming a software-level module. Later, one can replace this by a more resource-efficient unit designed at a lower level of abstraction. Similarly, a task that is not efficiently realizable at software level due to processor constraints, can be implemented fully or partly at register-transfer level. Furthermore, due to its fixed interface specifications, new functionality can also be acquired from other parties and "plugged" into the system.

3. PROTOTYPE AND EVALUATION

We implemented a prototypical MIXMAP packet processing system in order to show the architecture's feasibility and the achievable performance with current FPGAs. The structure of the system is depicted in Fig. 2. It supports up to 15 Gigabit Ethernet ingress and egress interfaces (we used three). To be able to perform full load tests also internal packet generators can be used, together with validation modules at the egress side. The packets are forwarded back to back into the actual processing pipeline consisting of register-transfer-level (RTL) modules and software modules. A slow path general-purpose processor, located on a second FPGA, performs control and management tasks.

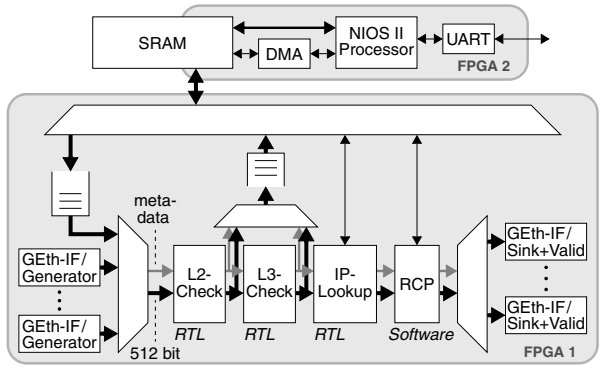

Figure 2: Prototypical MIXMAP system

Table 1: Software-implemented processing tasks

Processing task	Execution time (max.)
IP (w/o lookup)	43 cycles
TCP Quick-Start (non-critical)	55 cycles
Rate Control Protocol (RCP)	26 cycles

As stated in [4], software-level modules can only process a packet for $N \cdot F$ clock cycles, with N parallel processors with an F times higher clock frequency than the system pipeline. Hence, we designed small, specialized RISC-cores with an instruction set similar to that of the Intel (now Netronome) IXP2xxx microengines [6]. A multiplexer distributes the incoming data sequentially to the N processors writing it directly into the register set to save clock cycles. Place & route results show that more than 100 such processors can be accommodated on a current FPGA. Currently we achieve a maximum clock frequency of 90 MHz, optimizations should increase the possible rate. Also, modules with even smaller processors may be possible, or a module using a dataflow pipeline [3] as in the Xelerated NPs.

To evaluate the applicability of the MIXMAP architecture we implemented several functional units executing different packet processing tasks. Table 1 lists the processing tasks we implemented using software-level modules. $N = 16$ processors with $F = 4$ suffice to execute the required number of clock cycles. [5] shows that the parallel architecture of NPs is not suitable for the sequential bandwidth assignment steps of some future explicit congestion control schemes. Hence, we implemented the respective critical functions at register-transfer level and the non-critical part in a software-level module. This nicely shows the benefit of designing at a suitable level of abstraction. Using RTL-modules we implemented basic IP processing functions as well as an algorithmic and a TCAM-based IP address lookup module. They support clock rates up to 170 MHz, thus the system can process 170 million minimum-sized packets per second.

4. CONCLUSION

We implemented a prototypical MIXMAP system to show the architecture's feasibility as well as the achievable throughput. We further demonstrated its applicability to typical current and future packet processing tasks.

5. REFERENCES

[1] GENI – exploring networks of the future. http://www.geni.net, Aug. 2009.

[2] NetFPGA. http://www.netfpga.org, Aug. 2009.

[3] J. Carlstrom and T. Boden. Synchronous dataflow architecture for network processors. *IEEE Micro*, 24(5):10–18, Sept.-Oct. 2004.

[4] S. Hauger. A novel architecture for a high-performance network processing unit: Flexibility at multiple levels of abstraction. In *IEEE International Conference on High Performance Switching and Routing*, June 2009.

[5] S. Hauger, M. Scharf, J. Kögel, and C. Suriyajan. Evaluation of router implementations for explicit congestion control schemes. *Journal of Communications, Academy Publisher*. (to be published end 2009).

[6] Intel Corp.. IXP2400 network processor. data sheet. http://download.intel.com/design/network/datashts/30116411.pdf, Feb. 2004.