Volume 2, Issue 6, June 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Analysis of Network Processor Processing Elements Topologies

Devesh Chaurasiya[1], Pushpendra Singh[2], Ankita Joshi[3], Kumar Sambhav Pandey[4]
CSE Department & NIT Hamirpur
India
devu.chaurasiya@gmail.com

Abstract— **An exponential growth in high-speed requirement in computer networks due to increased number of users, servers, connections and demands for new applications, along with the tremendous growth in data traffic has claimed the development and deployment of high-speed telecommunication systems. A solution to this problem is to use specialized processors, called network processors (NPs). These application specific instruction processors (ASIPs) are specially designed to perform packet processing task and their architecture is usually a question of different trade-offs between performance, flexibility and price. So it is therefore important to study about NPs processing elements topologies, as well as investigate which topology is efficient in terms of performance (Throughput). In this paper, we form two types of NPs processing elements topologies, parallel and pipelined topology. By analysing these two topologies our analysis indicate that pipelined topology is efficient as compared to parallel topology in terms of performance (Throughput).**

Keywords— **Network processors, Processing elements, Parallel, Pipeline, Engine.**

I. INTRODUCTION

A network processor is an ASIP for the networking application domain. It is a software programmable device with architectural features and/or special circuitry designed specifically for packet processing at line rate of high speed networking. Network processors are emerging on the market to provide the performance of the ASICs and programmability of general-purpose processors. Current approaches for the design of network processors are different in terms of processing, capacity, flexibility and hardware architecture [3]. Despite these differences, their main purpose is to process packets at line rates of high-speed networking. To achieve this, they all exploit some kind of packet level and task level parallelism [1] [3].

First generation Internet routers use a microprocessor centric architecture, which is very similar to PC or workstation used for general purpose computing. In addition to management functions such as running routing protocol, the microprocessor is also responsible for processing required on a per packet basis, including routing table lookup and queuing management, the explosive growth of Internet leads to rapid increase in transmission capacity. In the last decade, network speed increased by a factor of 240 while CPU clock speed only increased by a factor of 12. A CPU centric architecture no longer can handle the increasing processing requirement [4] [7].

In order to meet demands on flexibility and performance, network traffic not only needs to be forwarded, but also processed inside the network. This type processing is performed on routers, where processors inside input port and output port can be programmed to implement a range of packet classification to complex payload modification functions.

To provide sufficient flexibility in network processors for increasingly complex network services at the edge. Currently router designs are moving away from the hard-wired ASIC solutions. Instead, programmable "network processors" (NPs) have been emerged in recent years. These NPs are mostly single-chip multiprocessors with high performance components. Network processors are usually located on a physical port of a router. Packet processing is performed on the network processor before the packets are passed on through the router switching fabric and through the next network link [2]. This is illustrated in Figure 1.

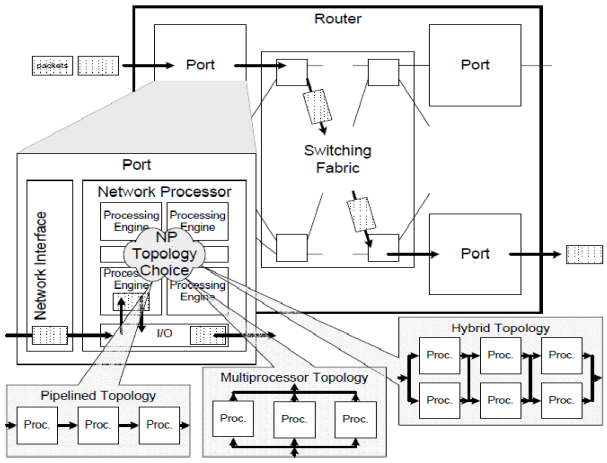

Figure 1: Router System with Network Processors.

One of the key architectural aspects of a network processor is the system topology that determines how processing engines are interconnected and how parallelism is exploited. This is an very important characteristic that determines the system performance as well as the flexibility and scalability of the architecture [2]. In principle, there are several approaches to arranging processing engines: in parallel, in a pipeline, or a hybrid configuration. The goal of this paper is to analyse the performance of these approaches. The analysis of these approaches allow us to understand the trade-offs between different topology.

II. RELATED WORK

There are three domains of related works, those addressing the modelling of network processors, those addressing the programming of network processors, and those addressing the evaluation of network processor applications.

Currently, there are a wide variety of network processor designs. The trade-offs of these designs are flexibility, programmability, performance, cost, development time and power consumption. The high demand for increased processing speed, Adaptability, High level of programmability in network devices drives the concept of network processor [3]. The packet forwarding is basic application of network processor. But besides packet forwarding, packet processing has gained more importance in today era. In network processor wide range of application could be termed as packet processing. The design space of network processors has been explored in several ways. Crowley et al. have evaluated different processor architectures for their performance under different networking workloads [5]. This work mostly focuses on the trade-offs between RISC, superscalar, and multithreaded architectures. In more recent work, a modeling framework is proposed that considers the data flow through the system [6]. Thiele et al. have proposed a performance model for network processors [8] that considers the effects of the queuing system.

III. DESIGN EXPLORATION

We have analysed various network processors, based on this analysis we have come to understand the network processor consist two different types of computational unit, first processing element and second micro engine. Computational units are individual processing unit of network processor.

A. Processing Elements (PE's)

Processing element is basis processing unit of network processors. These are instruction set processors that decodes its own instruction stream.

B. Micro Engine

These are special purpose hardware that is dedicated to perform one specific task also called engine. Micro engines are generally triggered by processing elements.

The basic building units of the network processor are processing blocks that can be seen as abstractions of PE's. The processing blocks are internally connected into a processing block topology. A processing block topology is formed by connecting processing blocks with unidirectional FIFO channels. A channel connects the output port of one processing block with the input port of another. Channels are also used to connect engines. The topology can be seen as a directed graph where processing blocks and engines are nodes and channels are edges. Packets entering a forwarding element may take different paths in the topology depending on the decisions made in the packet processing.

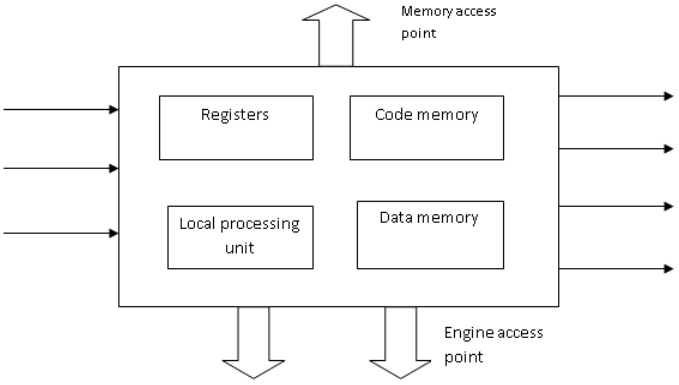

Figure 2: structure of processing block

The programming of a processing block is event-driven. When an event occurs, the local processing unit is triggered to run a particular program code in code memory. For example, a packet arrival may trigger the local processing unit to receive the packet from an input port. The packet is processed within the block, and is finally sent to an output port. Packet processing inside a network processor can be divided into two phases, ingress processing phase and egress processing phase. Packet enters from ingress phase then packet is processed by processing blocks and exit from the egress phase. Some packets need special treatment that is handling by the route processor, which is called slow path processing. Here we will primarily consider the fast-path processing where packets enter the ingress and are sent directly to the egress.

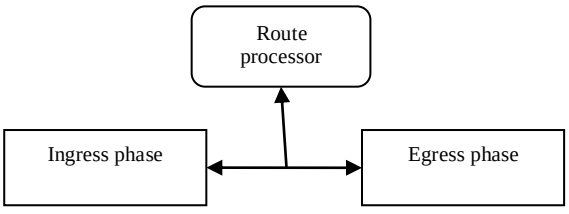

Figure 3: Phases of Network processer

Here we will focus only on the ingress processing because processing capacity required by egress phase is very less as compared to ingress phase.

IV. ANALYSING NPs PROCESSING ELEMENT TOPOLOGIES

The analysis of NPs processing element topology requires an abstract representation of NP architecture. We introduce a general NP architecture that allows us to analyse the NP performance and compare their different processing element topologies.

We use a general, parameterized network processor topology shown in Figure 4. The system topology is characterized by three components: processing elements, shared interconnects, and memory interfaces. The packets move from top to bottom. The key parameters are:

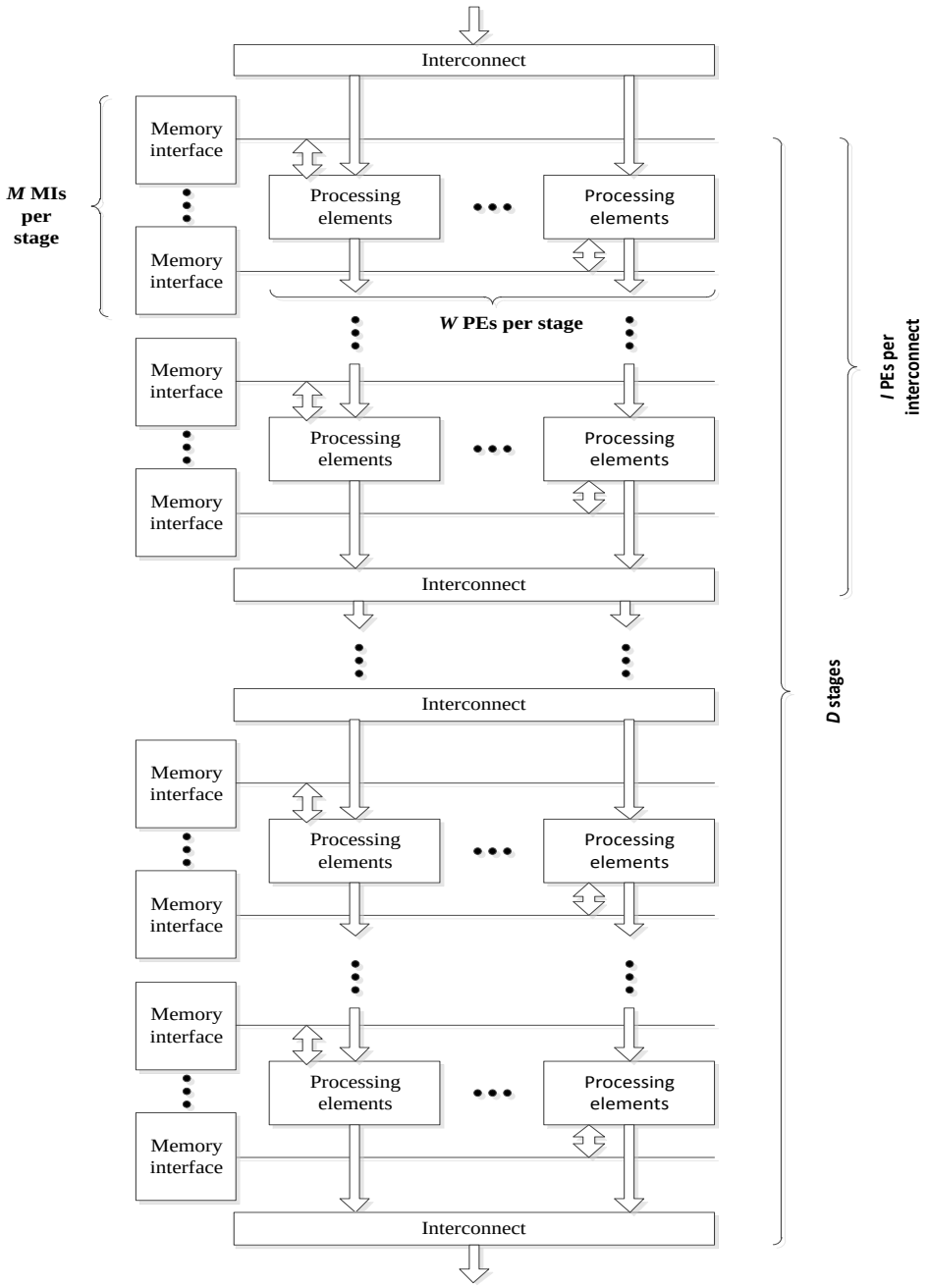

Figure 4: Generic Network Processor Architecture. The parameters which can be varied are shown in the figure.

The width of the pipeline stage (W), depth of the pipeline stage (D), number of stages per communication interconnect (I), number of memory channels shared by one row of processing elements (M). The variations in these parameters allow our generic NP architecture to represent a wide range of possible NP topologies.

A. Parallel Multiprocessor Based Topology

A parallel topology can be formed by setting the D=1 and I=1, while varying the pipeline width (W) shown in Figure 5.

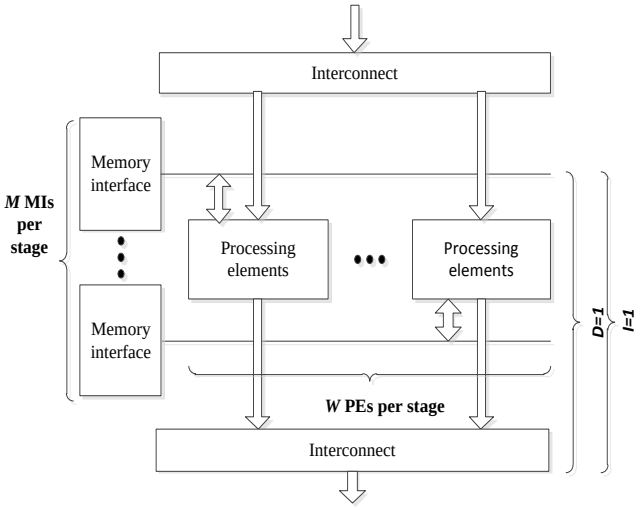

Figure 5: Parallel Multiprocessor Based Topology

B. Pipeline Based Topology

A pipeline topology can be formed by setting the W=1 and I=1, while varying the pipeline depth (D) shown in Figure 6.

The generic NP architecture model can represent a large range of NP topologies, but not all possible combinations are feasible. The available chip area limits the number of processors and interconnects; available power limits the processor clock rates; and packaging technologies limits the number of available pins for off-chip memory interfaces. We have just theoretically analysed the generic NP architecture in figure 4 for two types of processing element topologies. In this work, our analysis indicates that the pipelined based topology scheme outperforms the parallel based topology scheme in terms of throughput. Because packet drop happens in parallel topology even at lower line rate for bursty traffic whereas in the pipelined topology scheme no packets are dropped. This trend of higher packet drop in parallel topology scheme is even more pronounced at higher line rates. This is because under the parallel topology scheme, a single thread is responsible for the entire processing of packets and a thread is occupied during the entire period of packet processing. Whereas in the pipelined topology a dedicated set of micro engines and threads are available even though they are fewer in number, can move packets from ingress to egress with less number of packet drops as compared to parallel topology. Due to less number of less numbers of packet drops in pipeline topology throughput is also high as compared to parallel topology.

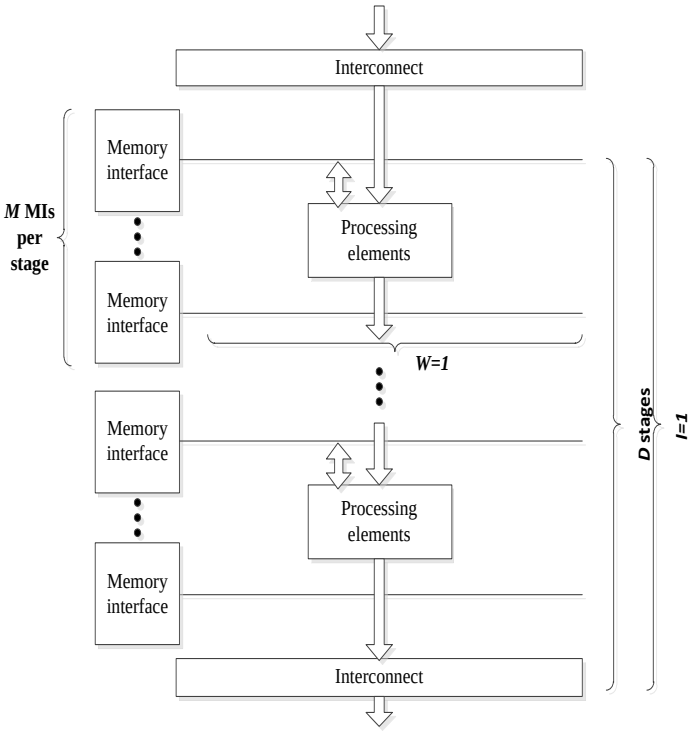

Figure 6: Pipeline Based Topology

V. CONCLUSIONS

In this work, we have analyzed the two different NP topologies parallel and pipeline topology by varying different parameters in generic NP architecture and our analysis indicates that pipeline topology outperforms the parallel topology because of less number of packet drops due to less number of less number of packet drops throughput is also high as compared to parallel topology.

ACKNOWLEDGMENT

This work was supported in part by Ministry of Human Resource Development (MHRD) and the Department of Computer Science and Engineering, N.I.T. Hamirpur (H.P.).

REFERENCES

[1] Danijela Jakimovska et al. "*Performance Estimation of Novel 32-bit and 64-bit RISC based Network Processor Cores*," (JSAT), May Edition, 2011

[2] Ning Weng et al. "*Pipelining vs. Multiprocessors–Choosing the Right Network Processor System Topology*," University of Massachusetts at Amherst, 2004

[3] Jing Fu and Olof Hagsand "*Designing and Evaluating Network Processor Applications*," IEEE 2005.

[4] Jingnan Yao et al. "*Optimal Network Processor Topologies for Efficient Packet Processing*," IEEE GLOBECOM, 2005.

[5] P. Crowley et al. "*Characterizing processor architectures for programmable network interface,*." International Conference on Supercomputing, May 2000.

[6] P. Crowley and J.-L. Baer, "*A modelling framework for network processor systems*," (HPCA-8),Feb 2002.

[7] Kanishka Lahiri et al. "*System-Level Performance Analysis for Designing On-Chip Communication Architectures,"* IEEE TRANSACTIONS, June 2001.

[8] L. Thiele, S. Chakraborty et al. "*Design space exploration of network processor architectures*," (HPCA-8), Feb 2002.

