Journal of Computer Science 2 (10): 770-774, 2006
ISSN 1549-3636
© 2006 Science Publications

Study of Network on Chip resources allocation for QoS Management

Abdelhamid HELALI, Adel SOUDANI, Jamila BHAR and Salem NASRI

Laboratoire d'Electronique et de Micro-Electronique (EµE),
Faculté des Sciences de Monastir, 5019 Monastir, Tunisia

Abstract: The increasing complexity of integrated circuits and application requirements drive the research of new on-chip interconnection architectures. A network on chip draws on concepts inherited from distributed systems and computer networks subject areas to interconnect IP cores in a structured and scalable way. The main goal pursued is to achieve superior bandwidth when compared to conventional on-chip bus architectures. The complexity of Systems-on-Chip (SoC) is growing; meeting real-time requirements is becoming increasingly difficult. Predictability for computation, memory and communication components are needed to build up real-time SoC. To achieve guaranteed throughput and bounded delivery delay, buffers in network interfaces (NIs) must be dimensioned to hide round-trip latency and rate difference between computation and IPs communication.. It is crucial to shape these buffers according to the network requirements and to bring out the right specification before the design step to provide desired performances in the SoC. In this field this paper describes and presents a performance analyses of NoC shaped on mesh architecture. The goal of this work is to quantify buffering requirements in the NoC nodes by the analyze of some QoS metrics such as drop, compute latency, and throughput. This study presented in this paper is based on simulation approach of a mesh (4 x 4) NoC behavior under multimedia communication process with MPEG-4 (Moving Picture Experts Group) flows.

Keywords: Network-on-chip; Quality-of-service, latency, buffering.

INTRODUCTION

Increasing transistor density, higher operating frequencies, short time-to-market and reduced product life cycle characterize today's semiconductor industry scenery [1]. Under these conditions, designers are developing ICs that integrate complex heterogeneous functional elements into a single chip, known as a system on a chip (SoC). As described by Gupta [2] and Bergamaschi [3], SoC design is based on intellectual property (IP) cores reuse. Gupta in [2] define core as a pre-designed, pre-verified hardware piece that can be used as a building block for large and complex applications on an IC. Examples of cores are memory controllers, processors, or peripheral devices such as MAC Ethernet or PCI bus controllers. Cores may be either analog or digital, or even be composed by blocks using technologies such as micro-electromechanical (MEMS) or optoelectronic systems [1, 4]. Cores do not make up SoCs alone; they must include an interconnection architecture and interfaces to peripheral devices [4]. The interconnection architecture includes physical interfaces and communication mechanisms, which allow the communication between SoC components to take place.

Generally, the interconnection architecture is based on dedicated wires or shared busses. Dedicated wires are effective for systems with a small number of cores, but the number of wires around the core increases as the system complexity grows. Therefore, dedicated wires have poor reusability and flexibility. A shared bus is a set of wires common to multiple cores. This approach is more scalable and reusable, when compared to dedicated wires. However, busses allow only one communication transaction at a time, thus all cores share the same communication bandwidth. System scalability is limited to few dozens IP cores [5].

The solution based on the use of separate busses interconnected by bridges may reduce some of these constraints mainly system bandwidth. Nonetheless, scalability remains a problem for hierarchical bus architectures.

According to ITRS [1], in 2018, ICs will be able to contain billions of transistors, with feature sizes around 18 nm and clock frequencies around 10 GHz [1]. In this context, a network on chip (NoC) appears as a probably better solution to implement future on-chip interconnects. A NoC is an on-chip network [8] composed by cores connected to switches, which are in turn connected among themselves by communication channels.

As described in [10, 11, 15, 16], NoCs are emerging as the best solution to the existing interconnection architecture constraints. They provide attractive characteristics mainly energy efficiency and reliability [7]. They also supply scalable bandwidth when compared to traditional bus architectures [16, 21]

Corresponding Author: Abdelhamid HELALI, Laboratoire d'Electronique et de Micro-Electronique (EµE), Faculté des Sciences de Monastir, 5019 Monastir, Tunisia

End to end communication in a system is accomplished by the exchange of messages among IP cores. Often, the structure of particular messages is not adequate for communication purposes. This leads to the concept of packet. [17] In the context of NoCs, packets are frequently a fraction of a message. Packets are often composed by a header, a payload, and a trailer. To ensure correct functionality during message transfers, a NoC must avoid deadlock, livelock and starvation [17]. Deadlock may be defined as a cyclic dependency among nodes requiring access to a set of resources so that no forward progress can be made, no matter what sequence of events happens. Livelock refers to packets circulating the network without ever making any progress towards their destination. It may be avoided with adaptive routing strategies. Starvation happens when a packet in a buffer requests an output channel, being blocked because the output channel is always allocated to another packet.

It appears of importance, to meet the required performances that in a network on chip hardware resources and management roles should be specified in an earlier step of the system design [9, 12] . The main attention should be given to the choice of the physical buffer size in the switching nodes.

In this idea range, this paper proposes a study of switching buffer considerations in NoC.

This paper is organized as follows. Section 2 details the network on chip specificities then we describe the NoC switcher architecture. A 4x4 mesh architecture using this switcher is then described using NS2 tool and simulated for IP MPEG communication process. In Section 4, some simulation results for the NoC architecture are presented and discussed to bring out some physical requirements enabling best QoS supply. We finish by the conclusions and directions for future work.

Network on chip specificities

Two parts compose an interconnection network: the services and the communication system. Rijpkema et al. [10] define several services considered essential for SoC design, such as data integrity, throughput and latency guarantees. The implementation of these services is often based on the specification of the OSI lower three layers (physical, link, and network).

The communication system, on the other hand, is what supports the information transfer from source to target. The communication system allows that every core send packets to every other core in the NoC structure. The NoC structure is a set of switches interconnected by communication channels. The way switches are connected defines the network topology. According to the topology, networks can be classified in one of two main classes: static and dynamic [18, 19]. In static networks, each node has fixed point-to-point connections to some number of other nodes. Hypercube, ring, mesh, torus and fat-tree are examples of networks used to implement static networks. Dynamic networks employ communication channels that can be (re)configured at application runtime. Busses and crossbar switches are examples of dynamic networks.

The communication mechanism, switching mode, and routing algorithm are functions of the network topology and are used to compose the services provided by the NoC.

The communication mechanism specifies how messages pass through the network. Two methods for transferring messages are circuit switching and packet switching [19]. In circuit switching, a path is established before packets can be sent by the allocation of a sequence of channels between source and target. This path is called a connection. After establishing a connection, packets can be sent, and any other communication using the allocated channels is denied, until a disconnection procedure is executed. In packet switching, packets are transmitted without any need for connection establishment procedures.

Packet switching requires the use of a switching mode, which defines how packets move through the switches. The most important modes are store-and-forward, virtual cut-through and wormhole [20]. In store-and-forward mode, a switch cannot forward a packet until it has been completely received. Each time a switch receives a packet; its contents are examined to decide what to do, implying per-switch latency. In virtual cut-through mode, a switch can forward a packet as soon as the next switch gives a guarantee that a packet will be accepted completely [11]. Thus, it is necessary for a buffer to store a complete packet, like in store-and-forward, but in this case with lower latency communication. The wormhole switching mode is a variant of the virtual cut-through mode that avoids the need for large buffer spaces. A packet is transmitted between switches in units called flits (flow control digits—the smallest unit of flow control). Only the header flit has the routing information. Thus, the rest of the flits that compose a packet must follow the same path reserved for the header.

The routing algorithm defines the path taken by a packet between the source and the target. According to where routing decisions are taken, it is possible to classify the routing in source and distributed routing [17]. In source routing, the whole path is decided at the source switch, while in distributed routing each switch receives a packet and decides the direction to send it. According to how a path is defined to transmit packets, routing can be classified as deterministic or adaptive. In deterministic routing, the path is uniquely defined by the source and target addresses. In adaptive routing, the path is a function of the network traffic [17, 20]. This last routing classification can be further divided into partially or fully adaptive. Partially adaptive routing uses only a subset of the available physical paths between source and target.

The general structure of a typical switch

In this section we describe the target architecture of the network on chip switch. This component will then be used to build up the NoC, that interconnect IP cores, subject of this study.

An on-chip switch has to provide correct transfer of messages between IP cores.
The internal on block structure of the switch is represented in Fig. 1. It is in general composed of complementary functional blocks such as; routing logic, FIFO manager, arbitration logic and communication ports directed to other switches or cores. The communication ports include input and output channels, which can have buffers for temporary information storage.

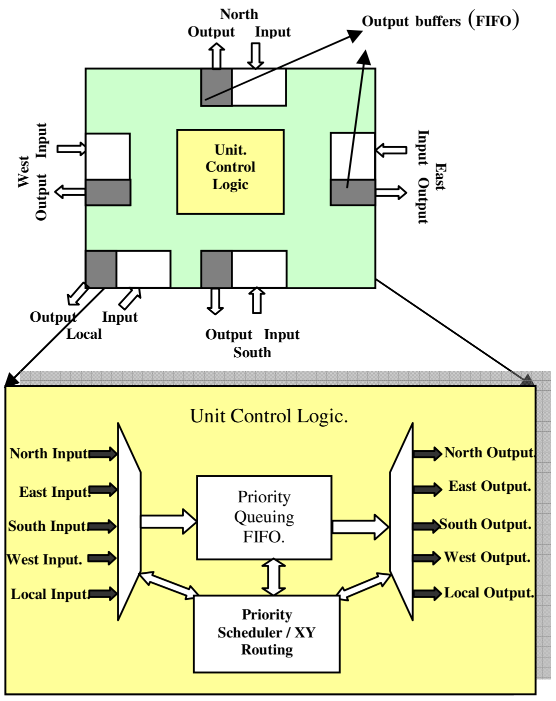

Fig. 1: General switch component architecture

The switch has routing control logic and five bi-directional ports: East, West, North, South, and Local. Each port has an input buffer for temporary storage of information. The Local port establishes a communication between the switch and IP core. The other ports of the switch are connected to neighbor switches, as presented in Fig. 1. The routing control logic implements the arbitration logic and a packet-switching algorithm.
The main critical part driving the switch performances is storage memory depth. In fact for embedded system one of the major constraints is optimizing circuit area. In addition the management approaches such FIFO queuing, or priority queuing

Target architecture of the Network on chip

NoC topologies are defined by the connection structure of the switches. The studied NoC assumes that each switch has a set of bi-directional ports linked to other switches and to an IP core. It is built around a mesh topology. Each switch has a different number of ports, depending on its position with regard to the limits of the network, as shown in Fig. 2. For example, the central switch has all five ports defined in Section 2. However, each corner switch has only three ports.

The use of mesh topologies is justified to facilitate placement and routing tasks as stated before [21, 15].

The general structure of the NoC being described with NS-2 simulator can also be applied to bring out similar studies for others NoC topologies such as torus and hypercube NoC topologies [21]. However, building such topologies implies changes in switch connections and, more importantly, in the routing algorithm.

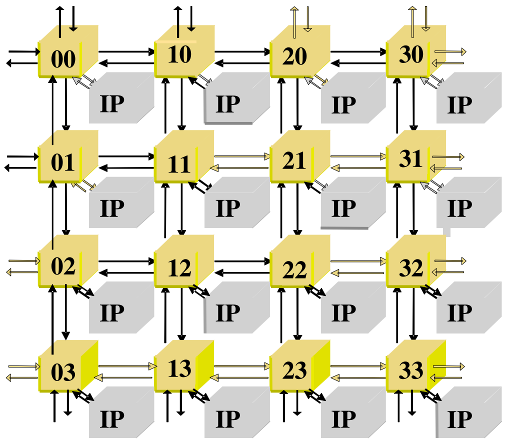

Fig.2: 4 x 4 Mesh NoC structure.

RESULTS

We used for this study the available network simulator ns [22, 23]. This tool is becoming one of the more popular platforms for performances analyses in the network research community. Developed as a part of the VINT project, ns is a multi-protocol, object-oriented, programmable simulator which also includes a number of facilities for large-scale simulations and the capability to interface the simulator to a live network. Its architecture is designed in a way intended to promote extension by users. The network simulator ns has been used to investigate a number of protocols, including TCP behavior, router queuing policies, multimedia, wireless and satellite networking and application-level protocols [22].
The studied architecture is described with this tool as 4x4 Mesh. The maximum bandwidth link is fixed to 2GB/s.
The purpose of the study is to optimise buffer size according to the general network loading states and also according to the interconnected IPs throughput.
The traffic simulated is MPEG4 shaped as an exponential traffic. This traffic is transferred over the network between tow IPs interconnected to the 00 switch (source) and the thirty third switch. The packet size is of 4 bytes (32 bits) based on a by 8 bits flits (4 flits by packet).

Application rates and network losses

As being discussed, a reasonable buffer size may drive on network on chip performances. The main idea is to keep the minimum buffer size to avoid packet drop and over-end to end delays. Otherwise, the hardware cost will be large accordingly, it may influences both network quality of service performances and also hardware designing resources optimization.

We mainly focus on per cent flit losses due to buffer congestion for a network loading. This helps us to identify the optimal buffer size for the switch design. The routing approach is based on the Wormhole routing method. It is mainly designed for on chip multiprocessor interconnect networks.

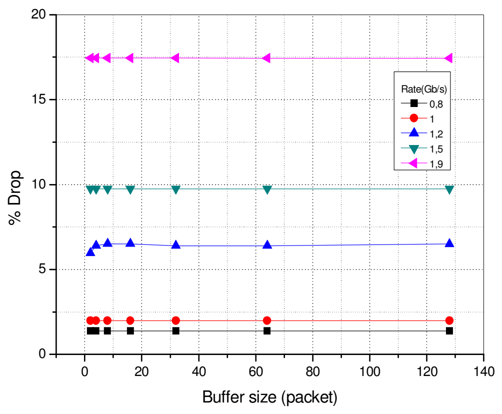

Fig. 3: % dropped packets according to the application rate and buffer size

Wormhole routing not only reduces the store-and-forward delay at each switch, but it also requires much less buffer spaces.

Figure 3 shows the relationship between % packet dropped and the available switch buffer size. It carries out boundary values of buffer size for different rates and also their associated per cent packet dropped. This figure shows that the % dropped packets increases with application rate. For 2 Gbits available bandwidth it is suitable to keep an application rate under then 1.2 Gbits.

End to end delay and buffer size

The end to end delay is one of the major critical QoS metrics. Some real-time applications bound up this value and require a specific hardware resources and particular management approaches in the NoC switch.

The best case is to provide the shortest constant end to end delay or at least with the minimum fluctuation. This can avoid synchronization between communication processes.

Figure 4 sums up the end to end delay when the switching buffer is managed without any scheduling approach.

This figure shows that the optimal buffer depth is about 32 packets. This size provides a range of end to end delay between 268 µs and 290 µs for different application rates.

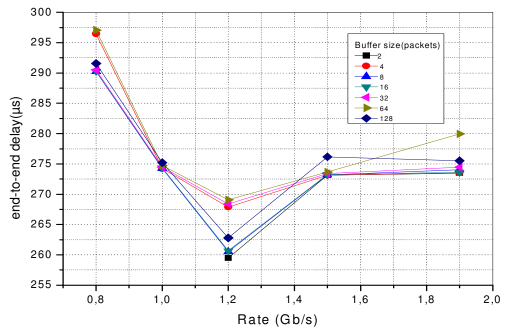

Fig. 4: End to end delay according to buffer size and application rate

Figure 5 shows that the overage end to end delay for different buffer size is quietly around 275 µs. It also shows that for to 2Gbits available bandwidth the better application rate should be nearly to 1,5 Gbits.

We think that this rate value can be hardly ameliorated with the development of an adequate packets scheduling algorithm for the management of the output switcher buffer.

As shown in Fig. 1, the scheduler should improve service quality according to the flit requirement, using priority queuing, fair approach or a combined technique.

In the flowing section we present the general structure of scheduling algorithm intended to manage the switch buffer during the communication process.

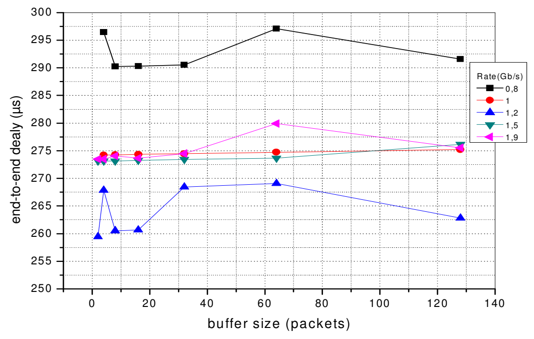

Fig. 5: End to end delay according to buffer

CONCLUSION

In this paper we have studied Quality of Service (QoS) for network on chip. We have focused the study on the switch buffering requirements.

In fact, we have clarified that buffer depth in switch drive on the cost of network area and then whole system on chip cost (hardware resources and power). Further more, metrics of quality of service during SoC communication processes can be widely affected according to the switch buffering capacity. The optimisation of the buffer size in network switching is then an important step before network design.

We have used for this purpose the NS tool to carry out a simulation study for a 4X4 mesh architecture.

We studied the optimal buffer size according to different rates and also with the network loading state.
We showed that the best buffer size should be greater nearly to 32 packets to avoid network congestion and to provide a wide application rate rang.
We think that the QoS in the switch and then in the network can be performed with the adaptation of an appropriate approach for packets queuing in the switch buffer such as priority queuing.
For this purpose we are now working on the specification of a buffer queuing with scheduling management approach. The idea is to meet both network required performances and embedded hardware system constraints.

REFERENCES

1 International Sematech. International Technology Roadmap for Semiconductors—2003 Update, Available at http://public.itrs.net.

2 R. Gupta, Y. Zorian, 1997. Introducing core-based system design, IEEE Des. Test Comput. 14 (4):15-25.

3 R. Bergamaschi, et al., 2001. Automating the design of SoCs using cores, IEEE Des. Test Comput. 18 (5) 32-45.

4 G. Martin, H. Chang, 2001. System on chip design, The Nineth International Symposium on Integrated Circuits, Devices and Systems (ISIC'01), Tutorial 2

5 S. Kumar, et al., 2002 A network on chip architecture and design methodology, in: IEEE Computer Society Annual Symposium on VLSI (ISVLSI'02), pp: 105-112.

6 M. Millberg, E. Nilsson, R. Thid, S. Kumar, 2004. A. Jantsch, The Nostrum backbone a communication protocol stack for networks on chip, Proceedings of the VLSI Design Conference

7 L. Benini, G. 2001 De Micheli, Powering networks on chips: energy-efficient and reliable interconnect design for SoCs, in: 14th International Symposium on Systems Synthesis (ISSS'01) pp: 33-38.

8 L. Benini, G. 2002. De Micheli, Networks on chips: a new SoC paradigm, IEEE Comput. 35 (1): 70-78.

9 P. Guerrier, A. Greiner, 2000. Ageneric architecture for on-chip packet-switched interconections, in: Design Automation and Test in Europe (DATE'00), pp: 250-256.

10 E. Rijpkema, K. Goossens, 2003. A. Radulescu, Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip, Design, Automation and Test in Europe (DATE'03), pp: 350-355.

11 E. Rijpkema, K. Goossens, P. Wielage, 2001. A router architecture for networks on silicon, The 2nd Workshop on Embedded Systems (PROGRESS'2001), pp: 181-188.

12 Yi-Ran Sun, 2001. Simulation and Performance Evaluation for Networks on Chip, Master's thesis, Department of Microelectronics and Information Technology, Royal Institute of Technology, Stockholm, Sweden

13 Adriahantenaina.A, Charlery.H, Greiner 2003. A, Mortiez.L and Zeferino.C, SPIN: A scalable, packet switched, on-chip micro-network, Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE03).

14 Benini, L. and De Micheli, G., 2001, Powering networks on chips, Int'l Symposium on System Synthesis (ISSS), pp. 33-38.

15 Bolotin, E., Cidon, I., Ginosar, R. and Kolodny, A., 2004, QNoC: QoS architecture and design process for network on chip, Journal of Systems Architecture 50(2-3), 105-128. Special issue on Networks on Chip.

16 Goossens, K., Dielissen, J., Gangwal, O. P., Gonzalez Pestana, S., Radulescu, 2005 A. and Rijpkema, A design flow for application-specific networks on chip with guaranteed performance to accelerate SoC design and verification, Proc. Design, Automation and Test in Europe Conference and Exhibition

17 J. Duato, S. Yalamanchili, L. Ni, 2002. Interconnection Networks, An Engineering Approach, Morgan Kaufmann, Los Altos, CA, p: 624

18 J. Hennessy, D. Patterson, 1996. Computer Architecture: A Quantitative Approach, Morgan Kaufmann, San Francisco, CA, p: 760

19 K. Hwang, Advanced 1992. Computer Architecture: Parallelism, Scalability, Programmability, McGraw-Hill, New York, p: 672

20 L. Ni, et al., 1993, A survey of wormhole routing techniques in direct networks, IEEE Computer. 26(2) 62-76.

21 Fernando Moraes, et al., 2004, HERMES: an infrastructure for low area overhead packet-switching networks on chip, Integration, the VLSI Journal 38(1) 69-93

22 K. Fall, K.Varadhan, (The Vint project), the ns Manual, February 28, 2005, (http:// www-isi.edu/nsnam/ns/ns-documentation).

23 G. Rodrigez, P. Merino, M.M. Gallardo, 2002. “An extension of the NS simulator for active network research”, Computer Communications, P:189-197.