Design Space Exploration Comparing Homogeneous and Heterogeneous Network-on-Chip Architectures

Márcio Kreutz kreutz@inf.ufrgs.br
César A. Marcon marcon@inf.ufrgs.br
Luigi Carro carro@eletro.ufrgs.br
Flávio Wagner flavio@inf.ufrgs.br
Altamiro A. Susin susin@eletro.ufrgs.br

Universidade Federal do Rio Grande do Sul – UFRGS/PPGC
Av. Bento Gonçalves, 9500. Prédio 43412 / Bloco IV - CEP: 91501-970 - Porto Alegre - RS – BRAZIL

ABSTRACT
Networks-on-chip (NoCs) are communication architecture alternatives for complex Systems-on-Chip (SoCs) designs, due to their high scalability and bandwidth. In this paper, we consider a heterogeneous NoC as an alternative to match performance and energy requirements for dedicated applications. By employing an optimized mix of different routers, a heterogeneous network optimized for latency and energy consumption is achieved. A dedicated data structure, the Application Communication Pattern (ACP), models the application, enabling the specification of the communication requirements among cores, together with their execution performance. ACP allows fast analysis, helping the system designer to evaluate the communication performance of a NoC-based system; this performance strongly depends on the placement of the cores, and it is computationally hard to find the optimal placement. An optimization algorithm mixes different router architectures - composing a heterogeneous NoC - and finds optimal placements for application cores. Therefore, a heterogeneous NoC can be achieved, which complies to the application requirements with minimum latency and energy, enabling one to obtain the Pareto curve relating latency and energy for a given application.

Categories and Subject Descriptors
B.7.1 [**Integrated Circuits**]: Types and Design Styles – *advanced architectures, algorithms implemented in hardware, VLSI (very large scale integration).*

General Terms
Design, Performance, Experimentation, Theory.

Keywords
Systems-on-Chip, Networks-on-Chip, Mapping and Optimization algorithms.

1. INTRODUCTION
Future Systems-on-Chip (SoCs) containing billions of transistors will allow the development of new applications, which will work in a distributed way and require reusable communication architectures offering scalable bandwidth and parallelism. Network-on-Chip (NoC) emerges as a potential tile-based architecture to meet such requirements. NoCs are communication infrastructures composed by a set of routers interconnected by communication channels, which can provide asynchronous communication between synchronous domains. An important issue for NoC-based system designers is to devise a solution of the core placement problem in order to satisfy the communication requirements.

In the upcoming years, a NoC is expected to accommodate more than 10 x 10 tiles [1]. The efficient implementation of tile-based architectures requires efficient strategies for mapping particular cores into tiles. The search for appropriate models and algorithms for this placement problem becomes mandatory. This paper introduces the *Application Communication Pattern* (*ACP*), a model of computation that enables to capture not only the amount of communication, but also the communication ordering, derived directly from the application itself. ACP enables the evaluation of latency and energy consumption in different NoC topologies.

This work presents a heterogeneous NoC called SoCINhet, which is based on the SoCIN network [2]. Its heterogeneity derives from the nature of the cores and from the architectures of the routers, since three different routers – Rasoc [2], Mago and Tonga – implement SoCINhet. These routers target different design constraints: Rasoc reaches higher performance, paying the price of more energy consumption and area; Mago reaches the minimum energy consumption, with the direct impact of higher latency. Finally, Tonga was designed to present a compromise between energy consumption and latency. A heterogeneous NoC can be developed by mixing these three routers. However, to achieve high quality results, we built an efficient algorithm to find the combination of routers that gives the right trade-off for latency and energy consumption. Furthermore, the placement of the cores also affects the communication, and thus latency and energy consumption. This way, in this paper we have used a Tabu-search method that performs the required design space exploration. Experimental results show that there is a trade-off to be explored in the field of heterogeneous networks, which increases the design space exploration. Moreover, we show a Pareto curve relating energy and latency, with a finer grain then other approaches, allowing better exploration of the design space.

The remaining of this paper is structured as follows. Section 2 presents an overview of current NoC design approaches. Section 3 describes the main architectural characteristics of the Tonga, Rasoc and Mago routers. Section 4 shows the mapping problem formulation. Section 5 presents the proposed optimization algorithm and experimental results. Finally, Section 6 draws conclusions and discusses future work.

2. RELATED WORK

Hu and Marculescu [3] showed that mapping algorithms may reduce over 60% of energy consumption when compared to ad hoc mapping solutions. Murali and De Micheli [4] presented an algorithm that maps the cores onto a mesh NoC architecture under bandwidth constraints, aiming to minimize the energy consumption and the average communication delay. Both works are based on application models that omit essential information to estimate the latency of the application, since these models do not capture the moment of each communication within the NoC, while the application is executing. On the other hand, our ACP formulation enables to model the communication *ordering*, which leads to better mappings with low extra computational effort, if compared to previous models.

Murali and De Micheli [5] extend the work presented in [4], by the introduction of a tool called SUNMAP, which builds its solution inside a predefined library of topologies and uses a multi-objective function, which encompasses average communication delay, area and energy consumption. The main objective of the tool is to select automatically the best topology for an application and to generate cores mapping onto that topology.

Marcon et al [6] introduced *communication dependence model* (CDM), which captures the message dependences. Comparing their approach with previous work [3][4], they achieve an average reduction of 42% in application execution time, at the same time reducing the total energy consumption of the system by 21% for state-of-the-art technologies. However, while the input data of ACP is easily extracted by simulation, CDM implies extra effort, since the designer has to describe the application and also its communication dependence.

Rijpkema et al [7] focused on designing a router architecture that has been adapted to trade-off cost and efficiency. They analyze and propose two orthogonal concepts for router operation: guaranteed throughput (GT) and best effort (BE). While GT services require resource reservation for worst-case scenarios, which can be expensive, BE services use resources in a better way, because they are typically designed for average-case scenarios instead of worst-case ones. Given that, in our approach we deal with the actual communication behavior, and we have different router architectures. This way, this work develops a more fine-grained solution to better explore the design space, combining latency, area, and power dissipation, enhancing the design space to look for a better solution, as it will be shown.

None of the mentioned works dealing with the mapping of applications onto NoCs has considered the complete set of variables related to the application behavior and to the NoC heterogeneity. The main contribution of this work is the development of an optimization algorithm, based on the analysis of NoC topologies and router architectures. The algorithm aims to find a trade-off among latency and energy constraints when NoCs are used as intercommunication resource. Our approach uses ACP, which models the ordering and volume of messages. This paper explores the design space for mesh NoC topology, considering different router architectures and appropriate analytical energy models in the search for an optimized NoC, concerning latency and energy consumption.

3. ROUTER ARCHITECTURES

This Section presents and compares the main architectural characteristics of the Rasoc, Tonga and Mago routers, which are used to compose the heterogeneous NoC.

Rasoc is a routing switch with up to five bi-directional ports (Local, North, South, West, and East). Rasoc implements a distributed approach for packet arbitration - there exists a round-robin arbiter at each output channel. The memory organization is based on FIFO buffer architecture [2]. An XY-routing scheme is used. A *handshake* protocol implements the NoC control flow between. Figure 1 (a) shows the Rasoc router architecture.

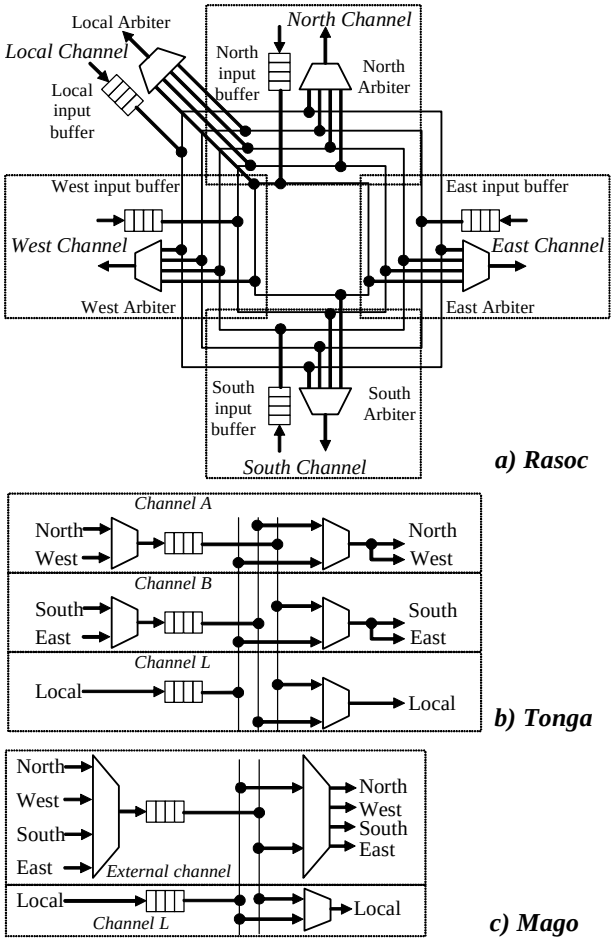


Figure 1 – Rasoc, Tonga and Mago routers architecture.

The architecture of the Tonga router is based on Rasoc. The main difference among them relies on the way channels are routed. Tonga can be viewed as low-cost Rasoc router, since a multiplexed architecture was employed to reduce the area and energy overhead. Figure 1 (b) shows the basic idea of Tonga architecture, highlighting input and output ports. The idea is to

use the same structure to manage North and West ports (named channel *A*), another for South and East (named channel *B*), and a third one for the Local port (channel *L*). Channels A and B include a switch that selects one of the multiplexed ports. The multiplexed operation of these channels is based on a counter, which reserves a configurable time-slot for each multiplexed port.

We have designed the Mago architecture to achieve minimum energy consumption. A single channel multiplexes all external channels, reducing the buffer area and energy consumption. On the other hand, Mago reduces the NoC parallelism, increasing the overall latency. Figure 1 (c) depicts the Mago architecture, highlighting input and output ports. The idea is to use the same structure to manage North, West, South and East (*external channel*), and a second one for the Local port (channel *L*). External channel include a switch that selects one of the multiplexed ports. The multiplexed operation of these channels is similar to Tonga.

We described the routers in VHDL and synthesized them to a 0.35µm technology by using Leonardo Spectrum 3. All routers have FIFOs of 4-word depth and different channel widths. Table 1 shows the area consumption, the mean energy consumption, and the average execution time obtained for the architectures. The area consumption was directly obtained from physical synthesis. The average power consumption was obtained with electrical simulation of a random data input, with a VDD of 3.3 volts. The average of overall execution time was obtained by functional simulation of complete homogeneous NoC with each router architecture and hypothetical applications.

Channel Width	Characteristic	Routers		
		Rasoc	Tonga	Mago
8	Area (µm2)	157,266	100,937	75,659
16		248,539	154,809	111,288
8	Average of power consumption (mW)	7.61	5.09	3.83
16		10.48	6.98	5.28
8 / 16	Average of overall execution time (cycles)	268,012	313,570	443,947

Table 1 – Area, energy, and execution time of the routers, according to different channel widths.

Table 1 confirms that, when using a more limited router, one can save area and power, but the price to pay is a reduction in the system performance, since the message passing mechanism will take more cycles, and the total communication time will increase.

At the system level, however, not all communications among cores must be implemented at full bandwidth. The challenge then is to find the best trade-off to fulfill the application requirements, while at the same time reducing area and power by the use of a low cost router in the right positions.

4. PROBLEM FORMULATION

Given a distributed application, we can state our mapping problem as the task of minimizing the latency and energy consumption by determining the better placement for cores and routers on different NoC topologies. The solution for this problem is a very complex task, because more than one variable is considered concurrently in the search space: position of each core, NoC topology, and tile size (with impact in router size).

We assume an application whose tasks were previously partitioned onto a set of cores, so we envision a scenario where functions are distributed among a number of cores in a SoC. We have defined an *application communication pattern* (ACP) to represent the semantics of communicating cores. Communications are described by the relationships among the cores, by the parallelism among messages, and by the temporal order in which the communications must occur. To model such behavior, a data structure is defined where each element carries a time tag and a pointer to a set of messages, performing a totally ordered set of messages.

Definition 1: An ACP is a list of sets. Let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of application cores, and $b_q \in \mathbb{N}$ the number of bits of the q-th message. Then $m_{ijq} = (c_i, c_j, b_{ijq}) \mid c_i, c_j \in C$ is the q-th message from core c_i to core c_j with b_{ijq} bits. Let $M = \{m_{ijq} \mid c_i, c_j \in C\}$ be the set of all messages between application cores and $\boldsymbol{m}_i$ be a subset of M. ACP = $\{T_i = (t_i, \boldsymbol{m}_i) \mid \boldsymbol{m}_i \neq \varnothing$ and $t_i \in \mathbb{N}\} \cup \{\text{START} = (0, \varnothing), \text{END} = (\infty, \varnothing)\}$ represents an ordered list of message sets, such that t is a time tag that marks the start time of all messagens of $\boldsymbol{m}_i$. START and END are 2-tuples, and (T_i, T_j) with $i \neq j$ implies $t_i < t_j$ is the set ordering criteria.

The communication architectures can also be modeled as a graph, whose vertices represent tiles and the set of oriented edges express all the links given by the network topology. This data structure is defined as the *communication resource graph* (CRG).

Definition 2: A $CRG = \langle \Gamma, L \rangle$ is a directed graph, where $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_p\}$ denotes the set of tiles, corresponding to the set of CRG vertices, and $L = \{(\tau_i, \tau_j) \mid \tau_i, \tau_j \in \Gamma\}$ designates the set of links from tile τ_i to tile τ_j, corresponding to the set of CRG edges. The way the edges are connected represents the network topology. Each directed edge $l_{ij} = (\tau_i, \tau_j)$ has associated a structure s_{ij} composed by parameters that express the link characteristics in a given topology, such as a link width.

Figure 2 illustrates the above definitions through a hypothetical application with four cores $C = \{a, b, c, d\}$, six messages, and a 2x2 NoC. Figure 2 (a) depicts an ACP = $\{(t_1, \{(a, b, 15), (c, a, 20)\}), (t_2, \{(c, a, 15), (a, d, 15), (b, d, 40)\}), (t_3, \{(d, b, 15)\})\}$. Figure 2 (b) depicts a CRG with an arbitrary valid mapping of C onto CRG, generating the following association: $\{(\tau_1, b), (\tau_2, a), (\tau_3, d), (\tau_4, c)\}$.

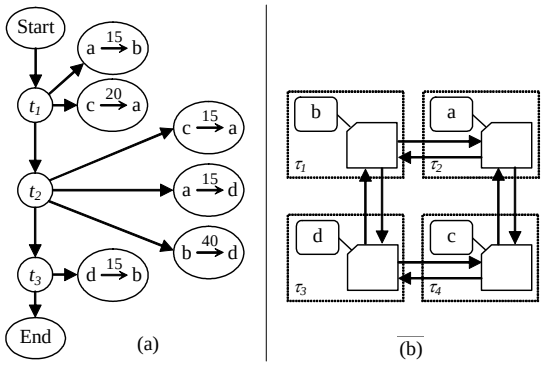


Figure 2 - ACP (a) and CRG (b) examples.

For each application message, it is necessary to find in the CRG a path between its sender and receiver vertices in order to determine

if the bandwidth offered by this path matches the one required by the application.

Definition 3: A path $p_{ij} = (\tau_i, l_{iy}, \ldots, \tau_w, l_{wj}, \tau_j)$ is an alternating sequence of CRG vertices and edges, to transport a message from core c_i to core c_j. A path is formed according to the routing strategy implemented in the network tiles the CRG represents.

Finally, in order to find a valid mapping, one must map each message of the ACP to links and local ports associated to tiles of the CRG.

Definition 4: Given a CRG, for each $m_{ijq} \in$ ACP there exists a corresponding $p_{ij} \in$ CRG, i.e. there is a mapping function F: ACP $\rightarrow$ CRG such that $\forall\ m_{ijq} \in$ ACP $\exists\ p_{ij} \in$ CRG.

4.1 Energy Model

This work uses an energy model similar to the ones presented in [3][4], where the concept of bit energy (*Ebit*) is used to estimate the dynamic energy consumption for each bit. *Ebit* is split into bit dynamic energy consumed on the buffers (*EBbit*), on the logic gates of each switch (*ESbit*), and on the links between tiles (*ELbit*), which is directly proportional to tile dimension.

To estimate the energy for mesh NoC topology, two assumptions are considered: *(i)* the router area is insignificant in comparison to the core area; and *(ii)* all tiles are regular and with the same square dimensions. These assumptions simplify the energy model since the energy consumption does not depend on the communication path, but rather in the communication traffic. Equation 1 computes the dynamic energy consumed by a single bit traversing the NoC, from tile τ_i to tile τ_j, where communicating cores were previously mapped, where η corresponds to the number of routers through which the bit passes from tile *i* to tile *j*.

$$Ebit_{ij} = \eta\,(ESbit + EBbit) + (\eta - 1)\,ELbit \quad (1)$$

Let n_{ijq} be the total amount of bits of a message $m_{ijq} \in M$ going from c_i to c_j. Then, we define $Ebit_q = n_{ijq} \times Ebit_{ij}$, and Equation 2 gives the total amount of *NoC dynamic energy consumption* (*EDyNoC*), which computes all bit traffic of all *k* messages.

$$EDyNoC = \sum_{q=0}^{k} Ebit_q \quad (2)$$

Given this energy model, an evaluation algorithm can be developed to optimize energy consumption of a NoC. In the next section, our optimization strategy and the associated algorithm are detailed.

5. HETEROGENEOUS NETWORK-ON-CHIP ARCHITECTURAL OPTIMIZATION

This section shows a heterogeneous network-on-chip, composed by Rasoc, Tonga and Mago routers. We apply an algorithm that automatically finds an optimized mix between these routers, regarding overall latency and energy consumption constraints.

5.1 Optimization Strategies

During the search process, either latency or energy consumption may have the highest priority, according to their upper bound limits: if latency is the stringent constraint, then the optimization algorithm prioritizes it, keeping energy consumption as small as possible; otherwise, energy consumption assumes the highest priority, while the algorithm tries to minimize latency.

To implement these alternative policies, two strategies were employed in order to find an optimized heterogeneous NoC architecture: *latency priority/energy minimization* (LP/EM) to keep up with latency constraints, minimizing energy as far as possible, and otherwise, *energy priority/latency minimization* (EP/LM).

In the EP/LM strategy, a network composed only with Mago is initially taken and the optimization relies on placing the cores in the best possible positions. As Mago demands less energy consumption than the others do, this homogeneous approach implies that if the latency is reached than also the lowest energy consumption is achieved. If the latency is not reached than some of the Mago are changed to Tonga or Rasoc, and a new placement optimization is tried. This process continues until the desired latency is reached or not all routers are changed to Rasoc.

In the LP/EM strategy, the network is initially only composed by Rasoc, and the algorithm replaces these routers by Tonga or Mago, trying to optimize energy consumption, while keeping the latency under the minimum limit required by the application. The intersection of both strategies, as exemplified in Figure 3, represents the possible solutions of mappings and routers that the designer can choose.

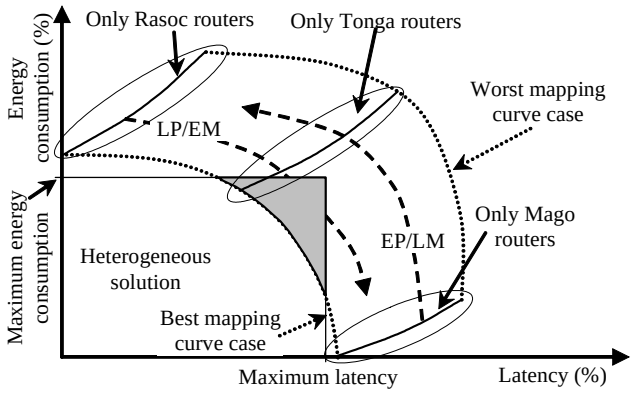


Figure 3 - Design space exploration (EP/LM and LP/EM).

Placing processing cores properly is crucial for latency improvement, since the mapping define sources and destinations of each message and the number of links among them. However, this can be classified as an instance of the *quadratic assignment problem*, which is proven to be NP-hard [8]. The search space increases factorially with system size: even for a small network configuration, such as 3x3 for instance, there are 9! possible places for processing cores. To cope with this problem we have chosen as heuristic the Tabu Search (TS) algorithm, which is being successfully used to solve this kind of problem in many areas [9].

The general strategy of TS is to explore, from a set of *p* resources $S = \{1, 2, \ldots, p\}$, all possible moves from the current solution to a neighboring one. The algorithm can accept the move leading to a neighboring solution even if this results in a worse value for the objective function. To prevent the search process from cycling, a

Tabu list (*T*) stores the last moves for a certain number of iterations. Thus, all solutions that the algorithm obtains by applying a move stored in *T* are excluded from the search. An aspiration criterion allows overriding the Tabu status of a move, for instance, if the move leads to a new best solution. A pseudo-code for the Tabu search algorithm is outlined in Figure 4, where *t* stands for one step of the algorithm, *k* is the cost obtained for energy consumption and latency, *Y* is the set of all possible solutions, *Y'* is a subset of *n* neighboring solutions, and *nt* is the total number of iterations the algorithm should execute.

```
Tabu_Search(resources S)  {
   select an initial solution:
       y ∈ Y e y* = y; k= 0; T = ∅
loop:
   if S(y)–T == ∅
       stop;
   else
       t = t + 1;
   select best Y' = OPTIMUM(s(y): s ∈ S(y)-T);
   y = Y';
   if k(y) < k(y*)  // y* → best solution
      y* = y;
   if t > nt
      stop;
   else
      update T;
   go to loop;
}
```

Figure 4 - Pseudo code for Tabu Search algorithm.

The OPTIMUM function finds each new network configuration *Y'*. However, if not all latency requirements can be met with this network configuration, the optimization algorithm employs a two-dimensional search: as the processing cores are moved around different places on the network local ports, the router types are simultaneously changed. For this purpose, two resources are defined for the Tabu search space: *S1* and *S2*. *S1* stands for cores placement and *S2* stands for the type of each router in the network: $S2 = \{r_i \mid i \in \mathbb{N} \text{ and } 0 \le i \le 2\}$, where r_0, r_1, r_2 represents routers Rasoc, Tonga and Mago, respectively. For instance, $S2[1] = \{r_2\}$ means that router "1" is of type Mago. Now, the Tabu has two *moves* in the search space: core swapping (for *S1*) and router type replacement (for *S2*).

In the LP/EM strategy, for instance, the network is initially configured with Rasoc routers only, as a result, $S2 = \{r_0, r_0, \ldots, r_0\}$. These routers are progressively replaced by Tonga or Mago ones, as new solutions are found, consequently $S2[n, m] = \{r_1$ or $r_2\}$, where *n* and *m* are *moves* for *S2*, and the router type (r_1 or r_2) is randomly chosen. The probability to choose Tonga starts higher than Mago and gradually decreases until only Mago routers compose all tiles of the network.

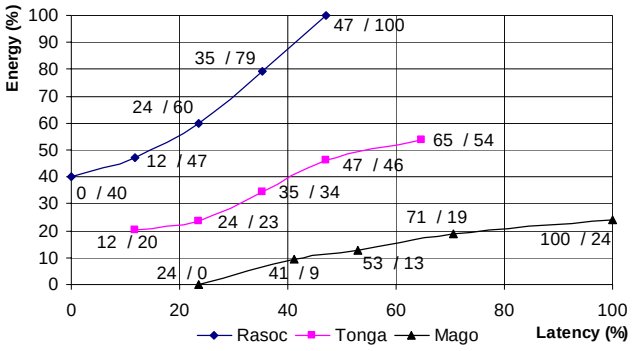
5.2 Results

In order to test optimizations on heterogeneous networks, we submit five applications to the evaluation tool: a distributed Romberg integration [10], an 8-point Fast Fourier Transform (FFT) [11], and an image application for object recognition and image encoding (Imag). The remaining applications (App1 and App2) are benchmarks randomly generated by a proprietary system, which is similar to TGFF [12].

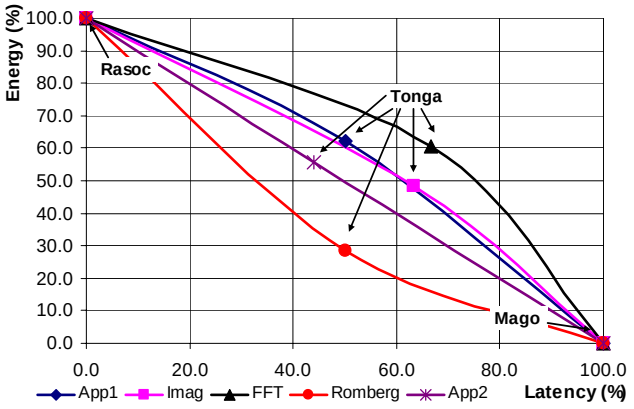
We described C++ models of Rasoc, Tonga and Mago routers to simulate the NoC on a dedicated C++ code-compiled simulator, to speed-up the search for optimized solutions.

Figure 5 (a) shows energy consumption and latency trade-offs for three homogeneous NoCs - Rasoc, Tonga and Mago – running the image application (Imag). The Tabu evaluation approach for this experiment relies on processors placement optimization for each router type. The results show the trade-off achieved for applications targeting different architecture types: the Mago router execution results on minimum energy consumption, the minimum latency concerns for the Rasoc one, while Tonga represents a compromise between both Mago and Rasoc. The points on Figure 5 (a) highlight the search space explored by the Tabu algorithm.

Figure 5 (b) depicts energy consumption and latency for five applications, based on placement optimization. The points show the best placement for each router. From Figure 5 (b), one can verify that the trade-off for latency priority (achieved with Rasoc) and energy consumption (achieved with Mago) remain constant for all applications. Besides, Tonga was designed to be a compromise between both routers, achieving in average 50% of energy consumption and 58% of latency. The behavior of the curves is not linear, since it is strongly dependent on the application characteristics, mainly due to the communication behavior that can induce network contentions.



(a)



(b)

Figure 5 – (a) Energy consumption and latency comparison for three homogeneous NoC. (b) Energy/Latency trade-off for five applications with homogeneous NoC.

Figure 6 depicts latency/energy trade-offs among homogeneous and heterogeneous NoCs architectures, taken for the "Imag" application evaluation. The points for the three homogeneous NoCs show the optimal mapping for each one. The combination of different routers in the heterogeneous NoC, i.e. a SoCINhet, increases the design space exploration, since it is possible to find a Pareto curve for the application. For instance, when using 10 Rasoc, 7 Tonga and 3 Mago, our algorithm achieved 25% of latency and 88,1% of energy consumption, if compared to other solutions (100% meaning that the worst solution was taken as the reference). Other points can be reached as well, since the plot in Figure 6 is actually the Pareto curve of the two variables (energy and latency) for the whole design with different routers. The proposed approach enables the designer to find the optimum trade-off between energy consumption and latency for a given constraint. We compare homogeneous and heterogeneous approach over others applications achieving similar results. Another interesting result is that we achieved in average less latency and energy consumption when using heterogeneous approach compared to homogeneous one.

Due to the code compiled simulation approach, all experiments have taken – in average – no more than 10 seconds, on a Pentium IV 2.8GHz, 512 MB memory machine, for 2000 iterations executed by the tabu algorithm.

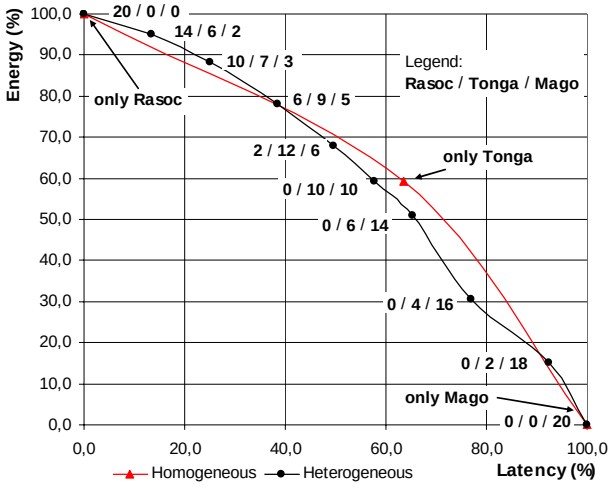


Figure 6 – The increase of design space exploration by using heterogeneous NoC.

6. CONCLUSIONS AND FUTURE WORK

This paper presented a strategy to find the best heterogeneous NoC architecture that can simultaneously meet latency and energy consumption constraints. To comply with the objectives, a dedicated design tool configures the internal components of a heterogeneous NoC architecture, comprised by three different types of routers. By exploring such heterogeneity, it is possible to trade-off between latency and energy consumption. The tool proposed is able to optimize NoC architectures, exploring, simultaneously, processing cores positions and router types on the target network, taking as design constraints energy consumption and latency.

Experimental results have shown that there is a large space to be explored for designs where a heterogeneous NoC is considered with dedicated applications constraints. The tool is able to prioritize either, latency or energy consumption, and finds the Pareto curve relating them for a certain application. As future work, we intend to study other routing strategies to obtain even more aggressive energy reductions.

7. REFERENCES

[1] S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä and A. Hemani. *A Network on Chip Architecture and Design Methodology*. **IEEE Computer Society Annual Symposium on VLSI**, pages 105-112, April 2002.

[2] C. Zeferino and A. Susin, *SoCIN: A Parametric and Scalable Network-on-Chip*, **16th Symposium on Integrated Circuits and Systems Design (SBCCI)**, pages 169-174, September 2003.

[3] J. Hu and R. Marculescu. *Energy-Aware Mapping for Tile-based NoC Architectures under Performance Constraints*. **Asia and South Pacific Design-Automation Conference (ASP-DAC)**, pages 233-239, January 2003.

[4] S. Murali and G. De Micheli. *Bandwidth-Constrained Mapping of Cores onto NoC Architectures*. **Design, Automation and Test in Europe Conference and Exhibition (DATE)**, volume 2, pages 896-901, February 2004.

[5] S. Murali and G. De Micheli. *SUNMAP: A Tool for Automatic Topology Selection and Generation for NoCs*. **41st Design Automation Conference (DAC)**, pages 914-919, June 2004.

[6] C. Marcon, A. Borin, A. Susin, L. Carro and F. Wagner. *Time and Energy Efficient Mapping of Embedded Applications onto NoCs*. **Asia and South Pacific Design-Automation Conference (ASP-DAC)**, January 2005.

[7] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage and E. Waterlander. *Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip*. **IEE Proc. Computers and Digital Techniques**, volume 150, issue 5, pages 294-302, September 2003.

[8] M. Garey and D. Johnson. **Computers and Intractability: A Guide to the Theory of NP-Completeness**. W. H. Freeman and Co. 1979.

[9] F. Glover. *Tabu search – Part II*. **ORSA Journal on Computing**, volume 1, pages 4-32, 1990.

[10] R. Burden and J. D. Faires. **Study Guide for Numerical Analysis**, McGraw-Hill, New-York, 2001.

[11] M. Quinn. **Parallel Computing- Theory and Practice**, McGraw-Hill, New-York, 1994.

[12] R. Dick, D. Rhodes and W. Wolf. *TGFF: task graphs for free*. **CODES/CASHE**, pages 97–101, March 1998.