Improving the Effectiveness of Network Processor to dynamically handle Heterogeneous Traffic

Dr.Satheesh.A
School of Computer Engineering(SCOPE)
VIT University, Vellore, Tamil Nadu, India

Dr.Kumaravelu.R
School of Computer Engineering(SCOPE)
VIT University, Vellore, Tamil Nadu, India

Abstract: Network Processors (NPs) are programmable multi-processor devices that offer both the flexibility and speed required for the network packet processing applications. It is specially designed for the efficient data plane processing and control-plane processing in networking applications. It is also provided with the necessary computing resources to meet the speed requirements of the network protocols. As the traffic fluctuations are inherent in packet networks, the workload on each pipeline stage in NP may vary significantly over time. Current NP's are not designed to manage with the fluctuating workloads even though it possesses abundant resources with high computing power. This sets the objective to make NP to handle the non-uniform traffic mix so that improved effectiveness in processor utilization and energy conservation of NPs can be achieved. To improve the effectiveness of Intel IXP2400 NP, the paper focuses on developing a method availing Self-Configuration, Adaptive Load Sharing (ALS) and Dynamically Reconfigurable Queue. The self-configuring method enables network elements to readjust themselves in the event of a network. The task of self configuring network processor constitutes both control plane and data plane processing operations. At control plane in X-Scale processor, based on the network traffic, reconfiguring the network processor takes place through dynamic code deployment. To make decisions about the deployment of code, the traffic is monitored and its statistic is maintained. Moreover, active network concept is used for reconfiguring the system, wherein the binary code is transferred to the NP through the SOAP protocol. Thus, this paper demonstrates the integration of control plane and data processing operations to handle the non uniform traffic mix.

Keywords – Dynamic code deployment, Energy estimation, NePSim, Network Processor, Self-configuration

I. INTRODUCTION

Network Processors (NPs) are programmable multi-processor device that offer both flexibility and speed of the packet processing applications. It is particularly designed for the data plane and control-plane processing of network applications. It is also equipped with the necessary computing resources to meet the process speed requirements of the network protocols.

Compared with GPPs, NPs have much simpler arithmetic and cache units as their speed is achieved through parallel packet processing in multiple execution threads. In addition, common packet-handling functions, such as tree lookup, classification, metering, policing, checksum computation, interrupt and timer handling, bit manipulation, and packet scheduling [6], are frequently supported by coprocessors or specific functional hardware units. Moreover, NPs are often teamed with an embedded GPP for more complex tasks [5]. Hence this research paper is undertaken to integrate control plane operations (X-Scale processor) with data plane operations (Microengine) to handle non-uniform traffic mix.

The NPs architectures are classified into three categories, (i) Run-to-Completion architecture, (ii) Pipeline architecture, and (iii) Hybrid High level architecture. Hybrid high level architecture only supports the run-time reconfiguration concept. The network processor (Intel IXP2400) adopts the features of hybrid high level architecture. We adopted the IXP2400 NP for dynamic reconfiguration to handle non uniform traffic.

By and large, Network performances are measured by the following metrics: Availability, Throughput, Utilization, Error Rate and Delay. This research paper uses Availability, Utilization, Throughput, and power consumption as the metrics for measuring the effectiveness of Intel IXP2400 NP.

Presently, the high-capacity devices used in network systems consume excessive power. For example, a router contains a group of line cards: each line card contains one or two NPs. These routers consume more power (375W

per line card [10]); the operating temperature can reach as high as 70^0C [11]. However, for improving the performance, the NPs are increasing the number of MEs with higher frequencies. For example, the Intel IXP2400 NPs consume approximately 13-16W and 9-12W [13] at the frequencies of 600MHz and 400MHz. Similarly, in the Intel IXP family, the Intel IXP1200 and Intel IXP 2800 NPs consume power of 4.5W and 14W at the frequencies of 232 MHz and 600 MHz [15]. Hence, designing NPs with low power consumption is a key challenge to the designer [7]. In this connection, we focus attention towards energy conservation and reduction of heat.

The network resources are underutilized, when the traffic fluctuates between heavy or low. Emerging new types of applications add to the awe that had resulted in the need for high speed network devices, routers, and switches, etc. These devices are being built with Network Processor (NP) for packet processing at wire speed. Obviously the NP designers have faced many design challenges. As a result, this research paper focuses on the methods to improve the effective availability, utilization and throughput of the network process devices under non uniform traffic.

At present, the dynamic code deployment and energy conservation have been two major challenging factors for NP designers. Even though at present powerful network devices are available, the computing power of NPs are being underutilized in handling the non uniform traffic. This has prompted this research to take up the dynamic reconfiguration for IXP2400 NP to solve the above said challenges.

The remainder of this paper is organized as follows: Section 2 discusses motivating objectives. Section 3 presents the overview of dynamic reconfiguration procedures, adaptive load sharing technique, energy conservation methods and mathematical model outcomes. Section 4 and 5 described the implantation details, results and discussion. Finally, Section 6 summarizes and concludes this paper.

II. MOTIVATING OBJECTIVES

The research area of network processors focuses on NP architecture [17] [18] [19] [20], NP applications [21] [22] [23] [24], control plane and data plane [25] [26], NP software [9] [16], analytical models [27] [28], NP simulators [29]. Previous research studies have extensively and explicitly addressed the static allocation resources for packet processing and very limited focus on energy conservation. The literature survey gives an eye-opening to fix the scope of the research work to adopt dynamic reconfiguration at runtime with a non uniform traffic for effective resource utilization and lower energy conservation.

The main objectives of this paper are:

- To dynamically reconfigure the IXP2400 NP, based on traffic fluctuation.
- To “put in” or “put off” additional MEs to dynamically share the load according to increase or decrease in the traffic fluctuation.
- To optimize – in fact- to minimize the energy /power consumption by incorporating MEs according to low, moderate and heavy traffic of packets.
- To create an empirical model to confirm and validate the adaptive load sharing of IXP2400 NP.

III. FRAMEWORK FOR DYNAMICALLY RECONFIGURABLE NETWORK PROCESSOR

The Figure.1 depicts the overall structure of this paper. The research work of this paper is divided into two categories of operations: Control plane and Data plane. The majority of the researchers in this field has addressed the issues of data plane operations only and very few have been contributed in the control plane operations. This is because the NP architectures are more complicated and writing a program is also equally a challenging task. The present study focused both control plane and data plane operations.

The control plane is used for dynamic reconfiguration with X-Scale processor [1]. The Adaptive Load Sharing (ALS) module [4], the mathematical model for ALS [3] and energy conservation model [2] are done with data plane operations with IXP2400 NP. The functions used in both planes are described in the following sections.

A. Control and Data Plane operations

In the dynamic reconfiguration, the process: Monitoring, Decision Making and Code Deployment are developed and deployed in the X-scale core to perform the control plane operations. The energy conservation, Adaptive Load Sharing (ALS) and mathematical model for ALS are developed for the data plane operations. The functions of all the process are discussed in detail as follows.

B. Monitoring module

The monitoring module is used for monitoring the traffic variation and the queue length (number of packets waiting in the queue). It could continuously monitor the packet arrival and departure rate, and store the results in the temporary memory. These operations are performed by X-Scale processor.

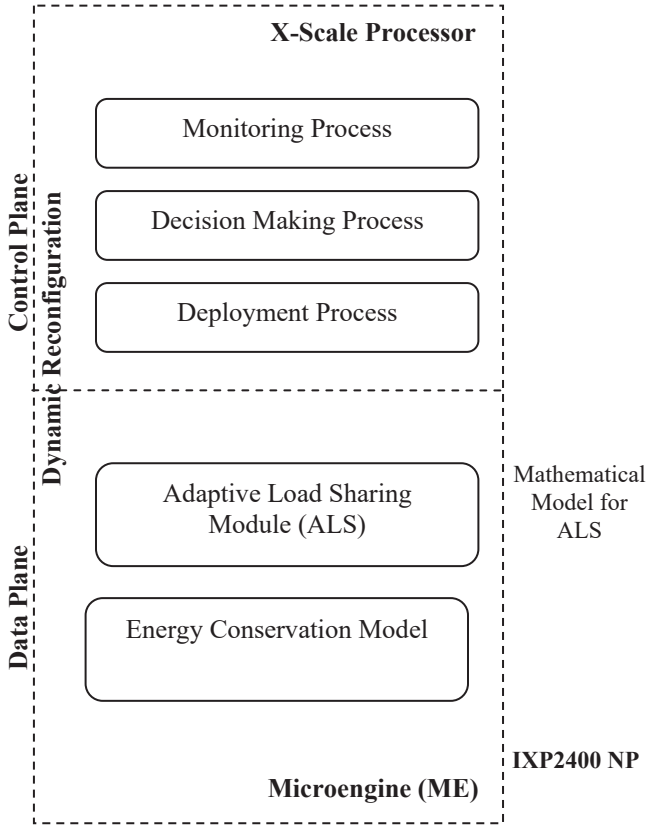

Figure.1 Framework for dynamically reconfigurable Network Processor

C. Decision making module

The decision making module is used to decide whether the code has to be dynamically deployed in the Microengine (ME) or the load is relieved from the current utilized ME. Based on the packet arrival and departure rate and the number of packets waiting in the intermediate queue, the decision making process will be switching on/off the MEs dynamically. This process manages the usage of MEs based on the queue length.

D. Deployment Module

The deployment module is used to deploy the code into MEs dynamically at run-time. The system has received two types of packets such as active and normal. The active packets are called as control packets generated by the system with the deployable code for ME. The normal packets are ordinary packets which can be processed by data plane itself. When the packets arrive in the system, it checks whether the packet is active or normal. When the active packet is received, it will be forwarded to X-Scale processor that to be parsed the contents. Then the code is stored in binary form, based on the instruction given by decision making module. Finally, the binary code will be deployed in ME.

E. Adaptive Load Sharing (ALS)

The NPs are using multiple processors for packet processing. The packets are scheduled among them according to a mapping computed at run-time. The goal of the adaption is to increase the resource utilization effectively and to reduce the system cost and energy consumption. ALS is all about sharing the load directed upon a single server across multiple redundant severs thereby reducing the per head load. Thus, the load sharing is made equitable and uniform, thereby utilizing the resources more effectively for achieving economical operation [4].

F. Energy Conservation

The energy conservation module is adopted for finding the energy consumption of IXP2400 NP at three different network traffics known as low, moderate and heavy. Finding the energy estimation at run-time is a challenging task in real time environment. Hence in this paper, NePSim2.0 open source simulator is used for energy estimation. It is an open source simulator and the literature survey concludes that, this is the only simulator available for energy estimation of IXP2400 NP. The research approach has used three applications namely IP forwarding (*ipfwdr*), network address translation (*nat*) and message-digest algorithm (*md4*) for finding the energy estimation. The throughput, service time and energy saving are calculated. The research study has achieved between 25-28% of energy saving in the heterogeneous traffic (1 hour) network system [2].

G. Mathematical Model

A mathematical model is developed for the dynamic adaptive load sharing module [3]. The queuing theory concept is adhered to. In the queuing model for ALS, the number of servers can be changed dynamically depending upon the queue length. This queuing model is so flexible that the number of servers can be increased or decreased depending upon the traffic congestion. The system load, reconfiguration time, service time and waiting time are calculated. One of the interesting findings is the usage of MEs are equal to the system load at a particular time. The average queue length and average waiting time are calculated and the average waiting time in the queue for a packet is 0.14*ms* and the reconfiguration time is 0.11*ms*.

IV. IMPLEMENTATION

The experimental setup contains the following hardware and software components for dynamic reconfiguration [Figure.2],

- ENP2611 board with Intel IXP2400 Network Processor
- Intel IXP2400 Developers Workbench
- Microcode Assembly language to write the code for microengines
- X-scale implementation – Embedded C programming language

The experimental setup containes a PC hosting the Radisys ENP-2611 board with Intel® 600MHZ network processor. The Intel IXP2400 Network Processor enable faster deployment of intelligent network services by providing high programming flexibility, code re-use, and high-performance processing. The IXP2400 has one channel of industry standard DDR DRAM running at frequency 150MHz, providing 19.2 Gbps peak DRAM bandwidth. The channel could address up to 2GB of DRAM. The DRAM is primarily used to buffer packets.

In addition to the DRAM, the IXP2400 also provide two channels of industry standard QDR SRAM running at frequency 200MHz, providing 12.8 Gbps peak bandwidth (6.4 Gbps read, 6.4 Gbps write). Up to 16MB of SRAM could be occupied on each channel. The SRAM is primarily used for forwarding table, next-hop table, packet descriptors, queue descriptors, counter and other data structures.

In ENP2611, two 4MB SRAM and one 512MB DRAM are build-in. The ENP2611 also includes 16 Mbytes of on-board Flash Memory for initialization and boot-up code, allowing the board to be self-provisioning at start-up. The ENP-2611 incorporates one PM3387 and one PM3386 Dual Gigabit Ethernet MAC controller to support up to three GE ports for fast-path data plane traffic. It also includes one Intel 82559 for 10/100 Mbps control plane traffic or debugging services.

Intel IXP2400 Developers Workbench is a simulation tool to execute the micro engine code and provide the performance measures for the developed application program. The developed application is ported onto the hardware. The execution of the code can be traced on an instruction-by-instruction basis using the simulation tool. There are provisions to watch the intermediate runtime values stored in various memory units like SDRAM, SRAM, Scratch Pad Memory, Local Memory and Micro Engine Registers. Intel's Software Development Kit (SDK) simulator does not support power model. Hence, this paper adopts the NePSim2.0 open source simulator for energy estimation.

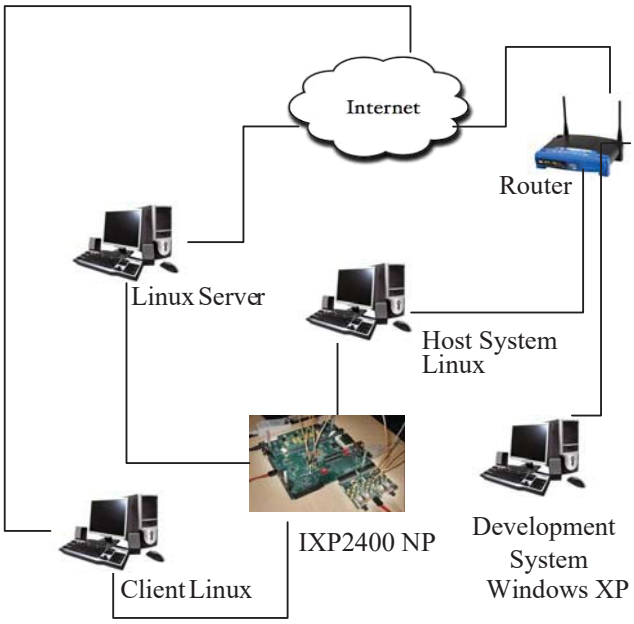

Figure.2 Skeleton of Experimental Environment

NePSim2 supports the IXP2400 network processor architecture [30]. NePSim2 contains source code written in C, Perl scripts and necessary *makefiles*. NePSim2 can be used to execute the benchmark applications on an IXP2400 network processor. In the research work, three benchmark applications: *ipfwdr*, *nat*, *md4* are used. In this continuation, the source codes of *Power.c* in NePSim2 have been modified to suite the need of energy estimation.

The real time network traffic trace is retrieved from [31] CAIDA (The Cooperative Association for Internet Data Analysis) laboratory: Bangalore-23-10-13 trace. The real traffic data (few second) is provided as an input to the simulator. The NePSim took more than an hour to simulate 1-s of real world trace. Three different arrival rates low, moderate and heavy are applied in this experiment. The packet arrival rate per second is less than 120, between 120 to 240 and more than 240 were designated as low, moderate and heavy traffic respectively. NePSim2 produce a large set of statistics data :sim_cycle, sim_time, sim_num_insn, num_insn_aborted, sim_pkts, sram_reads, sram_reads_words, etc.. The set of statistical data are used to calculate the throughput, service time and power consumptions.

Porting new benchmarks

The benchmark applications *ipfwdr*, *nat*, *md4* have been written in IXP microC that was compiled by Intel SDK . A *.list file is created for each microengine, to load the benchmark applications. Customize the respective Makefile and *.bin files is created. Finally, *.bin files is loaded to the corresponding microengines.

V. RESULTS AND DISCUSSION

This section deals with the result and discussion of the research study. The first outcome of this paper is dynamic reconfiguration.

In the dynamic reconfiguration, three MEs are allocated at the initial stage in the data plane for packet processing. It is assumed that the queue size is unlimited. The traffic traces taken from the real time network and it has been classified into low (less than 120 packets/second), moderate (121 to 240 packets/seconds) and heavy (more than 240 packets/seconds). The upper limit and lower limit value of packet arrival rates are fixed based on the network trace after analyzing the real time network traffic. The values are 120 and 240 for lower and upper limit for network traffic. The length of the queue is considered to be 50 % of the arrival rate (i.e., 60 for lower limit and 120 for upper limit value for the queue length).

Figure.3 shows that, upto 10 seconds the queue length is zero and the packet arrival rate is less than 120. Hence, the three MEs is sufficient enough to manage the traffic and packets are processed immediately and the queue is empty. When the traffic gets increased from low to medium (121 to 240) and queue length also crosses the lower limit (more than 60), the minimum resources allocated at the initial stage are unable to manage the workload. From the Fig.3, the queue length during 10-12 seconds is crossing the lower limit (more than 60) and packet arrival rate is

also more than the lower limit (120 to 240). In this stage, the system has to balance this situation by allocating one more additional resources (3+1 MEs). The monitoring module will monitor the packet rate from X-scale core and deploy the code at runtime in the REM (Runtime Environment Module) in order to tolerate the moderate workload. At 12-18 seconds, the queue length remain zero. The queue length during 19-22 seconds is increased more than the upper limit (120 and above) and packet arrival rate is also crossing the upper limit value (241 and above) (shown in Fig.3). In order to meet out the heavy traffic, the system deploys the code at runtime to allocate the maximum resources (i.e., 8 MEs) for packet processing.

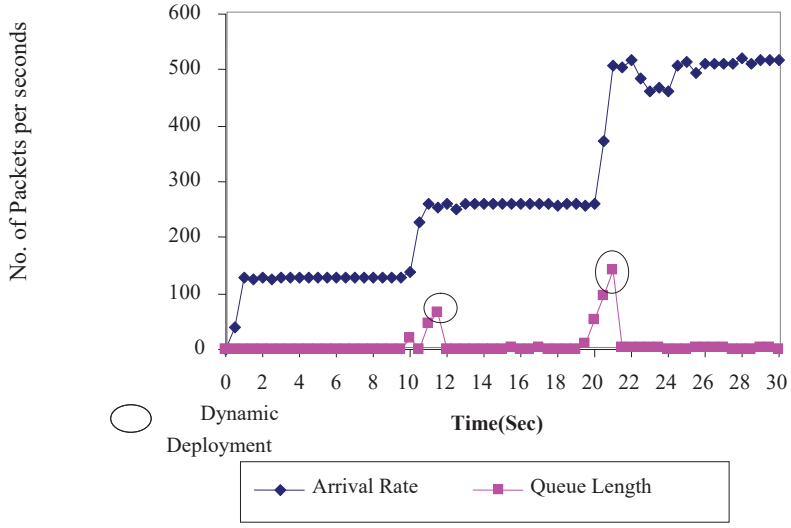

Figure.3.Arrival rate and lagging packet in Dynamic Reconfiguration

Moreover, the above result shows that the queue length of 60 and 120 are the system tolerates value for low and moderate traffic. Considering the value of K=60, then for every multiples of the K (ie., K, 2K, 3K and etc.) the additional MEs is deployed for packet processing at runtime. For the above said, a mathematical model [3] is derived and it is yet another outcome of this paper.

The important new findings of the mathematical model are:

i. The average number of active ME will be equal to traffic intensity (ρ). Here, traffic intensity (ρ) is 0.93 and the output of average number of active ME is 0.93.

ii. The maximum queue tolerant value is $k\rho$. Where k is the number of packets waiting in the queue or queue length and ρ is system load. The number of packets waiting in the queue is always less than the $k\rho$, hence the system avoids packet loss.

iii. The average number packets in the system are less than $k\rho$. Where k is the number of packets waiting in the queue or queue length. In the output, the average number of packets in the system is 7. This value is less than $k\rho$ ($k\rho$ = 16.74).

The third outcome of the paper is an energy estimation for dynamic reconfigurable IXP2400 NP. The research study explored the power savings for dynamic reconfiguration of IXP2400NP with three benchmark applications *ipfwdr*, *nat* and *md4* under different traffic mixtures of low, moderate and heavy regimens at frequencies of 400MHz (Figure.4) and 600MHz (Figure.5).

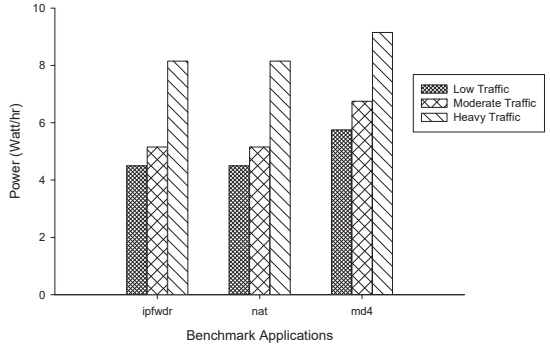

Figure. 4. Power consumption by different benchmarks in non-uniform traffic mixture (Core frequency is 400MHz)

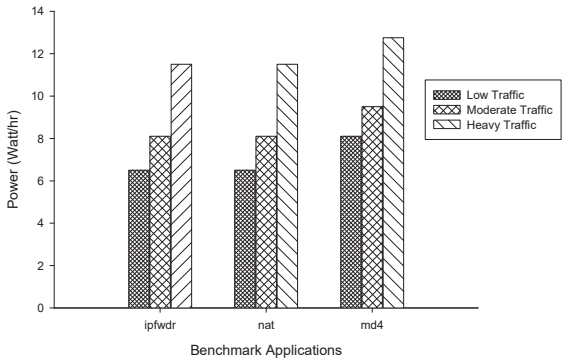

Figure5. Power consumption by different benchmarks in non-uniform traffic mixture (Core frequency is 600MHz)

The *ipfwdr* and *nat* are the header processing applications. The result shows that both the header processing application consumed approximately equal amount of power for packet processing. The *md4* is a cryptographic application found to be consuming more power than the header processing applications. This is because, the cryptographic applications are computationally more intensive and the control store and data path are used heavily.

Jia Yu et al [2004] proposed the work Assertion-Based Power/Performance Analysis of Network Processor Architectures. This work used IXP1200 NP with 6 MEs at a frequency of 232 MHz. It had applied in the low power technique, dynamic voltage scaling (DVS). The performance (Throughput) was dropped approximately by 2% to 3% for *md4* and *ipfwdr* benchmark. At the last, 2% to 3% of performance loss and only 10% of the dynamic power was saved for the benchmark applications except *nat*.

Yan Luo et al [2007] proposed to develop a low-power technique to reduce the activities of MEs in accordance with the varying traffic volume. This work also used IXP1200 NP with 6 MEs at a frequency of 232 MHz. At the lowest traffic load, upto only 30% of the power had been saved for *ipfwdr* and *nat*. The *md4* saved about 15%, respectively. At the high traffic load, power-reduction numbers were the lowest, but 6% - 17% of the total power saved for n*at*, *ipfwdr*, *md4*. In overall, 15- 27% of total power was saved without affecting the performance.

In this paper, the total power consumption of IXP2400 NP at 400MHz and 600MHz are approximately 4.5-9.0watts and 6.5-13 watts respectively. These results confirm to 50% of power saved in low traffic and 30-35% of power saved in the moderate traffic. In overall, 25- 28% of total power was saved without affecting the performance.

VI. CONCLUSION

The study extensively discusses about the dynamic reconfiguration of Network Processor. The present NPs are not proficient enough to handle the traffic fluctuation effectively under extreme conditions. This has lead to underutilization of resources, particularly during non-uniform traffic. Thus, a considerable amount of energy is also consumed by the resources unnecessarily. Hence, this paper has suggested the solution in the form of adaptive dynamic reconfiguration that can be implemented in IXP2400 NP with X-Scale processor operation.

The major contribution of this paper is in the control plane and data plane operations. The control plane is used for monitoring the traffic continuously, and the decision making module is used to decide whether the code has to be dynamically deployed or not in run-time. The X-Scale processor is used for the control plane operations. The deployment programmed and monitored values are stored in X-Scale processor for dynamic deployment. The active

networking concept is used for the dynamic reconfiguration. The code for deployment is carried by Simple Object Access Protocol (SOAP).

Adaptive load sharing (ALS) is an application for data plane operations. Energy estimation of ALS is performed in the data plane. The experiment is carried out in three types of traffic flow: *low*, *moderate* and *heavy*. The comparisons of the baseline configuration over dynamically reconfigurable NP as described in result showed the superior performance of dynamically reconfigurable NPs over baseline NPs. The result proved that the maximum utilization of resources whenever possible in the steady state.

The NePSim2.0 is an open source simulator used for energy estimation in the process of dynamic reconfigurable IXP2400 NP. The *nat*, *ipfwdr* and *md4* are the benchmark applications deployed in the NePSim. The throughput, service time and energy conservation are calculated. The NP has achieved 25-28% of energy savings with the implementation of dynamic reconfiguration.

The mathematical model for dynamic reconfigurable IXP2400 NP has developed. The proposed queuing system where the numbers of MEs utilization are dynamically adjusted based on the queue length. Kolmogorov differential equations were used to calculate the average waiting time, queue length and find the numbers of MEs need to be active. The simulation result shows a close match with a real-time experiment result, confirming the predictions of the research study.

REFERENCES

[1] Satheesh.A, Krishnaveni.S, and Ponkarthick.S "Self-configurable Environment for the Intel IXP2400 Network Processor" *International Journal of Computers and Applications*, Vol.31, No.4, pp.268-273, 2009.

[2] Satheesh.A, Kumar.D and Jayakumar.K "Energy Efficient Dynamic Adaptive Self- configurable Network Processor", *Journal of Theoretical and Applied Information Technology*, Vol. 63, No. 2, pp.477-485, 2014.

[3] Satheesh.A, Kumar.D, Dharmalingam.P and Lakshmipriya.TKS "Queuing System for Dynamically Reconfigurable Network Processor", *Far East Journal of Electronics and Communications*, Vol.13, No.1,pp.41-58 ,2014.

[4] Satheesh.A, Kumar.D, Vincent Jeyakumar.A "Run-Time Adaptive Processor Allocation of Self-Configurable Intel IXP2400 Network Processor" *International Journal of Computer Networks*, Vol.2, No.1, pp.16-33, 2010.

[5] Allen.J, B. Bass, C.Basso, R. Boivi, J.Calvignac, G.Davis, L.Frelechoux, M.Hedds, A. Herkersdorf, A.Kind, J.Logan, M.Peyravian, M.Rinaldi, R.Sabhikhi, M.Siegel, and M. Waldvogel, "IBM PowerNP Network processor: Hardware, software, and applications", *IBM Journal of Research and Development* , Volume (47), nos. 2/3 , pp.177-194,2003.

[6] Agere Systems Proprietary. The Challenge for Next Generation Network Processors. April 2001.

[7] Franklin.M and T. Wolf. Power considerations in network processor design. In *Workshop on Network Processors in conjunction with Ninth International Symposium on High Performance Computer Architecture (HPCA-9)*, pages 10–22, Feb. 2003.

[8] Andreas Kind, Roman Pletka, and Marcel Waldvogel. The role of network processors in active networks. *In Proceedings of IWAN 2003*, pages 18-29,Kyoto, Japan, December 2003.

[9] Andrew T. Campbell, Stephen Chou, Michael E. Kounavis, Vassilis D. Stachtos and John B. Vicente "NetBind: A Binding Tool for Constructing Data Paths in Network Processor-based Routers" *IEEE Fifth International Conference on Open Architectures and Network programming (OPENARCH' 02*), pp.91- 103,2002.

[10] Cisco. Cisco crs-1 carrier routing system 8-slot line card chassis system description 2012.Jia Yu; Wei Wu; Xi Chen; Hsieh, H.; Jun Yang; Balarin, F., "Assertion-based power/performance analysis of network processor architectures," *High-Level Design Validation and Test Workshop, 2004. Ninth IEEE*, pp.155-160, 10-12, Nov.2004.

[11] Kencl, L.; Le Boudec, J.-Y., "Adaptive load sharing for network processors," INFOCOM 2002.*Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings.* IEEE , vol.2, no., pp.545-554, 2002.

[12] "*IXP2400 Hardware Reference Manual*", June 2001, Intel Corporation.

[13] LakshmiPriya, T.K.S.; Hari Prasad, V.; Kannan, D.; Singaram, L.K.; Madhan, G.; Sundaram, R.M.; Prasad, R.M.; Parthasarathi, R.,"Evaluating the Network Processor Architecture for Application-Awareness," *Communication Systems Software and Middleware, 2007. COMSWARE 2007*. 2nd International Conference on, pp.1-9, 7- 12 Jan. 2007.

[14] Lee, K.; Coulson, G., "Supporting Runtime Reconfiguration on Network Processors," *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on* ,pp.721-726, 18- 20 April 2006.

[15] Moore.J, M. Hicks, and S. Nettles. Practical programmable packets. *In IEEE INFOCOM*, pp.41-50, Anchorage, Alaska, April 2001

[16] Ravi Kokku, Taylor L. Riché, Aaron Kunze, Jayaram Mudigonda, Jamie Jason, Harrick M. Vin, "A case for run-time adaptation in packet processing systems" *ACM SIGCOMM Computer Communication Review*, vol.34(1), pp.107-112, 2004.

[17] Raghunath, A; Kunze, A; Johnson, E.J.; Balakrishnan, V., "Framework for supporting multi-service edge packet processing on network processors," *Architecture for networking and communications systems*, 2005. ANCS 2005. Symposium on, pp.163-171, 26-28 Oct. 2005.

[18] Richard R. Weber "On the Optimal Assignment of Customers to Parallel Servers" *Journal of Applied Probability*, vol.15, pp.406-413, 1978.

[19] Ruf.L, R. Pletka, P. Erni, and P. Droz. Towards High-Performance Active Networking. *Proceedings of the Fifth Annual International Working Conference on Active Networks (IWAN 2003),* December 2003. Kyoto, Japan.

[20] Ruf.L, R. Keller, and B. Plattner. A Scalable High-performance Router Platform Supporting Dynamic Service Extensibility On Network and Host Processors. *In Proc. of 2004 ACS/IEEE Int. Conf. on Pervasive Services* (ICPS), Beirut, Lebanon. IEEE, Jul. 2004.

[21] Scott J. Harper "A secure adaptive network processor" Thesis 2003.

[22] Tsai.M, C. Kulkarni, C. Sauer, N. Shah, and K. Keutzer. A benchmarking methodology for network processors. *In 1st Workshop on network processors along with HPCA* 2002, February 2002.

[23] Troxel.I.A; George, AD.; Oral, S., "Design and analysis of a dynamically reconfigurable network processor," Local Computer Networks, 2002. *Proceedings. LCN 2002. 27th Annual IEEE Conference on,* pp.483-492, 6-8 Nov. 2002.
[24] Wolf, T.; Franklin, M.A, "Performance models for network processor design," *Parallel and Distributed Systems, IEEE Transactions* on , vol.17(6), pp.548-561,2006.
[25] Wolf, T.; Ning Weng, "Runtime Support for Multicore Packet Processing Systems," *Network, IEEE* , vol.2(4),pp.29,37,2007.
[26] Xinming Chen; Chasaki, D.; Wolf, T., "External monitoring of highly parallel network processors," *High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference* on , vol., no., pp.197-204, 8-11 July 2013.
[27] Xin Huang; Wolf, T., "A methodology for evaluating runtime support in network processors," Architecture for Networking and Communications systems, 2006. *ANCS 2006. ACM/IEEE Symposium on*, pp.113-122, 3-5 Dec. 2006.
[28] Yan Luo, Jia Yu, Jun Yang, Laxmi N. Bhuyan "Conserving network processor power consumption by exploiting traffic variability" *ACM Transactions on Architecture and Code Optimization*, Vol. 4(1),pp.1-25, 2007.
[29] Yan Luo, Jun Yang, Laxmi N. Bhuyan, Li Zhao, "NePSim: A Network Processor Simulator with a Power Evaluation Framework", *IEEE Micro*, vol.24, no. 5, pp. 34-44, September/October 2004.
[30] http://www.caida.org/data/