Mesh-based Network On Chip characterization:

A GALS approach

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 multiplication 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 architecture 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, this paper explores and discusses the impact of different architecture parameters on the achieved performance and presents an implementation of an asynchronous synchronization technique.

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 Existing communication architectures

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 present 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 [10] 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.

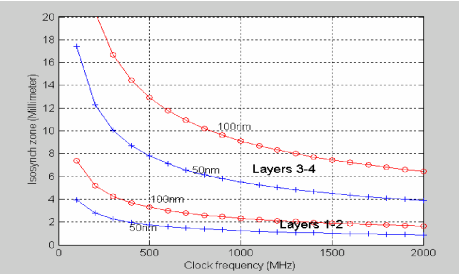

Figure 1 – Isosynchronous zone as a function of the frequency

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

- *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.
- Section 4 provides a synthetic analysis of the inter-clock domains communications problem and presents the implementation of a GALS version of the HERMES NoC.
- Section 5 is dedicated to the analysis of the impact of the GALS on different aspects, namely performance, power consumption and area.

2. NoCs: State of the Art

Several characteristics describe a Noc, we here only describe 2 main ones, for extensive inforation refer to [3], [5] and [18].

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] and network scalability. The different approaches are summarized in figure 2, refer to [18] for extensive information on each topology.

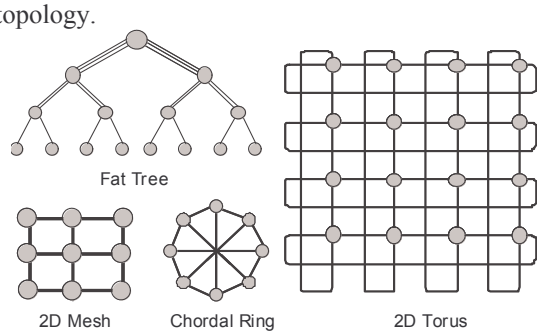

Figure 2 *– Exiting NoC topologies*

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 given 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 split 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 a lowered buffer 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 for some reason.

3. Hermes architecture

3.1 Overview

The Hermes project aims at defining a flexible environment allowing rapid evaluation of different

NoC realizations. 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 that was selected for our studies.

The Hermes network is based on a 2D Mesh switch (i.e. router) array [18]. 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.

Morever the switching mode routing algorithm used is based on whormole approach, allowing to decrease the latency.

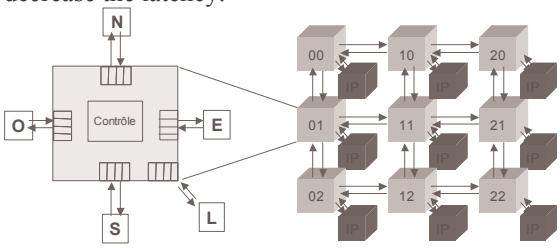

Figure 3 *– HERMES topology overview*

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.

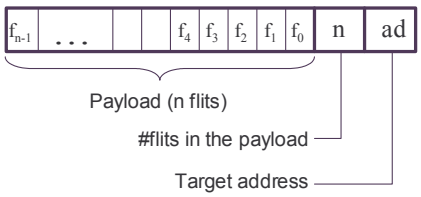

Figure 4 *– packet format*

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.

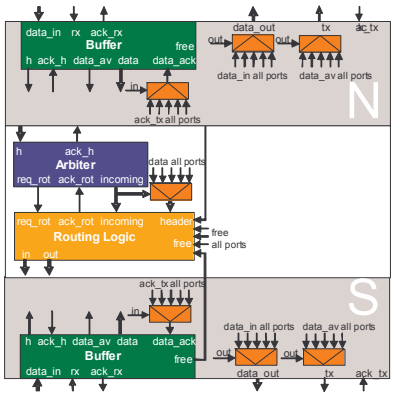

Figure 5 *– Partial block diagram of the switch, showing two of the five ports*

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 eight has been used for prototyping purposes.

4. GALS in a NoC

As briefly described in section 1, global synchronism won't be achievable anymore in future silicon technologies. The major challenge behind the GALS clocking scheme is the communication between different clock domains. We will consider in the rest of the discussion that no assumption at all can be made on the clock frequencies. The critical issue that must be addressed is *metastability*: the clock of the latching register and the data may switch simultaneously. The register output then settles into an undefined region – neither a logical high nor a logical low. Several solutions have been proposed to alleviate this problem [4], [15], [14], [11]. The quality of the solution depends on the mean time-to-failure. The common solution recommended by most ASIC designer guidelines is the use of at-least 2 synchronization stages for crossing clock domains, and also use metastability-hardened flip-flops whenever available on the target technology. The protocol to be used between the two clock domains is also important. Since no assumption at all were made on the frequencies, no common time basis can be taken into account. Therefore, all simple handshake-based protocols generating pulses (with the implicit notion of pulse duration) cannot be used. It is mandatory to use an edge-based protocol, like the Toggle protocol (figure 6). This protocol uses two toggle signals for the synchronization, a given data being considered as valid when a toggle is detected. When the data is latched, another toggle

is sent back to the sender to notify the acceptance (figure 6).

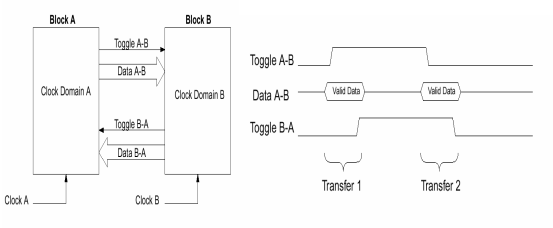

***Figure 6** – Crossing clock domains using the Toggle protocol*

Figure 7 depicts the used router synchronization system implementation. Notice that a flip-flop has been inserted in order to filter the glitches that might trigger the SR latch.

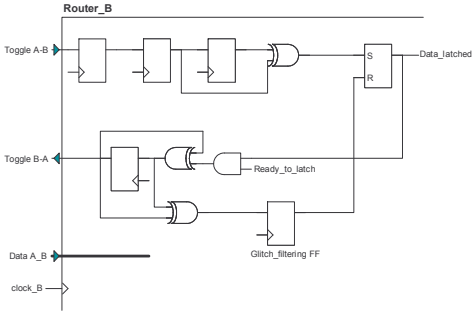

***Figure 7** – Synchronization structure*

This synchronization ensures efficient and reliable communication between unrelated clock domains. Electrical simulations have demonstrated the efficiency of this approach, still running from GHz down to kHz.

Figure 8 depicts a simple example illustrating how the NoC behaves in GALS mode. Only 3 routers are used, Routers A and B share the same frequency (500 MHz) as Router C is clocked at a much slower frequency (714kHz). The timing diagrams are given for Router B which has to handle the hard part of the job (from 500MHz down to 714kHz).

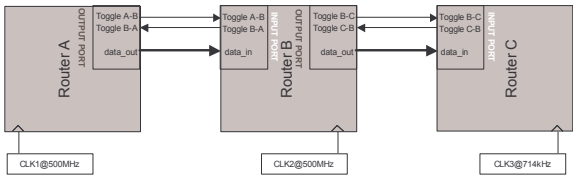

***Figure 8** – The GALS case study*

Figures 9 is a timing diagram covering the first 50 ns. During this period the Router is receiving the data at 500MHz frequency. Since a 5-positions only buffer is used, after a few cycles no more flit coming from Router A can be acquired. Figure 10 gives the same timing diagram at a higher time scale: Router C accepts the flits one by one, at 714kHz frequency. Each time a flit is sent to Router C a position is freed in the buffer; so a new flit coming from Router A is latched. According to figure 9, the first flit (x01) is the target router address, the second one (x08) the number of flits composing the payload(x70, x71, x72, etc.).

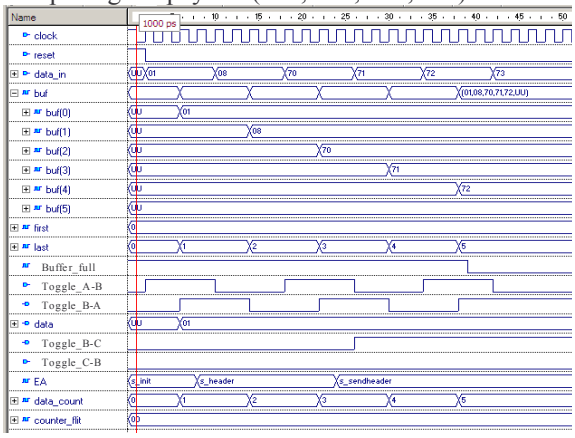

***Figure 9** – Router B Timing diagram (1/2)*

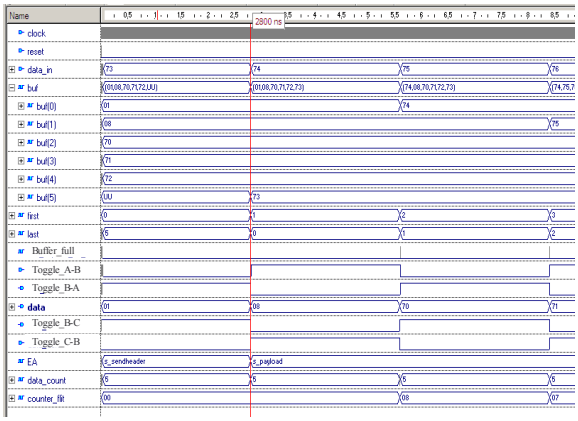

***Figure 10** – Router B Timing diagram (2/2)*

5. Results

Throughout this section we will mainly consider two structures, the original HERMES NoC and the aforementioned GALS realization. Both architectures exhibit comparable performances when all router/core couples are homogeneously clocked. Hence we will deliberately won't distinguish the two architectures in the first subsection and present the behavior under latency and bandwidth considerations. The second part of this section analyses the impact of GALS in terms of area and power consumption. All results are given for a 5x5 HERMES NoC.

5.1 Architecture model performance

5.1.1 Buffer sizing impact

Buffer sizing impacts on performance to deliver the message. Buffers featuring more than 6 positions (flits) enable compensate the routing time. This is due to the fact that the logic in charge of arbitration and routing in a router has a 6-cycles latency. Therefore no latency occurs when a new packet arrives in the router, the routing process taking place concurrently with the expedition of the last flits of the previous packet (full pipeline).

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). Table 1 and 2 show that data transfers occur faster for larger buffer sizes.

5.1.2 Latency, buffer sizing and performances under random traffic conditions

This experiment analyzes the NoC behavior under heavy random traffic conditions. Two different buffer sizes were used: 8 and 16 positions. Table 1 and Table 2 present the traffic results related to 500 packets randomly sent through the network, all results are expressed in clock cycles. Each initiator sends 20 packets composed of 39 *flits* to random targets. The two most relevant parameters are the *average* time to deliver a packet (first line), associated to the packet latency, and the *total time* to deliver all packets (last line), associated to the NoC throughput.

***Table 1** – Simulation results for a random traffic using 8 position buffers*

	Buffer size = 8			
	Traffic 1	**Traffic 2**	**Traffic 3**	**Average**
Average	260	275	271	**268**
Std. Deviation	170	199	167	**179**
Minimum	89	89	100	**93**
Maximum	1305	1618	1221	**1381**
Total Time	5346	5559	5142	**5349**

***Table 2** – Simulation results for a random traffic using 16 position buffers*

	Buffer size = 16			
	Traffic 1	**Traffic 2**	**Traffic 3**	**Average**
Average	312	324	326	**321**
Std. Deviation	203	208	201	**204**
Minimum	89	89	100	**93**
Maximum	1225	1644	1385	**1418**
Total Time	4686	5088	4908	**4894**

Note that some packets stay in the network for a much longer time (fourth line – *maximum*). This may arise if a set of packets is transmitted to the same target or simply because of random collisions. Further analysis of these data is under way, in order to develop adequate traffic models and associated switching algorithms to reduce this problem.

As in a pipeline, with an additional buffer capacity the latency increases (as mentioned before) and the throughput is improved (8% in this experiment). This improvement in throughput is due to the reduction in the network contention, since blocked flits use less network resources while waiting to be routed. The results indicate that buffers dimensioned with values near the minimum size for improving latency (6, in the this case as mentioned previously) represent a good trade-off between latency and throughput while keeping area consumption small.

Hermes versus an ideal BUS

Consider an ideal bus, able to send one word per clock cycle. Since the total number of words to be transmitted is 19500 (500 packets of 39 flits), 19500 clock cycles would be necessary to transmit all data.

The presented version of Hermes was able to handle that in roughly 5000 cycles. We estimate one order of magnitude of gain in performance for a real bus architecture.

5.1.3 Average latency

For this experiment we use a 4x4 switch array. We consider eight initiator and eight targets.

The *load* offered by a given simulated traffic is defined as the percentage of the channel bandwidth used by each communication initiator [ref][i]

If we consider that G is the time between 2 requests transmitted the load is defined by the ratio L/(L+GM) where L is the tail of the packets and GM the average of G (for a load of 100%, GM is equal to zero).

For this experiment, 26 000 packets of 16 flits have been considered. The figure 11 shows the latency (number of clock cycles) obtained for the PI Bus, the SPIN network and the HERMES NoC.

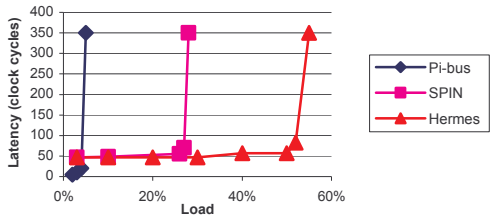

Figure 11 : latency as a function of load

We notice that saturation occurs later for Hermes when compared to busses, as expected. The SPIN network [4] performs better than Hermes, mainly thanks to its fat-tree topology which provides a smaller average distance between cores. This is however achieved to the detriment of the scalability as depicted in figure 2, since the fat-tree topology is less regular and implies to implement links of different capacities (function of the hierarchy level).

5.2 HERMES GALS results

Post-synthesis simulations have been performed for a wide range of frequency scenarios and have shown correct behaviour of the system. Electrical simulations (post-layout) are currently performed on a 0.35µ CMOS process to ensure correct operation of the proposed structure at the electrical level. Table 3 summarizes the obtained area and power consumption figures (carried out with Cadence tools).

As expected, the average latency of the GALS structure is greater, mostly due to the additional synchronization logic added. Similar simulations have been performed in order to ensure correctness of the behavior. When all the routers are clocked at the same frequency, latency curves are shifted up

from a few tenth cycles, while the bandwidth remains the same. The silicon area is also greater, an overhead of 10% when compared to the original NoC has been evaluated using logic synthesis tools (Cadence BuildGates). No placement/routing has yet been done, but since there are only systolic (local) connections between routers, no huge mismatch is expected, neither in area nor in power consumption.
Results are summarized in Table 3 for transferring 96 packets across a 3 routers-long path, for a 5x5 network clocked at 50 MHz. No other communication is taking place in the NoC.

Table 3 *– Area and power consumption overheads*

	Area	Energy	Average Latency
Hermes (fully synchronous)	0.21 mm²	323 nJ	51cycles
Hermes GALS	0.25 mm²	364 nJ	60.16 cycles

5.2 Implementation results
A first prototype of this NoC has been implemented on a Xilinx Virtex II FPGA, with 8 flits buffers. Results are summarized in the table 4. The silicon area overhead seems acceptable.

Table 4*– Hermes NOC module area report for FPGA and ASIC*

	FPGA Xilinx Virtex II		**CMOS 0.35µm**
	LUTs	**FFs**	**ASIC (gates)**
1 Switch	631	200	2930
SR	193	233	1986
Serial	91	93	752
Total	590	596	4587

5. Conclusion

This work has shown the advantage of NoC structures in comparison with traditional approaches.
A simple 2D Mesh realization providing a low area overhead as well as high performance has been presented and characterized. Furthermore, we have shown that providing an asynchronous interface between routers enables the use of completely unrelated frequencies on each core. This GALS version features comparable performances with an affordable cost overhead in terms of performance and silicon area.
Ongoing research works aim at:

- Exploring alternative GALS implementations and corresponding benefits for SoCs. We're currently investigating the benefits of using independent and dynamic clock frequency adjusting on each Router/core. Expected benefits in terms of power consumption are promising
- Providing Quality of Service support. Applications with hard real time or multimedia constraints need QoS enabled NoCs. We're exploring the possible solutions allowing to guarantee either latency or bandwidth. For that, bandwidth reservation, time division multiplexing techniques are evaluated.

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] J. Liang, S. Swarninathan, R. Tessier, "aSOC: A scalable, single chip communications architecture", International Conference on parallel architectures and compilation techniques, oct 2000, pp 37-46
[9] Dally, W.J.; Towles, B. “Route Packets, Not Wires: On-Chip Interconnection Networks” DAC’2001, pp 684-689.
[10] Maresceaux, T.; Bartic, A.; et al. “Interconnection Network Enable Fine-Grain Dynamic Multi-tasking on FPGAs.” FPL’2002. Sept. 2002; pp795-805.
[11] Leiserson C., “Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing”, IEEE Transactions on Computers, vol. C-34, no. 10, pp 892-901, October 1985.
[12] Greiner A., Andriahantenaina A. “Micro-réseau pour systèmes intégrés: réalisation d’un réseau SPIN à 32 ports”, GDR CAO’2002, pp71-74.
[13] 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.
[14] Karim,F.; Nguyen A.; Dey S. “An interconnect architecture for network systems on chips“, IEEE MicroV22(5), sept-oct 2002, pp36-45.
[15] Andriahantenaina A., “SPIN : a Scalable, Packet Switched, On-chip Micro-network”, DATE’2003, pp.1128-1129
[16] Charlery H., “SPIN, un micro-réseau d’interconnexiotn à commutation de paquets respectant la norme VCI. Concepts généraux et validation ”, SympAAA’2003, pp.337-344
[17] Dall'Osso M., Biccari G., Giovannini L., Bertozzi D., Benini L. "xpipes: a Latency Insensitive parameterized Network-on-Chip Architecture For Multi-Processor SoCs", pp. 536-539, Proc ICCD 2003
[18] 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.