Packet Processing at 100 Gbps and Beyond—Challenges and Perspectives

Simon Hauger*, Thomas Wild†, Arthur Mutter*, Andreas Kirstädter*,
Kimon Karras†, Rainer Ohlendorf†, Frank Feller*, and Joachim Scharf*
*Institute of Communication Networks and Computer Engineering (IKR),
Universität Stuttgart, Pfaffenwaldring 47, 70569 Stuttgart, Germany
Email: {simon.hauger, arthur.mutter, andreas.kirstaedter, frank.feller, joachim.scharf}@ikr.uni-stuttgart.de
†Institute for Integrated Systems (LIS),
Technische Universität München (TUM), Arcisstraße 21, 80290 Munich, Germany
Email: {thomas.wild, kkarras, rainer.ohlendorf}@tum.de

Abstract

The continuous growth of traffic volumes steadily raises the throughput requirements on the network infrastructure. Additionally, a transformation of the classical TDM-based backbone networks to packet networks with Carrier Ethernet as the target technology occurs. The standardization process of 100 Gbps Ethernet is under way. This not only poses big challenges to transmission but also to packet processing technologies. However, recent announcements from network processing unit (NPU) vendors promise that packet processing at 100 Gbps is feasible. The big question for system manufacturers now is, whether this trend will continue and finally lead to 1 Tbps packet switching, or whether there are technological roadblocks that inhibit this development path. In this paper, we address this question and identify packet processing performance, packet buffer throughput, chip-to-chip interface speed, and power dissipation as the most critical factors. We discuss their limiting factors as well as architectural and technological trends that can further increase their performance. Based on these investigations and extrapolating anticipated technological advances we expect that 1 Tbps packet processing and switching could be introduced in the network within several years. Since this, however, not only depends on technological but also on economical factors, we show how slight modifications of the network architecture and protocols could alleviate some implementation complexities and thus reduce the overall cost.

1 Introduction

New Internet services like IPTV and the forecasted Full/Ultra HDTV over IP as well as the applications in the framework of Web 2.0 together with the constantly increasing coverage of fixed and broadband access networks are leading to annual growth rates of backbone bandwidth demands in the order of 100 percent or even more. As by far the major part of the transported traffic is packet-based by its origin and due to advantages in terms of flexibility, we are currently seeing a shift from classical TDM based transport networks to packet-based architectures. A major step in this direction is the development and deployment of Carrier Ethernet and similar technologies like the Transport Profile for Multi-Protocol Label Switching (MPLS-TP) that the IETF and the ITU-T are jointly standardizing. First products for 100 Gbps Ethernet are commonly expected to enter the market in 2009/10 and are already highly desired by network operators, internet exchanges, and data center operators.
However, a major question is the sustainability of this trend towards packet-based transport networks: Will we still see an increase of networking speed beyond 100 Gbps Ethernet? Optical transmission technology is constantly evolving to the speed of 1 Tbps on a single wavelength and efficient methods are available to make use of parallel transmission over several wavelengths. However, a matching development allowing the electronic packet processing and switching to keep pace with the huge increase in transmission speed is still an open issue. We therefore investigate the scalability of the data plane towards speeds beyond 100 Gbps, disregarding the control and management planes of packet-switched transport networks, which also still require novel technological solutions.
In this paper, we therefore discuss the throughput limiting effects in the data plane of future packet transport networks. We focus on the network processing units (NPU) within the nodes handling layer 2.5 and 2 traffic (IP/MPLS and Packet Transport / Carrier Ethernet). Extrapolating the standardization timeline for 1, 10, and 100 Gbps Ethernet and looking at announced bandwidth growth data (e. g. from the Internet exchange points) we target the timeframe around the year 2015 for our evaluations of the technologies involved for a desired 1 Tbps Ethernet.
In the next section, we first discuss more generally the related aspects of network performance and evolution before defining a reference model for a generic packet network node. In section 3, we identify the critical technological

aspects and building blocks regarding their influence on the maximum achievable packet throughput in the nodes and show their possible evolution paths to line speeds of 1 Tbps. In addition to this, in section 4 we discuss possible slight network and protocol related modifications that have the potential to strongly reduce the requirements on the packet processing hardware at such high line speeds. Finally, we summarize our investigations in section 5.

2 Networks and Network Nodes

2.1 Today's Networks

Transport networks have traditionally been dominated by TDM over WDM technologies like SDH and SONET. Recently however, the operators became more interested in packet transport technologies. The reasons behind this move are twofold: On the one hand, the traffic streams flowing over transport networks originate in by far the majority of cases from packet-based traffic sources on the borders of the transport network and no longer from TDM-based telecommunication services and applications. Thus, strictly channel oriented TDM transport technologies lack the degree of flexibility and adaptability needed. On the other hand, enterprise and private LAN installations—especially on the basis of Ethernet—have reached such high total numbers of ports that their components show very interesting economies of scale. Thus, to save cost operators and manufacturers are increasingly interested to use components and technologies from the LAN domain also in transport networks.
With this migration towards fully packet based architectures even in the core of the network, the processing of packets at ultra-high rates is becoming the major challenge for network elements. Therefore, the packet processing rate defines an important performance metric—besides the ever increasing bit rate that has to be handled at the interfaces.
In carrier networks the forwarding effort per packet is constant, so the shortest packets define the worst-case load imposed onto the nodes. For system design it is generally accepted to use this assumption for the packet rates arriving at the network element under consideration. As most of the traffic at the borders of future transport networks is Ethernet based and since Ethernet technologies are more and more considered for their usage in transport networks, we can assume that the minimum packet size will be that of Ethernet: 64 bytes. For a 100 Gbps Ethernet link with a minimum frame size of 64 bytes the processing time budget per packet[1] is only 6.7 ns (including inter-frame gap and preamble) and would decrease accordingly if we move on to 1 Tbps Ethernet.
Such high packet rates resp. low processing time budgets demand a significant reduction of processing effort for the individual packets. As it is commonly accepted, classical IP header processing is not scalable to those speeds. High-speed network elements increasingly operate on the basis of label switching and virtual packet paths set up by network management or distributed network control plane approaches. The usage of label switching avoids the high processing requirements of longest prefix match lookups as needed for IP routing. Also forwarding table sizes can be reduced by setting up the paths efficiently. Originally, MPLS has been introduced for this purpose and has meanwhile evolved to a powerful traffic engineering concept on the layer 2.5.
Currently, this label-switching and virtual path concept is also extended to the emerging packet transport networks below IP/MPLS. Carrier Ethernet uses MAC header stacking in the form of PBB-TE to implement packet transport tunnels. Also, MPLS itself is extended by a Transport Profile (MPLS-TP) to fulfill the carrier grade requirements (OAM etc.) of transport networks.

[1]As datagram processing in network nodes is tranditionally called packet processing, in the following we use the term *packet* instead of *frame* irrespective of the processed layer.

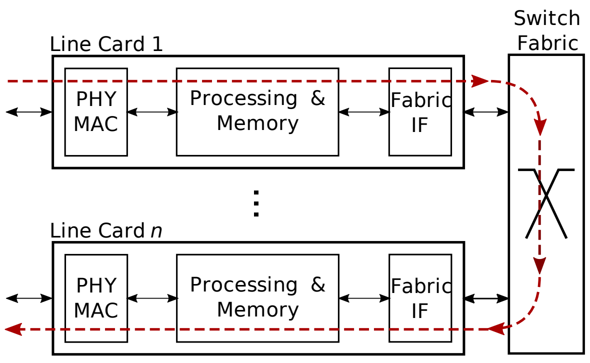

Figure 1 Principle packet processing path in a high-speed network node

2.2 Network Nodes

Current high-performance network nodes deployed in core networks have a modular structure. As depicted in Fig.1 they consist of several line cards, a switch fabric as well as one or more cards for control and management tasks (not shown in the figure) [1]. The individual bidirectional line cards receive, process, buffer and transmit the packets. All line cards are attached to the switch fabric that forwards each packet from its ingress line card to its egress line card. A line card itself comprises one or more line interfaces and its corresponding PHY (Physical Layer) and MAC (Medium Access Control) chips, a network processing subsystem, and a switch fabric interface. The network processing subsystem contains an NPU with integrated or separate traffic manager, other co-processors as well as on-chip and off-chip memories.
In the following we describe the way of a packet traversing the network node. The PHY chip receives the optical signal from the fiber and performs opto-electronic conversion, as

well as clock and data recovery. The MAC chip detects the frame boundaries, checks the frame's integrity and delivers it to the network processing subsystem. Here an NPU parses and classifies the packet, performs a route lookup to determine the outgoing interface, and modifies certain header fields. During processing, the packet is stored either on-chip or off-chip. An off-chip memory with queues of different classes of service (CoS) holds the packets in case of congestion. As soon as the corresponding line interface is available, the switch fabric forwards the packet to the egress line card. For this the switch fabric interfaces transform the packet data between the formats used on the line cards and in the switch fabric. The traffic manager schedules the packet's transmission according to its CoS. An NPU at the egress side may perform further processing steps on the packet, before it is transmitted by the MAC and PHY chips.

2.3 Network Processors

The architectures of current high speed NPUs with 10–40 Gbps throughput are based on two distinct configurations: a pool of processors and a pipeline of processors.

The first configuration, as depicted in Fig. 2a, is found in current products from Cisco [2], AMCC [3] and Cavium [4]. Here each packet is processed by a single processor out of a pool of identical processors. The individual processors communicate with the interfaces, memory and hardware accelerators (e.g. dedicated traffic managers, cryptographic modules, lookup engines) using bus systems or crossbar communication structures.

The second configuration, the pipeline, is shown in Fig. 2b. It is found in NPUs from Xelerated [5] or Bay Microsystems [6]. In such an architecture a packet is processed sequentially by all processors. An advantage of this configuration is its reduced communication effort.

Combinations of both configurations are found in products of EZchip [7] and Netronome [8] (that bought the NPU business from Intel [9]). Such configurations either feature a pipeline structure with each stage containing a pool of processors or a cluster of processors with special interconnects, which allow building a software pipeline.

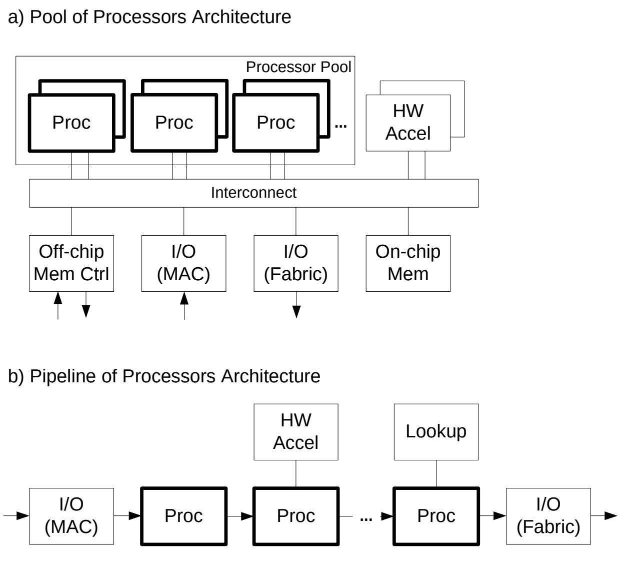

Figure 2 Pool-based and Pipeline-based Network Processor Architectures

3 Architectures and Technologies

The major functionalities that have to be provided by NPUs can be categorized in two main areas. Firstly, an NPU has to execute data plane functions for each incoming packet. This comprises to determine the output port, where the packet has to be transmitted, and if necessary to modify it according to the corresponding network protocol and the current network configuration. (In this paper, we disregard control plane functions, which usually have less stringent real time requirements.) Secondly, packets have to be buffered—depending on the architecture during processing, and in any case for the time needed to resolve congestion at the output port of the node.

Line speeds in the range of hundreds of Gbps pose big challenges to cope with these two major tasks in real time. Traffic loads consisting of minimum size packets make up the worst case for processing. An NPU that processes unidirectional (half duplex) traffic on a 100 Gbps Ethernet link has to cope with a packet arrival rate of roughly 150 Mpps, which means that it has to be ready to accept a new packet every 6.7 ns. Architectural alternatives for the data path of NPUs that cope with the processing requirements are discussed in the next section.

In addition to the actual forwarding-specific sub-functions, a packet buffer is needed to store and retrieve packets according to their arrival and departure (on the switch fabric or MAC interface) and depending on how packet processing is done. Memory bandwidth requirements for unidirectional traffic may be in the region of four to six times the I/O (input/output) speed in the worst case. This figure could even go higher if full duplex operation has to be supported or several ports on a line card share a single packet buffer. Therefore, the memory subsystem of NPUs is a decisive component to provide very high throughputs, an issue that is covered in section 3.2.

Besides these functional aspects, high data rates also cause big challenges to the implementability of appropriate NPUs. In section 3.3 we put the focus on the implementation of the interconnection of high speed NPUs to the rest of the chip set that makes up the line card. This has direct influence on feasibility and cost of a final implementation.

A further non-functional requirement is the limitation of total power consumption in order to restrict packaging cost and to allow reliable operation with air cooling in a router chassis. Assuming similar requirements as in

AdvancedTCA (Advanced Telecom Computing Architecture) [10], 200 W power dissipation would be allowed per line card. Taking into account two NPUs on the line card for full duplex operation and reserving a certain power budget for the rest of the chip set, the maximum power dissipation per NPU would have to be in the range of 40 to 80 W. Appropriate packaging solution will have to keep up with power consumption, the required I/O speed, and the resulting pin count.

In addition to describing current architectures and technologies, we discuss their scalability trying to answer the question: Will 1 Tbps packet processing be feasible? We assume that this line speed will be introduced in the year 2015. As technological background for our investigations, it can thus be expected that CMOS technology will have evolved from today's 40 nm to 20 nm minimum feature size.

3.1 Packet Processing

The throughput of a network processor is inversely proportional to the processing time. In order to achieve the required high throughput, NPUs make strong use of parallelization. As we introduced in section 2.3, almost all NPUs contain several processor cores that work in parallel: arranged as a pool, as a pipeline, or as a combination of both.

In a pool configuration—at first glance—the throughput seems to be linearly proportional to the number of parallel processors. However, a large number of processors that have to be connected to shared resources, like memory and coprocessors, require a large interconnection system between them. This directly maps to increased chip area and power consumption. Additionally, arbitration and locking mechanisms may degrade the system's throughput.

The throughput of a pipeline of processors is determined by the longest processing time within one of its stages. Task-specific resources can be assigned to single stages so that less access conflicts occur. Additionally, no complex interconnection mechanisms are necessary, as there are only point-to-point connections between the processors and to attached resources.

In all configurations, parallelization within a processor core is used to increase its throughput. Both superscalar architectures and Very-Long-Instruction-Word architectures are deployed. By providing multiple register sets, the processor hardware supports also multi-threading. Switching to another thread efficiently hides memory latencies.

A further way to speed up NPUs is to add coprocessors and hardware accelerators for compute-intensive tasks like ciphering, checksum calculation or classification. Additionally, optimized hardware is usually more power-efficient than performing the respective tasks in software. Here too, the interconnection to the processor cores is critical both in terms of speed and power.

The computational density within NPUs differs substantially. While the processors in a pool configuration have a rather general purpose instruction set, the processors of a pipeline are often optimized to certain tasks. Thus, the former configuration offers a higher flexibility, but the latter provides a higher throughput as fewer instructions are needed for a certain task.

Considering high-performance NPUs, we currently observe a major trend to pipeline-based architectures with equal, constant processing times in each stage. Such pipeline architectures offer a deterministic, high packet throughput, and a low interconnection effort. Due to the fixed processing time per stage and the given length of the pipeline, only a limited number of functions can be performed. However, this fits well to the fairly simple layer 2 to 2.5 processing functions that are needed in core networks.

Both EZchip [11] and Xelerated [12] recently announced 100 Gbps Ethernet capable NPUs that are based on a deterministic pipeline architecture. The pipeline of the EZchip NPU comprises four stages, each with a pool of several task-optimized processors. In contrast, the pipeline of the Xelerated NPUs consists of 512 special processors and each processor executes only one instruction per 64-byte word of a packet.

Also in terms of power consumption, pipelined architectures are preferable. Current lower speed grade NPUs in the range of 10 to 20 Gbps with a pool configuration have a power dissipation of several tens of Watt (Netronome IXP2855 32 W in 130 nm, Cavium OCTEON Plus CN58XX 40 W and Cisco Quantum Flow 80 W, both in 90 nm technology). Pipelined NPUs of even 40 Gbps performance like Xelerated X10q (130 nm) or Bay Networks Chesapeake (110 nm, including also a traffic manager) only consume about 10 and 16 Watt, respectively. Therefore pipeline architectures are much better scalable to high throughputs and we estimate a power consumption of 30 to 40 Watt for pipelined 100 Gbps NPUs using a 65 nm technology.

When even higher throughputs are needed in future networks, within an NPU similar to that of EZchip the number of parallel engines per pipeline stage can be increased. However, access problems as well as interconnection problems to shared resources within one stage might occur here. The NPU from Xelerated is clocked at 322 MHz using a 65 nm technology. Moving to even smaller geometries in the future as well as minimizing the combinatorial paths within the pipeline may further increase its throughput.

Generally, a pipeline based system that processes only a fixed amount of bytes (basically the header) of each packet and that forwards this data to the following stage in each clock cycle achieves a packet throughput corresponding to its clock rate. Thus, if a clock rate of e. g. 2 GHz will be achieved in some years—as is already achieved with

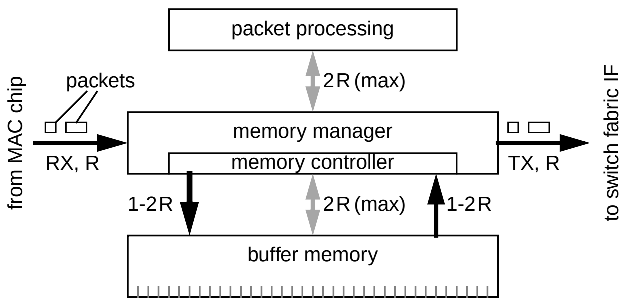

Figure 3 Principle packet buffer architecture – bandwidths depicted assume a line card with one port at line rate R

general purpose processors—such a system will support a packet throughput of 2 billion packets per second, corresponding to a line rate of at least 1.3 Tbps. Obviously, the system would have to be capable to buffer the remaining part of the packet fast enough.

3.2 Packet Buffer

A typical high-speed network node architecture as shown in Fig. 1 uses input packet buffering while maintaining virtual output queuing to prevent head-of-line blocking. Typically, the number of supported queues is in the order of thousands.

Fig. 3 shows the principle packet buffer architecture. The memory manager abstracts from actual memory organization providing N logical queues where packets are stored and retrieved from. Therefore it statically or dynamically assigns memory to the individual queues. In case of dynamic allocation, the memory manager divides the available memory into segments. It is common to build queues by concatenating these segments via linked lists, as done in [13]. Furthermore, a memory controller abstracts from the details of the physical memory.

In the following we derive the throughput and size requirements before presenting a practical solution and future trends.

3.2.1 Requirements

The packet buffer has to store packets as fast as they arrive and retrieve them as fast as they depart. With this, the minimal required memory bandwidth is 2R, where R is the line rate of the served port.

It is common for network nodes to segment packets into fixed size chunks to simplify memory management. 64 Byte is the common choice as it is the first power of two able to hold a minimal Ethernet or IP packet. Nevertheless, in the worst case the packet buffer must sustain a stream of 65 Byte packets, which all consume two 64 Byte chunks, leaving the second nearly empty. This is called the 65 Byte problem [14]. The consequence is a required overprovisioning of memory bandwidth by a factor of two on RX (receive) and TX (transmit) side of the packet buffer.

Furthermore, depending on the architecture, it can be necessary to access the packet headers for processing. These usually reside in the first memory segment of the packet. Assuming only minimum size packets an additional memory bandwidth of up to 2R is therefore required (grey arrow in Fig. 3), as due to the access granularity of the memory the complete segment is fetched for being processed and finally stored back.

Putting all together, a line card according to Fig. 1 with one port would require up to 6R of memory bandwidth, e. g. 600 Gbps for 100 Gbps Ethernet.

The required size of a packet buffer depends on many factors, like traffic characteristics, number of logical queues, target loss rate, etc. There is a widely used rule-of-thumb that says, for TCP to work well, the buffer should be dimensioned to $RTT \times R$, where RTT is the round trip time between active end hosts [15]. Assuming an RTT of 200 ms and a line rate of 100 Gbps the buffer size for a single port is 2.5 GByte. In [15] the authors show that buffer size could be reduced to $RTT \times R/\sqrt{N}$, where N is the number of major active TCP flows. This would reduce the buffer size to around 1 %. Nevertheless, for historical reasons and not to put the Service Level Agreements (SLA) at a risk [16] the network operators are likely to continue buying routers with buffers as large as possible.

In summary, both the required bandwidth and size of the packet buffer grow linearly with the line rate R and therefore pose big challenges to packet buffer design.

Reviewing throughput, density, power consumption and price of currently available memory types, only (external) dynamic RAM (DRAM) fulfills all these requirements. The drawbacks of DRAM are its rather long and non-constant access time T_{RC} of around 50 ns, the necessity for refresh of the DRAM cells and the complex control. T_{RC} is the maximum time to write to, or read from any memory location and limits the possible number of read/write operations. Already with 10 Gbps Ethernet, in the worst case the memory has to be accessed every 16.8 ns (assuming 4R total bandwidth) to store or retrieve a packet, which is far below T_{RC}. Exploiting the bank concept supported by DRAMs the effective T_{RC} can be reduced. Therefore the individual B banks of a DRAM are accessed in an interleaved manner, leading to an effective access time of T_{RC}/B in the best case [17]. Nevertheless, the level of reduction greatly depends on the access pattern.

3.2.2 Current Solution

In practice router line cards use multiple independent DRAM devices in parallel, firstly, to obtain the required aggregate memory bandwidth, and secondly, to reduce the random access time of the packet buffer to that dictated by the line rate. The latter is achieved by stripping the data statistically across all DRAM devices and its banks [18]. With M DRAM devices a random access time in the range of

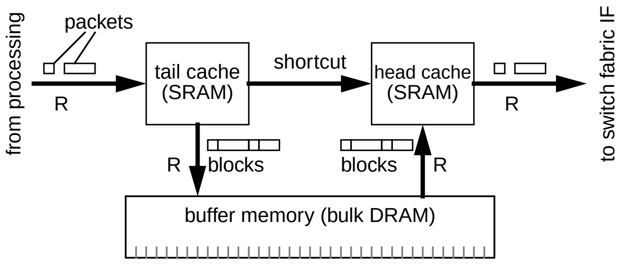

Figure 4 Memory hierarchy with head and tail cache

T_{RC} to $T_{RC}/(M \cdot B)$ can theoretically be achieved. Due to the dependence on the access pattern, the access guarantees are only statistical and so the system could loose throughput in an unpredictable way. The Intel IXP2800 NPUs use this approach with three parallel RDRAM (Rambus DRAM) memories [19].

3.2.3 Trends

There is a trend to lower the bandwidth requirements to be provided by the packet buffer. The number of ports per line card can be decreased to a minimum of one single port. Furthermore, the required bandwidth can be reduced to 4R by doing all the packet processing before buffering. This requires a second small packet buffer (preferably on-chip) to store packets for a bounded processing time. With this, the main buffer memory has only to be accessed in a first in first out (FIFO) manner, reducing memory manager's complexity.

A promising architectural approach to deal with access times and bandwidth requirements is to use a memory hierarchy [14, 17] similar to computer systems. This approach may already be used in current systems. Fig. 4 shows the basic architecture of a packet buffer using a memory hierarchy. Head and tail of all FIFO queues are stored in fast static RAM (SRAM). The remaining part is stored in the bulk DRAM. The tail cache initiates a data transfer to DRAM when enough data has arrived. Then a large data block is transferred from tail cache to the corresponding queue in DRAM. Similarly, in preparation for the packet departure, data from the corresponding queue is read in the same large block granularity into the head cache. This approach has guaranteed constant access times, simplifying overall system design. Due to the large blocks exchanged with the bulk DRAM, this can be ideally combined with memory parallelism on pin level, where all DRAM devices are operated with the same commands and addresses. Furthermore, as packet data can be seamlessly concatenated to build the blocks, the 65 Byte problem does not arise. With this, the effective required bandwidth reduces to a total of only 2R. A drawback of this approach is that the required size of head and tail caches (built from fast and expensive SRAM) increases linearly with the number of FIFO queues and the line rate.

3.2.4 Memory Technology and Outlook

Finally, we discuss the technological trends, limits and the feasibility of 1 Tbps Ethernet which we expect in 2015 (6 years from today).

The fastest currently available DRAM is GDDR5 (Graphics Double Data Rate SDRAM), with a data rate of 5 Gbps per pin [20]. Rambus [21] announced XDR2 memory with up to 12.8 Gbps per pin to be available in the next years. The 32 data pins of a GDDR5 memory chip achieve in total 160 Gbps of gross throughput. With this, two chips could suffice for the bandwidth demand of a 100 Gbps line card with one port, requiring at least 2R = 200 Gbps of memory bandwidth. With an approximate increase in DRAM bandwidth of 30 % per year [22], the gap to the deployed line rates will increase continuously, requiring more parallel memories. Extrapolation shows a supposed increase by a factor of 5 in DRAM bandwidth within 6 years. As Ethernet's data rate increases from 100 Gbps to 1 Tbps, i. e. by a factor of 10, in the same period the number of parallel memory chips has to be doubled to support the throughput. The access time of DRAMs fall by only 7 % per year [15]. In 6 years extrapolation shows still a supposed access time of 64 % of today's value. With the increase from 100 Gbps to 1 Tbps the time budget per packet decreases to 10 %. If stripping data over memory chips, then 6.4 times more parallel memories are necessary to achieve the required access time of below 1 ns. Using a memory hierarchy instead, the cache size would increase by a factor of 6.4, requiring several tens of MByte for head and tail cache assuming 10,000 queues.

With a large number of parallel memories pin count needs to be considered. According to [23] available enterprise server CPUs and boards feature three DDR3-DRAM channels with 240 pins each. With this, pin count is not awaited to be limiting in near future. Furthermore, as the computer industry, which is the main driver for DRAM development, asks for ever larger memories, buffer capacity is not awaited to be a problem in future.

Relaxing the size requirements to $RTT \times R/\sqrt{N}$ as proposed in [15], the buffer size could shrink to around 1 %, i. e. for a 1 Tbps Ethernet line card to some hundred MByte. With this, 2D-planar MCPs (Multi-Chip Package) or in future even 3D-stacked MCPs could be an alternative, avoiding external memories with their main drawbacks concerning pin count and line card layout. Furthermore, significantly higher bandwidths can be achieved with MCPs.

Concluding, with the assumptions and extrapolations made, 1 Tbps Ethernet seems to be feasible in 2015 with respect to packet buffers.

3.3 Chip Interconnects

For the connection of a high speed NPU to the interfaces of both MAC and switch fabric, chip-to-chip interconnect technologies with the appropriate speed are needed that restrict power consumption of the I/O circuitry and limit pin count in order to enable cost efficient packaging solutions while guaranteeing reliable data transfer. The traditional solution for this problem is to transfer data serially, using differential signaling techniques with low voltage swing (e.g. LVDS, Low Voltage Differential Signaling), which is both robust to noise and very power efficient.

Today's interface solutions for 10 Gbps NPUs use such differential interconnects, for example XAUI (10 Gigabit Attachment Unit Interface) and SPI-4.2 (System Packet Interface). XAUI is part of the IEEE Ethernet standard and uses 4 differential wire pairs, each transporting 2.5 Gbps. Taking into account an 8B/10B encoding scheme, this results in a frequency of 3.125 GHz per link. The interface supports both receive and transmit direction and thus requires 16 pins in total. A further well-established alternative for this speed range is SPI-4.2, which is standardized by the Optical Internetworking Forum. It uses two 16 bit wide data paths for receive and transmit with a minimum speed of 622 Mbps for each LVDS signal. In addition to the data path, each direction has flow control interfaces for signaling FIFO buffer status.

Currently, the standardization of XLAUI 40 Gbps and CAUI 100 Gbps interfaces based on 10 Gbps or even 25 Gbps differential signals is under way. There are also several approaches targeting at scalable interfaces with bandwidths of 100 Gbps and beyond. Interlaken is a new concept for packet-based inter-chip communication mainly in the range of 10 to 100 Gbps, but without an inherent speed limitation. Its interface is channelized and its throughput can be scaled by the number of parallel serial lanes and their speed. Compared to SPI-4.2 Interlaken requires 90% less pins. A similar approach is followed by the Scalable System Packet Interface (SPI-S), a successor of SPI-4. SPI-S, which provides interfaces for data transfer and associated flow control information, may be used for uni- or bi-directional and also asymmetric links.

The described concepts mainly aim at scalable high speed inter-chip packet communication and enable solutions that differ in number and speed of the underlying channels. As the concepts specify the protocol only, they can be realized using different high speed physical layer transmission technologies. To cope with 100 Gbps and higher, several of these serial interconnect lanes have to be operated in parallel. The actual number of signals and the power consumption needed for these packet interfaces is then mainly determined by the available transceivers.

Research in transceiver technology has a long tradition. Solutions heavily differ in I/O speed, power consumption and the required chip area, even when the same technology generation is used. Two examples of transceivers for 10 and 20 Gbps implemented in 90 nm CMOS have been shown at recent ISSCC conferences [24, 25]. They consume 200–300 mW and require less than 0.2 mm^2. A complete 12.5 Gbps SerDes (Serializer/Deserializer) device, which in addition to the transceiver also contains serial to parallel conversion to establish a connection to the chip-internal parallel data path, is presented in [26]. In [27] the transmitter side of a 20 Gbps SerDes is described. Both devices are implemented in 65 nm CMOS with power dissipation values in the same range as mentioned before, but the full SerDes is much bigger than the transmitter and has an area of 0.45 mm^2.

Considering these results, we assume that a full SerDes for 20 Gbps with a power consumption of roughly 300 mW and an area of about 0.4 mm^2 should be feasible in the near term. Thus, the 200 Gbps I/O performance needed by a 100 Gbps NPU would be feasible with 5 to 6 SerDes modules. The resulting power of 1.5–1.8 W and area cost of 2–2.4 mm^2 should make up no problem for the implementation. The same holds true for the required pins.

When looking at a 1 Tbps NPU, scaling throughput requirements would lead to roughly 10 times the area and power consumption, which might be critical considering the overall power budget. If I/O speed of SerDeses could be doubled to 40 Gbps, appropriate packaging solutions should be feasible for this speed [28]. Then 25 or 30 of such high speed SerDes instances would have to be run on a single die. In comparison to this, the I/O subsystem of today's 3rd generation SPARC processor [29], consists of 288 RX and TX SerDes channels with 4.08 Gbps each and is implemented in 65 nm technology (46 mm^2 and 21.2 W). Considering the 1.1 Tbps overall I/O throughput of this processor and taking into account the further miniaturization towards 20 nm, the I/O requirements of a future 1 Tbps NPU should be met as well, trading-off number and speed of then available SerDes modules.

4 Network and Protocol Aspects

The feasibility of packet processing at high line rates does not exclusively depend on technological constraints. Protocol design choices, along with criteria for network node dimensioning, influence both the required processing rate and the complexity of processing one packet. We detail on these aspects in the following.

4.1 Dimensioning Criteria

Frequently applied rules for network node dimensioning are based on worst-case assumptions. This applies both to the buffer size formula $RTT \times R$, as introduced in section 3.2.1, and the supported packet rate. However, the necessity of such stringent assumptions is questionable in transport networks. For instance, the buffer size formula provides for full link utilization by one single TCP flow [15], a situation neither realistic nor desired in core networks.

In today's transport networks, buffers allow to prevent packet loss in temporary congestion situations. Due to the highly multiplexed traffic, they are not required to compensate the behavior of individual protocol instances. In addition, target link utilizations generally range well below 100 %. High buffer levels are therefore indicative of poor network dimensioning or sub-optimal traffic engineering. Since buffering introduces additional packet delay, the buffering of large amounts of data in the network is not desirable. This particularly applies to real-time application data, where excessive delays are equivalent to packet loss. Consequently, small buffers should be sufficient in network nodes. If buffers would be reduced to few MBytes, they could be realized on-chip, providing considerable benefits: reduced design complexity, very wide words, increased bandwidth and no need for I/O pins.

Only a long sequence of minimum size packets transmitted back to back results in the worst-case packet rate today's network nodes are dimensioned for. However, the occurrence of such a sequence is highly unlikely. Consequently, NPUs could be dimensioned to less stringent requirements, which mostly benefits the actual packet processing. The target processing rate should range between the average and the worst-case packet rate. For instance, a reduction to 50 % of the worst-case packet rate would already cut processing requirements by half.

4.2 Switching Granularity

Besides adapting dimensioning rules, we can reduce the actual processing load by decreasing the packet rate. There are three different approaches. A first one is the increase of the maximum size of packets transported between end systems. It particularly reduces the number of packets required for the transfer of large blocks of data, and thus the mean packet rate. On the downside, the maximum packet rate remains unaffected.

If dimensioning is done on worst-case assumptions, the increase of the minimum packet size is of more interest. This is achieved by extending smaller packets with padding data to the required minimum size. Since the worst-case packet rate is close to inversely proportional to the minimum packet size, we can thereby bound the packet rate as desired.

The third approach is to concatenate packets following the same path through the transport network at this network's edge. Within the network, the resulting aggregates are processed instead of the individual packets they contain. Thus, less processing is required. On the downside, aggregation induces additional delay. However, it can often be kept in uncritical regions negligible compared to the overall transmission delay [30].

It should be noted that all of these approaches mainly alleviate the requirements on the packet processing unit. In contrast, the bandwidth requirements for the packet buffer as well as the packet reception and transmission over the switch fabric and line interfaces, remain largely unaffected by changes of the packet rate.

If processing in network nodes has to be reduced further, we may go back to circuit switching techniques like TDM. Due to pre-defined treatment of the contents of time slots, processing and buffer requirements are kept to a minimum. Besides, optical switching, e. g. by future optical packet switching systems, could completely replace electrical processing in some nodes.

4.3 Processing Tasks

Standardization of networking protocols offers further potential to ease packet processing and thus to contribute to achieving higher packet throughput. The appropriate choice of protocol specific functionalities and an optimized layout of packet headers can reduce the effort for implementing the basic operations to be carried out in the data path of NPUs. Many approaches already have reduced the processing complexity of today's protocols, especially those used in the high speed core of the network. Two examples are the use of label based concepts, which heavily simplify lookups to identify the next hop, and the omission of checksums under the assumption of ever more reliable transmission.

A further measure at the network core is to strictly limit processing to packet headers, as payload inspection would lead to additional processing load, which may even not be strictly bounded depending on the payload content. In most network concepts that apply content-dependent treatment of packets, the associated processing is done for moderate link speeds, before traffic aggregation on highest speed links.

However, further coordinated standardization that considers the consequences of the later implementation could lead to solutions with higher performance or less implementation effort. One example could be reducing the width of fields used as lookup keys. If the width corresponds to the number of lookup entries actually used in a network or on a single network link and if it is small enough, the key could directly be used as lookup address in a standard memory. If this is not possible and fields have to be wider than the actual lookup key, the difference should be bounded or rules about field usage (possible values) should be defined, which both could allow to reduce hash collision probability or even avoid this situation. Such a measure would encourage standard RAM based lookup memory architectures and avoid using additional TCAMs, which are quite power hungry.

An extreme case for easing packet processing would be to avoid table lookups at all. This could be achieved by a special addressing scheme in which the next hop can be determined from the destination address, e. g. geographic routing. Another approach would be to apply source routing. Here, the edge node computes the complete path and the core nodes only need to forward the packet to the next nodes which are specified within the packet. However, the suitability of such approaches for core networks is at least questionable.

Even minor measures like the definition of fixed positions—relative to the packet start—for all fields that are relevant for processing would simplify packet parsing and thus enable higher frequencies in hardware implementations or reduce the number of software operations. Prohibiting the wrapping of fields over word boundaries would also contribute to this effect.

Especially with respect to strictly pipelined architectures, which by design have a bounded number of processing stages, a limitation of functional depth, i.e. the number of operations to be performed on a packet, is an important protocol design goal. For label based schemes this would mean to restrict the size of the label stack or at least the number of labels to be processed in one network node.

Boosting processing throughput by implementing parallel data paths is hampered if the processing state for each traffic flow needs to be maintained. In such a case state information in the memory is usually locked for the duration of the "read-modify-write" sequence in order to guarantee data consistency. Parallel processing of packets belonging to the same flow therefore stalls until the state information is unlocked. The time needed for state processing relative to the complete processing time would then limit the potential to parallelize processing of such protocols, if back-to-back sequences of packets belonging to the same flow make up the traffic input.

Even for pipelined architectures with a highly parallel data path, e.g. in the order of magnitude of a minimum size packet, stateful processing is critical. If minimum packets follow directly one after the other, in each clock cycle a new packet has to be processed. If these packets are from the same flow and if stateful processing is required in a pipeline stage, reading the state information of a subsequent packet would collide with the "modify-write" phases of the previous one. Therefore, from the perspective of both architecture types, stateful processing should be avoided in the high-speed core of the network.

In addition, protocol functions requiring multiplication or even division of arbitrary values, unless limited to powers of 2, are highly undesirable because of the associated time and/or resource complexity. An example for this is the implementation of a generic leaky bucket mechanism as part of a metering and marking function where the calculation of the current bucket size requires a multiply operation before the decision about the packet's acceptance.

These examples show that protocols that are applied in highest speed packet processing should be lean, avoiding as much functionality as possible, and be implementable with very simple operations.

5 Conclusion

The current transition from circuit-switched to packet-switched transport networks together with the introduction of 100 Gbps Ethernet within the next 12 to 18 months raises the question, whether and how packet processing will be feasible at even higher speeds in the future. Product announcements of NPU vendors indicate feasibility for a line rate of 100 Gbps. In this paper we investigated the main challenges towards packet processing beyond 100 Gbps—up to 1 Tbps, which we estimate to be introduced in about six years from now, i.e. in 2015. The results of our studies concerning data plane processing, packet buffering and chip-to-chip interconnects lead to the expectation that these aspects should not inhibit scaling to 1 Tbps line rates.

Pipelined NPU architectures with very wide data paths and inherently short combinatorial delays are scalable to Tbps rates. They will have to be operated at Gigahertz processing frequencies comparable to current state of the art CPUs. Memory bandwidth evolution of advanced DRAM technologies and the application of sophisticated buffering strategies, which restrict the required throughput to a minimum and optimally exploit the available memory bandwidth, will help to meet the throughput requirements of packet buffers. Scalable packet interfaces, which rely on high speed SerDes modules with a throughput of several tens of Gigabits per second, together with an appropriate number of parallel channels should provide sufficient I/O capacity. Furthermore, future packaging technologies should offer sufficient pin density as well.

Our study, however, could not shed enough light on the overall power dissipation and the chip size of potential 1 Tbps NPUs. We are not able not give serious estimates concerning these aspects, since a lack of basic data for today's NPUs hinders a sufficiently deep assessment. Both issues usually have no hard limits that demarcate a region of feasibility, but they make up soft optimization criteria that directly influence the economical competitiveness of a product.

Finally, in addition to the technological feasibility, the question of the associated cost will decide on the introduction of 1 Tbps packet networks. We therefore make non-technological suggestions towards standardization how to alleviate the requirements for future ultra-high speed packet processing in order to relieve the implementation effort and thus cost—as well as possibly power consumption. Our suggestions reach from re-thinking long-established dimensioning rules, over changes of the supported switching granularity, to hints how to reduce the actual processing complexity.

In summary, we expect that 1 Tbps packet switching will become technologically feasible and hopefully also economically and ecologically viable in the years to come. The extrapolated advances in technology along with the application of some of the proposed non-technological aspects should provide the technical prerequisites. The economic impetus is likely given by the ongoing traffic growth of the Internet and the associated need to continuously expand network capacities.

Acknowledgments

The work presented in this paper was partly funded within the 100GET projects 100G ARP and NPU100 by the German Bundesministerium für Bildung und Forschung under the contract numbers 01BP0768 and 01BP0778.

References

[1] H. J. Chao and B. Liu. *High Performance Switches and Routers*. Wiley-IEEE Press, May 2007.

[2] Cisco. The Cisco QuantumFlow Processor: Cisco's Next Generation Network Processor. http://www.cisco.com/en/US/prod/collateral/routers/ps9343/solution_overview_c22-448936.pdf, 2008.

[3] AMCC – Applied Micro Circuits Corporation. nP7310 – 10-Gbps Network Processor with Integrated Traffic Manager. Product Brief, https://www.amcc.com/MyAMCC/retrieveDocument/SNP/nP7310_060822.pdf, 2006.

[4] Cavium Networks. OCTEON Plus CN58XX 4 to 16-Core MIPS64-Based SoCs. Product Brief, http://www.caviumnetworks.com/pdfFiles/CN58XX_PB%20Rev%201.5.pdf, 2008.

[5] Xelerated. Xelerator X11 Network Processors Product Brief. http://www.xelerated.com/uploads/files/5.PDF, 2008.

[6] Bay Microsystems. Chesapeake - Product Overview Page. http://www.baymicrosystems.com/products/network-silicon/chesapeake.php, 2009.

[7] EZChip. Network Processor Designs White Paper. http://www.ezchip.com/Images/pdf/ezchip_white_paper.pdf, 1999.

[8] Netronome. Network Topology Offload with Intelligent NICs. http://xen.org/files/xensummitboston08/2008-06-23-net-topo.pdf, 2008.

[9] Intel Corporation. IXP2800 Product Brief. http://download.intel.com/design/network/ProdBrf/27905403.pdf, 2004.

[10] PCI Industrial Computers Manufacturers Group PICMG. AdvancedTCA PICMG 3.0 short form specification. http://www.picmg.org/pdf/PICMG_3_0_Shortform.pdf, 2003.

[11] EZchip Technologies, Inc. NP-4 Network Processor – 100G NPU with TM. http://www.ezchip.com/products.htm#NP4, 2008.

[12] Xelerated AB. Xelerated HX300 family. http://www.xelerated.com/templates/page.aspx?page_id=329, 2009.

[13] A. Nikologiannis et al. An FPGA-based queue management system for high speed networking devices. *Microprocessors and Microsystems*, 28(5-6):223 – 236, 2004. Special Issue on FPGAs: Applications and Designs.

[14] S. Iyer et al. Designing packet buffers for router linecards. *IEEE/ACM Trans. Networking*, 16(3):705–717, June 2008.

[15] G. Appenzeller et al. Sizing router buffers. *SIGCOMM Comput. Commun. Rev.*, 34(4):281–292, 2004.

[16] J. Sommers et al. An SLA perspective on the router buffer sizing problem. *SIGMETRICS Perform. Eval. Rev.*, 35(4):40–51, 2008.

[17] J. Garcia-Vidal et al. A DRAM/SRAM memory scheme for fast packet buffers. *IEEE Trans. Comput.*, 55(5):588–602, 2006.

[18] R. Giladi. *Network Processors: Architecture, Programming, and Implementation*. Morgan Kaufmann, 2007.

[19] Intel. *Intel Internet Exchange Architecture Portability Framework Developers Manual, SDK 3.5 Release*. Intel corporation, November 2003.

[20] Qimonda AG, www.qimonda.com.

[21] RAMBUS, www.rambus.com.

[22] Samsung Semiconductor. http://www.samsung.com/global/business/semiconductor/products/dram/Products_DDR3SDRAM.html.

[23] Intel roadmap overview. http://download.intel.com/pressroom/kits/events/idffall_2008/SSmith_briefing_roadmap.pdf, 2008.

[24] M. Meghelli et al. A 10Gb/s 5-Tap-DFE/4-Tap-FFE transceiver in 90nm CMOS. *IEEE International Solid-State Circuits Conf., 2006. ISSCC 2006. Digest of Technical Papers.*, pages 213–222, Feb. 2006.

[25] J. Lee et al. A 20Gb/s duobinary transceiver in 90nm CMOS. *IEEE International Solid-State Circuits Conf., 2008. ISSCC 2008. Digest of Technical Papers.*, pages 102–599, Feb. 2008.

[26] M. Harwood et al. A 12.5Gb/s SerDes in 65nm CMOS using a baud-rate ADC with digital receiver equalization and clock recovery. *IEEE International Solid-State Circuits Conf., 2007. ISSCC 2007. Digest of Technical Papers.*, pages 436–591, Feb. 2007.

[27] R.A. Philpott et al. A 20Gb/s SerDes transmitter with adjustable source impedance and 4-tap feed-forward equalization in 65nm bulk CMOS. *Custom Integrated Circuits Conf., 2008. CICC 2008. IEEE*, pages 623–626, Sept. 2008.

[28] H. Shi et al. Study of fundamental limit and packaging technology solutions for 40-gbps transceiver package design. *Electronic Components and Technology Conf., 2008. ECTC 2008. 58th*, pages 1128–1131, May 2008.

[29] J. Nasrullah et al. A terabit/s-throughput, SerDes-based interface for a third-generation 16 core 32 thread chip-multithreading SPARC processor. *2008 IEEE Symposium on VLSI Circuits*, pages 200–201, June 2008.

[30] W. Lautenschläger et al. Frame aggregation in packet core networks – overview and experimental results. In *Proc. 10. ITG Symp.Photonic Networks*, Leipzig, May 2009.