Synthesis of Low Power NOC Topologies under Bandwidth Constraints

Alessandro Pinto
Luca Carloni
Alberto L. Sangiovanni-Vincentelli

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2006-137
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-137.html

October 24, 2006

Synthesis of Low Power NOC Topologies under Bandwidth Constraints

Alessandro Pinto, University of California at Berkeley
Luca Carloni, Columbia University
Alberto Sangiovanni-Vincentelli, University of California at Berkeley

October 24, 2006

Abstract

We propose an efficient design flow for the automatic synthesis of Network-on-Chip (NOC) topologies. The specification of the problem is given as a netlist of IP cores and their communication requirements. Each IP is characterized by its area. A communication constraint is denoted by its source and destination IP and a minimum bandwidth requirement. Together with the specification, the users provides a percentage of the chip area that they want to allocate for the communication network. Then, given the clock period of the network (that we assume to be synchronous), and a target technology, the proposed design flow explores the entire topology space and returns an optimal NOC where each router has a position and a routing table assigned. We consider two optimality criteria: power consumption and power delay product. Our design flow, which is based on an approximation algorithm, is efficient and highlights the delicate trade-off balance between cost of communication and cost of switching.

1 Introduction

With the advent of chip multi-processors (CMP) and multi-core systems-on-chip (SOC), the network-on-chip (NOC) paradigm has been proposed as the solution to the problem of connecting the increasing number of processing cores that are integrated on a single die [3, 6, 7]. While some of the ideas developed for macro-level interconnect networks (local area networks, supercomputer networks) can be adapted to NOC design, the challenges are in leveraging the intrinsic characteristics of on-chip communication to achieve both energy efficiency and high performance [11]. At the specification level, NOC design is made complex by the variety of the processing cores that can be integrated on a chip, (CPUs, micro-controllers, accelerators, memories,...) and the heterogeneity of bandwidth and latency requirements among them. At the implementation level, each target silicon technology offers a multitude of options to the NOC designers who, for instance, must decide the number and positions of network access points and routers as well as which metal layer to use for implementing each given channel. In particular, choosing the network topology is challenging as the space of possible topologies is very large. *Hence, it is very difficult to guess the right communication topology only by experience, taking into account the heterogeneity of the requirements and the constraints imposed by the silicon technology.*. Consequently, the development an automatic tool for optimal topology selection for on-chip networks (the topic of this paper) is of great help to the NOC design paradigm.

Related work. To the best of our knowledge, most of the research efforts in the NoC domain have been devoted to the study of *ad hoc* routing, switching and flow control protocols for high performance and minimum power consumption while scant have been the approaches to synthesizing an optimal topology. One approach, implemented in the NetChip design flow, is described in [4]. The topology selection is done by an algorithm that tries many fixed topologies (described in a library) and that selects the best one. In this technique, the granularity of the library is the entire communication infrastructure. In [14] the idea is to have a library of communication primitives that capture schemes like gossiping, broadcast etc. This approach is very interesting but does not fully consider the silicon properties like the critical sequential length. In [13], the authors start from a standard topology and perform a local search to improve its performance by inserting a certain number of long links. In [17] the authors present a very interesting approach that we consider the closer to ours. They explore a much larger design space using mixed-integer linear programming techniques. The design flow is the same that we adopt (floorplan, generation of admissible router positions, optimal routing).

Compared to the works in [4] we focus on the synthesis of optimal heterogeneous network topologies by assembly components from a fine grained library. Compared to [13, 14] we use a more detailed model for the communication components and we don't neet to assume an underlying basic network topology. Compared to [17] we rely on an approximation algorithm that exploit the graph structure of the problem instead of using an MILP formulation. This allows us to explore a larger design space (in terms of admissible positions for the routers) more efficiently (a comparable solution for the same problem instance is given in less than a minute instead of few hours).

2 Formal Setting

We represent networks with labeled graphs $G(V, E, p, \omega)$ where V is the set of vertices, E is the set of edges $p : V \rightarrow \mathbb{R}^2$ is a function that associates a position to each vertex and $\omega : E \rightarrow \mathbb{R}_+$ is a function that associates a bandwidth to each edge. The set of vertices V is partitioned in the sets of source ports S, destination ports D and routers R. For a source $s \in S$, $indegree(s) = 0 \wedge outdegree(s) > 0$ for a destination $d \in D$, $outdegree(d) = 0 \wedge indegree(d) > 0$.

This simple model, which has been commonly adopted in this research field [12, 13, 15], can be used to describe both the design specification of a NOC (as a set of point-to-point communication requirements) and its final implementation (where different communication flows across source-destination pairs may share multiple channels in the network). The difference lies in the interpretation of the two functions p and ω. A *point-to-point network specification* is given as a graph $G(V, E, p, \omega)$ where $R = \emptyset$ and each edge represents a point-to-point unidirectional link between a source and

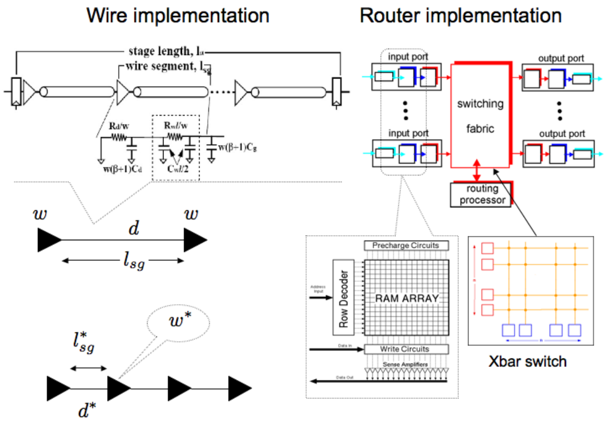

Figure 1: Optimally repeated wire model (left) and block diagram of a router (right).

destination. Here, a function value $p(v) = (x_v, y_v)$ captures the position (x, y) for a vertex $v \in V$ that must be maintained in the final implementation. Meanwhile, $\omega(e) = b_e$ is a constraint requiring the network to provide a paths between vertices $s, d \in V$ that can sustain a bandwidth of at least b_e. If $G'(V', E', p', \omega')$ is a *network implementation* for G, then typically $R \neq \emptyset$ is the set of instanced router and function p' must be an extension of function p capturing also the router positions. Instead, ω' describes how much bandwidth can be supported by each edge. Edges in G' connect pair of elements of V' (including routers). The set of all edges must guarantee the presence of a directed path for each point-to-point unidirectional link in G, while each edge must have enough bandwidth to accommodate all paths that are sharing it.

The synthesis algorithm must satisfy all these constraints while minimizing a given cost function. If $\mathcal{G}$ is the set of all such graphs, let $F : \mathcal{G} \rightarrow \mathbb{R}_+$ denote a cost function with associates a positive real number to each of its elements. After setting the cost of sources and destinations equal to zero (since they are given), we assume that the cost function is separable in two terms: $F(G) = F_E(E) + F_V(V) = \sum_{e \in E} F_e(e) + \sum_{v \in R} F_v(v)$. In Section 3 we characterize this cost function in detail.

3 Modeling of NOC Components

In this section we present the physical models for wires and routers, which are the basic "building blocks" that we use to synthesize NOCs. We use a classic physical model of optimally-repeated wires but we follow also the authors of [16] in accounting for the increasingly important role played by the *critical sequential length* with nanometer technologies. For the routers we rely on the careful models that have been developed as part of the ORION project [18]. The models that we adopt are depicted in Figure 1.

Wires. While the on-chip propagation delay increases proportionally to the square of the interconnection length because both capacitance and resistance increase linearly with the distance, designers can insert an optimal number of optimally sized repeaters in order to divide the interconnect wire into smaller subsections, thereby making the delay linear with its length [2]. Figure 1 shows the classic first-order RC model of a repeated wire [2, 8, 9] where R_d is the driving repeater resistance, w is the width of the repeater NMOS transistor normalized by the minimum technology width, β is the PMOS-to-NMOS sizing ratio, C_d and C_g are diffusion and gate capacitance per unit width, R_w and C_w are wire resistance and capacitance per unit length, and l_{sg} is the length of the repeated segment. This first order model leads to a delay of the wire segment l_{sg} equal to

$$d = 0.7\Big[\frac{R_d}{w}\cdot\big[w(\beta+1)(C_d+C_g)+l_{sg}\cdot C_w\big]+l_{sg}^2\frac{R_w+C_w}{2}+l_{sg}\cdot R_w\cdot w(\beta+1)C_g\Big]$$

For a given technology process and a chosen metal layer, there exists a minimum length l^*_{sg} (*critical repeater length*) beyond which inserting an optimal-sized repeater makes the interconnect delay smaller than that of the corresponding un-repeated wire [16]. The delay d^* of a critical repeater length is called *critical delay*. Figure 1 shows also the minimum-sized flip-flops that are used to drive and sample the signal on the wire [1]. These flip-flops are separated by a stage length l_{st}. Therefore, one can define the *critical sequential length* as the maximum distance that a signal can travel in an interconnect that has been optimally sized and optimally buffered uniformly within a single clock period T. For a given process technology and metal layer we can compute d^* and l^*_{sg}. Hence, assuming a synchronous network implementation with clock frequency f_{clk}, the maximum distance that can be spanned by a wire is equal to

$$l_{st} = \frac{l^*_{sg}}{d^* f_{clk}} \tag{1}$$

The power dissipated by an optimally-repeated line of length l, where $l_{sg} < l < l_{st}$, running at frequency $\frac{1}{T}$ with an activity factor α (the fraction of repeaters that are switched during an average clock cycle) is:

$$P = l\cdot\Big[\frac{\alpha}{T}V_{dd}^2\cdot\big(\frac{k_{opt}}{h_{opt}}(C_d+C_g)+C_w\big)+\frac{V_{dd}}{2}\cdot\big(\frac{k_{opt}}{h_{opt}}(I_n^{off}+2I_p^{off})W_n^{min}\big)\Big]$$

where V_{dd} is the power supply voltage, k_{opt} and h_{opt} are the optimal repeater size and the optimal inter-buffer interconnect length, I_n^{off} (I_p^{off}) are the leakage currents per unit NMOS (PMOS), and W_n^{min} is the width of the NMOS transistor in minimum sized inverted. The first term of the equation is the switching power and the latter is the leakage power. [2] Traditionally the former has dominated the latter, but the trend is changing with nanometer technologies as the leakage power is scaling up super-linearly.

In summary: the delay of an optimally repeated wire is linear in its distance; the power consumption is the sum of the leakage power that is linear in the distance and

[1] In this paper, we assume traditional on-chip signalling while we leave the extension of our approach to more advance circuit techniques [10] for future work.

[2] We simplified the formula omitting the contribution of the short-circuit current which is negligible with respect to the other two.

the dynamic power that is linear in the distance and in the bandwidth; moreover, wires cannot be longer that the critical sequential length.

Routers. The task of a router in a packet-switched network is to receive data streams from input ports and switch them to the output ports by means of a crossbar circuit. A packet arriving at an input port is temporarily stored in a local memory, which for an input-queued router is typically made of a set of queues (Figure 1). The head of queue packets are switched to the proper output ports in accordance with the configuration of a routing table. If more that one packet must be sent to the same output port at a given time, a scheduler decides which one goes first depending on their priority. The area of a NOC router is dominated by the buffer space [6], but the total area overhead of the NOC with respect to the whole SOC is estimated to be minor (about 6.6% in [6]). The energy required by a router is equal to the sum of three terms, each associated to one of its main components [19]:

$$E_{router} = E_{buffer} + E_{xbar} + E_{scheduler}$$

To evaluate this quantity we relied on the results of the ORION project [18]. The authors of ORION have developed detailed parameterized models for various technology processes by decomposing each contribution in order to account directly for the energy dissipation of their "atomic components" down to the transistor level. In particular, their analysis demonstrated that the overall energy dissipation of an input-queued router is linear in the packet injection rate, i.e. in the total communication bandwidth of the incoming channel [20] because the contributions of the input queues and crossbar switches are dominant. [3] Since each flit arriving at a router has to be stored and then switched through the cross-bar, the energy is proportional to the number of flits and therefore the power consumption is proportional to the aggregate bandwidth traversing the router.

While the router buffering power depends also on the size of the input buffers and the crossbar power depends on the number of its input/output lines and its implementation, during the synthesis step we express the power dissipation just as variable that is proportional to the total amount of data bandwidth seen by the router. This abstraction allows us to simply distribute the cost of a router on its input edges.

In summary: the power consumption of a router is proportional to the total bandwidth carried by its input edges and, therefore, can be seen as an additional cost of the corresponding wires.

Cost functions. The cost functions that we use in our synthesis algorithms are abstractions of the models above. The cost of an edge $e = (u, v)$ is defined as

$$F_e(e) = c_1 \cdot \omega(e) \cdot ||p(u) - p(v)||^{\alpha} + c_2 \cdot z(\omega(e)) \cdot ||p(u) - p(v)||$$

where c_1 and c_2 are two coefficients that depend on the technology. The first term accounts for the operation cost of the edge and depends on the value of data that flow through it. By setting the norm exponent value, we can either optimize the wire power consumption ($\alpha = 1$) or the power delay product ($\alpha = 2$). Recall that both delay and

[3] The energy model of a router includes the model of a memory element such as a flip-flop that can be used for wire pipelining. For more details we refer to [19].

power are linear in the distance). The second term accounts for the leakage power and can be seen as an *installation cost*. Function z evaluates to one when $\omega(e) > 0$ and to zero otherwise. The cost of a router $r \in R$ is defined as:

$$F_v(r) = c_3 \cdot \sum_{(u,r)\in E} \omega(u,r)$$

where c_3 depends on the technology and the architecture. [4]

Normalizing both terms by c_1 (since optimizing a cost function or a scaled version of it leads to the same solution) gives the objective cost function for the synthesis problem:

$$F(G) = \sum_{(u,v)\in E} (\omega(e)||p(u) - p(v)||^{\alpha} + \lambda z(\omega(e))||p(u) - p(v)||) + \\ +\gamma \cdot \sum_{r\in R} \sum_{(u,r)\in E} \omega(u,r)$$

where $\lambda = c_2/c_1$ and $\gamma = c_3/c_1$. These two parameters play an important role in evaluating alternative NOC implementations: λ is the *coefficient of leakage impact* and captures the relative cost of leakage versus dynamic power for a unit wire; γ is the *coefficient of communication/switching tradeoff* and captures the relative cost of transmitting directly a bit of data from one point to another of the chip versus temporarily storing it in an intermediate memory element (either a router or a switch).

4 NOC Topology Synthesis

Figure 2 illustrates the proposed design flow that comprises of two main steps: (1) the physical placement of the IP cores with the preallocation of the area for the NOC components is followed by (2) the synthesis of the NOC topology. Given a point-to-point specification graph G as described in Section 2 and the area characteristics of each IP core, we use PARQUET [1] to derive a floorplan of the entire chip by determining the dimension and the position of each core. Notice that a core can at the same time be the source for some communication links and the destination for others. Since NOC routers and repeaters are active elements that cannot be placed on the area that is already occupied by an IP core, we reserve some additional area for the communication infrastructure during floorplanning. In fact, this quantity is given as input parameter $a_{com} = A_{com}/A_{chip}$ denoting the portion of chip area that can be used to design the NOC. In particular, for each core IP_i of area A_i, we allocate an amount of *communication area* equal to $a_{com} \cdot A_i$. Such area is disposed uniformly around the block as illustrated in Fig. 3, where the actual dimensions of the cores are shown by the gray boxes while the floor-planner considers a larger area (shown by the surrounding thick black line).

[4] For instance, the value of coefficient c_3 can be very different if the memories in the routers are implemented with registers or with SDRAM.

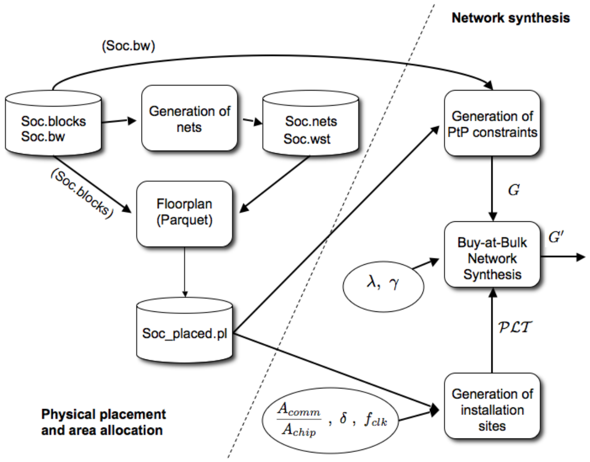
Network synthesis
(Soc.bw)
Soc.blocks
Soc.bw
Generation of nets
Soc.nets
Soc.wst
Generation of PtP constraints
(Soc.blocks)
Floorplan (Parquet)
G
Buy-at-Bulk Network Synthesis
G′
λ, γ
Soc_placed.pl
PLT
Generation of installation sites
A_comm/A_chip , δ , f_clk
Physical placement and area allocation

Figure 2: The proposed design flow.

The communication area is discretized both in the x and y directions with a step equal to an input parameter δ. The discretization defines a set of possible positions to install the routers called *installation sites*, which are represented by the red dots in Figure 3. Let $\mathcal{P}$ denote the set of installation sites. A smaller value of δ may give a more refined solution at the price of increasing the run time of the synthesis step. Not all installation sites can be directly connected. Given a target process technology, clock frequency f_{clk}, and metal layer, only sites closer than l_{st} can be directly connected by a wire. A graph $\mathcal{PLT}(V, E, p, \omega)$ that captures all possible nodes (routers) and all possible links (edges) for a given silicon platform can be defined as follows: for each position $(x, y) \in \mathcal{P}$ there is a vertex $v \in V$ such that $p(v) = (x, y)$. We also add the set of sources and destinations form the point-to-point specification graph G to $\mathcal{PLT}$. Given two vertices u and v in V, there is an edge $e - (u, v) \in E$ if and only if $||p(u) - p(v)|| \leq l_{st}$. *By removing or contracting edges of this graph, we can obtain any other network topology that satisfies the constraints imposed by the silicon platform.* In other words, $\mathcal{PLT}$ can be seen as the union of all the possible network implementation graph of G as defined in Section 2. Therefore the *NOC synthesis problem* consists in finding a network implementation graph G' that is contained in $\mathcal{PLT}$, satisfies the constraints specified by G and optimize the chosen cost function as defined in Section 3.

Solving the synthesis problem. If one did not consider the wires' installation cost (seeSection 3) then the globally optimal solution would be given by a simple algorithm that routes each point-to-point constraints along the shortest path. With installation costs, however, the problem becomes NP-Hard. Hence, we solve it using a variation of the *buy-at-bulk algorithm*, a well-known approximation algorithm [5], which we adapted in order to account for the cost of each instanced router. In particular, we kept as input parameters for the cost specification the α and λ that abstract the silicon properties of the NOC components. The modified buy-at-bulk remains a randomized algorithm with approximation guarantee of $e^{O(\sqrt{(\ln N \ln \ln N)})}$ (where N is the total

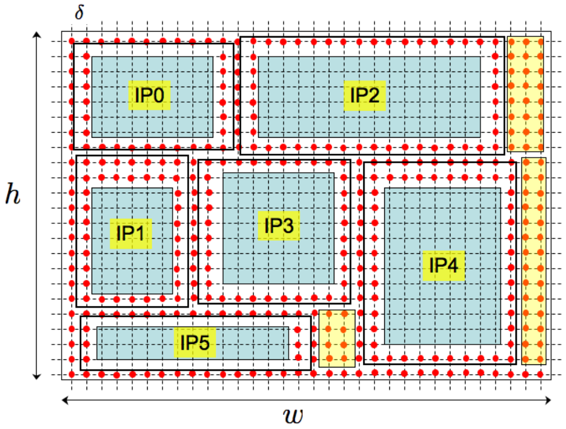

Figure 3: Admissible router installation sites.

bandwidth normalized with respect to the minimum bandwidth in the system). The running time is proportional to the number of installation sites and to the edges of $\mathcal{PLT}$. Notice that in practice graph $\mathcal{PLT}$ can be dynamically explored without building it in its entirety.

The buy-at-bulk returns an implementation graph $G'(V, E)$ where V is the of sources, destinations and installed routers and E is the set of installed connections. As a byproduct of the synthesis we also obtain the full set of source-destination paths (computed by the shortest path algorithm that is used by the buy-at-bulk algorithm) from which we can easily derive the routing tables. All this information is used to synthesize a cycle-accurate SYSTEMC NOC specification by instancing pre-designed components from a parameterized fine-grained communication library.

5 Experimental Results

In this section we present three experimental results.

1. Analysis of the buy-at-bulk algorithm behavior. The first set of experiments show the influence of the parameters λ and γ on deriving final solution of the network synthesis problem. While sweeping their values across given ranges we compute the following four quantities:

$$\widehat{B} = \max\{\omega(e) : e \in E\} \qquad \overline{B} = \frac{\sum_{e \in E} \omega(e)}{|E|}$$

$$\widehat{H} = \max\{l(p) : p \in P\} \qquad \overline{H} = \frac{\sum_{p \in P} l(p)}{|P|}$$

The values of $\widehat{B}$ and $\overline{B}$ give information on the network link congestion. while the values of $\widehat{H}$ and $\overline{H}$ give information on the latency of the optimal solution.

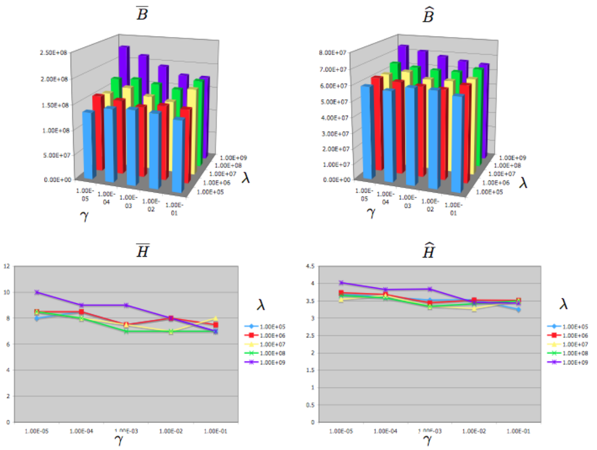

Figure 4: Power optimization case.

In order to estimate these quantities we used a random SoC generator that takes the following parameters: the number N of IP cores; the minimum A_{min} and maximum A_{max} area for a core; the average core input degree $\overline{indeg}$ and output degree $\overline{outdeg}$; and the minimum and maximum bandwidth B_{min} and B_{max} are for a communication constraint. We report here experiments obtained with the following configuration: $N = 20$, $A_{min} = 4mm^2$, $A_{max} = 25mm^2$, $\overline{indeg} = \overline{outdeg} = 1$, $B_{min} = 20Mb/s$ and $B_{max} = 100Mb/s$. Furthermore we set the previously-defined parameters as $a_{com} = 5\%$ and $l_{st} = 5mm$.

For each pair (λ_i, γ_i) we took the average of the four quantities across twenty randomly generated SoC. For each instance we performed a floorplan, computed the set of router installation sites and run the NOC synthesis algorithm. The results are shown in Figure 4 for the case of power optimization and in Figure 5 for the case of power-delay product optimization. In the first case we note two important things. First, the average bandwidth increases with λ showing that when the leakage power (or, more generally, installation cost) is not negligible the optimization algorithm steers the source-destination paths to share as many wires as possible. While this leads to an optimal solution in terms of power consumption, the average congestion (defined as the ratio $\overline{B}/f_{clk}$) increases which could prevent the architecture platform to accommodate for other communication paths. Secondly, the end-to-end latency is not affected by the parameters. This is due to the fact that when the cost function is linear in the distance there is no gain in breaking up a long link in smaller links. Hence, the algorithm tends to use always wires whose length is the maximum determined by the critical sequential length.

The results are different when the cost function considered for the wires is the power-delay product (Figure 5). In this case, the parameter γ, that models the cost of communication with respect to storage and switching, has an impact on the latency of the optimal topology. In particular, the number of hops decreases when the cost of routers increases. In both cases, if the cost of the routers, modeled by γ, increases,

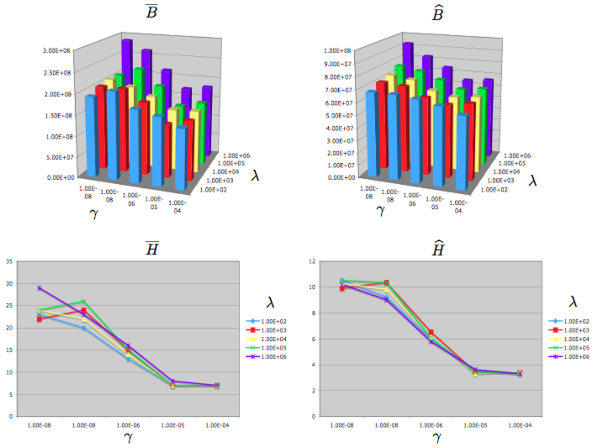

Figure 5: Power-delay optimization case.

the algorithm tends to avoid router instantiations and tries to implement many point-to-point connection. This is the reason why $\widehat{B}$ and $\overline{B}$ decrease.

2. NOC synthesis (heterogeneous SOC). We applied our design flow to an SOC composed of an H263 decoder and an MP3 decoder taken from [17]. Figure 6(left) shows the communication constraint graph G with 14 cores and each edge is labeled with its bandwidth in Kb/s. Figure 6(right) show the final result, where the dark square are routers while the light square are repeaters. The total chip area is $10mm^2$. We estimated by inspection the power consumption of the NOC reported in [17] and compared it with the power consumption of the NOC in Figure 6 (using the same number for the power consumption of wires and input/output ports as reported in [17]. We obtained approximately the same result of $12\mu W$. While the authors of [17] report an execution time of $36,000$ second for their MILP-based heuristic, our approach took approximately 30 seconds on an Intel Pentium Core Duo @ $2GHz$.

3. NOC synthesis (homogeneous SOC). Figure 7 shows a different application, a chip multiprocessor composed of sixteen identical processing cores. Each core communicates with every other core with a rate of $133Mb/s$. The result of the synthesis flow is shown in Figure 7 on the right. The critical sequential length is $5mm$ and the parameters are referred to a $130nm$ technology. We emphasize that this regular topology (a mesh) is generated automatically by our tool by composing wires and routers and that, differently from other approaches, we don't use a library of pre-designed complex topologies (meshes, tori,...).

6 Concluding Remarks

We presented a design flow that generates an energy-efficient NOC topology given a set of IP cores and the communication constraints among them. Our optimization uses a modified version of the buy-at-bulk algorithm [5] and takes into account parame-

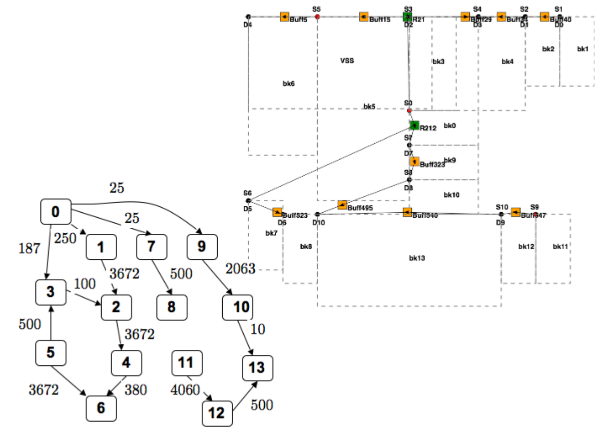

Figure 6: Result for H263/MP3 SoC.

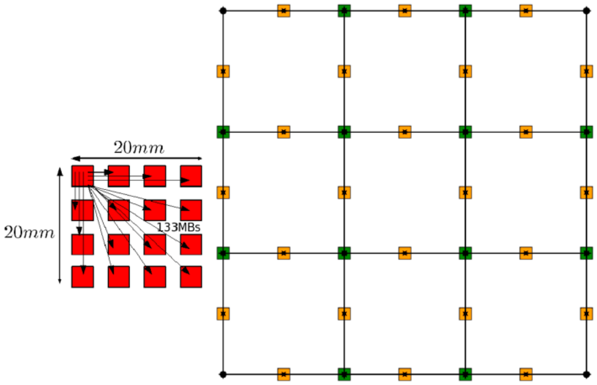

Figure 7: Result for the homogeneous CMP.

ters that are directly connected to the physical implementation of NOCs. In particular we consider both dynamic and static power consumption of wires, trade-off between communication and switching power, as well as the role played by the critical sequential length. We reported on experimental results (for both irregular and regular input specifications) that are obtained with a run time of the order of tens of seconds.

References

[1] S. N. Adya and I. L. Markov. Fixed-outline floorplanning : Enabling hierarchical design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(6):1120–1135, December 2003.

[2] H. B. Bakoglu. *Circuits, Interconnections, and Packaging for VLSI*. Addison-Wesley, Reading,MA, 1990.

[3] L. Benini and G. De Micheli. Networks on chip: A new SoC paradigm. *IEEE Computer*, 2002.

[4] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli. NoC synthesis flow for customized domain specific multiprocessor systems-on-chip. *IEEE Transactions on Parallel and Distributed Systems*, 16(2):113–129, February 2005.

[5] M. Charikar and A. Karagiozova. On non-uniform multicommodity buy-at-bulk network design. In *STOC '05: Proc. of the 37-th Ann. ACM Symp. on Theory of Computing*, pages 176–182. ACM Press, 2005.

[6] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *Proc. of the Design Automation Conf.*, June 2001.

[7] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Vberg, M. Millberg, and D. Lindqvist. Network on chip: An architecture for billion transistor era. In *Proc. of the IEEE NorChip Conference*, November 2000.

[8] S. Heo and K. Asanovic. Replacing global wires with an on-chip network: a power analysis. In *Proc. of the Intl. Symp. on Low Power Electronics and Design*, pages 369–374, 2005.

[9] R. Ho, K. W. Mai, and M. A. Horowitz. The future of wires. *Proceedings of the IEEE*, pages 490–504, April 2001.

[10] A. P. Jose, G. Patounakis, and K. L. Shepard. Near speed-of-light on-chip interconnects using pulsed current-mode signalling. In *Symposium on VLSI Circuits*, June 2005.

[11] G. De Micheli and L. Benini. *Networks on chip*. Morgan Kaufmann, 2006.

[12] S. Murali and G. De Micheli. SUNMAP: A tool for automatic topology selection and generation for NOCs. In *Proc. of the Design Automation Conf.*, pages 914–919, June 2004.

[13] U. Ogras and R. Marculescu. Application-specific network-on-chip architecture customization via long-range link insertion. In *Proc. Intl. Conf. on Computer-Aided Design*, November 2005.

[14] U. Ogras and R. Marculescu. Energy- and performance-driven noc communication architecture synthesis using a decomposition approach. In *Design, Automation and Test in Europe*, March 2005.

[15] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli. Constraint-driven communication synthesis. In *Proc. of the Design Automation Conf.*, pages 783–788, New Orleans, LO, June 2002. IEEE.

[16] P. Saxena, N. Menezes, P. Cocchini, and D.A. Kirkpatrick. Repeater scaling and its impact on CAD. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(4):451–462, April 2004.

[17] K. Srinivasan, K. S. Chatha, and G. Konjevod. Linear-programming-based techniques for synthesis of network-on-chip architectures. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14(4):407–420, April 2006.

[18] H. S. Wang, X. Zhu, L. S. Peh, and S. Malik. Orion: A power-performance simulator for interconnection networks. In *Proceedings of the 35th International Symposium on Microarchitecture (MICRO)*, pages 294–305, November 2002.

[19] Hangsheng Wang. A detailed architectural-level power model for router buffers, crossbars and arbiters. Technical report, Princeton University, January 2004.

[20] H.S. Wang, L.S. Peh, and S. Malik. A power model for routers: Modeling Alpha 21364 and InfiniBand routers. *IEEE Micro*, 23(1):26–35, 2003.