Network Processors: Challenges and Trends

Mahmood Ahmadi and Stephan Wong

Computer Engineering Laboratory

Electrical Engineering, Mathematics and Computer Science Department

Delft University of Technology

Mekelweg 4, 2628 CD, Delft, The Netherlands

Tel: +31 15 27 85021, Fax: +31-15-27-84898

{mahmadi, stephan}@ce.et.tudelft.nl

Abstract— **The aim of this paper is to provide a survey of network processors (NPs), which are a new type of special microprocessors intended for networking equipment, mainly switches and routers. We will describe many aspects in the network processor area. First, we introduce network processors together with their functionalities and requirements. Subsequently, we describe the basic definitions and concepts involved in the network processing area. Second, we describe the architectural specification and implementation of NPs and present comparison between different commercial NPs. Third, we describe network processors software tools including network processor simulators, benchmarks, and other related tools. Fourth, we highlight several challenges and trends in network processing area.**

Keywords: Network processor, network processor simulator, benchmark, grid computing

I. Introduction

The bandwidth growth of networks increased almost exponentially in the past couple of years and is expected to continue to do so for years to come. This has been fueled by emerging new technologies that are capable of achieving higher bandwidths. Consequently, new applications are being developed that take advantage of the new capabilities. In turn, move consumers are starting to use these applications and thereby increasing the demand for higher bandwidth. The technological advances must also be accompanied by improved network processing capabilities within routers and switches that connect the networks. Therefore, network processors have been incorporated within these devices to cope with the continued increasing demand for higher performance. In addition, the multitude of applications and services that require support lead to the introduction of many different protocols that govern the transmission, forwarding, and communication of data (in the form of packets). Therefore, improved flexibility is needed to cope with the many existing and future protocols.

Consequently, the design of network processors remains an ongoing research and development effort. The aim of this paper is to present the recent state-of-the-art of network processors (requirements, software tools, existing architectures) and to discuss the future challenges and trends that we are facing in this field. By no means we intend to be complete as the field in still in movement, but we intend to describe the main recent and possible future developments in the field.

This paper is organized as follows. Section II gives a description of what constitute a network processor and present the requirements that such a processor must meet. Furthermore, several design approaches will be highlighted followed by a survey of existing network processors. Section III presents the software tools used in the design of network processors that include benchmarks and simulators. Section IV describes possible future challenges and trends in the field of network processing. Section V presents the conclusions of this paper.

II. Network processors

In this section, we first present a short description of what constitute a network processor. Subsequently, we present the general, functional, and implementation requirements of such processors. Finally, several design approaches are highlighted and several commercial architectures are presented.

A. Description

A network processor is an application-specific instruction processor (ASIP) for the networking application domain with architectural features and/or special circuitry for packet processing at wire speed [1][2][3]. The network processor differs from traditional microprocessors in three ways:

- The instruction set of many network processors is based on existing RISC processor instruction sets.
- The network processor's instruction set contain special instructions intended for, e.g., bit manipulation, CRC calculation, and search and lookup operations.
- Special hardware function blocks are present to accelerate specific packet processing tasks.

Besides a functional description of a network processor (given by its instruction set architecture), it is equally important to understand at what levels of the network (similar to a protocol stack) the network processor can be utilized: depicted in Figure 1.

- The **core level** includes high-speed components to carry and transfer large amounts of data. The nodes
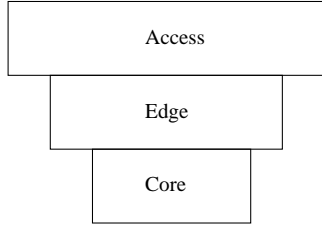
Fig. 1. Simple model of operational levels in network processor.

linking up to form the core implement rudimentary services as routing, tag-switching, and access control.

- The **edge level** of the network forms the ingress and egress to the core. Services at the edge are complex and run at medium to high speeds, services at this point include routing, switching, net-flow, access control, and QoS features.

- The **access level** of the network covers all the delivery points of the Internet. The end-user accesses the Internet through campus networks, broadband connections and dial-up lines. At this level, there are several different protocols and technologies inter-operating with one another at relatively low speeds.

Finally, a network processor can be utilized in two different planes that that differ in the speed and manner they handle incoming packets; namely data plane and control plane.

In the **data plane** simple tasks are performed, and most packets follow the fast path through the NP that required very little processing. In the **control plane** exceptional packets and complex routines are handled. In this plane some packets we sent over to follow slow path [3]. This structure is depicted in Figure 2.
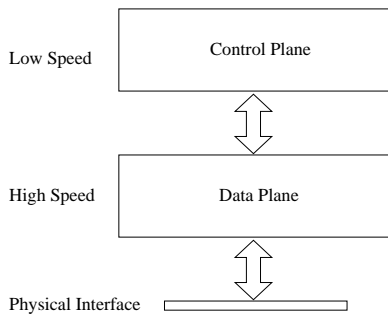

Fig. 2. Packet processing model in network processor

B. General requirements

In this section, we describe the general requirements of network processor.

- **Performance:** By executing key computational kernels in hardware, NPs are able to perform many applications at wire speed. Network processors must be able to support high bandwidth connections, multiple protocols, and advanced features without becoming a performance bottleneck.

- **Flexibility and programmability:** Having software as a major part of the system allows network equipment to easily adapt to changing standards and applications. The network processor should be easily programmable in order to support customization of feature sets and the rapid integration of new and existing technologies. In order to meet this demand, network processor manufacturers must strive to supply programming and testing tools that are as simple as possible to use. These programming tools should be based on a simple programming language that allows for reuse of code wherever possible. In addition, programming tools must provide extensive testing capabilities that provide intelligent debugging features, such as descriptive codes and definitions, as well as code level statistics for optimization. Testing tools must be able to simulate real world conditions and provide accurate measurements of throughput and other performance measurements [4].

- **Fast time to market (TTM):** Time to market has become a critical factor in achieving success with network equipment, it is the time required for system vendor to bring a product from demand to commercial availability and has known as a factor that determined the success or failure of the product in the market [1].

- **Serviceability:** Users are demanding services such as real-time video, secure private networks and voice over IP, these will require lot of serviceability at the access and edge network elements [5].

C. Functional requirements

Typical functions performed by network processors are summarized below:

- **Lookup and pattern matching:** This function compares packet header fields with specific patterns to classify the type of packets, for example perform a table lookup to return the relevant table entry or determining type of incoming packets are an IPv4 or an IPv6 packet.

- **Forwarding:** This function is defined as determining the output path for incoming packets. It is implemented using hardware prefix tree structure and special hardware [5].

- **Access control and queue management:** Once packets have been identified, they are placed in appropriate queues for further processing. Packets are also checked against security access policy rules to see if they should be forwarded or discarded.

- **Traffic shaping and control:** Some protocols or applications require that, as traffic is released to the outgoing wire or fiber, it is shaped to ensure that it meets delay or delay variation requirements. Other requirements specify the priority of traffic between different

channels or message types [2].

• **Data Manipulation:** This is where the packet is modified in some way, this could be decrementing the Time To Live (TTL) field in a IP packet, recalculating the CRC check, performing packet segmentation and reassembly and encryption or decryption of packets.

D. Implementation requirements

Each network processor is combination of many different elements, that are described in the following:

• **The processing engine:** Many network processors are multi-processors, meaning that they are not built as one big RISC processor. The basic programmable unit in the network processor is a processing engine (PE). The PE may be clustered in a group of two or more PEs. Different network processor use different architectures for their PEs, and amount of PEe also differs. The PEs may be grouped into functional blocks or be independent. Moreover, next to network processors sometimes co-processors or hardware accelerators are utilized. A hardware accelerator is the finite state machine that operates independently of the PEs and can be called a functional unit. If a hardware accelerator is programmable it is called a co-processor. The abstract model of network processor is depicted in Figure 3.
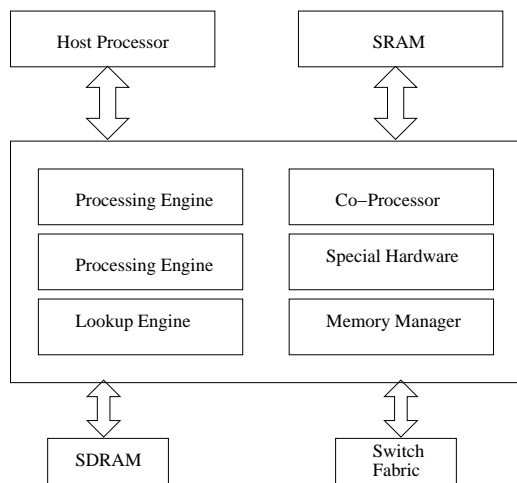

Fig. 3. Abstract model of network processor architecture

• **Exploiting parallelism:** All network processors are using parallel techniques and pipelining. Basically, They use three type of parallelism:
  1. Instruction-level parallelism (ILP).
  2. Thread level-parallelism (TLP).
  3. Packet level-parallelism (PLP).
In ILP, the compiler or hardware instruction scheduler determines simultaneous execution of program instructions. In PLP, a mechanism should be used for packet ordering to allow parallel processing of packets. In TLP, different threads are executed to avoid idle time in memory references and processing engines, i.e, if a thread waits for the memory it is stalled and then another thread is started.

• **NP memory architecture:** A critical resource in NPs is the memory architecture. There are three types of memories in NPs including: instruction memory, packet memory, and route table memory. Instruction memory usually is small because the number of instructions in NP is low. Packet memory that handles the buffered arrival packets, queued, modified packets and read forwarding packets must be designed carefully with a minimum delay. Routing table memory includes routing entry that is read by the NP. The routing table require update operations and lookups thus it must be designed as fast as possible. One solution for mentioned aim is use intelligent data structure, hardware accelerator for lookup, content addressable memory (CAM) and SRAM.

• **Dedicated hardware:** All network processors incorporate special hardware and integrated co-processors to perform common networking tasks. Typical hardware functionality include CRC calculation, queue management, forwarding engine and lookup engine.

• **Network interface:** The most important feature next to a network processor is the network interface. This is the point where packet enter and exit the network processor. In the past, some of the manufactures developed their own network interface but now most network processors implement standard interfaces such as UTOPIA level 2, 3 and SPI-3, SPI-4.

• **Software support:** Thus far, the considerations were all related to the hardware of the network processor, but of equal importance is software support. It is no good selecting the fastest most powerful network processor available, if it is impossible to program it effectively. However, manufactures are now paying much more attention to the software support. Now the more network processor softwares allow to be written in C and certain core routines are written in microcode. This factor relates high programmability degree and will decrease the time to market period.

E. Design approaches

In the previous section, we reviewed the requirements that must be met in network processor architecture design and highlighted common elements accompanying NPs. In this section, we presents common approaches in designing NPs[1][4]:

1. **ASIC (application specific integrated circuit):** Any hardwired solution including a microchip that has been designed from scratch for a specific application. In ASIC technique all of NP element is designed by hardwire and lack of programmability and flexibility is main problems in ASIC, meantime designing by ASIC increased TTM and performance.

2. **ASIP (application specific instruction processor):** An instruction set processor specialized for a particular application domain.

3. **Co-processor:** A hardwired, possibly configurable solution with a limited programming interface.

4. **FPGA (field programmable gate array):** A device that can be reprogrammed at the gate level.

5. **GPP (general purpose processor):** A programmable processor for general purpose computing.

ASICs are the most hardwired (least flexible), but provide the highest performance. GPPs are the most general (flexible) at the cost of the lowest performance. FPGAs provides an interesting value proposition in the absence of ASIPs or co-processors as they are higher performant than GPPs with more flexibility than ASICs.

F. Commercial architectures

In this section we present the architectural specification for some of commercial network processors such as: Agere (PayloadPlus), Alchemy (Au1xxx), Applied-Micro circuits, Formerly MMC Networks (nP7xxx), BRECIS Communications (MSP5000), Broadcom, Formerly-SiByte (Mercurian SB1250), ClearSpeed Network Processor, Cognigine, IBM (PowerNP), Intel (IXP12xx, IXP24xx, IXP28xx), Vitesse and Formerly SiTera (PRISM IQ2000).

Various architectural techniques have been used in NP structure. These techniques are described in the following [1][4][2]:

• **Traditional RISC-based approach architectures:** The RISC architectures are the basic infrastructures of many existing NPs. RISC architectures have simple instructions for fast execution. However, due to RISC's simple ISA, compilers have to generate complex routines using a large number of simple instructions. To overcome this, time consuming tasks are identified and implemented as new instructions in a standard RISC processor ISA. These instructions include bit matching operations, lookup tables, and checksum calculations. The first generation of NPs were designed based on a RISC architecture. AU1000&AU1550 (Alchemy network processor) [6] BROADCOM, BRECIS, Vitesse-Sitera NP and Applied-Micro circuit [1][4]. The major performance bottlenecks in RISC-based architectures are the necessary bus bandwidth to handle instructions, data traffic, and processing cycles required to extract packet data, performing a forwarding table look up, modifying, and transmitting data. Several methods have been employed to overcome these bottlenecks. these

methods are: function partitioning, special instruction and cache optimization [7].

• **Special processor architectures:** Some NPs use special processor architecture techniques, such as modified co-processor and special functional units to improve their performance. For example, Agere routing switch processor and Cisco-PXF TOASTER use VLIW architectures to exploit ILP [8][9][10]. Another method exploits ILP at run time using a superscalar approach that issues several instructions per clock cycle. Cognigine uses this method to find the ILP based on the dynamic behavior of the program. Multi-threading is another technique that has been implemented in some NPs, including the Intel IXP1200/2400/2800, and the IBM PowerNP [1][2][11].

• **Massively parallel architectures with modern RISC approach:** Some NPs utilize multiple PEs to exploit parallelism for example, ClearSpeed network processor is a cluster of several multi-threaded array processors (MTAP). Each MTAP contains up to 256 processing engines (PEs), which are all simple 8-bit processors with a 32-bit wide 4 kB memory unit of their own. The IBM PowerNP, IXP are other instances of this approach [12][13].

The information provided in Tables I and II describe that all NPs have some kind of special hardware to execute common time-consuming tasks faster and executes special instructions to speed up network processing functions. Furthermore the utilization of multiple packet processing engine hides latency and results in reaching higher throughput.

III. Software tools

In this section, we review different network processor software tools including: simulators, benchmarks and related tools. (Which may be used in other networking area fields).

A. Network processor benchmarks

In NP area, no agreed upon standard benchmark exist and existing general purpose benchmarks can not be utilized. The goal of a NP benchmarks to allow for an objective and quantitive comparision between different architectures and to cover the NP application domain. Still, there are sevral issues in designing NP benchmark including: different NPs have different architectures, programming models and languages, and need to support a varying NP application domain from edge to core. Consequently, four levels have been identified in which network processing benchmarks can be defined:

1. **System level:** NP benchmarking in this level covers performance of complete systems such as routers and include both control plane and data plane functionalities. Examples of system level benchmarking are benchmarks for routers and firewalls.

NP	Special Hardware	Special Instructions	Layering Support
Agere Payload-Plus	FFP (Fast Patten Processor), ASI (Agere System interface), RSP (Routing switch processor)	For traffic management, QoS and packet modification	L2-4
Intel IXP 1200	Specialized functional unit for hashing and queue management	yes	L2-4
IBM PowerNP	Co-processor to accelerate tree search and frame manipulation	yes	L2-4
Motorola C-5	Fabric processor, table lockup unit, and queue and buffer management	yes	L2-7
Ezchip NP1	Four special processors, MAC queue, and search engine	Each TOP(Task Optimized Processor) has its ISA	L2-7
Cisco PFX	16 processor packet forwarding function	yes	L2-4
Cognigine	16 Processing element or reconfigurable communication unit	yes	L2-7
Alchemy AU1xxx	MIPS processor	yes	L2-4
BRECIS (MSP5000)	2 DSP processor	yes	L2-4
Broadcom(SB-1250)	2 MIPS 64 bit	no	L3-7
Applied Micro circuit	Packet transform, search, and policy engines	yes (Optimized ins.)	L2-4
ClearSpeed	Table lookup engine	no	L2-4
Virtese Sitera	Co-processor for lookup, classification, and queue management	yes	L2-3

TABLE I

ARCHITECTURAL COMPARISON FOR SOME NPS.

NP	Multiple packet processing	Number of Threads
Agere PayloadPlus	Three (FPP, RSP which one has three VLIW compute engine, and ASI)	64 FPP
Intel IXP 1200	6 programmable engine(four thread each one)	4
IBMPower NP	16 functional unit.	2
Motorola C-5	16 channel processor	4
Ezchip NP1	64 task optimized processor	1
Cisco PFX	16 Processor engine	2
Cognigine	16 RCU each one has 4 parallel unit	4
Alchemy AU1xxx	MAC(Multiply &Accumulate)	1
BRECIS (MSP5000)	Co-Processor (ADPCM)	1
Broadcom(SB-1250)	No	1
Applied Micro Circuit	No	8
ClearSpeed	Multi Thread Array processor include 256 PE	32
Virtese Sitera	Co-processor, 4 RISC processor	1

TABLE II

MULTI PROCESSING COMPARISON FOR SOME NPS

2. **Function level:** In this level, benchmarks cover the performance of specific NP application functions and include data plane functionalities. Example of function level are IP forwarding filtering and NAT applications.

3. **Micro level:** In this level, benchmarks cover elementary operations that are commonly put together to make up a function such as string searcher, CRC calculation, and longest prefix match table lookups.

4. **Hardware level:** In this level, benchmarks measure the latencies and throughputs for accessing the various hardware resource in the NP. This level is architecture specific and it can be defined for a given architecture, therfore this level does not provide a metric for comparing the performance of two different network processors.

Network processor benchmarks are described in the following:

• **CommBench**: A methodology for studying performance of network hardware is CommBench. The CommBench is similar to SPEC for traditional architectures [14][15]. The benchmark is composed of eight programs, four of them oriented towards packet header processing and four oriented towards data stream processing. The instruction mix of CommBench are similar when compared across the entire benchmark, but the payload processing applications execute significantly more add/sub, shift, and logic operations [14].

• **NPBench:** NPBench is a set of benchmarks targeted towards control plane (traffic management and quality of service) as well as data plane workloads. The char-

acteristics of NPBench workloads, such as instruction mix, parallelism, cache behavior and required processing capability per packet are described in[16].

• **EEMBC:** Embedded microprocessor benchmark consortium, was formed in 1997 to develop meaningful performance benchmarks for the hardware and software used in embedded systems. It defines a set of 34 application benchmarks in the areas of networking, communication, telecommunication and industry domains. In the networking domain, EEMBC define only the benchmarks including Patricia route lookup, Dijkstra's OSPF, and packet flow between queues [17][1].

• **NetBench:** NetBench is a suite of benchmarks that allow users to approximate the performance of processors tasked with moving packets in networking applications. The suites seven benchmark kernels are the following: IP packet check, IP reassembly, IP network address translator (NAT), route lookup, open shortest path first (OSPF), quality of service (QOS) and TCP [18]. It defines and classifies a set of nine network processor benchmarks at the microlevel [1].

• **NPF Benchmarking Work Group (NPF-BWG):** NPF-BWG defines benchmarks at microlevel, function level, and system level applications.

• **Intel NP benchmark:** This benchmark specifically focuses on the IXP1200 NP, it operates at four levels: Hardware level, micro level, function level, and system level. It provides some results for IP forwarding on Intel IXP1200 [1].

B. Network processor simulators

In this section, we present several network simulators for network processor simulation and other networking area.

• **NEPSIM simulator:** the NEPSIM simulator is an integrated infrastructure for analyzing and optimizing NP design and power dissipation at architectural level. NEPSIM is based on IXP1200 Intel architecture [19][20].

• **Component network simulator (ComNetSim):** The goal of this simulator is to model the behavior of network processing components. This simulator implemented and simulated an abstract model of the Cisco-Toaster network processor and it is implemented by object oriented programing in c++ environment [1][2].

• **The network simulator (NS 2.0):** This system is used to study the behavior of algorithms and protocols at the network level with interaction between different nodes. NS2.0 provides substantial support for the simulation of TCP, routing and multi-cast protocols over wired and wireless (local and satellite) networks [21].

• **The network animator:** The network animator began in 1990 at LBL as a simple tool for animating packet trace data. This trace data is typically derived as output from a network simulator like NS2.0 or from real network measurements. It supports topology layout, packet level animation and various data inspection tools [22].

• **The CNET network simulator (v2.0.9):** This simulator enables experimentation with various data-link layer, network layer, routing and transport layer networking protocols in networks consisting of any combination of point-to-point links and IEEE 802.3 Ethernet segments. It was developed at the university of Western Australia, CNET runs on a variety of Unix and Linux platforms [23].

• **National Chiao Tung university simulator (NCTU):** The NCTU network simulator is a high-fidelity and extensible network simulator and emulator capable of simulating various protocols used in both wired and wireless IP networks. The NCTUns can be used as an emulator, it directly uses the Linux TCP/IP protocol stack to generate high-fidelity simulation results, and it has many other interesting qualities. It can simulate various networking devices. For example, Ethernet hubs, switches, routers, hosts, IEEE 802.11 wireless stations and access points, WAN (for purposely delaying/dropping/reordering packets), optical circuit switch, optical burst switch, QoS DiffServ interior and boundary routers. It can simulate various protocols for example, IEEE 802.3 CSMA/CD MAC, IEEE 802.11 (b) CSMA/CA MAC, learning bridge protocol, spanning tree protocol, IP, mobile IP, Diffserv (QoS), RIP, OSPF, UDP, TCP, RTP/RTCP/SDP, HTTP, FTP and telnet [24].

• **Georgia-tech network simulator (GTNetS):** The Georgia-tech network simulator is a full-featured network simulation environment that allows researchers in computer networks to study the behavior of moderate to large-scale networks, under a variety of conditions. The design philosophy of GTNetS is to create a simulation environment that is structured much like actual networks are structured. Simulation objects representing network nodes have one or more interfaces, each of which can have an associated IP address and an associated link. layer 4 protocol objects in GT-NetS are bound to ports. Connections between protocol objects at the transport layer are specified using a source IP, source port, destination IP, destination port tuple just like actual TCP connections. The interface between applications and transport protocols uses the familiar connect, listen, send, and send to calls much like the ubiquitous sockets API in Unix environments [25].

• **The packet lookup and classification (PALAC) simulator:** This simulator developed at Stanford university provides a framework for studying behavior of packet classifying algorithms. The system includes a traffic generator, packet classifier algorithms, and statistics collection mechanisms. The goal of this simulator seems to have been to design and implement efficient packet classification algorithms [26].

- **Scalable wireless network simulator (SWANS):** the scalable wireless network simulator was created primarily as a validation of the virtual machine-based approach to simulator construction. SWANS is organized as independent software components that can be composed to form complete wireless network or sensor network configurations. Its capabilities are similar to ns2, but is able to simulate much larger networks. SWANS design to achieve high simulation throughput, save memory, and run standard Java network applications over simulated networks. In addition, SWANS implements a data structure called hierarchical binding for efficient computation of signal propagation [27].
- **OMNeT++:** The OMNeT++ simulator is a component-based, modular and open-architecture simulation environment with strong GUI support and an embeddable simulation kernel. Its primary application area is the simulation of communication networks and because of its generic and flexible architecture, it has been successfully used in other areas like the simulation of IT systems, queuing networks, hardware architectures and business processes as well. OMNeT++ is rapidly becoming a popular simulation platform in the scientific community as well as in industrial settings. Several open source simulation models have been published, in the field of Internet simulations (IP, IPv6, MPLS, etc), mobility and ad-hoc simulations and other areas [28].

C. Network processor related tools

The one of interesting work related to network processor refered to by E. Kohler in MIT as the click modular router. It provides a good framework to study the router functionalities.

The click modulator router: Click is a new software architecture for building flexible and configurable routers. Click software runs in the Linux kernel on conventional PC hardware. A click router is assembled from packet processing modules called elements. Individual elements implement simple router functions like packet classification, queuing, scheduling, and interfacing with network devices. A router configuration is a directed graph with elements at the vertices's, packets flow along the edges of the graph. Configurations are written in a declarative language that supports user-defined abstractions. This language is both readable by humans and easily manipulated by tools [29][30][31].

IV. Challenges and Trends

A. Challenges

Some of important challenges in network processor area are:

- **Maintain high programmability and flexibility:** Due to attention exponential growth in bandwidth and variety in applications, NP must be able to perform many application in wire speed, in other word NP must be able adapt to software and hardware changing.

- **Achieve high performance:** Network processors must be able to support large bandwidth connections, multiple protocols, and advanced features without becoming a performance bottleneck. Network processors must be able to provide wire speed, nonblocking performance regardless of the size of the pipe, the type of protocol or the features that are enabled.

- **Achieve lower TTM (time to market):** Time to market has become a critical factor in achieving success with network equipment, it has known as a factor that determined the success or failure of the product in the market and relates to programmability and flexibility.

The most important challenge for network processor designer is to have flexibility in software and to archive high speed using ASIC or SOC-based network processor.

B. Trends

The most important trends in network processors area are:

- **FPGA and reconfigurable architecture network processor:** With due attention to much complexity in higher level of protocol stack, each network processor must have elements that can dynamically run different part in higher layer of protocol stack. In other words future network processors include two parts [32]: an ASIC part for the data plane part, and an FPGA part for the control plane part. Two FPGA network processor were designed that show FPGA architectures are at the starting point in this area and the next generation of high speed network processors are moving towards being based on FPGA [33][34]. Some instances FPGA network processors are ProPars and VALUE network processor [34].
- **Parameterizable hardware platform for multi-NPs architectures:** A trend exists to have dynamic elements in network processor architecture. Dynamic parameters in NPs structure are done in two levels including: register level and processing element level.
Register Level: There are different implementations from mentioned alternative. In first case one of interesting work has been done in Georgia university [35], in this work, the base platform is IXP1200 network processor and the properties of applications running on the network processor have been studied. It has been observed that their imbalanced register requirement across different threads at different program points could lead to poor performance so it uses register allocator aiming to distribute available registers to

different threads according to their need [35].

Processing element level: A method to achieve more flexibility in network processor architecture using reconfigurable computing. An effort has been done combination reconfigurable elements to IXP network processor using BONeS simulator [36]. In other works has been designed network processor with parameterizable processing element in cluster and number of clusters that can be adaptable to different application domain including core and edge area [37][38].

- **Packet differentiation filtering and processing:** This is a trend to separate redundant packets from the normal packet stream. In other words, redundant packets in the processing phase are separated and inserted again after processing. Some of these packets are acknowledgment and hello packets. The FlexPath NP was designed to map network processor application sub-function onto both software programmable processor resources and reconfigurable hardware building blocks, such that different packet flows are forwarded via different, optimized processing path through the NP. In FlexPath there are different packets path to CPU, co-processor and traffic manager that path selection operation is done by path dispatcher [39].

- **Designing quantitative evaluator software:** Network applications vary as fast as NP architectures do, each application has specific packet pattern. Designing a NP based application is complicated hence it is necessary to study the structure of packets and traffic patterns when designing network processors [26]. It is better to perform quantitative studies of the time consumption of the different stages in packet processing, along with statistics of the distribution of packets in Internet traffic and using of these criteria for optimizing and designing NPs.

- **Designing operating system in network processors:** One of most important challenges in network processor area is programmability and flexibility in software. To make full use of the capabilities of network processors, it is imperative to provide the ability to dynamically adapt to changing traffic patterns and to provide runtime support in the form of a network processor operating system. The differences to existing operating systems and the main challenges lie in the multiprocessor nature of NPs, their on-chip resources constraints, and the real-time processing requirements. Network processing is inherently a dynamic process. Changing traffic patterns, new network services and protocols, new algorithms for flow classification, and changing defenses against denial of service attacks present the dynamic background that a programmable router needs to accommodate. This requires that the router can:

  – Implement multiple packet processing applications at the same time.

  – Quickly add and remove processing functions from its workload.

– Ensure efficient operation under all circumstances. In particular, the management of various system resources is important to avoid performance degradation from resource bottlenecks. The complexity of NP multiprocessor architectures has limited the use of existing operating system concepts in this domain [40][41]. It is important to note that the goals of an operating system for network processors are very different. Some differences between network processor operating system and conventional operating system are: separation between control and data path, limited interactivity, regularity and simplicity of applications, processing dominates resource management. There is no reference model for network processor operating systems some efforts has been done related to data plane [40] and other in NIC architecture [41].

- **Asynchronous design of network processors:** Some RISC processor architecture have been designed using asynchronous techniques but these products are not available. One limitation in network processor area is the clock rate limitation. This limitation can be eliminated by asynchronous design [42][43][44]. Examples are high-speed non-linear asynchronous pipeline, an asynchronous dataflow FPGA architecture pipelines, an asynchronous super-scalar architecture for exploiting instruction-level parallelism and asynchronous implementation of parallel architectures [45][46][47].

- **Small-scale and ad-hoc networks:** In an ad-hoc network, a set of wireless stations communicate directly with one another without using an AP(access point). The devices in ad-hoc networks do not communicate with a central base station or server, but rather with its peers in the local environment. The suggestion is a network processor designed specifically for such an environment. Compared with its larger cousins in high-end fixed networks, such a mobile NP would not need parallel processing engines. At a couple of megabits per second, there are several microseconds for each packet even at the worst case (all ACKs) and one packet processing engine could be enough for that purpose. Other issues are of interest though. Since the protocols used for communication over a wireless channel need to compensate for the lossy nature of the medium, those parts of the protocol suite could be reflected in the NP design. Then there is the ad-hoc part. Routing tables need to be updated frequently. In a full-scale NP, this is commonly done by the host GPP, but in the mobile case there is reason for putting it in the NP. One way to design a mobile NP is thus a set of two simple processors on one chip: a simple general-purpose RISC core and one PE with packet handling instruction. The GPP takes care of routing table management and exception handling. It could also be responsible for invoking the host CPU if unresolvable packets appear. The PE does classification and calls on the look-up engine to get the next-hop address. Since memory latency always is a problem,

it would be wise to equip the PE with support for two simultaneous threads, so that two packets could be processed at the same time [2].

- **Grid computing and network processors:** Applying the resources of many computers in a network to a single problem at the same time usually a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. Grid computing uses software to divide and farm out pieces of a program to as many as several thousand computers [48][49]. With NPs in the network infrastructure performing the routing of packets, it would be matter of software to let the NPs participate in the forming of the grid by having them, the NPs, take care of the load balancing and the distribution of computing tasks to different computers in the grid. Just as a regular router forwards packets along the supposedly best way to a destination, a Grid router could forward computing tasks to the part of the grid and eventually to the computer or the computers where they are best served [2][50]. It is interesting to investigate what such a network, where the network does more than forwarding data, but also takes part in the processing of tasks, would demand from the equipment that performs the necessary functions. If the NPs enable a new form of grid how would that grid affect the NPs? How can optimized NPs for grid processing?

V. Conclusion

In this paper, we introduced the field of network processors by first presenting a description of what network processors are. Subsequently, we described the general, functional, and implementation requirements to such processors must meet. Afterwards, a brief survey of design approaches was discussed in conjunction with a summary of commercially available network processors. Second, we described in detail benchmarking approaches and presented several instances of such benchmarks. This was followed by survey of existing network processor(-related) simulators. Finally, we presented an overview of several challenges and identified many trends that we are witnessing nowand that continue in the near future.

References

[1] P. Clowley, M. Franklin, and H. Hamidioglu, *Network Processor Design: issues and practices.* Morgan Kaufmann, 2003.

[2] D. Suryanarayanen, "A Methodology for Study of Network Processing Architecture, M.Sc. Thesis," Master's thesis, North Carolina State University, 2001.

[3] "White Paper for Roke Manor Research An Introduction to Network Processors," http://www.roke.co.uk/ download/ white_papers/ network_processors_introduction.pdf.

[4] N. Shah, "Understading Network Processors," Master's thesis, Berkeley University, 2001.

[5] "White Papaer Challenges in Building Network Processor Based Solutions," www.futsoft.com/pdf/ NPwp.pdf.

[6] "Product Brief, AMD Alchemy, Au1550 Processor," http://www.amd.com/usen/Connectivity Solutions/Product Information.

[7] F. Khunjush, M. Watheq, E. Kharashi, F. Li, and N. Dimopoulos, "Network Processor Design: Issues and challenges ," in *Proc. IEEE Intl. Conf. on Communications, Computers and signal Processing.*

[8]

[9] "White Paper, 10G Network Processor Chip-Set," http://www.agere.com.

[10] A. Company, "Product Brief Agere System PaylaodPlus APP750NP," October 2003.

[11] "Intel IXP2850 Network Processor," http://www.intel.com /design/ network/ products/npfamily/ ixp2850.htm.

[12] "Clearspead Network Processor," http:// tech.nplogic.com/ survey/ clearspeed.html.

[13] "Clearspead Network Processor," http://www.clearspeed.com .

[14] T. Wolf and M. Franklin, "Commbench: A telecommunications benchmark for network processor," in *IEEE Intl. Symp. on Performance Analysis of Systems and Software (ISPASS-2000).*

[15] B. Liljeqvist, "Vision and Facts, Asurvey of Network Processors, M.Sc Thesis," Master's thesis, Chalmers University, 2003.

[16] B. K. Lee and L. K. John, "NpBench: A Benchmark Suite for Control Plane and Data Plane Applications for Network Processors," in *Proc. IEEE Intl. Conf. on Computer Design (ICCD'03).*

[17] "Network Animator Network Simulator," http:// www.eembc.org/ TechLit/Index.asp.

[18] "EEMBC Network Processor Benchmark," http://www.eembc.org/benchmark/networking2_sl.asp.

[19] Y. Luo, J. Yang, L. N. Bhuyan, and L. Zhao, "NepSim: A Network Processor Simulator with Power Evalution Framework," *IEEE Micro Special Issue on Network processors for Future High-End System and Applications*, pp. 33–44, September 2004.

[20] "NepSim Simulator," http://www.cs.ucr.edu/ yluo/ nepsim/.

[21] "The Network Simulator ns.2.0," Http:// www.isi.edu /nsnam/ns/.

[22] "Embedded Microprocessor Bencmark Consortium," http://www.isi.edu/nsnam/nam/.

[23] "CNET Network Simulator," http:// www.csse.uwa.edu.au /cnet/.

[24] "NS Network Simulator," http:// nsl.csie.nctu.edu.tw/ nctuns.html.

[25] "GTNets Network Simulator," http:// www.ece.gatech.edu/ research/labs/ MANIACS/ GT-NetS/.

[26] J. Balkman and N. Mckeown, "The Packet Lookup And Classification (PALAC)," http:// klamath.stanford.edu/tools /PALAC/SRC/.

[27] "SWANS Network Simulator," http:// jist.ece.cornell.edu/ swans-user/ index.html.

[28] "OMNETPP Network Simulator," http:// www.omnetpp.org/ index.php.

[29] E. Kohler, "The Click Modular Router," Ph.D. dissertation, MIT University, 2000.

[30] E. Kohler, R. Morris, B. Chen, J. Jannoti, and M. F. Kaashoek, "The Click Modular Router," *ACM Transaction on computer systems*, pp. 263–297, August 2000.

[31] "Click Modular Router," http://pdos.csail.mit.edu/click/.

[32] X. Nie, U. Norvviqst, L. Gazsi, and D. Liu, "Network Processors for Access Network Trends and Chalenges," *Proc. IEEE Conf. on Parallel and Distributed Computing Systems*, 2004.

[33] M. Coss and R. Sharp, "The Network Processor Decision," *Bell Lab. Technical Journal*, pp. 177–189, September 2004.

[34] "FPGA Network Processor," http://www.roke.co.uk/ networks/ hardware/ network_processor_cores.asp.

[35] A. Zhuang and S. Pande, "Balancing Register Allocation Across Threads for a Multithreaded Network Processor," in *Proc. Conf. on Programming language design and implementation.*

[36] L. A. Troxel, A. D. George, and S. Oral, "Design and Analysis of a Dynamically Reconfigurable Network Processor ," in *Proc. IEEE Local Computer Networks.*