Packet-switching Network-On-Chip Features Exploration and Characterization

Gilles Sassatelli, Séverine Riso, Lionel Torres, Michel Robert
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier
Université de Montpellier II / CNRS
161, rue Ada – 34392 Montpellier Cedex 5, France
Email: <lastname>@lirmm.fr

Fernando Moraes
FACIN PUCRS
Av. Ipiranga, 6681 – Prédio30/ BLOCO 4
90619-900-Porto Alegre-Brasil-
Email: moraes@inf.pucrs.br

Abstract

The era of bus-dominated communication architectures for SoCs might end soon: the increasing number of cores used on a single die used in response to the power-hungry applications tend to make SoC designs more and more communication-centric. Communication structure synthesis is an increasingly challenging problem; bus-based structures are still leading but tend to become the cornerstone of successful SoC designs: among others issues, they hardly allow parallel communications and scale badly from the electrical point of view. Based on a previously published Network-on-Chip [18], this paper presents and discusses the performance/cost tradeoffs achieved through different hardware solutions, like Quality of Service (QoS) support.

1. Introduction

A communication architecture is usually described using two characteristics: Bandwidth and latency. Depending on the behavior of the communication architecture under those two aspects, the overall system performance is greatly affected. A high bandwidth is usually expected in raw data processing systems (multimedia for instance) while achieving a low latency is a must for reactive systems (real-time control/operation for instance).

1.1 Communication architecture families

Two main communication structure families exist, namely:

- Point-to-point
- Time / space multiplexing

Point-to-point communication structures

These structures are usually dedicated wires between existing cores. This ad-hoc solution provides a guaranteed throughput and latency but has two main drawbacks: this is non-scalable since the number of physical links increases rapidly with the number of cores, and the bandwidth is wasted: a physical link is instantiated for each possible communication, even if the data to be transferred is limited.

Time / space multiplexing based structures

Time / space multiplexing structures are known as bus-based structures. A single bus usually provides a single physical link which is shared among all connected cores: this is a time-multiplexing access method to the medium. The bandwidth allocation is usually dynamic, a given core issuing asynchronously a request on the bus for a given communication. Some improvements like hierarchical buses provide parallelism (space multiplexing), however limited.

1.2 Technology scaling

Besides intrinsic performance considerations (i.e. bandwidth and latency), other aspects are to be taken into account when considering below 100nm CMOS processes.

Synchronism issues

Completely synchronous systems are no-more feasible. Figure 1 [1] depicts the maximum foreseeable dimensions of a synchronous system versus target frequency. This underlines the need of asynchronous-based communications, GALS designs (Globally Asynchronous Locally Synchronous) are considered as an interesting alternative.

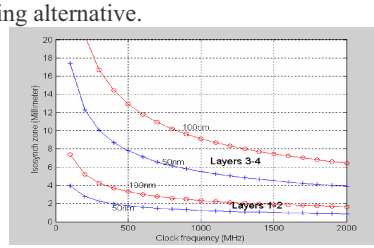

Figure 1 – Isochronous zone dimensions

Therefore shared busses nor point to point connections can be used anymore, since global wires spanning the whole chip are forbidden [2].

Power consumption issues

Deep submicron CMOS technologies exhibit a growing current leakage. If a shared bus is used, each core connected implies an additional capacitance, decreasing the electrical performance and increasing the power consumption. Some research works are investigating solutions aiming at reducing the power consumption, like continuous operating system controlled frequency and Vdd, power-down, etc.

1.3 Network on chips

From the aforementioned considerations, an interesting solution stands in between the two existing communication architecture families: structures providing both point-to-point connexions and time/space multiplexing. Numerous topologies exist, a few of them are depicted in figure 2. Usually a core is attached to each router in the structure. Data are then to be routed from an initiator to a target. The routing can be either static or dynamic. An interesting characteristic of those structures is that they allow to be designed according to the GALS principle: each core/router couple is fully synchronous whilst inter-router communications can take place asynchronously.

This paper is organized as follows:

- Section 2 is a synthesis of the state of the art giving an overview of the existing principles and realizations.
- Section 3 presents the Hermes architecture developed by the GAPH group of the PUCRS University, Brazil and the proposed modification in order to improve performance under several aspects including a quality of service (QoS) technique.
- Section 4 presents the characterization results for several modifications performed on the NoC including the QoS figures (performance, area, power).
- Section 5 summarizes the works presented in this paper and provides an overview of the ongoing work within this project.

2. NoCs: State of the Art

2.1 Classification

Several features describe a NoC, we here only describe 2 main ones, for extensive information refer to [3], [4], [5] and [8].

Topology

Different topologies are described in the literature. The predominant topology is the 2D Mesh array. The reason for this choice derives from its three advantages: facilitated implementation using current IC planar technologies, simplicity of the XY routing strategy [6],[7] and network scalability. The different approaches are summarized in figure 2, refer to [10], [12] and [16] for extensive information on each topology.

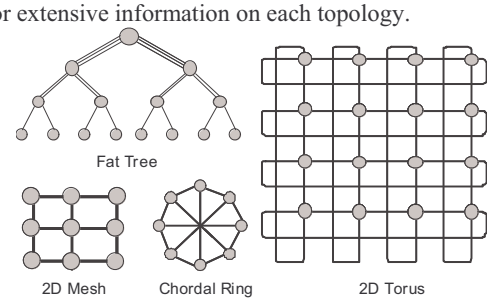

Figure 2 – Exiting NoC topologies

Routing

The routing can either be static or dynamic. Static routing architectures are less area-consuming while dynamic routing architectures usually behave more efficiently under heavy load conditions. Routing strategies are very topology-dependant, we will describe in section 3 a few existing strategies for 2D Mesh NoCs.

Three inter-router packet routing strategies exist:

- Store-and-forward

Each packet received in a router is stored and then forwarded as soon as possible. That reduces the network contentions but at the cost of increased latency and silicon area due to the important buffering needed.

-Virtual cut-through

Each packet is sent only if the next router can store it. The memory needed for this strategy is also important, however the latency is lower than in the Store-and-forward strategy.

-Wormhole routing

Each packet is splitted in several words called *flits*. All flits are using the same route which is determined by the first flit. Hence, a route (set of adjacent links) is reserved for transferring the whole packet. This technique needs less buffering memory in comparison with the two previous ones, but is subject to the *Head-of-line* problem: a set of physical links forming the route might be blocked (and so unusable for other communications) if the first flit is blocked.

2.2 Quality of Service

The lack of a general control unit in NoCs (bus arbiter) provides better scalability but has a major drawback: without any additional hardware support, no quality of service (QoS) can be provided, namely neither *maximal latency*, nor *minimal bandwidth is* guaranteed. This feature is mandatory for many application domains, like real-time systems.

Two different solutions for supporting QoS exist:

Circuit switching

A router is reserved between two cores, no other communication but the one which has initiated the

reservation can make use of it. It of course provides a fully predictable and guaranteed communication link behavior.

Virtual channels

Also known as TDM (Time Division Multiplexing), this technique is also based on reservation, some timeslots on a given route being reserved to a given communication. While more area consuming than the circuit-switching technique, this solution enables link sharing, non priority communication can share the same physical links through using different timeslots.

3. Hermes architecture

3.1 Overview

The Hermes project aims at defining a flexible environment allowing rapid evaluation of different NoC structures. The core of this environment is a set of tool allowing i) automatic VHDL RTL code generation for a given set of NoC parameters ii) graphical traffic analysis allowing to identify contentions, buffering problems, etc. From now on, we will focus on the 2D mesh topology which was selected for our studies.

The Hermes network is based on a 2D Mesh switch (i.e. router) array [16]. The Hermes 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 its local IP core. The other ports of the switch are connected to neighbor switches, as presented in Figure 3. The routing control logic implements the arbitration logic and a packet-switching algorithm.

Moreover the switching mode used is based on wormhole approach, allowing to decrease the latency.

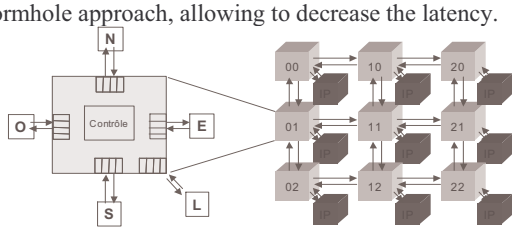

Figure 3 – HERMES topology overview

3.2 Packet format

A Hermes packet (described in figure 4) is composed by a header and the payload. The header specifies both the target address and the number of flits in the payload.

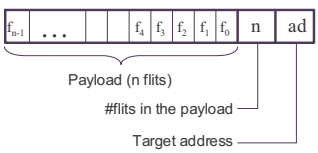

Figure 4 – packet format

3.3 Control logic

Two modules implement the control logic: *routing* and *arbitration*, as presented in Figure 5. When a switch receives a header flit, the arbitration is executed and if the incoming packet request is granted, an routing algorithm XY (route according to X-axis and then to the Y-axis) is executed to connect the input port data to the correct output port.

A switch can simultaneously grant requests for establishing up to five connections. Arbitration logic is used to grant access to an output port when one or more input ports simultaneously require a connection. A dynamic arbitration scheme using a round robin algorithm is implemented.

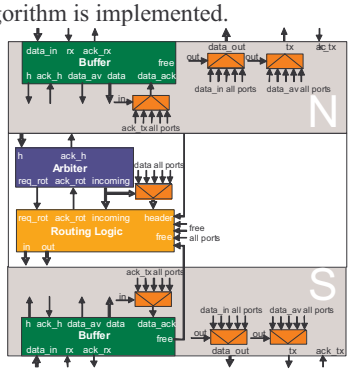

Figure 5 – Partial block diagram of the switch, showing two of the five ports

3.4 Message buffering

When a flit is blocked in a given switch, the performance of the network is affected, since the flits belonging to the same packet are blocked in other switches. To lessen the performance loss, a buffer is added to each input switch port, reducing the number of switches affected by the blocked flits. The inserted buffers work as circular FIFOs. In Hermes, the FIFO size is parameterizable, and a size of eight flits has been used for prototyping purposes.

3.5 QoS support

We have implemented an alternative QoS technique within the Hermes network, based on option packets. Inspired by the IPv4 standard [17][18] which features an option field, we've modified the Hermes packet format and switch structure. As per the IPv4 standard, IP option packets specify some directives for routing, like route to take, latency or bandwidth priority. Hermes packets only specify a priority field, which is a binary value.

As depicted in figure 6, two queues per port have been used: one for normal traffic, the other one for prioritized traffic. Whenever an option packets arrives in a given switch, the last normal packet transmission (if any) is completed and option packets are then immediately processed. While usually more area consuming, we have opted for output buffering because it exhibits better performance due to better use of

buffering space: an incoming packet is routed to the destination port and then buffered. On average, that decreases the congestions on the considered input port since for input buffering a packet blocked due to a congestion on its destination port does block the following ones (head-of-line problem).

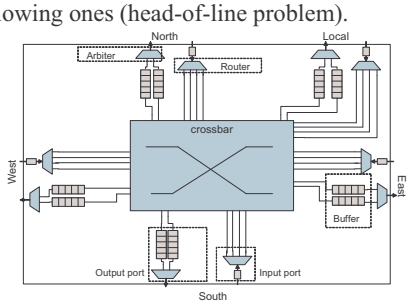

Figure 6 – Ouput buffering QoS structure

4. Results

4.1 Preliminary considerations

All figures provided throughout this section aim at characterizing the NoC under latency, bandwidth, area and power aspects:

Load is often used as a parameter in the following plots. For a NoC architecture, *load* is defined as the percentage of aggregated data that can be fed into the network per time unit, as to say the bandwidth per port multiplied by the number of ports. The experimental protocol used is based on infinite fifos: for instance, a 50% load means that *each* two cycles, *each* core connected is emitting a send request to its router for sending a packet. If the packet is not accepted, it is buffered into the testbench (infinite fifos) until the router accepts it. Unless specified, all experiments are using a 8x8 NoC, output buffering, and the generated traffic is random. Half of the cores are master, half slaves.

4.2 Performance results

4.2.1 Architecture comparisons

Figure 7 depicts the latency versus load figures for different architectures in comparable scenarios.

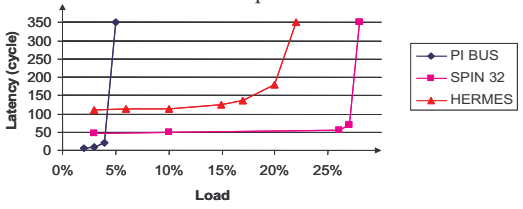

Figure 7 – Latency vs load for 3 architectures

As depicted in figure 7, the SPIN micronetwork [14] and [15] behaves better than Hermes at both low loads and high loads. This is due to the Fat-tree structure it uses which provides a hierarchy in the communications. The drawback of this solution relies in its poorer scalability compared to regular 2D mesh structures. Both NoCs exhibit better performance than the PI-bus (Processor Interconnect bus)[19] at high loads as expected. Nevertheless, the PI-bus dominate for lower loads where communications are achieved in very few cycles.

4.2.2 Topology impact

We have explored two different topologies, 2D mesh and torus. While the first one is our favorite mainly for scalability reasons (better adapted to silicon planar technologies, neighbor to neighbor connections only), the torus exhibits better performances (figure 8) because it reduces the maximum distance between two cores (divided by 2). Additionally it also helps reducing the congestions because of the opportunity to route packets in two opposite directions for reaching a given core.

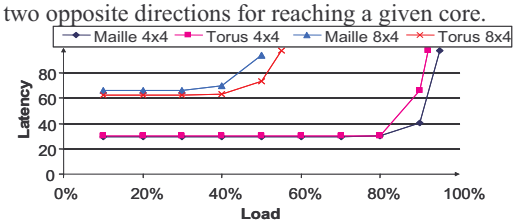

Figure 8 – Latency vs load for different topologies

4.2.3 Buffer sizing impact

Buffer sizing impacts on performance to deliver the message. Using large buffers improve performance under heavy load: since more packets can be buffered into routers fewer physical links are blocked (a packet is buffered on fewer routers) when a congestion occurs. Figures 9 and 10 show respectively the throughput and packet latency (number of cycles for delivering a packet) versus network load.

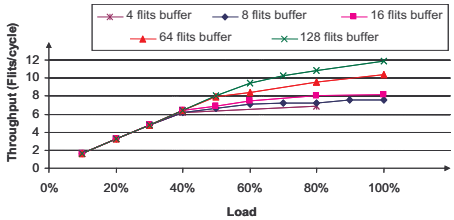

Figure 9 – Throughput vs load for different buffer sizes

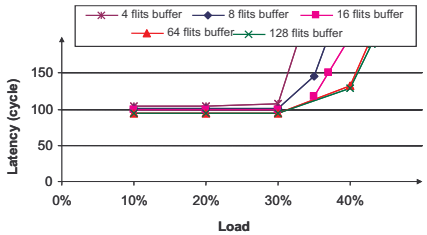

Figure 10 – Latency vs load for different buffer sizes

4.2.4 Input buffering versus output buffering

As explained in 3.5, output buffering tends to improve the performance of the NoC, due to the ability to route packets to the destination port before buffering

them. This is shown in figures 11 and 12. Among all the simulations we have performed, on average, the routing time inside the router is decreased from 8 cycles to 3.5 cycles. Output buffering is also more area consuming, as shown in section 4.3.

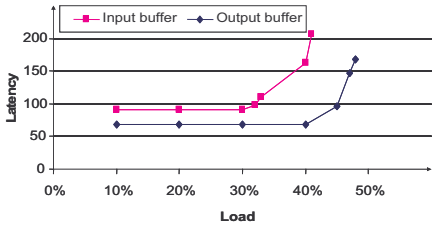

Figure 11 – Latency vs load for different buffer sizes

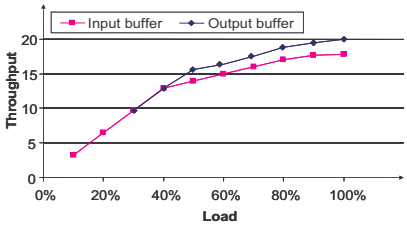

Figure 12 – Throughput vs load for different buffer sizes

4.2.5 Quality of service

The technique presented in section 3.5 has been implemented and simulated according to the same protocol used for the previous versions. While the proposed solution does not strictly guarantee neither a maximum latency nor a minimum bandwidth, it nevertheless systematically improves the delivery time for option packets (figure 13). As of now, the structure does not limit the maximum number of option packets, it has been verified that using only option packets leads to exactly the same performance as a non-QoS Hermes NoC.

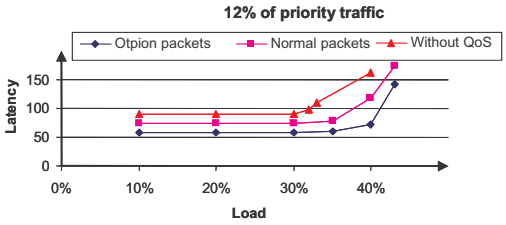

Figure 13 – Latency vs load for QoS NoC

As depicted in figure 13 and 14, the delivery time for option packets is largely improved when compared to normal traffic. This statement remains valid until 25% of priority packets. Furthermore, normal traffic is not much affected by the QoS usage. This is probably due to the fact that two queues are employed. The drawback of that approach relies in the potential under use of priority queues when no QoS traffic is used.

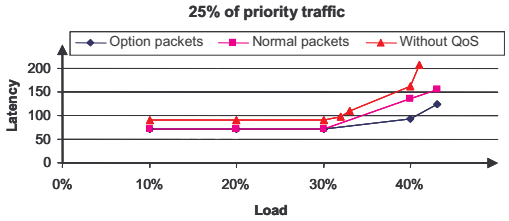

Figure 14 – Latency vs load for QoS

Figure 15 and 16 depict the percentage of packets *immediately* accepted by the NoC (emitted from cores) per time unit for 2 average values of priority traffic (1% and 10%). It is interesting to note that priority traffic (1% of total traffic, figure 15) remains unaffected until 60% of load while normal traffic has almost the same behavior as for non QoS Hermes. That is for a given scenario however and has not been verified extensive. When priority traffic reaches 10% of total traffic (figure 16), the benefits are largely deteriorated, the critical threshold being the same for both priority and normal traffic (35%).

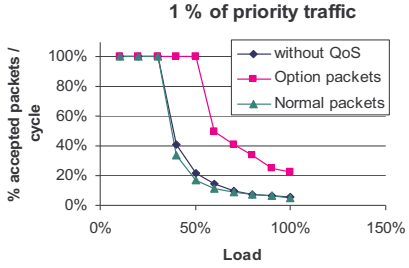

Figure 15 – Percentage of immediately accepted packets in QoS and non-QoS modes

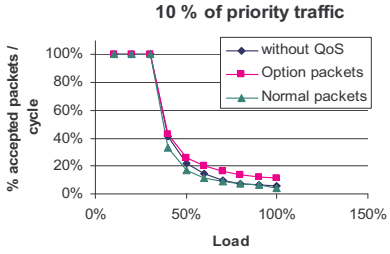

Figure 16 – Percentage of immediately accepted packets in QoS and non-QoS modes

4.3 Area et power results

Considering NoCs as a viable alternative to bus-based system leads to consider the induced cost overhead, both in term of area and power consumption. The following results summarize the values obtained for the structures used in the previous sections, as to say 8x8 2D mesh.

4.3.1 Area results

Area figures (Table 1) were achieved with Cadence BuildGates logic synthesis tool targeting a 0.35µm

CMOS technology. We've noticed that the area figures reported by the synthesizer are not a fully linear function of the number of routers, this is due to the clock tree needed for ensuring the full synchronism at the chip level. The die area of the QoS Hermes is increased, due to the additional logic needed for managing arbitration and routing.

Table 1 – Area results in mm²

	Router		Total (8x8)
	logic	Buffers(flits)	
Input buffering	0,028	0,72(30)	48,46
Output buffering + QoS	0,39	0,72(30)	71,36
Output buffering	0,03	0,72(30)	66,6

4.3.2 Power consumption results

Power estimations have been carried out using the same tool which performs cycle-based simulations on the synthesized netlist (0.35µ CMOS technology). Those estimations have been conducted for a 100 % load (defined in 4). Results exposed in Table 2 represent the worst case in a 64 cores system where all cores are intensively sending and receiving data.

Table 2 – power consumption results

	with QoS	Without QoS
Input buffering	48 mW	40 mW
Output buffering	47,7 mW	40 mw

5. Conclusion

This work has shown the advantage of NoC structures in comparison with traditional approaches. The main drawback of NoCs often relies on the lack of quality of service, mandatory for numerous applications. That has been addressed through the implementation of QoS hardware support in the Hermes NoC.

The conducted experiments have shown that prioritizing some traffic can be done with an affordable area overhead while retaining most of the advantages of the NoC approach, like scalability and increased available bandwidth.

In response to the increasing difficulty to achieve global synchronism (section 1.2), we are currently developing a GALS (Globally Asynchronous Locally Synchronous) version of Hermes with a fully asynchronous communication protocol between cores (toggle-based). Employing that version removes the need of using a fully synchronous clock on the chip and therefore contributes to lower both area and power consumption. Future works also aim at implementing error detection/correction at different levels in the NoC, in response to the emerging problem of soft-errors in new silicon technologies.

6. References

[1] Lauwereins R. "Creating a world of Smart Re-configurable Devices", *Field Programmable Logic* FPL'2002, pp790-794.

[2] Rabaey J., "Busses and Networking", Computer Science 252, Spring 2000.

[3] Benini L; et al. "Powering Network on Chip", *ISSS'2001*, pp.33-38.

[4] Guerrier P., Greiner A., "A Generic Architecture for on-chip Packet Switched Interconnections" *Date'2000*, pp 250-256.

[5] Benini L; et al. "Network on Chips: A New SOC Paradigm". *IEEE Computer*, 35, Jan. 2002, pp70-78.

[6] Christopher, J. Glass and Lionel M. Ni, "The Turn for Adaptive Routing". Association for Computing Machinery ACM 1994, Vol 41, No 5, pp 874-902.

[7] Sgroi M.; and al. "Addressing the System-on-a-Chip Interconnect Woes Through Communication-Based Design" DAC 2001, pp667-672.

[8] Dally, W.J.; Towles, B. "Route Packets, Not Wires: On-Chip Interconnection Networks" DAC'2001, pp 684-689.

[9] Maresceaux, T.; Bartic, A.; et al. "Interconnection Network Enable Fine-Grain Dynamic Multi-tasking on FPGAs." FPL'2002. Sept. 2002; pp795-805.

[10] Leiserson C., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", IEEE Transactions on Computers, vol. C-34, no. 10, pp 892-901, October 1985.

[11] Greiner A., Andriahantenaina A. "Micro-réseau pour systèmes intégrés: réalisation d'un réseau SPIN à 32 ports", GDR CAO'2002, pp71-74.

[12] Karim,F.; Nguyen A.; Dey S.; Rao R ."On chip communication architecture for OC-768 network processors" 38th Design Automation Conference (DAC'01), Jun 2001, pp 678-683.

[13] Karim,F.; Nguyen A.; Dey S. "An interconnect architecture for network systems on chips", IEEE MicroV22(5), sept-oct 2002, pp36-45.

[14] Andriahantenaina A., "SPIN : a Scalable, Packet Switched, On-chip Micro-network", DATE'2003, pp.1128-1129

[15] Charlery H., "SPIN, un micro-réseau d'interconnexion à commutation de paquets respectant la norme VCI. Concepts généraux et validation ", SympAAA'2003, pp.337-344

[16] Moraes, F. G.; Mello, A. V. de; Möller, L. H.; Ost, L.; Calazans, N. L. V.. "A Low Area Overhead Packet-switched Network on Chip: Architecture and Prototyping.", IFIP VLSI SOC 2003, Darmstadt. International Conference on Very Large Scale Integration. 2003.

[17] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[18] Stevens, W., "TCP/IP Illustrated, Volume 1: The Protocols" Addison-Wesley, Reading, Massachusetts, 1994.

[19] Siemens, OMI 324: PI-Bus – Ver.0.3d. Munich, Siemens AG, 1994.