Multistage Interconnection Network for MPSoC: Performances study and prototyping on FPGA

B. Neji [1], Y. Aydi [2], R. Ben-atitallah [3],S. Meftaly [4], M. Abid [5], J-L. Dykeyser [6]

[1] CES, National engineering School of Sfax , Tunisia. Email: neji.bilel@gmail.com.
[2] CES, National engineering School of Sfax , Tunisia. Email: yassine.aydi@oous.rnu.tn
[3] INRIA, University of Lille, France. Email: rabie.ben-atitallah@lifl.fr
[4] INRIA, University of Lille, France. Email: samy.meftali@lifl.fr
[5] CES, National engineering School of Sfax , Tunisia. Email: mohamed.abid@enis.rnu.tn.
[6] INRIA, University of Lille, France. Email: dekeyser@lifl.fr

Abstract—**Multiprocessor System on Chip is a concept that aims to integrate multiple hardware and software in a chip. Multistage Interconnection Network is considered as a promising solution for applications which use parallel architectures integrating a large number of processors and memories. In this paper, we present a model of Multistage Interconnection Network and a design of prototyping on FPGA. This enabled the comparison of the proposed model with the full crossbar network, and the estimation of performance in terms of area, latency and energy consumption. The Multistage Interconnection Networks are well adapted to MPSoC architecture. They meet the needs of intensive signal processing and they are scalable to connect a large number of modules.**

***Key wordss* — MPSoC, NoC, MIN, FPGA.**

I. INTRODUCTION

Technological evolution enables the integration of billions of transistors on a chip. This allows an efficient exploitation of resources and increasingly complex and varied design [1]. The Multiprocessor Systems on Chip (MPSoC) represent a new paradigm emerged from this development. They can include several heterogeneous components like processors (Scalar, Risc, VLIW…), memories, hardware accelerators (DMA, DCT, FFT…) and peripheral input/output. The communication between these various components is provided by an interconnection network called Network on Chip (NoC).

A NoC can be characterized by topology, routing algorithm, arbitration technique, the area used on the platform, latency and energy consumption. We target in this work intensive signal processing applications, requiring a lot of parallelism in the calculation. Multistage Interconnection Networks (MINs) have been used in classical multiprocessor systems. As an example, MINs are frequently used to connect the nodes of IBMSP [2] and CRAY Y-MP series [3]. Further on, MINs are applied for networks on chip to connect processors to memory modules in MPSOCs [4]. These architectures provide a maximum bandwidth to components (processors, DSP, IP...), and minimum delay access to memory modules.

Performance evaluation is a key step in any MPSOCs design and especially of the selected communication architecture, allowing for decisions and trade-offs in view of system optimization. It is determined by modeling, using simulation [5], formal methods in which we use model checking and/or theorem proving techniques to verify instances of MINs [6] or direct execution which consists in the use of programmable logic circuits for fast performance evaluation [7].

A design prototype and performance analysis of MINs for MPSOC architectures is investigated in this paper.

The next section presents a state of the art of some industrial and academic NoC and presents some research on MIN. Section III introduce the MIN proposed model for MPSoC. Performance estimation and comparison between MIN and full crossbar network are presented in section IV. Section V introduces the results found by Integrating MIN in MPSoC on FPGA. Finally, Section VI concludes this work and gives perspectives.

II. STATE OF THE ART

The Network based on shared bus is very interesting for the industry because it is a relatively cheap and simple to implement. However, it has drawbacks that will no doubt crucial in the coming years.

With the complexity of applications and the increase of the modules number, the shared bus can not solve these problems. Therefore, we have to use new forms of interconnection networks for MPSoC. The idea of applying techniques from networks between computers to NoC has emerged since 2000 [8], [9], [10]. Thus, the problem of network on chip has been transformed into the problem of modules network with specifics functions; it is a prevalidated module connection. In this context, several NoC have been developed at industrial and academic. SPIN (Scalable Programmable Integrated Network) is an example of NoC which was originally developed at LIP6 [11]. The nodes are connected through routers R-SPIN [12]. Nostrum is a NoC developed by the LECS (Laboratory of Electronics and Computer Science) within the Royal Institute of Technology in Suède3. This network has a regular topology based on tow dimensions mesh [13]. Xpipes is the result of the DEIS laboratory work (Department of Electronics and Computer Science) at the University of Bologna in Italy. It is a combination of soft macros written in SystemC5 witch can be adapted to any type of application (through a generic router) [14]. The MINs are often used to connect a large

number of modules. The work in [15] present a model of reconfigurable MIN described in SystemC and an estimation of it performance. The aim of this work is to propose and develop on FPGA MPSoC architecture with a MIN network and to assess the performance of the design in terms of area, latency and energy consumption.

III. MODEL OF A MIN FOR MPSOC

The multiprocessors architectures become an inevitable solution for intensive signal processing applications. The performances of calculation of such system are proportional to the number of processors it contains, the individual performance of these processors and access times to data in memory. That memory can be shared or distributed. The access of data from processors to memory always passes through an interconnection network. It follows that the overall performance of the machine can be degraded by the NoC. An interconnection network is defined by its topology, its routing algorithm, its switching strategy and mechanism for flow control.

In this context, MIN networks are proposed to connect a large number of processors to establish a multiprocessor system.

A. Topology

The MIN networks are based on routing and arbitration circuits conducting connections between their inputs and outputs (in our case boxes of size 2x2). The entire network contain n (n=log2N) floors, each one with N/2 routing and arbitration circuits and one connection block. The usefulness of the connection blocks is to define a MIN type like Omega, Butterfly, Baseline, and so on. As a result, we just change the connection blocks for a new type of MIN.

B. Routing algorithm

The routing algorithm allows nodes to direct a message to its destination. A message is composed of a destination address and the data to be transmitted.

To achieve the routing in MIN, we used the self routing algorithm; it's a simple algorithm that depends only on the destination address. The principle of this algorithm is: the Ind bit of the destination address determines the activation switch on the floor i. Thus, if this bit is 0, the road passes through the switch top. Conversely, if the bit is 1, the bottom of the switch is used. The floors are numbered from the source to the destination.

C. Technical arbitration

The network must be able to manage conflicts way (two messages simultaneously wishing to use the same channel), so it is able to store temporarily (using FIFOs) a message to be send further once Track is free.

The used technical arbitration is Round Robin. It is based on the following principle: the port of entry with the highest priority will be in the next iteration the lowest priority. For this, we used a static variable (it takes the values 1 or 0) in the case we have two complaints that arise and seeking access to the same output port.

D. Prototyping on FPGA

The used prototyping platform is an Altera Stratix II EP2S60. The advantage of using FPGA is essentially motivated by their reconfiguration and the ease and speed of implementation. Two types of networks have been addressed in our work: the MIN and the full crossbar network. As a result, we can show the effectiveness of MINs for MPSoC and compare the performance with the full crossbar network.

We have developed a generic code of a MIN for N processors and N memory using VHDL language and Altera Quartus II [16]. The following figure represents an example of MIN (size: 8x8) described at RTL level.

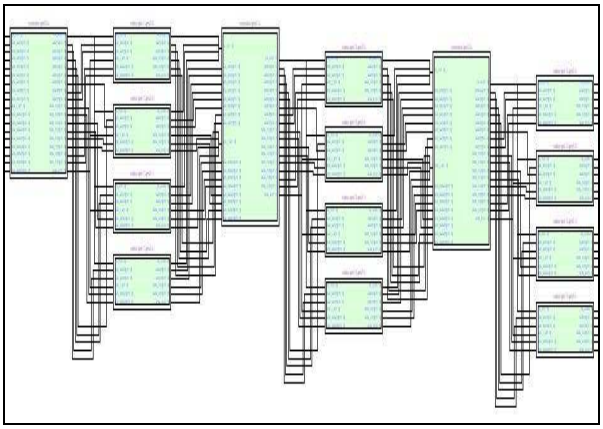

Fig.1 Multistage interconnection network (8x8)

The resources used in the FPGA card for this network (MIN 8x8) are:

5% of logical elements: 603 / 48,352 ALU and 2,017 / 48,352 logics register.

2% of memory blocks: 49,152 / 2,544,192 blocks.

We treat the result of other MIN sizes and we compare their performance against the networks full crossbar.

A full crossbar network allows simultaneously connecting any pair of nodes unoccupied. Practically, this type of network used to connect a limited number of processors and memory.

We have described this architecture in VHDL and we simulated it using Altera tools. The network is composed mainly of n routers and n arbitrators. Each router can connect a node (CPU) to one of the n memories. The role of the arbitrator is to connect one signal among those who wish to access at the same time to a memory. The technical arbitration that was used is the round robin. In order to don't lose expectations data; we made in each entry of an arbitration module a FIFO (First In First Out memory).

The figure 2 shows a schematic of a full crossbar network (8x8). It was generated from the code using Altera RTL Viewer [16]. We note the complexity of interconnections in this type of network.

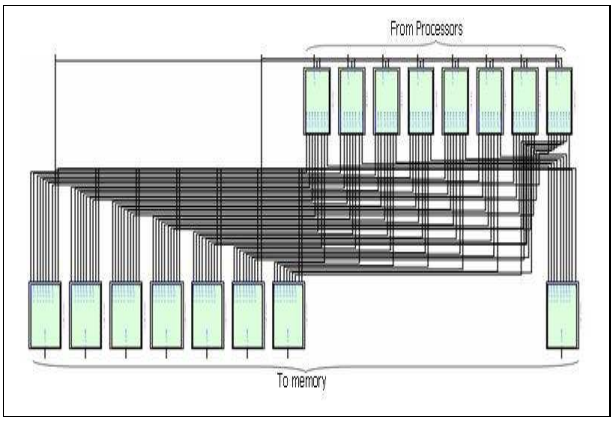

Fig.2 Full crossbar network (8x8)

The performance of this network will be compared with the MIN performance and more precisely the omega network.

IV. PERFORMANCE ESTIMATION AND COMPARISON

After the implementation of the Omega and full crossbar network, we identified the following performance depending on their size (number of processors and memory to connect):

- The area on FPGA (number of logical elements).
- Analysis and synthesis time.
- Latency.
- The consumption.

These performances have been identified using Altera Quartus II and Modelsim.

A. Area on FPGA

The area of interconnection depend on the topology (topology weakly connected will have a area smaller than topology completely connected), located services (more the established mechanisms are complex and numerous, more resources area is important) and the size of buffers included in the routing resources.

We note that the area of a full crossbar network is increasingly important as MIN while increasing the size of two networks (Figure 3).

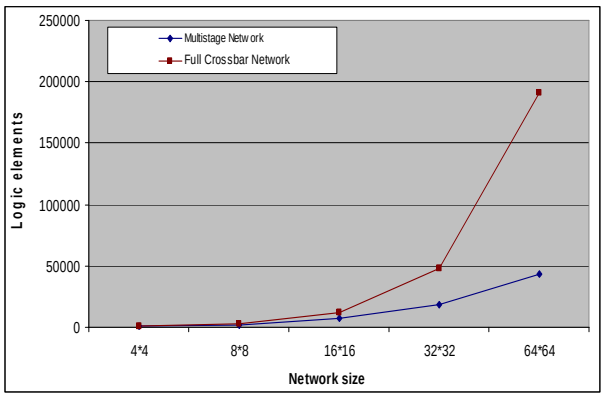

Fig.3 Area on FPGA

As a result, MIN are performing in terms of area to connect a large number of processors to memory.

The area of MIN is proportional to Nlog2N, compared to N2 for full crossbar networks.

B. Analysis and synthesis times

The analysis and synthesis represent the first phase in the design flow of the Altera Quartus II environment. The running time of this stage depends mainly on the power of the tool (the version of the tool), the target platform and performance of the machine that runs the software (CPU frequency, size of memory, etc.). The analysis and synthesis time (calculated in seconds) can give an idea about the complexity of design, more time, the greater the complexity of the circuit is great. In our case, we want to have an idea about the complexity of two networks types, which are MIN and the full crossbar network.

The following table represents the time of analysis and synthesis for the two networks.

TABLE I
ANALYSES AND SYNTHESIS TIME FOR MIN AND FULL CROSSBAR NETWORKS

	4x4	8x8	16x16	32x32	64x64
MIN	29	56	151	432	2428
Full crossbar network	25	60	235	Hours	Hours

We note that increasing the size of networks, the analysis and synthesis time for the full crossbar is more important than MIN. As a result, we see that the architecture of full crossbar networks is very complex. This complexity is a limit for the network capacity, so it can connect a few tens of nodes (processors and memory). The networks MIN overcome this constraint and can be used to connect a large number of nodes, hence the importance of these networks.

C. Latency

The latency is the time to transfer a package from a transmitter to a receiver. We distinguish between two types of latency: minimum latency and maximum latency.

In the simulation, we considered that the processors are blockers. The period of the clock is 10ns (P). As a result, we found the following results (Figure 4):

- For MIN network, minimal latency is (3 * log2 (n)) * P and maximum latency is (3 * log2 (n) + n-1) * P, with n the number of processors to connect the memories, P the clock period and n-1 is the maximum number of conflict in the network, knowing that the transfer of a package in each stage lasts 3 clock cycles.
- For full crossbar network, the minimum latency is 3 * P and the maximum latency is (3 + n-1) * P.

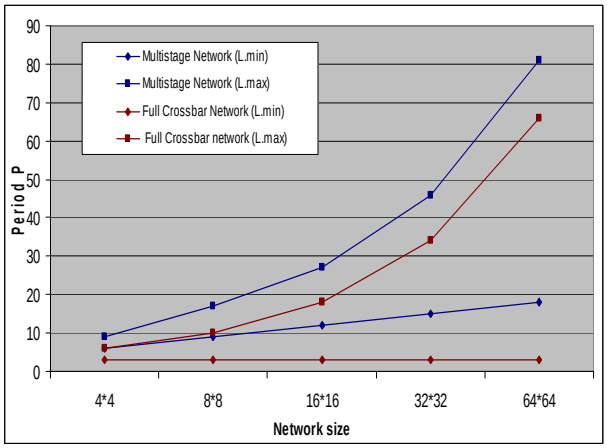

Fig.4 Latency for MIN and Full crossbar networks

According to the figure, we note that the full crossbar network is a little more efficient than the MIN network in terms of latency. But MIN still more interesting as we want to interconnect a large number of node.

D. Energy consumption

In this part, we are interested to present some results in terms of energy consumption for MIN while increasing the size of the network. The Figure 5 shows the results for the same application (with different sizes of NoC) using the Altera power analyzer tool.

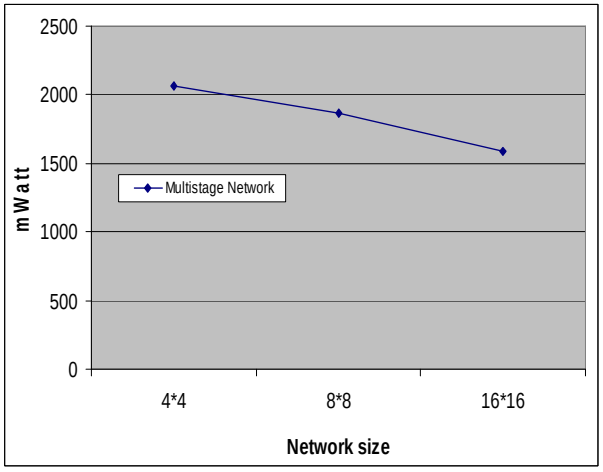

Fig.5 Energy Consumption for MIN network

Based on these results, we find that energy consumption is reducing while increasing the size of the network. By increasing the size, the running time of the application and the use of the clock decrease. As a result, energy consumption decreases. In this case, we should reduce the use of clocks and therefore use combinatorial blocks (asynchronous), but without degrading the frequency of the circuit.

V. APPLICATION: INTEGRATION OF MIN IN MPSOC

To validate the developed MIN, we integrated the it in an MPSoC architecture and we felt the performance of the architecture composed of processors, memory, MIN…

A. Proposed architecture

Our scope is the intensive signal processing. This requires real-time constraints. As a result, the trend is moving to use parallelism in the execution of application.

In this context, the proposed architecture contain mainly N miniMIPS processors [17], N data memories, N instruction memories of and a NoC composed of two networks, one to send queries and another to receive Answers (Figure 6).

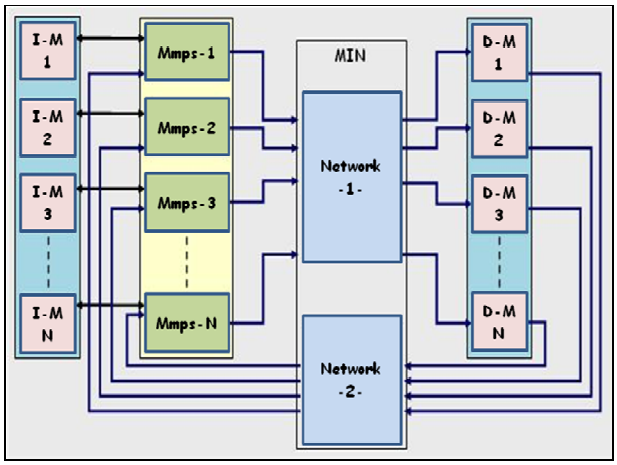

Fig.6 Proposed MPSoC architecture

All processors operate in parallel; each one is linked directly to an instruction memory through local bus connection. On the other hand, each processor can access to any data memory across the MIN to read information or write data. The role of MIN is to manage access to data memories and avoid conflicts.

B. Implementation on FPGA

After developing all components, we specified in VHDL a generic multiprocessor architecture for n processors, n memories and a MIN (omega) network.

The area available on FPGA is a very important constraint, so we have made analyses and syntheses of our generic architecture code for different values of n (number of processors) and we got the percentages of used FPGA resources (circuit EP2S60) as shown in the table below.

TABLE II
PERCENTAGES OF USED FPGA RESOURCES

n value	Nb of logic blocks	Nb of DSP blocks	Nb of memory blocks
4	28 %	24 %	16 %
8	53 %	44 %	21 %
16	102 %	79 %	34 %

According to the results, we can implement on FPGA an MPSoC architecture composed of 8 processors, 16 memories, an interconnection network, and so on.

At this stage, the architecture is complete, it remains to develop an application to test and validate the proper functioning of processors and the smooth delivery of data in our network.

C. MIN validation in MPSoC

To check our architecture, we developed in assembly language an application that is the product of two matrixes (NxM).

Each processor multiplies m lines of the first matrix with the second matrix. The assembly code is converted into binary code and exactly MIF format (Memory Initializing File).

As a result, all processors perform their instructions from their instruction memories. The memory initialization was made with the Altera Mega Wizard tool. The result is a matrix; it is distributed on data memories with our choice.

For matrix 8x8, the simulation shows that the last request (writing in a data memory) was done at the 17 micro seconds. As a result, the running time of the application on the proposed MPSoC architecture is 17 micro seconds.

Different simulations have been made for MPSoC architectures with different values of n and using the same application (Figure 7).

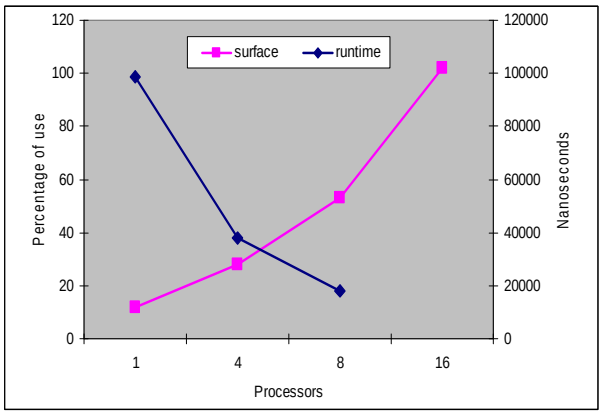

Fig.7 Compromise area / runtime

We note that there is a compromise area / runtime we have a space exploration and we should be located according to the needs of the application. Our case requires real-time constraints, so we should use the maximum of processors in MPSoC architecture, we will win in the execution time but the area and the cost increase.

D. Estimated performance of different types of MIN in MPSoC

In this part we are interested to use different types of MIN in the MPSoC architecture. Then we compare the number of blockages (conflicts), and the execution time of an application.

We treat omega, butterfly and baseline networks, the difference between these networks is the interconnection between floors. The scenario to be carried out is as follows: After reading instruction, all processors send requests (to recover an element of matrix) to the data memories across the MIN network. There will be activation of arbitrators and routing modules, thereafter routing data from the queues to the appropriate files, applications blocked pass in the next cycles. Then the memories are activated and a test takes place. When it comes to reading, memory takes care to send the required data to processors with an acquittal, and when it comes to writing, recording data is carried out with an acquittal processor.

A conflict in a Delta network occurs when tow messages want to use the same output channel. A message can be blocked if he tries to use the same route on a switch when another message is in transit.

The table below shows the simulation results of different MIN networks using the multiplication of two matrix application.

The total number of blockages (total Bl) is the sum of the number of request network (Bl in N1) and response network (Bl in N2) blockages.

TABLE III
BLOCKAGES NUMBER AND RUNTIME USING DIFFERENT TYPES OF MIN

Network	Bl inN1	Bl inN2	total Bl	runtime
Omega	349	193	542	17040
Butterfly	236	257	483	15125
Baseline	135	372	507	15827

We note that the Butterfly network is the most efficient network for the used application; it shows the smallest number of blockages, and a runtime less weak.

The performances of different types of MIN vary from one application to another, so we should find a technique to improve the performance of Delta networks. A solution is the use of dynamic reconfiguration of the architecture connection blocks, so the appropriate type of network will be used.

VI. CONCLUSION AND PROSPECTS

The complexity and growing number of heterogeneous modules in MPSoC motivated designers to consider new interconnection architectures. In this context, MIN represents a solution to increase the performance of communication and thereafter overall system performance. We have shown that they use fewer resources on FPGA compared to other network topologies (Full crossbar network.), and they are effective in terms of runtime compared to other types of topologies. Then, we integrated the MIN in MPSoC and we implemented it on FPGA. Finally, we compared the performance of different types of MIN in terms of number of conflicts between packages and the runtime of a given application.

One of the future work is to use the dynamically connection blocks reconfiguration in MPSoC architecture in order to use the type rather appropriate as required by the application.

VII. REFERENCES

[1] P.M. Zeitzoff and J. E. Chung, A perspective from the 2003 ITRS, IEEE circuits and devices magazine, February 2005.

[2] B. Stunkel, D. G. Shea, B. Aball, M. G. Atkins, C. A. Bender, D. G. Grice, P. Hochschild, D. J. Joseph, B. J. Nathanson, R. A. Swetz, R. F. Stucke, M. Tsao, and P. R. Varker, "The SP2 High-Performance Switch", IBM Systems Journal, Vol. 34, N° 2, IBM Corp., Riverton, USA, 1995, pp. 185–204.

[3] T. Cheung, and J. E. Smith, "A simulation study of the CRAY X-MP memory system", IEEE Transactions on Computers, Vol. 35, N° 7, IEEE Computer Society, Washington, 1986, pp. 613–622.

[4] S. Duquennoy, S. Le Beux, P. Marquet, S. Meftali, and J-L. Dekeyser, "MpNOC Design: modeling and simulation", In 15th IP based SOC Design Conference (IP--SoC 2006), 2006.

[5] Y. Aydi, S. Meftali, M. Abid, and J-L. Dekeyser, "Dynamicity Analysis of Delta MINs for MPSOC Architectures", STA'07, 2007.

[6] Elleuch M., Aydi Y., Abid M., « Formal Specification of Delta MINs for MPSOC in the ACL2 Logic », in Proceedings of Forum on Design and specification Languages - FDL '08, 2008.

[7] Christophe Bobda and Ali Ahmadinia. Dynamic Interconnection of Reconfigurable Modules in FPGA, In IEEE Design & Test of Computers – Special Issue Networks on Chip. Vol. 2, No. 25, pp. 443-451, September-October 2005.

[8] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli. Addressing the system-on-a-chip interconnect woes through communication based design. IEEE / ACM Design Automation Conference, 2001.

[9] W. J. Dally and B. Towles. On-chip interconnection networks. IEEE / ACM Design Automation Conference, 2001.

[10] L. Benini and G. De Micheli. Networks on Chips: A new SoC Paradigm. IEEE Computer, January 2002.

[11] A. Andriahantenaina and A. Greiner. Micro-network for SoC: Implementation of a 32-port SPIN network. IEEE / ACM International Conference on Design, Automation and Test in Europe, Germany, 2003.

[12] P. Guerrier and A. Greiner. A Generic Architecture for On-chip Packets witched Interconnections. IEEE / ACM International Conference on Design, Automation and Test in Europe, France, 2000.

[13] Z. Lu, R. Thid, M. Millberg, E. Nilsson, and A. Jantsch. NNSE: Nostrum networkon-chip simulation environment. Swedish System-on-Chip Conference, Sweden, 2005.

[14] D. Bertozzi and L. Benini. Xpipes: A network-on-chip architecture for gigascale systemson-chip. IEEE Circuits and Systems, 2004.

[15] Y. AYDI, MEFTALI S., M. ABID and J. DEKEYSER: Design and Performance Evaluation of a Reconfigurable Delta MIN for MPSOC. In 19th International Conference on Microelectronics (ICM'07), Cairo, Egypt, December 2007

[16] Altera: Home page, http://www.altera.com/, 2007.

[17] miniMIPS OpenCores: miniMIPS overview. http://www.opencores.org/projects.cgi/web/minimips/, 2006.