# Using the NS-2 Network Simulator for Evaluating Network on Chips (NoC)

Muhammad Ali[1], Michael Welzl[1], Awais Adnan[2], Farrukh Nadeem[1]

[1]*Institute of Computer Science, University of Innsbruck, Austria*
[2]*Institute of Management Sciences, Peshawar, Pakistan*

**{Muhammad.Ali, Michael.Welzl, Farrukh.Nadeem}@uibk.ac.at**

**Owais_Adnan@yahoo.com**

**Abstract:** Networks on chips (NoCs) have been introduced as a remedy for the growing problems of current interconnects in VLSI chips. Being a relatively new domain in research, simulation tools for NoCs are scarce. To fill the gap, we use network simulator NS-2 for simulating NoCs, especially at high level chip design. The huge library of network elements along with its flexibility to accommodate customized designs, NS-2 becomes a viable choice for NoCs. We have used NS-2 to simulate our prototype of a fault tolerant protocol for NoCs.

**Key words:** *Network on chips (NoCs), Network Simulator (ns-2), simulation and modeling*

## INTRODUCTION

Chip integration has reached a stage where a complete system can be placed on a single chip. The integration has been made possible because of the rapid developments in the field of VLSI design. These chips, commonly termed as System on a chip (SoC), are primarily used in embedded systems. While designing an SoC, a vendor may use a library of cores designed by external designers in addition to using cores from in-house libraries. Cores are basically predesigned models of complex functions termed as Intellectual Property Blocks (IP Blocks), Virtual Components (VC) or simply micros. One key issue in the SoC design is heterogeneity; components of various vendors with significantly distinct characteristics lie on the same chip, making the design process even more complex [14].

According to the International Technology Roadmap for Semiconductors (ITRS) before the end of this decade, chips will be designed with billions of transistors [07]. Such a development means that in near future we will have Application Specific Integrated Circuits (ASIC) that will comprise of hundreds of heterogeneous components integrated together to provide full functionality of an application.

However, development of such chips is not an easy task as the number of transistors increases on-chip, so does the complexity of integrating them. Studies have shown that buses --- *the current SoC interconnects* --- are unable to scale effectively beyond a certain number of communicating resources, hence, becoming a bottleneck toward the billion transistor chip [01]. To cope with the inefficiency of buses, VLSI researchers, having explored the areas of parallel computing and computer networks, came up with a novel packet-based interconnect architecture for future SoCs --- *Network on a chip (NoC)*. The idea is to connect different resources on a chip through a network where communication takes place using packets instead of connecting the resources via dedicated wires.

NS-2 is an open source, object-oriented and discrete event driven network simulator written in C++ and OTcl. Its a very common and widely used tool to simulate small and large area networks. Due to similarities between NoCs and networks, NS-2 has been a choice of many NoC researchers to simulate and observe the behavior of a NoC at a higher abstraction level of design. It has a huge variety of protocols and various topologies can be created with little effort. Moreover, customized protocols for NoCs can easily be incorporated into NS-2. The parameters for routers and links can easily be scaled down to reflect the real situation on a chip. Based on this fact, we have successfully simulated a hundred node 2D mesh based NoC using our reliable protocol for safe delivery of packets. As we will see in section 4 that we are not the only ones to use NS-2 for simulating a NoC.

The purpose of this paper is to show the network community the similarities that exist between general networks and NoCs and show how NS-2 is facilitating the NoC designers to realize new design paradigms for this novel communication architecture. Furthermore, we hope that this paper would motivate network researchers to make a valuable contribution toward NoCs, hence opening a new dimension of research.

The paper is organized as follows; section 1 gives a brief overview of NS-2 network simulator. Section 2 introduces a NoC along with its similarities and differences with computer networks. Section 3 discusses the simulation environment for NoCs and compares the usage of NS-2 at various SoC design levels. Section 4 lists the authors who have used NS-2 to simulate their design of NoCs followed by a case study of fault tolerant NoCs in section 5. Finally, we conclude the paper in section 6.

## 1. NS-2 network simulator

NS-2 is an object-oriented, discrete event driven network simulator developed at UC Berkely and written in C++ and OTcl [22]. NS-2 is a very common tool used for simulating local and wide area networks. It implements network protocols such as TCP and UPD; traffic source behavior such as FTP, Telnet, Web, CBR and VBR; router queue management mechanism such as Drop Tail, RED and CBQ; routing algorithms such as Dijkstra, and a lot more. NS-2 also implements multicasting and some of the MAC layer protocols for LAN simulations. The simulator is open source, hence, allowing anyone and everyone to make changes to the existing code, besides adding new protocols aand functionalities to it. This makes it very popular among the networking community which can easily evaluate the functionality of their new proposed and novel designs for network research. The simulator is developed in two languages: *C++ and OTcl*[1]. C++ is used for detailed implementations of protocols like TCP or any customized ones. TCL scripting, on the other hand, is the front-end interpreter for NS-2 used for constructing commands and configuration interfaces. For example, if you want to develop a new routing protocol, you have to write it in C++ and add it into the NS-2 library. In order to check the functionality of this protocol, you use TCL scripting through which you can create the required topology, define parameters for links and nodes, and perform simulations to realize your own protocol in action.

Besides above-mentioned functionality of NS-2, a Network AniMator (NAM) is also provided with NS-2 in order to visualize and interact with the system at run-time. Finally, graphs can be created from the produced results to evaluate and analyze the performance of the system.

## 2. Network on a chip (NOC)

A NoC has been proposed as a viable alternative for the inefficient buses of today's SoCs. A NoC is viewed as a collection of computational resources connected through a network where they communicate using packets. Many topologies have been proposed for NoCs including a 2D mesh [15], a fat tree [11], and a honeycomb [02]. However, most common proposals of all is a 2D mesh due to its simplicity and easiness in implementation. A typical mesh topology is shown in figure 1 where circles and squares represent routers and resources respectively. Resources are shown here in black as they can be anything; a MIPS processor, a Digital Signal Processor (DSP), a memory module or even a combination of all these. Hence, a NoC is viewed as a regular, tightly coupled network of resources on a silicon chip using a packet based communication infrastructure making them identical to a regular network but on a much smaller geographical scale.

[1] Tcl script language with object-oriented extensions developed at MIT

Since a NoC is composed of heterogeneous components from various vendors with their own IPs (Intellectual Property), a network interface (NI) is needed which can act as a middle layer transforming streams of bits from the resource into packets before sending them to router and vice versa. When a resource has something to send, it starts transmitting the data to the NI, which creates data packets of predefined sizes and then forwards them to the connected router. Each router along the path checks the destination address and forwards the packet accordingly until it reaches the destination. Likewise, on receiving packets from the network, the last router simply passes them to the connected NI which extracts the bits from it and forwards it to the connected resource.

The interest of NoC researchers in computer networks clearly shows that there is a great deal of resemblance between the two domains. Computer networks have been at the core of research for decades and are still progressing at a very high pace. Similarly, a significant amount of work is being done in parallel computing area which is considered as the parent domain for NoCs. Design paradigms and protocols that are obsolete in networks (or the Internet) can be a good choice for NoCs because of their size and regular structure. However, besides the similarities, there are certain differences between the networks and NoCs due to which various design choices for NoCs need to be re-evaluated. Following is a brief description of both similarities and differences between computer networks and NoCs.

### 2.1. NoCs and Networks: *Similarities*

As the chips scales, it is becoming increasingly difficult to maintain global synchrony among various components of the chip with a single clock [07]. Due to this reason, the future SoCs are shifting toward the notion of Globally Asynchronous Locally Synchronous (GALS) [06] model where self-timed blocks (resources) will communicate with each other. This scenario is analogous to distinct computers in a traditional network which are connected together through a network. This, in turn, paves the way for decoupling communication from computation, which is the one of the key goals of NoCs [20].

In order to address the challenge of growing number of components on a chip, a 5-layered protocol model is proposed to govern the communication in a NoC in comparison to the OSI model in the [07]. Being said, a NoC abstractly resembles a traditional computer network in the sense that it has resources, routers, links connecting resources to routers and routers to routers, routing strategies (static or dynamic) and protocols, packet forwarding mechanism (flow control), and finally data in the form of packets to be transported on the network. All these characteristics of NoCs are influenced by computer networks making them a perfect candidate to utilize the facilities provided by NS-2 simulator. The authors of [13]

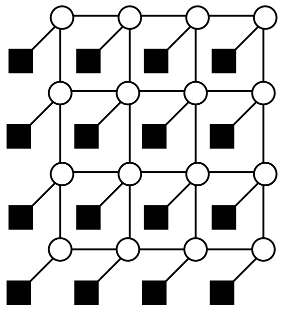

**Figure 1: A 2D mesh NoC**

performed simulations using *NS-2* and *chpsim* simulators for various network topologies and traffic patterns. Their argument is based upon the fact that due to scalability of chips, the SoCs are asynchronous, and therefore, on-chip operations can be event-driven instead of controlled by a single clock signal. The results they obtained showed that the statistics measured by *chpsim* were scaled versions of statistics obtained from NS-2, thus, endorsing the fact that the two networks --- *regular networks and NoCs* --- behaved in identical manner.

### 2.2. NoCs and Networks: *Differences*

The primary difference between NoCs and general networks is in the size of the two; networks can range from a single room to a city and even to the entire globe (the Internet) whereas NoCs would fit in a 50-60 nanometer silicon die (or even less than that). Computational and storage constraints are tighter in a NoC than in networks. On-chip memory is expensive in terms of the space it occupies. Likewise, limited amount of buffers could be allowed on-chip. Computation also comes at a cost; in a computer network a Network Interface Card (NIC) is usually equipped with a dedicated processor which implements a part of the protocol stack. This approach is infeasible for NoCs, again because of small dimensions. Since NoCs are embedded into portable hand-held devices like PDA's or mobile phones, the most important concern is the power consumption. Unlike traditional networks, where various equipment is supported by sophisticated electrical facilities, embedded system devices operate on batteries with small voltages. Consequently, complex computational logic and communication infrastructures are infeasible for NoCs in order to avoid excessive power consumption. Another significant difference is the abundance of wires in a NoC. A typical point-to-point NoC communication may be 300-bit wide providing large bandwidth for communication. Similarly, on-chip wires are very short in distances providing tight synchronization between the components. This way we can have smaller buffers in routers since communication can be done at smaller granularity.

Another aspect of NoCs is the implementation of various routing and communication protocols. TCP, for instance, is not a good choice for NoCs as its too complex to be implemented on a micro scale network. It means that lighter protocols need to be designed for NoCs which require minimum logic and memory resources. The regular and fixed nature of on-chip network makes it relatively easier to design protocols which are simpler and can be effectively implemented without incurring extraneous resources.

## 3. Simulating NoCs with NS-2

A SoC design process involves three major stages; *Behavioral design*, *Structural design and Physical design* [14]. The *behavioral design* specifies the functionality of the system at higher level of abstractions, whereas *structural* and *physical design* view reduces the abstraction level to logic gate and transistor level respectively. At *behavioral design level*, a SoC is realized as a collection of components which are modeled as blocks and connections along with protocols that govern the communication. Considering the above-mentioned scenario, it is clear that NS-2 is a perfect candidate for simulating and evaluating NoCs at *behavioral design level*. The individual blocks of a NoC are defined as ``nodes" and connections as ``links" in NS-2. Similarly, protocols can be defined over the blocks as ``agents" with relevant applications if any. Using the graphical animation of NS-2 (NAM), the behavior of the protocols can be observed interactively. It is quite convenient to realize various regular as well as irregular topologies using the TCL scripting language used in NS-2. Any form of topology ranging from mesh, torus, fat tree to even a fully connected network[2] can easily be created in NS-2. In contrast to traditional networks, a NoC has considerably short distance wires *(4.5 mm in a 20mm x 20mm chip, for instance*) and very large bandwidth (*ranging from 8 Gbits/sec to 16 Gbits/sec*). This can be realized by setting the link delay and bandwidth attributes of the links accordingly in NS-2.

### 3.1. Limitations of NS-2 with respect to NoCs

While NS-2 best suits to simulate a NoC at the behavioral design level, it is simply not possible to obtain a *structural* and *physical design* view of a NoC with NS-2. Logic gates and RTL logic design cannot be realized in NS-2. Failures like stuck-at faults that can cause permanent failures or transient faults that can scramble data on a chip, are irrelevant to computer networks, and thus cannot be modeled in NS-2. Similarly, calculating power estimations for various proposed mechanisms on chips is also not possible with NS-2. Currently, we do not have any simulator particularly for NoCs, however, existing software tools like Verilog or VHDL are being used to synthesize the chips and study the gate level behavior. These shortcomings of NS-2, however, are far outweighed by the facilities and benefits it offers for general simulations

[2] where each node is connected to every other node in the network, hence requiring very high wiring demands

of NoCs in a broader way. Since NoCs are still evolving and new designs and protocols are being proposed, it is rather convenient to use NS-2 for checking and validating these new designs for NoCs. It's only after a generic design for a NoC is being agreed upon that specialized tools down the line can be used to further realize the NoC functionality at a lower abstraction level.

## 4. Related Work

In [19], Ngo et al. performed experiments with NS-2 to compare various performance metrics of a 2D mesh with a Fat-Tree topology for NoCs. The authors of [21] have constructed a prototype using NS-2 to evaluate design options for a specific 2D NoC. They analyzed the series of simulation results about the relationship between buffer size in switch, communication load, packet delay and packet drop probability arguing that these results are very helpful in designing an appropriate switch for NoCs. Similarly, in [03], Vahdatpour et al. have proposed a new architecture for NoCs based on hierarchical graph. They performed simulations using NS-2 to show that the hierarchical graph performs better that mesh topology especially in terms of local traffic. In [05], Fummi et al. have presented a modeling and simulation methodology that is based on a timing accurate integration of SystemC and NS-2. They have argued that the two simulators should have common notion of time and should have the possibility of exchanging data bidirectionally. This, according to them, will yield a faster modeling of a system, provided the synchronization overhead is not too heavy. In [12], the authors have given a NoC-based implementation of a 4G modem. They proposed a modeling environment based on NS-2 to evaluate its performance and set critical parameters.

## 5. Case Study: Fault tolerant NoCs

As the number of transistors increase on a chip, so does the probability of faults, making reliability a major issue [10]. Failures can occur due to a variety of reasons, for example, crosstalk faults can lead to permanent or transient failures of the communication links [08]. Transient failures, in particular, can have many reasons for example, alpha particles emitted by trace uranium and thorium impurities in packages and high-energy neutrons from cosmic radiations can cause soft errors in semi-conductor devices[17]. In addition to this, implementing packet-based communication on-chip brings new reliability related challenges along with it. A transient fault may cause a bit-flip in the packet header due to which packet gets routed to a wrong destination. Similarly, in case of permanent faults, one or many links may go down, causing congestion in the alternate paths. Thus, it is extremely important to deploy mechanisms in NoCs that can handle both permanent and transient errors to ensure reliable packet delivery over shared communication channels.

Various fault tolerant designs have been proposed for NoCs, like stochastic communication model which proposes probabilistic flooding algorithms for NoCs [18]. This model is based upon randomized rumor spreading protocol where if a node has a data packet to send, it will be forwarded to a randomly chosen set of tiles in the neighborhood. Every node will forward the packet to all others it is connected to until the packets reach the destination. It is very likely that the destination receives multiple copies of the same packet. In such a case, after receiving the packet in correct form, the receiver discards all the other incoming copies of the same packet from other sources. Also if received packet is corrupt, it will be discarded and since multiple copies are received so no retransmission is required. Other proposals are concerned with providing a fault tolerant mechanism based upon retransmission of corrupt or missing packets [04, 16]. These proposals provide error tolerant protocols at different layers ranging from link level, then in the router and network level and finally in the end-to-end system. This enhances the error resilient capabilities of the NoC, but adding to the complexity of the overall design.

We have simulated a mesh based NoC with NS-2 to realize our fault tolerant protocol to cope with faults caused by permanent failures in a NoC [09]. The protocol follows a deterministic routing path to deliver packets until it detects a broken link or malfunctioning router. In such a case, routing tables are updated and communication resumes on newly available paths. Currently, besides improving the existing protocol, we are working on a reliable packet delivery mechanism to deal with failures caused by transient errors. As it's obvious that when lots of communicating partners are sharing the same links, congestion is very likely to occur. Our future work is mainly directed to handle congestion in the links and routers and provide mechanism that is well suited for NoCs. We, at the moment, are using NS-2 for all the NoC protocol designs.

## 6. Conclusion

This paper presents arguments to effectively use NS-2 network simulator for simulating NoCs. We have given an overview of the NS-2 simulator and NoCs followed by a general description of the similarities and differences between NoCs and traditional networks. Our argument is that NS-2 is a feasible candidate for simulating NoCs at a higher abstraction level since no specific NoC simulator exist to-date. The in-built facilities of NS-2 can effectively facilitate in the design of new protocols for NoCs. However, energy estimation metrics which are very significant for optimized design of NoC protocols cannot be modeled with NS-2. We have argued that although NS-2 is a perfect tool to observe the protocol behavior at behavioral design level of SoCs, it is of no use at the structural and physical design level. We further discussed how scaling chips are becoming more vulnerable to both transient and permanent errors and until dealt properly, realizing chips with billions of transistors would be too far from reality. We discussed various fault tolerant proposals for NoCs and described about our own approach followed by future work. We believe that identical nature of networks and NoCs may motivate network researchers to contribute effectively toward the design of NoCs.

## REFERENCES

[01] Ahmed Jerraya, Hannu Tenhunen and Wayne Wolf, ``Multiprocessor System-on-chips'', *Magazine, IEEE Computer*, July 2005 (Vol. 38, No. 7), pp. 36-40.

[02] Ahmed Hemani, Axel Jantsch, Shashi Kumar, Adam Postula, Johny Oberg, Mikael Millberg, Dan Lindqvist, ``Networks on a chip: An Architecture for billion transistor era'', *Proc. IEEE NorChip Conference*, November 2000.

[03] Alireza Vahdatpour, Ahmadreza Tavakoli, Mohammad Hossein Falaki, `Hierarchical Graph: A New Cost Effective Architecture for Network on Chip'', *Proc. International Conference on Embedded And Ubiquitous Computing*, Nagasaki, Japan, December 2005.

[04] D.Bertozzi, G.De Micheli, L.Benini, ``Energy-reliability trade-off for NoCs``, *Networks on Chip*, edited by A. Jantsch, H. Tenhunen, Kluwer KAP, March 2003.

[05] Franco Fummi, Giovanni Perbellini, Paolo Gallo, Massimo Poncino, Stefano Martini and Fabio Ricciato, ``A Timing-Accurate Modeling and Simulation Environment for Networked Embedded Systems'', *Proc., 40th Design Automation Conference (DAC)*, June 2 6, 2003, USA.

[06] J. Muttersbach, T. Villiger, H. Kaeslin, N. Felber and W. Fichtner, ``Globally-Asynchronous Locally Synchronous Architectures to Simplify the Design of On-Chip Systems'', *Proc. ASIC/SOC* pp. 317-321, 1999.

[07] L. Benini and G. De Michelli, `Networks on Chip: A new paradigm for component-based MPSoC design'', *Multiprocessors Systems on Chips*, edited by A. Jerrraya and W. Wolf, Morgan Kaufman, 2004, pp. 49-80.

[08] Ming Shae Wu and Chung Len Lee, ``Using a Periodic Square Wave Test Signal to Detect Cross Talk Faults'', *Journal, IEEE Design & Test of Computers*, Volume 22, Issue 2, March-April 2005, pp. 160-169.

[09] Muhammad Ali, Michael Welzl, Sybille Hellebrand, ``A dynamic routing mechanisn for Network on Chip'', *Proceedings, 23rd NORCHIP Conference*, Nov. 2005, Finland, pp. 70-73.

[10Partha Pratim Pande, Cristian Grecu, Andre Ivanov, Resve Saleh, and Giovanni De Michelli, ``Design, Synthesis, and Test of Networks on Chips'', *IEEE Design and Test*, September/October 2005 (Vol. 22, No. 5), pp. 404-413.

[11] Pierre Guerrier, Allen Greiner, ``A generic architecture for on-chip packet switched interconnections'', *Proc., Design, Automation and Test in Europe*, pp. 250-256, 2000.

[12] R. Lemaire, F. Clermidy, Y. Durand, D. Lattard, A. A. Jerraya, ``Performance Evaluation of a NoC-Based Design for MC-CDMA Telecommunications Using NS-2'', *Proc., 6th IEEE International Workshop on Rapid System Prototyping (RSP'05)*, pp. 24-30, 2005.

[13] R. Manohar and C. Kelly, ``Network on a Chip: Modeling Wireless Networks with Asynchronous VLSI'', *IEEE Communications Magazine*, November 2001.

[14] Rochit Rajsumman, ``System-on-a-chip: Design and Test'', *Artech House Publishers*, 2000.

[15] Shashi Kumar, Axel Jantsch, Juha-Pekka Soininen, Martti Forsell, Mikael Millberg Johny Oberg, Kari Tiensyrja and Ahmed Himani, ``A network on chip Architecture and Design Methodology'', *Proc., IEEE computer society annual symposium on VLSI*, 2002.

[16] Srinivas Murali, Luca Benini et al. ``Analysis of Error Recovery Schemes for Networks on Chips'', *IEEE Design and Test*, Vol. 22 ,Issue 5, Sept. 2005.

[17] S.K. Shukla and R.I. Bahar, ``Nano, Quantum and Molecular Computing, Implications to High Level Design and Validation'', *Kluwer Academic Publishers*, Boston, 2004.

[18] Tudor Dumitras, Radu Marculescu, ``On-Chip Stochastic Communication'', *Proc. Design Automation & Test in Europe (DATE)*, March, 2003.

[19] Vu-Duc Ngo, Hae-Wook Choi, ``On Chip Network: Topology design and evaluation using NS2 '', *Proc., 7th International Conference on Advanced Communication Technology (ICACT 2005)*, Phoenix Park, Korea, Feb. 21-23, 2005.

[20] William J. Dally and Brian Towles, ``Route packets, not wires: on-chip interconnection networks'', *Proc., Design Automation Conference (DAC)*, pp. 684-689, Las Vegas, NV, June 2001.

[21] Y.-R. Sun, S. Kumar, and A. Jantsch, ``Simulation and evaluation of a network on chip architecture using ns-2'', *Proc., IEEE NorChip Conference*, November 2002.

[23] http://www.isi.edu/nsnam/ns/