Parallel Processing Letters

ON DESIGN AND APPLICATION MAPPING OF A NETWORK-ON-CHIP(NOC) ARCHITECTURE

JUN HO BAHN, SEUNG EUN LEE, YOON SEOK YANG, JUNGSOOK YANG, and NADER BAGHERZADEH *
Department of Electrical Engineering and Computer Science
University of California, Irvine - Irvine, California 92697-2625, U.S.A.

Received August 2007
Revised February 2008
Communicated by C. R. Jesshope

ABSTRACT

As the number of integrated IP cores in the current System-on-Chips (SoCs) keeps increasing, communication requirements among cores can not be sufficiently satisfied using either traditional or multi-layer bus architectures because of their poor scalability and bandwidth limitation on a single bus. While new interconnection techniques have been explored to overcome such a limitation, the notion of utilizing Network-on-Chip (NoC) technologies for the future generation of high performance and low power chips for myriad of applications, in particular for wireless communication and multimedia processing, has been of great importance. In order for the NoC technologies to succeed, realistic specifications such as throughput, latency, moderate design complexity, programming model, and design tools are necessary requirements. For this purpose, we have covered some of the key and challenging design issues specific to the NoC architecture such as the router design, network interface (NI) issues, and complete system-level modeling. In this paper, we propose a multi-processor system platform adopting NoC techniques, called NePA (*Ne*twork-based *P*rocessor *A*rray). As a component of system platform, the fundamental NoC techniques including the router architecture and generic NI are defined and implemented adopting low power and clock efficient techniques. Using a high-level cycle-accurate simulation, various parameters relevant to its performance and its systematic modeling are extracted and analyzed. By combining various developed systematic models, we construct the tool chain to pursue hardware/software design tradeoffs necessary for better understanding of the NoC techniques. Finally utilizing implementation of parallel FFT algorithms on the homogeneous NePA, the feasibility and advantages of using NoC techniques are shown.

Keywords: chip-multiprocessor; parallel processing; network-on-chip (NoC); interconnection network; wormhole routing; adaptive router

1. Introduction

In order to meet the design requirements for computation-intensive applications and highly integrated low power solutions, the number of computing resources on a single-chip has rapidly increased. This is mainly because of the current trends in VLSI technology where billions of transistors are available to the designers. While

*E-mail : {jbahn, seunglee, yangys, jyang12, nader}@uci.edu

adding many computing resources such as CPUs, DSPs, and application specific IPs, to build a System-on-Chip (SoC), interconnection among resources becomes another challenging issue. For some applications SoC design must handle a mix of data streams such as video and wireless communication simultaneously. Although these streams may be different, they have to share common resources that are not optimized for the complete system integration of a SoC platform [1]. As a system platform for this purpose, currently either Multi Processor System-on-Chip (MPSoC) or Chip Multi Processor (CMP) are considered. MPSoC environment is the advanced SoC paradigm with multiple processing units. It consists of multiple homogeneous/heterogeneous processors such as CPU, DSP, IP blocks, and the associated communication capabilities. On the other hand, in a CMP environment, multiple programmable processors are located on a single chip. The CMP environment is similar in functionality to a small scale distributed computer systems.

As the semiconductor processing technology makes sub-micron level progress, there are several design limitations that need to be addressed. One of the critical issues is the wiring delay. While the speed of basic elements such as gates becomes much faster, the wiring delay is growing exponentially because of the increased capacitance caused by narrow channel width and increased crosstalk. According to [2], by 2018 the nominal logic gate delay of a high performance NAND gate is estimated to be less than 3 ps while the interconnect RC delay for a 1 mm copper intermediate wire is to be more than 3500 ps. That is, the interconnect delay is estimated to be 1000 times greater than the gate delay. Therefore, if this trend continues, the wiring delay will be one of the critical issues of the future VLSI designs.

In this paper, we propose the multi processor system-on-chip architecture named NePA (*Ne*tworked *P*rocessor *A*rray). The fundamental NoC techniques including the router architecture and generic NI are defined and implemented adopting low power and clock efficient techniques. A processing element based on OpenRISC is introduced for the homogeneous system. By using a high-level cycle-accurate simulation, various parameters relevant to its performance and its systematic model are extracted and analyzed. Finally, combining various developed systematic models, we construct the tool chain to pursue hardware/software design tradeoffs necessary for better understanding of the NoC techniques. The major contributions of this paper are: 1) a multi processor SoC platform, 2) design of fundamental components for NoC techniques, and 3) tool chain for NoC implementation.

The rest of this paper is organized as follows. Section 2 will introduce the backgrounds on NoC. Related works are addressed in Section 3. Section 4 will explain our platform in detail. Section 5 will provide some implementations on NePA as a benchmark. Finally conclusions are drawn in Section 6.

2. Backgrounds

In communication between a small number of cores for a System-on-Chip (SoC) environment, available options have been bus-based architectures or point-to-point communication methodologies. For simplicity and ease of use, the bus-based architectures are the most common methods. Busses have been manually designed

around either a specific set of features related with a narrow target need, or support for a specific processor. A bus-based architecture has fundamental limitations in terms of bandwidth and scalability. For instance, as the number of components attached to a bus increases, physical capacitance on the bus wires grows and as a result its wiring delay grows even further. Because of these serious limitations, designers have introduced advanced techniques such as: split and retry techniques, removal of tri-state buffers and multi-phase clocks, pipelining, and various methods of defining standard communication sockets. As a solution for overcoming some of these limitations, industry has introduced advanced bus architectures such as ARM^{TM} AMBA [3], OpenCore's WISHBONE System-on-Chip (SoC) interconnection [4], and IBM CoreConnect^{TM} [5]. Most of the advanced bus architectures adopt a hierarchical structure to obtain scalable communication throughput and partition communication domains into several groups of communication layers depending on bandwidth requirement such as AMBA high-performance bus (AHB) vs. AMBA peripheral bus (APB), and other relevant requirements. In many cases, these different bus architectures have required many changes in the bus implementation, but more importantly in the bus interfaces, with main impacts on IP reusability and new IP design. Bus architectures do not separate the activities generally divided as transaction, transport and physical layer operations. This is the key reason why they cannot adapt to changes in the system architecture or take advantage of the advanced silicon process technology.

Another approach to exceeding communication limitations and overcoming wiring delay problems for future technologies is to adopt a network-like interconnection which is called Network-on-Chip (NoC) architecture. By applying network-like communication which inserts routers in-between each communication object, the required wiring can be shortened. Therefore, the switch-based interconnection mechanism improves scalability and freedom from the limitation of complex wiring. Replacement of SoC busses by NoCs will follow the same path as data communication systems where from the economics point of view NoC can potentially reduce SoC manufacturing cost, time to market, time to volume, and design risk and at the same time improve performance. According to [1], the NoC approach has a clear advantage over traditional busses and most notably as far as the system throughput is concerned. And hierarchies of crossbars or multilayered busses have characteristics somewhere in between traditional busses and NoC. However, they still fall far short of the NoC with respect to performance and complexity.

Although the network technology in computer network is already well developed, it is almost impossible to apply those same techniques to a chip-level intercommunication environment without any modification or SoC optimization. For that reason, many researchers are trying to develop appropriate network architectures for on-chip communication. To be practical for an NoC architecture, the basic functionality should be simple and light-weight and the implemented components of an NoC architecture should be small enough to be economically feasible. On the other hand, for certain mobile applications it must meet the low power requirements as

well as performance and cost. In order to be low powered one has to consider many parameters such as clock rate, operating voltage, and power management schemes for the system design.

3. Related Works

In designing NoC systems, there are several issues that need to be considered such as: topologies, routing algorithms, performance, latency, and complexity. As topology variations, many researchers have proposed various interconnect architecture such as SPIN [6], CLICHE (Chip-Level Integration of Communicating Heterogeneous Elements) [7], 2D torus [8], OCTAGON [9], and a Butterfly Fat-Tree (BFT) [10]. As a feasible topology for NoC systems, the mesh is getting more popular for its modularity; it can be easily expandable by adding new nodes and links without any modification of the existing node structure. As the mesh nodes can be used as basic components in on-chip communication, they are potentially important components to accomplish a scalable communication model in NoC environment [11]. Another reason behind this popularity is the notion of being partitioned into smaller meshes, which is a desirable feature for parallel applications [12].

Another issue in NoC environment is the routing algorithm and the switching techniques. There are different types of switching techniques such as circuit switching, packet switching, and wormhole switching [13]. In terms of the way of choosing a path among the set of possible paths from source to destination, the routing algorithms are classified as deterministic/oblivious and adaptive ones [14]. The oblivious/deterministic routing algorithms choose a route without considering any information about the network's present condition, resulting in relatively simple design complexity. Adaptive routing algorithms use the state of the network like the status of a node or link, the status of buffers for network resources, or history of channel load information. Adaptive routing algorithms are refined as minimal or fully adaptive routing ones depending on the degree of adaptivity. Even though the adaptive routing algorithms utilize the flexibility in routing paths, the design complexity should be increased. DOR (dimension-ordered routing) [15], ROMM [16], and O1TURN [17] are examples of deterministic or oblivious routing algorithms. Some researchers have developed better performance routing algorithms using adaptive routing algorithms [18][19][20][21][22][23].

On the other hand, the adoption of virtual channel (VC) has been expanding because of its versatility. By adding virtual channels and proper utilization, deadlock-freedom can be easily accomplished. Network throughput can be increased by dividing the buffer storage associated with each network channel into several virtual channels [19], resulting in increase of channel utilization. By proper control of virtual channels, network flow control can be easily implemented [24]. Also to increase the fault tolerance in a network, the concept of virtual channel has been utilized [25][26]. In order to maximize its utilization, the method of allocating virtual channels is a critical issue in designing routing algorithms [27][28]. Another

optimization approach is the clock boosting mechanism that proposed in order to increase the throughput of an adaptive router [29].

Network interconnects implement interfaces such as AXI, OCP, and DTL to connect IP modules within an NoC. AMBA Extended Interface (AXI) [32] is the next generation, high performance on-chip interface technology developed by ARM to support ARM11 family-class processors. The configurable AXI interconnection components provide data-efficient, highly-optimized link from the processor and data bursting in ARM core-based NoC systems. Furthermore the AXI configurable interconnect supports a multi-layer topology that guarantees the necessary bandwidth and low latency for all connected IPs and it provides related ARM technologies, such as IEM for voltage and frequency scaling. The Open Core Protocol (OCP) [33] is a plug and play interface for a core having both master and slave interfaces. The OCP signals of the functional IP blocks are packetized by a second interface. All signals are synchronous, simplifying core implementation, integration and timing analysis. It defines a point-to-point interface between two communicating entities and each component acts as master and slave. The OCP integrates all inter-core communications, including dataflow and sideband control signals. The Device Transaction Level (DTL) [34] is one of standards for interconnection researched by Philips Semiconductors to interface IPs existing on an SoC. The DTL allows easy extension to other future interconnection standards.

4. NePA Architecture

4.1. System Platform

Our approach is to construct a scalable, flexible, and reconfigurable multi-processor platform which meets the high-performance and low-power requirements. As a result, we designed NePA (*Ne*twork-based *P*rocessor *A*rray) system platform which is a mesh based multi-processor SoC as shown in Figure 1. This reconfigurable multi-processor platform includes multiple programmable processors, memory modules, and several specific IPs which are required as part of the specification. By virtue of scalability of NoC, the number of connected processors or IPs is not fixed in this platform. A heterogeneous and scalable multi-processor architecture allows parallel processing for several applications in a multi-processor system. The multi-processor system consists of a large number of homogeneous or heterogeneous processing elements executing multiple tasks concurrently. Current multi-processor SoC architectures designed to run a relatively small number of parallel applications has a significant limitation on the number of cores used because of low scalability and efficiency. Communication requirements for the conventional SoC system made of numerous cores will not be possible using a single or a multi-layer of busses due to their poor scalability and bandwidth limitation between the cores comprising the architecture. Future multi-processor architectures will be composed of hundreds of multi-processors by means of various interconnection methodologies, low-power,

and high performance processing element.

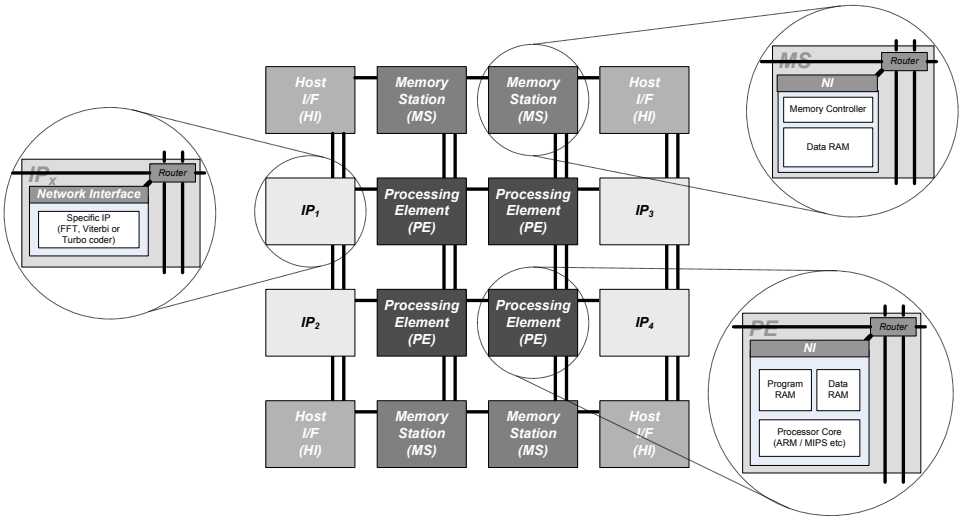


Figure 1: Reconfigurable NePA platform adopting the NoC techniques

Since one of the main goals of this research is to promote the performance of embedded computing systems by incorporating an NoC architecture. The research is concerned with the trends and characteristic behavior of on-chip interconnection networks and how NoC can be appropriately targeted for SoC designs. This strategy will result in a better understandings of the design and implementation of on-chip interconnection network that have the potential to improve the performance of modern chip-multi processors.

The outcome of this research will not only be a set of new architectural ideas and realization methodologies, but also a set of tools, strategies, and guidelines to design future chip-multi processor and their tool chain. The basic module of interconnection network in NoC is composed of router and network interface. In this section, we will investigate the high performance router architecture, general network interface for homogeneous/heterogeneous chip-multiprocessor, and low power architecture. Besides, we will provide the associated tool chain for the NoC generation and verification.

4.2. High-Performance Router Architecture

The design of a high performance router for on-chip interconnection has tight resource constraints such as router's area, power, and speed. Adaptive routing has been proposed as a method of improving network utilization by using information about the network state to select a path among alternative paths to deliver a packet, potentially reducing network latency [14]. While a good adaptive routing algorithm can balance network occupancy and enhance its maximum throughput, it also suffers

from the design cost in terms of additional sophisticated logic and performance degradation due to the routing decision time.

Wormhole flow control has increasingly been advocated as a means of reducing latency by routing a packet as soon as its head flit arrives at a node. The routing of a head flit enables switches to establish the path and body flits are simply forwarded along the path in a pipelined fashion. Wormhole flow control has some disadvantages. A major problem is that a router stops a packet when its head flit is blocked. When the link requested by a head flit is busy, the head flit could not advance and the remainder of the flits are also stopped holding the buffers and channels along the path that have already formed leading to significant link congestion. One of the most serious disadvantages of an adaptive wormhole router is the performance degradation due to routing decision time because routing flexibility requires additional resources.

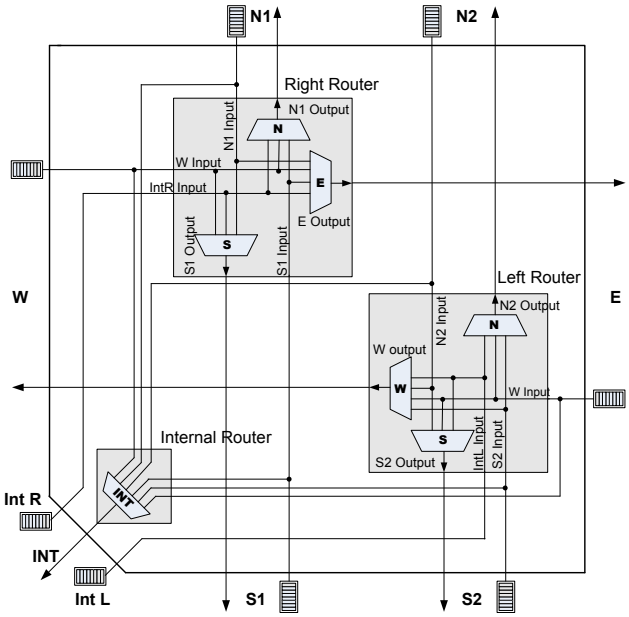
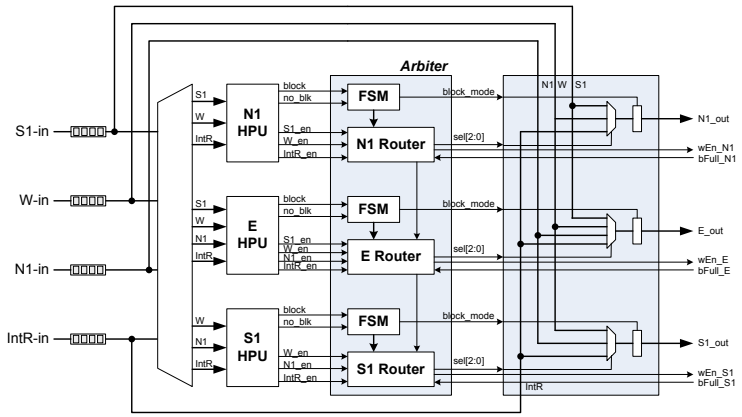


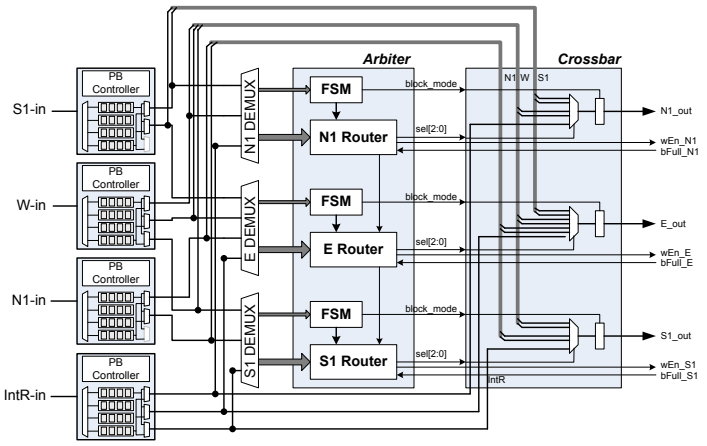
Figure 2: Adaptive router architecture

Our goal is not only to design a fast fully adaptive router which will provide more throughput than current wormhole routers, but also to provide flexibility in choosing an optimal router architecture depending on the applications. As a baseline router architecture, we developed the near-optimal router architecture as shown in Figure 2, which adopts a minimal adaptive routing algorithm and is deadlock-/livelock-free, and its performance has been already verified with simulation with standardized traffic patterns [35][36]. Also by prototyping the developed router architecture in RTL, the hardware complexity has been evaluated.

On the other hand, in order to enhance the maximum throughput and link utilization, a virtual channel (VC) approach was considered. In adopting VCs as a buffer model, their requirement of large number of FIFOs and additional delay caused by deep pipelining associated with complicated control mechanism of VCs such as VC allocation and switch arbitration, discourage the use of VCs regardless of versatility of VC approach. By optimizing the number of virtual channels (VCs) and developing a simple and independent management scheme in architectural level, we developed a low-latency adaptive VC router. In addition to VCs, by applying a dynamic priority scheme in managing routing paths, the link utilization was enhanced.



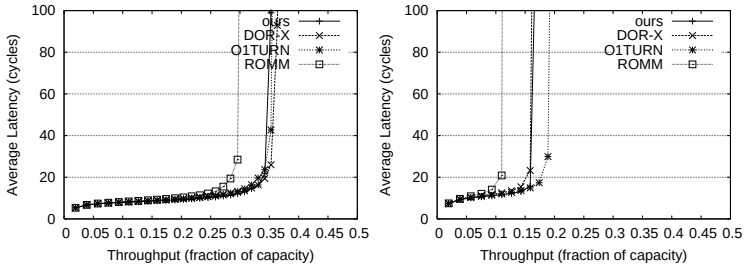
(a) Baseline router



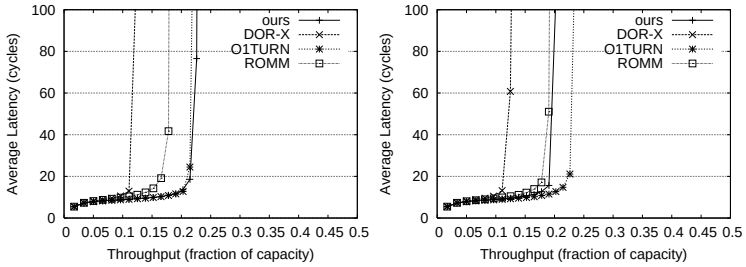
(b) VC router

Figure 3: Block diagrams of sub-routers of baseline and VC routers

Figure 3 shows the block diagram of sub-router which is either right or left router

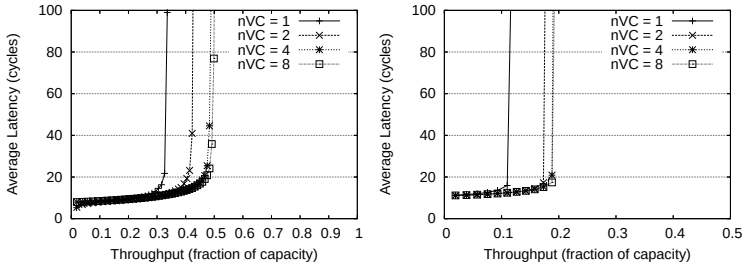


(a) Uniform random traffic (b) Bit-complement traffic

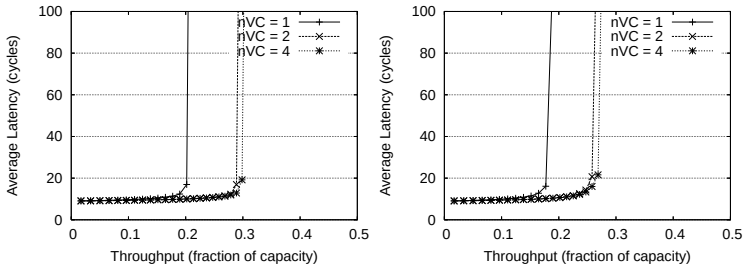


(c) Matrix-transpose traffic (d) Bit-reverse traffic

Figure 4: Baseline router performance in 8×8 mesh



(a) Uniform random traffic (b) Bit-complement traffic



(c) Matrix-transpose traffic (d) Bit-reverse traffic

Figure 5: VC enhanced router performance in 8×8 mesh

in Figure 2. Figure 3(a) illustrates the baseline router which is based on FIFOs while Figure 3(b) represents the VC enhanced router. For each router, the overal performance for different traffic patterns is shown in Figures 4 and 5. As shown in Figure 4, the baseline router shows competitive performance with comparison to other routing algorithms. Furthermore, in VC enhanced low-latency router, the overall performance is improved with respect to latency and maximum throughput as shown in Figure 5.

Our aim is not only to improve the performance of on-chip interconnection network in terms of latency, throughput, and power consumption, but also to provide flexibility depending on the required bandwidth of target applications. Thus, the exploration capability of several developed router architecture is added to our tool chain to find out the optimal NoC architecture for the different application requirements. Effective parameters for various router configurations are kind of routing algorithm, type of buffer between FIFOs or VCs, number of FIFOs or VCs, and different VC allocation.

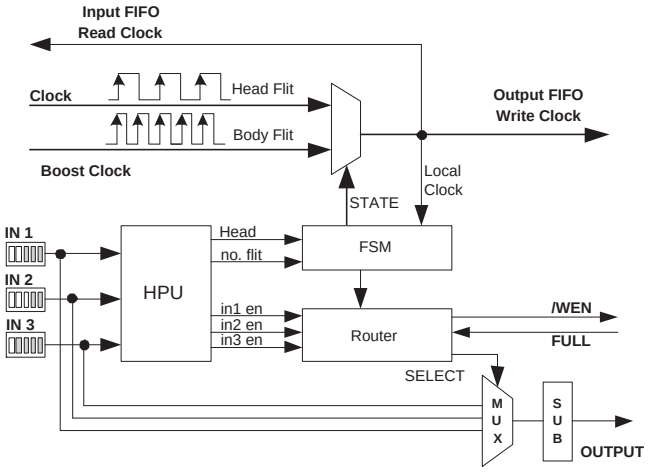
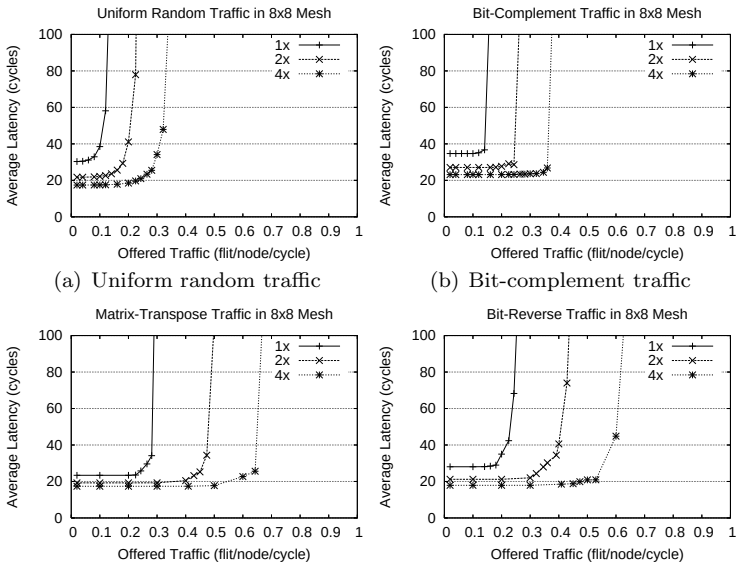


Figure 6: Conceptual block diagram of clock boosting router

We also investigate the router architecture which enhances the channel utilization by using multi-clock domains. As an initial step, we developed a clock boosting mechanism which uses different clocks in a head flit and body flits [29]. We expect that these techniques reduce the latency and greatly increase the throughput independent of routing algorithms. According to [29], the clock boosting mechanism is motivated by the fact that routing decision time for the head flit is the critical path of an adaptive router, determining the operating frequency of overall router while body flits just advance along the reserved path that is already established by the head flit. Therefore, while a relatively slower clock is applied to handle the head flit, a faster clock is used to deliver the rest of body flits as shown in Figure 6. Then it



(a) Uniform random traffic (b) Bit-complement traffic

(c) Matrix-transpose traffic (d) Bit-reverse traffic

Figure 7: Clock boosting router performance in 8×8 mesh

reduces the probability of congestion as well as latency. As a result, it increases the maximum throughput as shown in Figure 7 where the ratio of increasing maximum throughput is almost proportional to the ratio of boosting clock to the orignal one.

4.3. Generic Network Interface

The network interface is an essential part of a generic NoC design where XML machine-readable specification can be used to interconnect IPs. It provides a more flexible and programmable interface for NoC systems. Also it not only enables integration into EDA tools but also specifies the configuration of the network interface. To increase the performance and reliability of the NoC, development of an efficient on-chip network is essential.

Currently we have developed a simple NI module, one side of which is directly connected to the router and the other side is connected using a general bus interface as shown in Figure 8. In order to accelerate the speed of data transfer between router and the connected PE, most of NI operations are initiated by a minimal set of control registers and the rest of data transfer between router and PE is done automatically like conventional DMA controllers.

4.4. Processing Element

As an MPSoC system platform, each of the processing elements (PEs) can perform kernel-level operation in parallel depending on the applications. In our re-

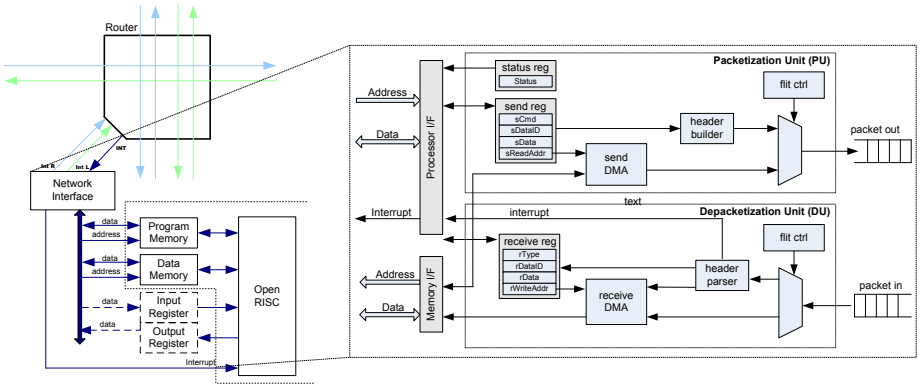


Figure 8: Block diagram of network interface between router and PE

search, we adopted an OpenRISC core as a homogeneous PE while one could use any embedded CPU or DSP. Original OpenRISC is a general RISC processor which includes I/D-caches, MMU, power management unit and debugging unit. After extensive refinement of this processes, we used the modified architecture in our design.

4.5. Tool Chain

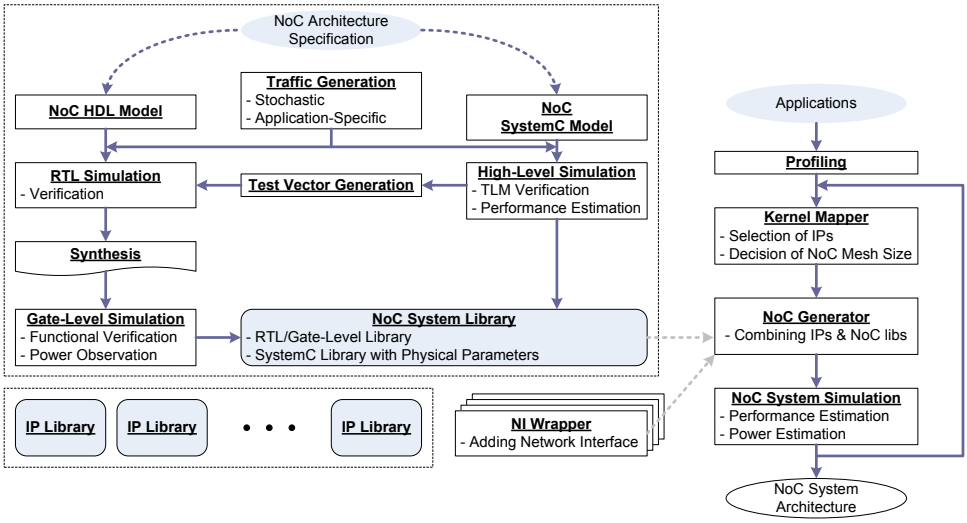


Figure 9: NoC design flow as a tool chain.

Figure 9 illustrates a prospective NoC design flow as a tool chain. The right side of Figure 9 represents the basic design flow based on the given design libraries such

as several IPs, an NoC system library and other relevant sources. Different from SoC environments, MPSoC design flow explores the parallelism in a given target application. For this purpose, the profiling is used as the first step to estimate the required performance using all the parallel processing resources available for the target architecture. Based on the profiling information for a given set of application kernels, the choice of IPs utilized and the mesh dimensions are evaluated. Once the MPSoC IP components and the mesh dimensions are selected, the high-level NoC system simulation, the validity of performance margin and its power budget is evaluated. Depending on the results of the NoC system simulation, the procedure from kernel mapping to NoC system simulation is repeated. When the expected requirements of performance and power are met, the final description of the NoC system architecture is generated.

As part of this effort, we developed NoC specific system libraries. From the NoC architecture specification, both SystemC model and HDL description have already been completed. Specifically, a cycle accurate SystemC NoC simulator with various router specifications such as different type of routing algorithms, different types of buffer models like FIFOs and VCs, and their configurations, has been developed. In order to evaluate the performance of the target NoC architecture, several traffic pattern generators using stochastic models have been modeled in SystemC and tested. Although stochastic models are well defined and widely used for evaluation purposes, there are limitations regarding real traffic patterns, therefore a way of application-specific traffic generation is needed. From the SystemC model and its simulation with the traffic generation block, timing-accurate test vectors are generated for the RTL/gate-level simulation and verification. Throughout the synthesis and gate-level simulation, the necessary parameters are extracted and applied to build up the NoC system library.

5. Benchmark

In order to evaluate the NePA system, parallel FFT algorithms are used as benchmark. Many mapping scenarios of small point FFT have been introduced for different kinds of NoC topologies such as star [30], mesh, fat-tree and torus NoC [31]. For the NePA system, which is a $2D$ mesh NoC, we implemented 3 different parallel FFT algorithms and tested on our cycle-accurate SystemC simulator varying the input data size from 32 to $32,768$. The components of the simulator are various number of compact OpenRISC processing elements and the associated routers and shared communication channels which connect the routers. The compact OpenRISC processing element has local program/data memory which is directly connected with OpenRISC core for fast access. The C program for each OpenRISC core was automatically generated from the kernel C which is described based on the proposed methods. Finally, the binary code for each PE was compiled using OpenRISC toolchains. The router model used in this system is the baseline router. The overall operating clock frequency is assumed to be 100 MHz. The proposed parallel FFT

Table 1: Cycle counts for parallel FFT algorithms

# of FFT	single	2×2			4×4			8×8		
		pFFTv1	pFFTv2	pFFTv3	pFFTv1	pFFTv2	pFFTv3	pFFTv1	pFFTv2	pFFTv3
32	3149	2629	2802	2267						
64	7073	3989	3846	3311	3762	4609	2680			
128	16269	7120	6250	5715	4793	5289	3360			
256	37424	14364	12527	11939	6917	6717	4788	5503	7265	4827
512	85237	29546	24856	24322	11528	10230	8760	7114	8138	5702
1024	191885	62774	52379	51885	22427	18293	16718	10050	10357	7726
2048	427190	134761	113186	112544	44993	35119	33560	16780	14977	12273
4096	941762	289806	245595	244903	91498	69591	68057	31882	25600	22944
8192	2058984	621878	532609	531676	190101	144607	142613	62630	47456	44802
16384	4469507	1329997	1150615	1149152	398403	305511	302882	128194	94995	92147
32768	9380584	2834265	2474675	2472211	836961	649386	645440	258356	188763	172303

algorithms are implemented and run by varying system parameters such as the number of PEs and the size of FFT. Data and the twiddle factors are represented in 16-bit real and 16-bit imaginary in 2's complement format. 14 bits are used for the fractions and scaling is performed to avoid overflow. The input data are assumed to be fed in packet format from the external host which is connected to PE[0] which is located at the upper left corner in 4×4 mesh. Therefore, the corresponding input data for each PE are delivered throughout network.

The three different parallel FFT algorithms differentiated by the degree of balanced computation and communication, were tested and the execution time for completing all stages was measured. The result of the simulation is shown in Table 1 where pFFTv1, pFFTv2, and pFFTv3 represent the reference parallel algorithm method I, II, and III, respectively. Method I is the same as [37], and the other two were our own optimized ones. Method II is the version that has well-balanced kernels with regular communication pattern while the method III is the optimized version with further reduced communication overhead than method II.

The simulation shows that parallel FFTs require fewer clock cycles for computing transform on the same size of data as the number of processing elements increases. Also, the proposed algorithms such as methods II and III outperform the reference parallel FFT algorithm, method I. After running the simulation with different system parameters, the performance gain of method II over the reference parallel FFT method I is greater than 24% in 8×8 PE model when the data size is larger than 2^{13}. The parallel FFT method III shows 33% of performance gain when the data size is 2^{15} and the average performance gain is 25%. As a result, the proposed algorithms are effective in utilizing the parallel processing capability of the architecture and achieve scalable performance.

When the performance results are compared with the cycle counts of FFT implemented on TI C62x and C67x architecture [38][39] where the formula of cycle counts are $(2N+7)\log_2 N + 34 + N/4$, and $2N\log_2 N + 42$, respectively, our proposed algorithms spend less clock cycles. According to benchFFT [40], the results can be normalized by MFLOPS which is a scaled version of the speed, defined by:

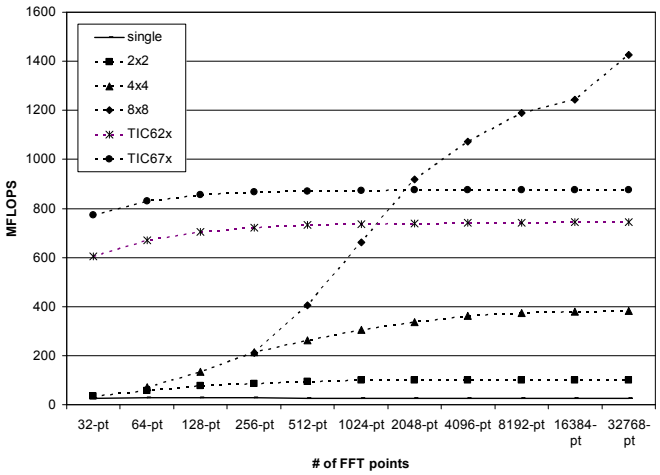


Figure 10: benchFFT comparison in speed

$$MFLOPS = \frac{5N \log_2 N}{time\ for\ one\ FFT\ in\ \mu sec.} \tag{1}$$

where N is the total number of FFT points. Figure 10 shows the normalized performance graph of method III with TI DSP's based on benchFFT. In Figure 10, TIC62x and TIC67x are assumed to run at 300 MHz and 350 MHz, respectively. Therefore, even though the operating clock frequency is much lower than TI DSP's, i.e. 100 MHz, our proposed algorithms show better performance starting with 2048-point FFT in 8×8 mesh. If the operating clock frequency is set to be as TI's, then overall performance of the proposed methods in 4×4 and 8×8 mesh outperform.

6. Conclusions

In this paper, we have shown that the results of this research provide a new avenue for the design of next generation highly integrated embedded systems. The notion of NoC as a replacement for bus central design is another key contribution of this work. The tool chain and the modeling effort discussed here for the target platform is one of the principle results for design exploration of future systems. By applying the designed system platform to practical embedded system design, an easy integration of different IPs or processors is possible. Also our application mapping approach is a conceptual basis for automatic kernel mapping, scheduling, and parallel programming models.

[1] *A Comparison of Network-on-Chip and Buses*, http://www.arteris.com/noc_whitepaper.pdf, 2005
[2] *International Technology Roadmap for Semiconductors 2004 Update*, ITRS, 2004.
[3] *AMBA Specification Rev. 2.0*, http://www.arm.com, 1999.

[4] *Specification for the: WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores*, OpenCore, 2002.

[5] *The CoreConnect Bus Architecture*, http://www-03.ibm.com/chips/products/coreconnect/, 1999.

[6] P. Guerrier and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections," in *Proc. Design and Test in Europe (DATE)*, pp. 250–256, Mar. 2000.

[7] S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä and A. Hemani, "A Network on Chip Architecture and Design Methodology," in *Proc. Int'l Symp. VLSI (ISVLSI)*, pp. 117–124, 2002.

[8] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," in *Proc. Design Automation Conf. (DAC)*, pp. 683–689, 2001.

[9] F. Karim, A. Nguyen and S. Dey, "An Interconnect Architecture for Networking Systems on Chips," *IEEE Micro*, vol. 22, no. 5, pp. 36–45, Sept./Oct. 2002.

[10] P. P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "Design of a Switch for Network on Chip Applications," in *Proc. Int'l Symp. Circuits and Systems (ISCAS)*, vol. 5, pp. 217–220, May 2003.

[11] J. Duato, S. yalamanchili and L. M. Ni, *Interconnection Networks: An Engineering Approach*(IEEE Computer Society Press, 2003).

[12] H. H. Najaf-abadi, H. Sarbazi-azad and P. Rajabzadeh, "Performance Modeling of Fully Adaptive Wormhole Routing in 2-D Mesh-Connected Multiprocessors," in *Proc. Int'l Symp. Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS)*, pp. 528–534, Oct. 2004.

[13] P. P. Pande, C. Frecu, M. Jones, A. Ivanov, and R. Saleh, "Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures," *IEEE Trans. Computers*, vol. 54, no, 8, pp. 1025–1040, Aug. 2005.

[14] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*, (Morgan Kaufmann Publishers, San Francisco, CA, 2004).

[15] H. Sullivan and T. R. Bashkow, "A Large Scale, Homogeneous, Fully Distributed Parallel Machine," in *Proc. Symp. Computer Architecture*, pp. 105–117, ACM Press, 1977.

[16] T. Nesson and S. L. Johnsson, "ROMM Routing on Mesh and Torus Networks," in *Proc. ACM Symp. Parallel Algorithms and Architectures*, pp. 275–287, ACM Press, 1995.

[17] D. Seo, A. Ali, W. Lim, N. Rafique and M. Thottethodi, "Near-Optimal Worst-case Throughput Routing for Two-Dimensional Mesh Networks," in *Proc. Int'l Symp. Computer Architecture (ISCA)*, pp. 432–443, June 2005.

[18] J. Hu and R. Marculescu, "DyAD - Smart Routing for Network-on-Chip," in *Proc. Design and Automation*, pp. 260–263, ACM Press, 2004.

[19] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. Computer*, vol. C-36, no. 5, pp. 547–553, May 1987.

[20] J. Duato, "A New Theory of Deadlock-free Adaptive Routing in Wormhole Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 12, pp. 1320–1331, Dec. 1993.

[21] G. Chiu, "The Odd-Even Turn Model for Adaptive Routing," *IEEE Trans. Parallel and Distributed Systems*, vol. 11, no. 7, pp. 729–738, Jul. 2000.

[22] C. J. Glass and L. M. Ni, "The Turn Model for Adaptive Routing," *Journal of ACM*, vol. 31, no. 5, pp. 874–902, Sep. 1994.

[23] C. J. Glass and L. M. Ni, "Maximally Fully Adaptive Routing in 2D Meshes," in *Proc.*

Int'l Conf. Parallel Processing, I: 101–104, 1992.

[24] W. J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. Parallel and Distributed Systems*, 3(2): 194–205, Mar. 1992.

[25] R. V. Boppana and S. Chalasani, "Fault-Tolerant Wormhole Routing Algorithms for Mesh Networks," *IEEE Trans. Computers*, vol. 44, no. 7, Jul. 1995.

[26] J. Zhou and F. C. M. Lau, "Adaptive Fault-Tolerant Wormhole Routing with Two Virtual Channels in 2D Meshes," in *Proc. Int'l Symp. Parallel Architectures, Algorithms and Networks (ISPAN)*, pp. 142–148, May 2004.

[27] A. S. Vaidya, A. Sivasubramaniam and C. R. Das, "Impact of Virtual Channels and Adaptive Routing on Application Performance," *IEEE Trans. Parallel and Distributed Systems*, vol. 12, no. 2, pp. 223–237, Feb. 2001.

[28] M. Rezazad and H. Sarbazi-azad, "The Effect of Virtual Channel Organization on the Performance of Interconnection Networks," in *Proc. Int'l Parallel and Distributed Processing Symposium (IPDPS)*, Apr. 2005.

[29] S. E. Lee, N. Bagherzadeh, "Increasing the Throughput of an Adaptive Router in Network-on-Chip(NoC), " in *Proc. of Third Int'l Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pp. 82–87, Oct. 2006.

[30] D.Kim, M.Kim and Soberlman G.E, "Parallel FFT computation with a CDMA-based network-on-chip",*Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 1138-1141 Vol. 2, 23-26 May 2005.

[31] M. Kreutz, C. Marcon,L. Carro, N. Calazans and A.A. Susin, "Energy and latency evaluation of NoC topologies," *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 5866-5869 Vol. 6, 23-26 May 2005.

[32] *AMBA Advanced eXtensible Interface (AXI) Protocol Specifcation*, Version 1.0, ARM, 2004. http://www.arm.com.

[33] *OCP International Partnership*, Open Core Protocol Specification. 2.0 Release Candidate, 2003.

[34] Device Transaction Level (DTL) Protocol Specification, Version 2.2, Phillips Semiconductors, 2002.

[35] N. Tabrizi, N. Bagherzadeh, A. H. Kamalizad and H. Du, "MaRS: A Macro-pipelined Reconfigurable System," *ACM Computing Frontiers*, pp. 343–349, Italy, 2004.

[36] J. H. Bahn, S. E. Lee and N. Bagherzadeh, "On Design and Analysis of a Feasible Network-on-Chip (NoC) Architecture," in *Proc. ITNG 2007*, pp. 1033–1038, Las Vegas, 2007.

[37] C. Zhong, G. Han and M. Huang, "Some New Parallel Fast Fourier Transform Algorithm," in *Proc. of the Sixth Int'l Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT)*, pp. 624–628, 2005.

[38] *TMS320C62x DSPs C62x Core Benchmarks from Texas Instruments*, http://www.ti.com

[39] *TMS320C67x Floating Point DSPs C67x Core Benchmarks from Texas Instruments*, http://www.ti.com

[40] *FFT Benchmark Results*, http://www.fftw.org/speed