A simulation environment for network processor based on simultaneous multi thread architecture

Mahdi Koohi[1], Hassan Bayadi[2] and Mansour Nourmohamad Khaless[3]

[1] *Department of Electrical Engineering, Islamshahr Branch, Islamic Azad university, Tehran, Iran*
[2] *Department I.E Engineering Sience&Technology, Mazandaran university, Mazandaran, Iran*
[3]*Department of electrical engineering, East Tehran Branch IslamicAzad University, Tehran, Iran*
m.kohy@yahoo.com; Bayadi@bayadi.com; mansour.khaless@gmail.com

Abstract

This study is centered on the architectural efficacy of network processors based on simultaneous multi thread (SMT) architecture that support multiple active concurrent hardware threads at the same time with the goal of sharing processor resources such as functional units and memory. This thread-level-parallelism can be exploited in packet processing devices called network processors. The programs running on network processors have different characteristics depending on where the processor is used and the input workload. In this paper , we introduce our simulation environment, called SNP (Simulator for Network Processor), for simulating a typical SMT network processor which includes a connected network controller and a packet generator. The simulator presented in this study is a process based "simulator for Network processor" (SNP) usable for different modes and applications based on "clock – by – clock" method and is described according to SMT Structure. The simulator architecture is performed by C++ language optimized as efficient as possible. Such simulator is similar with simple scalar simulator.

Keywords: Network Processor, Router, Packet, Simulator, architectural efficacy

1. Introduction

The fast growth of network lines speed and the need for new services in routers, have lead to appearance of a new generation on microprocessors called "Network processor". In this article we present a highly fixable and exact simulator designed and performed in order to study the architectural efficacy of network processor based on simultaneous multi thread (SMT) architecture proposed by us in another related article. This study details different components of this simulator and simulation basis along with related algorithms. We started by defining the proposed simulator structure with its adaptable parameters. Then we presented the Network controller unit, arrangement of packet in memory , Software models of simulation , Classifier program, Processes programs along with Memory classification and Patterns and parameters of simulation. The goal of study is to introduce a "Simulator for Network processor" (SNP) in order to studying the architectural efficacy of network processor based on simultaneous multi thread (SMT) architecture.

1.1 Definition of the proposed simulator structure

The simulator presented in this study is a process based "simulator for Network processor" (SNP) usable for different modes and applications .This simulator is based on "clock – by – clock" method and is described according to SMT Structure. The simulator architecture is performed by C++ language optimized as efficient as possible . Such simulator is similar with simple scalar simulator (Jing Fu Hagsand, 2005; Wolf Tilman & Franklin mark 2006; Burger & Austin,1997) used in most simulator. One of the major differences of SNP with simple scalar simulator is related to its multi threads nature. Another difference is that, in addition to simulating architecture and practice, SNP simulates the network environment, network traffic producer and the controller of network connected to processor, and performs most statistics needed in network processing. The performance accuracy of such simulator is studied by performing different programs and compared with similar cases. One of the uncompleted projects in this simulator is producing high level languages compiler (such as C) for it and for such current version. Assembly input is directly given to the simulator. This simulator achieves the assembly file program and configuration file of the system as input andafter simulating the program performance produces different reports about its performance as output. Some of these report include cach log , icahe leg , issue .log , write back log , tus logs , fuslog, instlog and system log different parts of such simulator and their relationship.

1.2 Adaptable parameters of the simulator

The system cfg (configuration system file) includes many properties in the system and by its regulating, the system may be activated with different properties and different result will be studied in it.

1.3 Entrance traffic production

This process is very important without which calculation may be incorrect. Such traffic may beproduced by complex and difficult methods but another method is using archives in internet which is used in our study. The used archive here is traffic DEC-PKT from the complex ITA(Gries , M. Kulkarni ,C. Sauer ,C. Keutzer, k2003).including traffic company DEC with the rest of the world, in special time field. This file includes different files (TCP,UDP) stored as ASCIand stores important data for each packet(except the body components).To be unusable, some data in these files are changed for example ,IP address of the packets origin and destination in these files are virtually changed. Of course such change is provided in such a way not to change these packets. Since we need modeling fast lines traffic (appropriately l0 Gbs) and most current archives are not in this limit, we aced a type of indexing to reach a desired speed. It is evident that such traffic is not real. For regulating such unit, parameters shown in table 1 are adaptable inconfiguration file. Parameter Comment MBPS The speed of the line connected to processor based on Mbit/s Traffic-file Entrance traffic in sanitize frame.

Table 1. *Parameters related to traffic and network speed*

Parameter	Comment
MBPS	The speed of the line connected to processor based on Mbit/s
Traffic-file	Entrance traffic in sanitize frame

Table 2. *Parameters related to packets memory management*

Parameter	Comment
PACKET-STORAGE-ADDRESS	Address of packets memory start in process memory
The size of packets memory (firstly empty cites) based on packet. Each of them is equal with maximum size for each packet (for an example 1600 byte when the layer two is Ethernet).	PACKET-STORAGE-SIZE

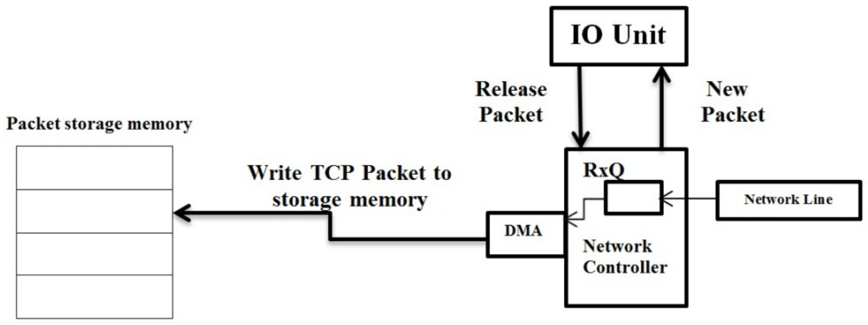
1.4 Network controller unit

According to Fig.1, another unit is connected to our processor working as the network controller unit. This unit reads IP packets produced by traffic producer and transfers to processor memory by using one of memory gates and then informs the processor. The communication type of the processor with this unit,the volume of the used memory, and its management can provide a direct effect on simulation result and their attention level .In this part, for the considered model in performed simulation, packets memory is considered as a part of processor memory. For simplification, the sizes of the cites which the network controller in the memory uses it for writing packets is considered as constant amount and the numbers of total packets in memory is limited. A table defines the occupied parts in network controller unit. This unit uses this table for suitable managing of packets memory. Figure 1 shows the performance of this unit. Such unit which is assumed as an entrance – exit tool by the processor, reads the produced IP packets, finds the suitable empty part based on the described table and puts the packet in this part by DMA using one of the processor memory gates .After complete transferring of the packet to memory, thenetwork controller, informs processor about the new packet arrival by IOU unit. When processor finished its job with received packets, when there isn't any need for its preservation in memory, it instructs network controller to free it and update packets memory. The size of packets memory in system and its parts number is dependent with processing speed. In practice, in most real products, bonding lists is used including property of parts in packets memory. In this method we try to define the empty site with little cost for new packet. In conclusion, performing simple algorithm in this simulator makes the simulation result similar with real condition although practically such method may not be used.

The method of using equal sized parts is similar with contiguous buffering method in Cisco (Franklin & Joshi, 2004; CISCO,1999). Although memory management is simple in this model, but the major problem of this unit is difficulty of some needed practices for creating another version of the packet .In this model, for creating a similar version with a little difference, it is necessary all packets be copied Opposite to this method is buffering particle method in Cisco (Franklin & Joshi, 2004; CISCO,1999) in which packets are stored in parts with smaller sizes and each one occupies one or several sites. Like other parts, such unit is described by configurable parameters.

In SNP simulator, MAC layer and its signaling process are not modeled and in this study we do not talk about problems and challenges of layer 2. Simulation on each packet is started when that packet isreceived by network controller and should be submitted to packets memory in processor. In fact, the goal of considering such unit in simulating is real time modeling in competition for memory (between processor and DMA) of real network processors. In other word, the memory related part is modeled from the network controller in this simulator

. **Fig.1.** *Network controller unit*



1.5 Important hints regarding the site and arrangement of packet in memory

The size of packets memory sites should be more than maximums packets size transferred to memory. For example, regarding Ethernet, packets size is between 44 to 1518 byte. By considering Ethernet header, packets component which is transferred to memory should be between 48 to15000 byte and then, assuming a constant size for houses, their size should at least be 1500 byte. Packets transfer should be performed fast and in fact, this part of packets should be processed by more preference regarding to body part and this is because of header importance in addressing and access to important data of packet. If with each time access with data memory (cache) we guarantee that all packet header is transferred into cache, we can achieve the goal. If TCP/IP is the packet under processing, the maximum size of TCP and IP header is equal with 40 byte (20 byte TCP and 20 byte IP if the packet includes both headers and is not fragmented). By assuming L740 byte for cache line and by assuring this packets are lined in memory in situation that their header is always in one cache line, the related goal is achievable. as a result the length of packets memory sites should be a multiplication of L (L multiplication). Fig.2 shows the data structure in packets memory, used in this simulation. For each site of the above memory 36 byte in this example, (24 byte at first and 12 byte finally) is totally unusable.

1.6 Simulations software models

Fig.2. *Packets arrangement in packets memory*

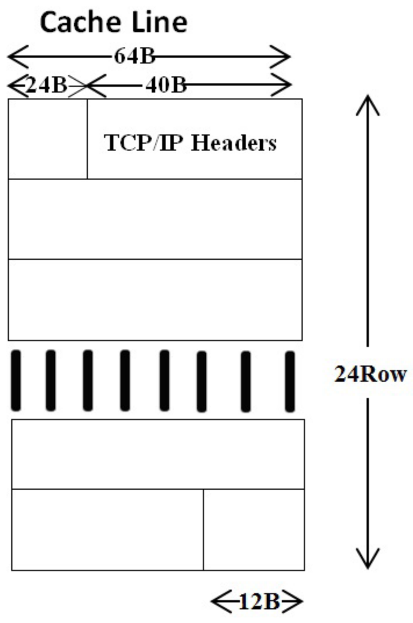
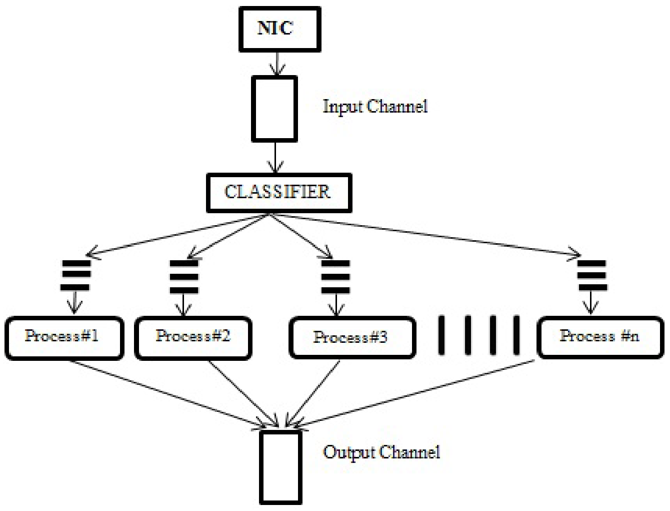


Fig.3. *Processing software model*



Our software model for simulation is shown in Fig.3.

According to Scheme 1, after transferring packets by network controller using classifier program, thesepackets are studied. This program performs some primary processing on packets and then classifies them. Packets are transferred to different programs called process based on performed classification on them. Each program includes a separated line and related packets address is cited in that line by classifier. Each process performing in a separated process takes packets and performs it's on processing on them. After finishing processing, transfers it for sending to exit canal. The pattern of completing packets processing in this simulator is the completing the related processing on them and their transfer on the line by processor is notexactly simulated and really the basic point is theperformed process on packets. This software model isused in most similar studies, especially in cases which MT processors are used as network processors (Spalink *et al*., 2001; Jianhua Huang, 2003; Noonan, 2004; Niemann *et al*., 2005; Wolf Tilman & Franklin mark, 2006). Semi codes shown in Scheme 1 shows the general performance of classifiers and processes.

Scheme1. *General performance of classifier and process*

```
Classifier()
Loop{
        Wait for Packet arrival
        Read PacketAddress
        Phrase PacketAddress
        qid = Classify (PacketAddress)
        Enque(qid.PacketAddress)}
        Process#i( )
Loop{
        De-queue queue#i into PacketAddress
        Process (PacketAddress)
        Send Packet at PacketAddress into output
        Channel
 }
```

1.7 Classifier program

Although different methods and algorithms are used for classifying packets but practically limited methods are used for software implementing of this unit in network processors (Burger & Austin,1997). For example in cases in which, IXP processor is used, often a type of internal hashing installed in this processor is used (Lekkas, 2004). Another example is application of an algorithm called DFC for classifying in two dimension (TOS , protocol) in SMT networkprocessors (Jianhua Huang, 2003). In some studies, the dominance of software method has been shown.The important property of this program includes low volume, high speed and high preference. Because this program is regarded as the common pathway for packets processing, its slowness may lead to packed drop and increasing total delay of packets.

1.8 Process programs

So far, there hasn't been a clear pattern for diagnosing the efficacy of network processors. Since each of these situations needs different processing on packets ,anotherpoint is to use such processors in different situations. For example, in core pathfinders, exactly after packets entrance, their destination should be defined and they should be sent to suitable gate. In this reason, the performed programs should be as small and fast as possible processed in minimum possible time. In this study, three different processing are studied (as IP-look,HTTP-parser and MD5).

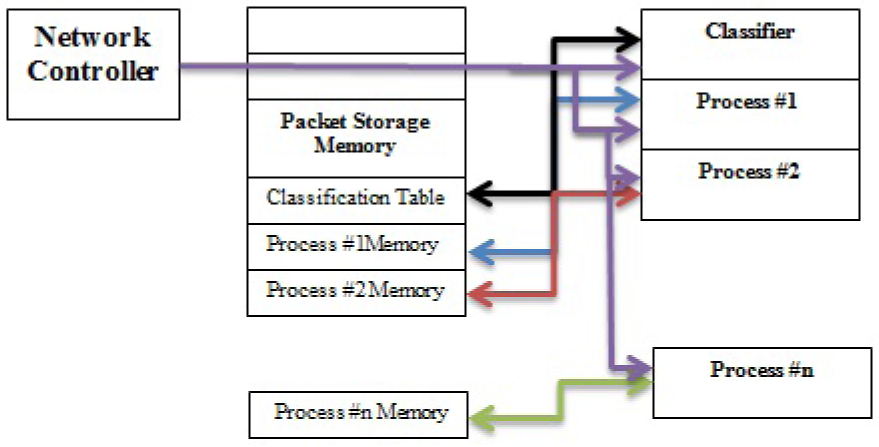
1.9 Memory classification

According to described software model, system memory is dividend among different sources. For example program (IP-look up) uses forwarding table to find packet destination gate and program (MD5) should store different parts of the packet in temporary memory. Fig.4 shows the general schema of memory division in this software model.

1.10 Patterns and parameters of simulation

In this study, two major patterns are considered for assessing system efficacy. One is "the number ofprocessed packets in second" based on MPPS or out put and another is "average time for packets processing in system or packet wait time". The first parameter is the sum of packets which are read in one second by network controller and transferred to packetmemory .packet is the time The second parameter for each between packet recall by network controller and finalizing its processing in related process. This is the sum of packet writing time in packets memory, its classification in classifier and its processing. Both parameters should be met in a suitable network environment. In addition to above two parameter, the number of filled sites of packets memory is another suitable parameter for simulation the forth parameter is the traditional pattern of processors as the number of demand orders in each clock on ICP. Other parameters used for SMT architecture include line speed, number of parallel threads, memory size and (cachle. log,icache,log) types and numbers of practical units and superscalar parameters.

Fig.4. *Memories in software model*



2. Conclusion

The simulator presented in this study is a process based "simulator for Network processor" (SNP) usable for different modes and applications .This simulator is based on "clock – by – clock" method and is described according to SMT Structure. The simulator architecture is performed by C++ language optimized as efficient as possible. One of the major differences of SNP with simple scalar simulator is related to its multi threads nature and another difference is that, in addition to simulating architecture and practice, SNP simulates the network environment, network traffic producer and the controller of network connected to processor, and performs most statistics needed in network processing. This simulator also produces high level languages compiler (such as C). Network controller unit in this simulator fetches IPpackets produced by traffic producer and transfers to processor memory by using one of memory gates and then informs the processor. Although memory management is simple in network controller unit connected to simulator, just the opposite to buffering particle method in Cisco ,the major problem of network controller unit is difficulty of some needed practices for creating another version of the packet .In this model, for creating a similar version with a little difference, it is necessary all packets be copied . In fact, the goal of considering such unit in simulating is real time modeling in competition for memory (between processor and DMA) of real network processors. In other word, the memory related part is modeled from the network controller in this simulator. In SNP simulator, MAC layer and its signaling process are not modeled and in this study we do not talk about problems and challenges of layer2.

Simulation on each packet is started when that packet is received by network controller and should be submitted to packets memory in processor .The size of packets memory sites is very important regarding the site and arrangement of packet in memory and they should be more than maximums packets sizetransferred to memory and packets transfer should also be performed fast and in real condition. The processing software model in this simulator which is used in most similar studies, especially in cases which SMT processors are used as network processors ,performs some primary processing on packets and then classifies them. Packets are transferred to different programs called process based on performedclassification on them. Each program includes aseparated line and related packets address is cited in that line by classifier. Each process performing in a separated process takes packets and performs its on processing on them. After finishing processing, transfers it for sending to exit canal.

In this study, two major patterns are considered for assessing system efficacy. One is "the number of processed packets in second" based on MPPS or out putand another is "average time for packets processing in system or packet wait time".

3. References

1• Burger D and Austin T (1997) The SimpleScalar Tool Set, v2.0. Technical Report UW-CS-97-1342, University of Winsconsin, Madison.

2• CISCO (1999) Router architecture session 601, cisco system ,Inc,

3• Franklin MA and Joshi V (2004) SimplePipe: a simulation tool for task allocation and design of processor pipelines with application to network processor. The IEEE Computer society 12th annual international symposium on modeling analysis and simulation of computer and telecommunications systems, Washington Univ., USA. pp: 59-66..

4• Gries M, Kulkarni C, Sauer C and Keutzer K (2003) Comparing analytical modeling with simulation for network processor. Design Automation and Test in Europe Conference and Exhibition, California Univ., Berkeley ,CA ,USA. pp: 256-261.

5• JianhuaHuang (2003) Network processor design, Proc. 5th Int. Conf. on ASIC2003, pp:26-33.

6• Jing Fu Hagsand (2005) Designing and evaluating network processor applications. Workshop on high performance switching and routing. Institute of Technol. Kista, Sweden. pp:142-146.

7• Lekkas P (2004) Network processor architectures, Protocol,platforms. McGraw-Hill.

8• Niemann JG, Porrmam M and Ruckert U (2005) A scalable parallel SoC architecture for network processor. IEEE Computer society annual symposium on VLSI, Heinz Nixdorf Inst. Paderborn Univ., Germany, pp: 311-313.

9• Noonan L (2004) Modeling a network processor using object oriented techniques. Euromicro Symposium on Digital System Design. Dept. of Electron. Eng. Limerrick Univ., Ireland. pp: 484-490.

10• Spalink T, Karlin S, Peterson L and Gottlieb Y (2001) Building a robust software-based router using network processors. Proc. of the 18th ACM Symposium on Operating Systems Principles (SOSP01), Chateau Lake Louise, Banff, Alberta,Canada. October. pp:216-229,

11• Wolf Tilman, Franklin mark (2006) Performance models for network processor design. IEEE Transact. on parallel and distributed systems.pp:548- 561.

12• Wolf Tilman, Franklinmark E. Ayguade, Valero M and Navaux P (2006) A simulator for SMT architectures: Evaluating instruction cache topologies. 12th Symposium on Computer Architecture and High Performance Computing. pp: 279- 286.