Research Article

Area and Power Modeling for Networks-on-Chip with Layout Awareness

Paolo Meloni,¹ Igor Loi,^{1,2} Federico Angiolini,² Salvatore Carta,³ Massimo Barbaro,¹ Luigi Raffo,¹ and Luca Benini²

¹ *Department of Electrical and Electronic Engineering (DIEE), University of Cagliari, 09123 Cagliari, Italy*

² *Department of Electronics and Information Science (DEIS), University of Bologna, 40136 Bologna, Italy*

³ *Department of Mathematics and Information Science, University of Cagliari, 09123 Cagliari, Italy*

Received 1 November 2006; Revised 2 February 2007; Accepted 1 March 2007

Recommended by Maurizio Palesi

Networks-on-Chip (NoCs) are emerging as scalable interconnection architectures, designed to support the increasing amount of cores that are integrated onto a silicon die. Compared to traditional interconnects, however, NoCs still lack well established CAD deployment tools to tackle the large amount of available degrees of freedom, starting from the choice of a network topology. "Silicon-aware" optimization tools are now emerging in literature; they select an NoC topology taking into account the tradeoff between performance and hardware cost, that is, area and power consumption. A key requirement for the effectiveness of these tools, however, is the availability of accurate analytical models for power and area. Such models are unfortunately not as available and well understood as those for traditional communication fabrics. Further, simplistic models may turn out to be totally inaccurate when applied to wire dominated architectures; this observation demands at least for a model validation step against placed and routed devices. In this work, given an NoC reference architecture, we present a flow to devise analytical models of area occupation and power consumption of NoC switches, and propose strategies for coefficient characterization which have different tradeoffs in terms of accuracy and of modeling activity effort. The models are parameterized on several architectural, synthesis-related, and traffic variables, resulting in maximum flexibility. We finally assess the accuracy of the models, checking whether they can also be applied to placed and routed NoC blocks.

Copyright © 2007 Paolo Meloni et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. INTRODUCTION

Current and future Systems-on-Chip (SoCs) achieve increasingly better functionality and performance by integrating larger amounts of processing elements. This growth in computing resources must be matched by a corresponding evolution of the interconnection infrastructure. Traditional communication fabrics exhibit scalability issues in terms of performance and physical circuit design. The Network-on-Chip (NoC) paradigm, which brings networking concepts to the on-chip domain, answers such concerns with a scalable design, at both the architectural and physical levels.

NoC design is a discipline with a large amount of degrees of freedom. While this is an advantage, in that the interconnect can be optimally tailored to match the application at hand, it creates a critical issue when exploring the design space from the hardware overhead point of view. Given the flexibility of NoCs, an exhaustive exploration would require impractical amounts of synthesis runs, and a thorough

characterization of switching activity in every candidate topology to properly assess power consumption.

An answer to the NoC customization complexity lays in the deployment of robust CAD tools, introducing the ability for automated design space exploration according to some fast optimization algorithm. In turn, however, such an approach requires the availability of accurate analytical models of NoC area and power consumptions to drive the optimization algorithms. Models allow the designer to quickly preestimate the area requirements and power consumption overhead introduced by the candidate interconnects. However, it must be noted that, as with any hardware component, the hardware cost of an NoC switch depends on several kinds of parameters, including (i) architectural (e.g., amount of buffering), (ii) synthesis tool-related (e.g., target operating frequency), (iii) operating (e.g., traffic flows).

It is also sometimes easy to underestimate the complexity of synthesis flows, which involve multiple tools, increasingly complex libraries, a large amount of heuristics and

several approximations or models of the behavior of physical on-chip devices. Experience shows that one wrong assumption may severely impact the properties of a whole design. For example, in recent years, the increasing importance of wiring resources has sometimes been impacting the final performance of circuits in unexpected ways, for example, by introducing higher parasitic capacitances and therefore power consumption, or by forcing redesign iterations due to higher transition latencies. This scenario demands for careful assessment of the accuracy of any predictive model at the lowest available level of abstraction; if possible, designers should try to validate their assumptions on placed and routed netlists. This obviously requires a large effort and may be impractical.

As a major contribution of this work, we propose an NoC modeling methodology which takes advantage of the designer's knowledge of the target architecture and synthesis library. It is of course impossible to devise an accurate yet fully generic model for the hardware cost, in power and area, of any given NoC. Our focus is instead on how such a model can be built for a specific NoC instance; we will illustrate the challenges and opportunities involved in this flow, in terms of accuracy and characterization time.

For our analysis, we choose the $\times$ pipes NoC switch as a case study due to its parameterizability (Section 3). Key properties of our approach include accuracy and explicit modeling on several parameters of the design, like switch cardinality, flit width, buffering, traffic, and synthesis parameters. These properties make the approach suitable for fast exploration of large parts of the fabric design space, flexible and applicable in real life, for example, by accounting for the behavior of the synthesis tools when the target operating frequency approaches the limits of the design. The characterization is dependent on the target technology library, but can be easily scripted and automated. A major novel feature of our study, improving on our previous work [1], is that we explore the accuracy of our modeling style against placed and routed test instances; we feel that this is an essential step given the uncertainties intrinsic in today's technology processes. We also show that model coefficients can be made even more accurate by using a placed and routed training set for characterization, albeit at a modeling effort cost, and that the remaining inaccuracies can mostly be attributed to the intrinsic variations induced by synthesis tools.

Our approach starts from an existing RTL description of the NoC components, which are then synthesized and characterized under multiple architectural configurations and traffic conditions. A mathematical formulation of the area and power models is derived from empirical evidence and from the designer's knowledge of the NoC. Eventually, the coefficients of the model are fitted to the experimental results, guaranteeing accurate results for the given architecture. We present two different ways of characterizing the coefficients, with varying accuracy/effort tradeoffs, and two models to account for the dependency of synthesis results on the target synthesis frequency. The synthesis process can optionally include the placement and routing step for maximum thoroughness of the assessment.

2. RELATED WORK

NoCs have recently been proposed as a way to overcome the scalability issues of traditional interconnects [2–4]. Research has focused on multiple design levels. From the architectural point of view, a complete scheme is presented for example in [5], while specific topics are tackled in several works, such as quality of service (QoS) provisions [6] and asynchronous implementations [7]. The complexity in tackling the configurability of NoCs has been made clear by [8], where synthesis results for switches show widely different hardware requirements. Hence, the need for the development of algorithms and CAD tool-chains for NoC instantiation and optimization, as found for example in [9, 10]. A requirement of such works is the availability of reliable power and area models.

Power models and simulators for processors and memories have been proposed in an extremely large body of research [11, 12]. Interconnects have also become the focus of research [13], due to their increasing role in the hardware budget of recent and future systems; for example, the on-chip network of the MIT RAW chip multiprocessor is taking 36% of the chip power budget on average [14].

Some models of NoC hardware cost have already been proposed in previous literature. Results in [15] are derived from a mix of results on template circuits and from technology trends, and are specifically aimed at wide applicability. Therefore, even though they have been used for design space exploration [16] and in association with high-level traffic injection models [17], they do not guarantee maximum accuracy within an architecture-specific CAD flow. The main advantage of these techniques is flexibility and fast deployment. We see them as complementary to our approach, especially for initial exploration when the NoC component library is not available yet.

The approach in [18], on the other hand, attempts to build a cycle-accurate power model of a target router instance. However, several major points differentiate our approach. First, we build a model which is parametric not only on traffic-related events, but also on the architectural knobs of the design. Second, we include an area model in the exploration. Third, our model can be more readily adopted within a CAD mapping flow; this is both because we express the model as a function of architectural parameters, and because we provide a high-level dependence on traffic variables, instead of a cycle-by-cycle one. Fourth, we strive to make our approach as applicable as possible in real-world conditions, including the hard-to-model peculiarities of the behavior of synthesis tools when aiming for maximum frequency operation, and placement and routing issues. Fifth, we propose a fast characterization mechanism, by means of which model coefficients can be quickly derived with a minimal amount of synthesis runs.

In [19], a framework for NoC exploration is presented; the framework includes a power modeling flow. The power model features very limited dependence on architectural parameters and does not seem to account for the configuration knobs of synthesis tools. No area model is provided.

FIGURE 1: 4×4 xpipes switch architecture.

In [20], a bit energy modeling flow is proposed to compare different switch fabrics in IP network routers. The approach is focused on the cost for transmitting bits from input to output ports, and while bit pattern-accurate, it is only focused on comparing router topologies against each other. The authors of [21] propose a model based on transistor count, while in [22], which is focused on FPGAs, switch cardinality is the main parameter. None of these models is meant for simultaneously accurate, parametric, and fast representation of power consumption, that is, suitable for design space exploration within a CAD environment.

3. THE xPIPES SWITCH ARCHITECTURE

We choose the switch architecture defined in the xpipes NoC component library [8, 23] as a case study, due to its customizability. The xpipes switch (Figure 1) is output buffered; FIFOs of configurable depth are instantiated at each output, while inputs feature a single register. The flit width can be arbitrarily set. The number of input and output ports is also a parameter; full connectivity is provided in the central crossbar. An arbiter is attached to each output port to handle contention issues. We test the switch with its default ACK/NACK flow control mechanism, which leverages the output buffer resources. Since xpipes performs source routing, the switch does not include routing LUTs.

4. PROPOSED MODELING METHODOLOGY

Our modeling activity is composed of five main phases. First, we devise a set of parameters that are relevant to the accuracy of any model which aims at practical applicability. Second, we define a general model formula for area and for power, relying on the knowledge of the target architecture as explained in Section 3. Third, we synthesize several configurations (*training set*) of the target switch architecture in a $0.13\ \mu\text{m}$ technology library with Design Compiler [24], and measure the corresponding area and power consumption. The configurations are chosen so as to uniformly but sparsely cover the design space of interest, therefore allowing for an accurate yet quick construction of the model. Fourth, we use experimental results to numerically quantify the coefficients

FIGURE 2: Outline of our characterization activity. The placement and routing step is optional both for the training and the test set.

of the model. As outlined later, we propose two different ways of performing this step. Fifth, we assess the quality of our models against configurations (*test set*) outside of the training set. The first four steps will be covered in Sections 4.1, 4.2, 4.2.3, 4.3, while the fifth will be discussed in Section 5. An outline of how we handle steps 3–5 is provided in Figure 2.

4.1. Parameters of interest

A key phase of the approach is devising a model that matches the architecture under consideration and its properties. However, considering the architecture alone does not guarantee that the model will be applicable and accurate enough in practice. For example, synthesis tools play a primary role in defining the area and power efficiency of a component. Therefore, we first summarize the parameters of interest when assembling our model.

4.1.1. Architectural Parameters

The main parameters are

- (i) switch cardinality (number of ports); to account for rectangular switches, we separately consider the amount of input ports (np_i) and output ports (np_o);
- (ii) amount of buffering devoted to flow control handling and performance optimization, also called buffer depth (bd) (expressed in terms of single-flit buffering elements);
- (iii) number of bits of the incoming and outgoing elementary data blocks, also called flit width (fw).

FIGURE 3: Area requirements versus target operating frequency.

4.1.2. Implementation flow parameters

It is possible to tune synthesis tools, among other things, for

- (i) target frequency of operation;
- (ii) target area;
- (iii) target power consumption.

Tuning these parameters differently in the synthesis tools yields, as expected, a widely different quality of results. For example, when performance demands are extreme, synthesis tools are forced to create netlists containing large amounts of buffers and fast gates, which are not area- and power-efficient. To mimic a typical industrial flow, where an application performance constraint must be satisfied, we impose as the primary objective a certain target operating frequency (which is a parameter of our model), while area and power minimization are given to the tool as secondary optimization objectives. As a result, area and power requirements, expressed as a function of the target operating frequency, exhibit a characteristic flat behavior followed by a steeply rising trend after an inflection point. This trend is well known, and can be explained by the fact that, above some target operating frequency which can be achieved with minimal circuitry, synthesis backends are forced to insert extra gates to comply with increasing performance demands.

Figure 3 shows a linearized and a parabolic approximation of this trend, and at the same time summarizes the ways we modeled this effect. For each device configuration (e.g., 4×4 32-bit switches with 6-deep FIFO buffers), a “native” frequency f_n can be identified. This frequency is that achieved by the synthesizer with relaxed timing constraints. Under this condition, the tool is free to fully pursue its secondary objectives, hence creating minimum area ($A(f_n)$) and power ($P(f_n)$) netlists. Configuring the tools for target frequencies lower than f_n does not result in further decreases of area or power dissipation. For each switch instance, it is also possible to find a frequency $f_{\max}$, that corresponds to the fastest achievable synthesis result. Under this timing constraint, the module has $A(f_{\max})$ area and $P(f_{\max})$ power consumption. We approximate the dependency of area and

power overheads as linear or parabolic in the range $(f_n; f_{\max})$. This assumption allows us to characterize devices only twice, at f_n and $f_{\max}$ (under various combinations of the other architectural parameters), while being able to estimate results over the whole range of frequencies achievable by the module. Since this analysis is not correlated to other model parameters, in the following, for simplicity of notation, we will not explicitly mention the dependency of coefficients on the synthesis target frequency; the characterization of this parameter will be implicitly assumed.

The linearized or parabolic approximation is a way of abstracting away from low-level details of the logic synthesis process, which are impossible to capture in a high-level model. The experimental results that will be shown in Section 5 will be based on a test set which is also spread in terms of target operating frequency, therefore providing a metric of the accuracy of such a model. Section 5.5 will compare the accuracy of the linear versus the parabolic models.

Please note that developing area and power models which are a function of the target frequency of operation up to $f_{\max}$ also implies making available a model of the timing properties of the switches.

4.1.3. Traffic condition parameters

These parameters are only relevant to power models, since area models are clearly static. They include downstream congestion and internal congestion (i.e., arbitration conflicts). They will be explained in more detail in Section 4.2.2.

4.2. Area and power models

4.2.1. Area model

In general, the area equation must be of the form:

$$A = f(bd, fw, np_o, np_i). \quad (1)$$

We identify as suitable the area model:

$$\begin{aligned} A(fw, bd, np) = & A_1 \cdot np_o \cdot fw \cdot bd + A_2 \cdot np_i \cdot fw \\ & + A_3 \cdot np_o \cdot np_i + A_4 \cdot fw \cdot np_o \cdot np_i. \end{aligned} \quad (2)$$

The rationale of this formula is that the area of the target switch can be rendered as the sum of four contributions (Section 3): (i) output buffers, (ii) input buffers, (iii) arbitration and flow control logic, (iv) crossbar. Each contribution strongly depends on a known combination of architectural parameters as follows.

- (i) Output buffers, which are dominated by flip-flop area, can be supposed to depend linearly on flit width fw and buffer depth bd ($\times$ pipes switches are output-buffered), which, respectively, represent the width and depth of the buffer (Figure 4), there are np_o such buffers.
- (ii) Input buffers are similar to the case above, but since they have a constant depth, they do not depend on bd . Obviously np_i is used in place of np_o .

FIGURE 4: Dependency of the output buffer area on fw , bd .

- (iii) Since a distributed arbitration technique is used in the target switch, one arbiter is instantiated at each output port. Each arbiter has a complexity proportional to the number of candidate input ports np_i , therefore the overall contribution is the product of the input and output cardinalities. The arbiter logic is clearly independent of datapath parameters such as flit width and buffer depth.
- (iv) The area overhead due to the crossbar must have a linear dependency on flit width, must be independent of the buffering resources, and must have a linear dependency on the product of input and output cardinalities.

4.2.2. Power model

The power consumption of a module depends on the switching activity of the cells, so, to express the power consumption of an NoC switch, a term that accounts for traffic conditions must be present. The most general way to model the power consumption thus becomes

$$P = f(bd, fw, np_o, np_i, T) \quad (3)$$

with T being a generic variable that summarizes the traffic conditions. Since sequential components exhibit a power consumption even if they are not performing computation, due to the clock switching, a static (traffic-independent) term must appear. After analyzing the possible traffic flows in the $\times$ pipes router, we propose (4) as a general power model:

$$\begin{aligned}
 P(bd, fw, np_o, np_i, T) = & P_A(\dots) + \sum_{j=1}^{np_o} [P_B(\dots) \cdot T_{Oj}] \\
 & + \sum_{j=1}^{np_o} [P_C(\dots) \cdot T_{OCj}] \\
 & + \sum_{j=1}^{np_i} [P_D(\dots) \cdot T_{ICj}],
 \end{aligned} \quad (4)$$

where the dots express dependencies on bd , fw , np_o , np_i which will be analyzed in more depth in the following. The first term models the power dissipated by inactive, but still

FIGURE 5: Example traffic in a 4×2 switch.

clocked, registers. The remaining terms depend on traffic conditions. An accurate representation of the traffic conditions requires a separate analysis of the state of each input and output port. Therefore, we define np_o traffic variables T_{Oj} and T_{OCj} , to model the lack or presence of external congestion, and np_i traffic variables T_{ICj} , to model internal contention for resources. More specifically, we define the following.

- (i) T_{Oj} : percentage of time during which the output port j is successfully transmitting flits. This coefficient models traffic in absence of congestion.
- (ii) T_{OCj} : percentage of time during which the output port j is trying to transmit, but flits are rejected. This coefficient models external congestion due to traffic spikes.
- (iii) T_{ICj} : percentage of time during which the input port j of the switch is trying to transmit flits through one of the output ports, but arbitration is denied by the switch logic. This coefficient models the contention for the same output port inside of the switch.

This set of traffic percentages is linearly independent, since the complex arbitration and flow control patterns within an NoC switch make it very easy for some of these time windows to overlap. Please consider the following.

Example 1. A 4×2 switch (see Figure 5) may feature one established input-to-output connection where traffic is freely flowing (which is expressed by the condition T_{O1}), another established input-to-output connection which is stuck due to congestion in the downstream switch (modeled within T_{OC2}), while the third input port is unsuccessfully trying to transmit to one of the two output ports, which in this example are already busy (T_{IC3}), and the fourth is simply idle (this contribution is therefore included in the coefficient P_A).

The coefficients P_A , P_B , P_C , P_D depend on architectural parameters, as for the area model. They account for the power consumption in the traffic states, described above, as follows.

- (i) P_A accounts for the static power dissipated by the switch and it is due to the noncombinational logic in

TABLE 1: Dependency on architectural parameters of the static power coefficient P_A .

Contribution	Dep. on fw	Dep. on bd	Dep. on np_i	Dep. on np_o
Output buffering	Linear	Linear	None	Linear
Input buffering	Linear	None	Linear	None
Spare registers	None	None	Linear	Linear

the design. Therefore, it simply depends linearly on the number of flip-flops in the design, which are

- (a) input buffers;
- (b) output buffers;
- (c) state registers in the control logic;

whose dependencies on architectural parameters are summarized in Table 1.

- (ii) P_B accounts for the dynamic power dissipated by flowing packet streams, due to the enabled registers and to the switching activity of combinational logic. We identify four contributions to the power dissipation:

- (a) output buffers: in these buffers, during every cycle, one of the flit registers (fw bits wide) samples a new piece of data; a $bd \times 1$ multiplexer then brings a flit to the output port therefore, this contribution is itself the sum of two terms;
- (b) input buffers;
- (c) control logic;
- (d) selected crossbar branch.

The dependencies of these contributions on the architectural parameters are summarized in Table 2.

- (iii) P_C accounts for the dynamic power dissipated by the switch under a scenario where downstream congestion is preventing a free flow of packets. Although numerically different, the P_C coefficient is similar to P_B , in that it still involves an established input-to-output channel, and therefore its dependency on architectural parameters is the same (see Table 2).
- (iv) P_D accounts for the power dissipated by the switch when an incoming stream requires the access to an output port, but the arbitration is denied. The contributions to this portion of the power consumption are related to the following logic blocks:

- (a) input buffers;
- (b) control logic.

We can summarize the dependencies on this contributions as shown in Table 3.

The dependencies of the power coefficients are thus summarized in Table 4.

We would like to stress that some coefficients, which could be intuitively expected to quadratically depend on parameters, are instead linearly dependent, because they characterize a single input or output port. The quadratic behavior is indirectly restored by the summation symbols in (4).

4.2.3. Choice of a relevant training set

To characterize the coefficients of our area and power models, we define a training set, composed of switch configurations chosen in such a way as to uniformly cover the relevant design space for the particular NoC under study. In the case of the $\times$ pipes NoC, which is focused on the highest customizability of topologies, we choose to study a design space spanning over a large variety of cardinalities (np_i and np_o of 4, 10, 16, and 20). Since $\times$ pipes is also focused on the best performance/overhead tradeoff point [23], and therefore on low hardware cost, we consider moderate buffer depths bd of 5 and 7 FIFO locations and flit widths fw of 21, 28, and 38 bits.

In the modeling approach called full factorial design, all the possible permutations of the values of the independent design parameters should be studied to create the training set. This is often impractical due to the quick rise in the number of instances as soon as new design knobs are added, leading to approaches to select only a subspace of the characterization set (fractional factorial design). In our case, based on the knowledge of the target architecture, we choose a very simple way of pruning the training set. The rationale is based on the observation that rectangular switches add a smaller amount of information to the training set; for example, when studying the power consumption, a rectangular switch is by design unable to simultaneously feature traffic flows on all of its input and output ports (see Figure 5), and is therefore behaving similarly to a square switch of smaller cardinality. Our preliminary internal testing confirms this property, at least for the $\times$ pipes NoC. Therefore, we simply choose to coalesce the np_i and np_o axes for the generation of the training set, and only include 4×4 , 10×10 , 16×16 , and 20×20 instances.

We finally permute all the possible parameter values, resulting in 24 (4 cardinalities times 2 buffer depths times 3 flit widths) configurations being synthesized.

4.3. Fitting model coefficients

4.3.1. Fitting area model coefficients

To estimate A_1 , A_2 , A_3 , A_4 , we propose two different methods.

- (i) Methodology 1: coefficients can be derived directly from synthesis reports, which hierarchically list every switch subblock. For example, once the area cost of an output buffer which is bd_0 flits deep and fw_0 bits wide is gathered from one report, it can be called $A_{\text{obuf}}|_{bd_0, fw_0}$. Since A_1 is expected to increase linearly with both bd and fw , it can be approximately derived as:

$$A_1 = \frac{A_{\text{obuf}}|_{bd_0, fw_0}}{bd_0 \cdot fw_0}. \quad (5)$$

Other coefficients can be similarly computed.

TABLE 2: Dependency on architectural parameters of the dynamic power coefficients P_B , P_C .

Contribution	Dep. on fw	Dep. on bd	Dep. on np_i	Dep. on np_o
Output buffer (reg.)	Linear	None	None	None
Output buffer (mux)	Linear	Linear	None	None
Input buffer	Linear	None	None	None
Control logic	None	None	Linear	None
Crossbar branch	Linear	None	Linear	None

TABLE 3: Dependency on architectural parameters of the dynamic power coefficient P_D .

Contribution	Dep. on fw	Dep. on bd	Dep. on np_i	Dep. on np_i
Input buffer	Linear	None	None	None
Control logic	None	None	Linear	Linear

TABLE 4: Dependency of power coefficients on architectural parameters.

Model coefficient	Dep. on fw	Dep. on bd	Dep. on np_i	Dep. on np_o
P_A	Linear	Linear	Linear	Linear
P_B	Linear	Linear	Linear	None
P_C	Linear	Linear	Linear	None
P_D	Linear	None	Linear	Linear

Advantages: with this methodology, each contribution in the formula keeps a strict physical meaning. Only one synthesis run is needed to extrapolate coefficients for any switch instance; we arbitrarily choose a 10×10 , 28-bit switch as a reference. This instance is close to the center of the design space of interest (see the previous subsection); its choice will be further discussed in Section 5. Disadvantages: this simplified approximation discards any constant offset that may be present in the coefficients. Further, the nature of synthesis tools introduces unpredictable fluctuations in the netlist area and power trends under different architectural configurations. This noise does not have any easily characterizable property. Thus, the model incurs a nonnegligible error when compared against actual switch instances. Moreover, the choice of the specific switch instance for characterization might skew the computed coefficient values.

- (ii) Methodology 2: coefficients can be derived by leveraging the multivariate nonlinear regression algorithms natively provided by several mathematical and statistical packages. In this case, the input is a set of characterization syntheses (the training set described in the previous subsection). The target polynomial for the regression is chosen based on insight of the dependency of area on the architectural design parameters (see (2)). Advantages: the model fits actual synthesis results better. Disadvantages: longer characterization time;

with a thorough characterization set like that chosen in Section 4.2.3, experiments must be performed in 24 device instances, against just one. The actual improvement in accuracy depends on the smoothness of the native behavior of the synthesis tools. Some coefficients may lose their physical meaning (e.g., they may become negative).

Both methodologies can be readily adapted to any parameterizable NoC architecture.

4.3.2. Fitting power model coefficients

To characterize the P_A , P_B , P_C , P_D coefficients, we first inject traffic into the switch netlists under test, one at a time. This is achieved by ModelSim [25] simulation of the Verilog netlists (please refer to Figure 2), to which traffic generators are attached. The traffic generators are configured to inject into the switch one of the four patterns described above (idle, free flow, downstream congestion, internal contention) at a time. The switching activity is logged and fed as an input to Synopsys PrimePower [26], which provides a hierarchical report of the power consumption of the switch subblocks. For each netlist of the training set, four hierarchical reports are therefore generated.

At this point, the power model coefficients are determined by using either of the techniques just outlined for the area models. The P_A , P_B , P_C , P_D scenarios are separately accounted for; the fitting polynomials are directly derived from Table 4. For each of them, either by direct derivation or by nonlinear regression, we extract the coefficients modeling the dependency on architectural parameters.

5. EXPERIMENTAL RESULTS

To evaluate the accuracy of the proposed techniques, we first randomly choose a test set of 70 switch configurations spread across the design space of interest (both in terms of architectural parameters and target synthesis frequencies), and not

overlapping with the training set previously used for characterization. Each switch is synthesized with Design Compiler to extract its area requirements, then stimulated with traffic streams within ModelSim and studied in PrimePower to evaluate its power consumption (Figure 2). A reference set of experimental results is therefore collected. The area and power consumption of the same set of switches is then estimated according to our methodology, and the statistical distribution of the resulting error is plotted to study the behavior of both coefficient fitting strategies.

The implementation flow of any circuit design is composed of several steps, among which the two major ones are logic synthesis (i.e., mapping functionality onto elementary cells from a technology library), which results in a netlist, and placement and routing (i.e., placing and interconnecting the netlist within a target floorplan), generating a layout as the outcome. Netlists can be generated in a relatively short time, but they do not include any information about the placement of the cells, and thus do not give any information about the length of the wires needed for the interconnections. This is a key missing piece of information, especially as designs become wire-dominated. Therefore, logic synthesis tools try to model the effect of wires by means of predictive models provided by the vendor of the technology library. These models are necessarily simplistic, and therefore may impact the accuracy of any area and power evaluation at the netlist level. On the other hand, creating the layout of a complex circuit provides more accurate estimations of its area and power cost, but this extra step is at least as time-consuming as the initial logic synthesis. Therefore, designers would clearly like to avoid performing this extra phase repeatedly during a modeling activity, if at all possible.

To assess the usefulness of our models, we investigate their inference and their application to both netlists and layouts. This can be seen in Figure 2, where the placement and routing step is optional.

5.1. Experiments with netlist-based models and a netlist-level test set

In this subsection, we generate our models starting from synthesized (but not placed and routed) switch instances, and check their accuracy against a test set which is also at the netlist level. The results are depicted in Figure 6, where the vertical axis reports the number of occurrences of inaccuracies comprised in the ranges listed on the horizontal axis. As can be seen, in around 80% of the cases, our models result in an error margin less than 10% of the actual value. Sporadically, relatively high error rates of up to 20% are detected; however, as can be seen for example in Figure 7, the distribution of the errors is quite randomly spread over the design space, and comprises both under- and overestimations. The figure reports modeling inaccuracy for a subspace having as axes the flit width and the switch cardinality; these numbers are thus only a subset of the whole test set. Similar plots can be derived for varying buffer depths and target synthesis frequencies, and we omit them due to space constraints. Therefore, we can attribute inaccuracies to the

(a)

(b)

FIGURE 6: Modeling inaccuracy (percentage deviation among predicted and actual values of (a) area, (b) power, for the test set) under two different characterization policies for the coefficients. Vertical axis represents the occurrence frequency of a given inaccuracy range. Models and test set are at the netlist level.

unpredictability which is intrinsic in the behavior of synthesis tools, and not to a problem of our modeling approach.

Comparing the results of the two techniques for coefficient fitting presented in Section 4.3, we see that the tails of the inaccuracy distributions drop more sharply for Methodology 2, indicating a lower chance of large modeling errors. However, Methodology 1 exhibits just marginally worse average inaccuracy rates: 6.26% against 5.30% for power models and 5.97% against 5.45% for area models. In terms of characterization effort, in our experience, we can roughly assume that one hour may be needed in average for the analysis of an instance of the training set; therefore, Methodology 1 requires one hour of runtime, while Methodology 2 needs 24 hours to provide numerical values of coefficients (the actual time depends on how thoroughly the design space

FIGURE 7: Distribution of the area modeling inaccuracy over a subset of the design space for Methodology 2. Dark color: underestimations; light color: overestimations.

is covered). Due to the drastically lower effort, Methodology 1 becomes a natural candidate for fast yet accurate modeling. However, this approach leverages upon a single switch instance to characterize all the coefficients. The choice of the reference switch configuration is therefore key, and may impact the robustness of the flow. Internal testing, that we omit due to space constraints, shows that coefficients are quite accurately rendered under a wide range of possible choices of the reference switch. However, when manually picking an “outlier” instance as the reference, errors over the whole design space turn out to be large. As a possible workaround, Methodology 1 could be applied to multiple switch instances to minimize the chance of choosing bad references; outliers could be effectively discarded. This hybrid approach provides better reliability, but requires a modeling effort which is progressively closer to that of Methodology 2 as its robustness is increased. Methodology 2 remains the most accurate and reliable, and its characterization time can still be assumed to be fully acceptable for both academic and industrial environments.

5.2. Test case: a complete NoC topology

To further validate the most complex part of our methodology, that is, the power modeling, we study a whole NoC topology, such as a 5×3 mesh. The mesh includes switches with three different cardinalities of 4×4 , 5×5 , and 6×6 . We then inject functional traffic, namely, that required to drive a multimedia application from the MPARM [27] suite, in the topology, and compare the resulting power consumption against that predicted by our model (characterized with Methodology 2). Traffic patterns in the mesh are irregular, due to application needs, causing the switches to spend variable amounts of time in each possible state. The results are

FIGURE 8: Distribution of the power modeling inaccuracy for the switches of a 5×3 NoC mesh.

plotted in Figure 8. The average inaccuracy is 5%, with only two switches out of fifteen (about 13%) exhibiting inaccuracies greater than 10%. Since the power consumption of some switches is overestimated while that of others is underestimated, the margin of error on the consumption of the whole mesh is as low as 1.3%. This result confirms the usefulness of our modeling strategy for integration within a CAD mapping and design space exploration flow.

5.3. Experiments with netlist-based models and layout-level test set

We try to apply the previously mentioned models, which are based on netlist-level analyses, to a layout-level test set, by placing and routing the test set described above. This activity generates a very realistic test set, and is a demanding metric for the accuracy of the models, since extra unpredictable

FIGURE 9: Modeling inaccuracy (percentage deviation among predicted and actual values of power for the test set) under two different characterization policies for the coefficients. Vertical axis represents the occurrence frequency of a given inaccuracy range. Models are at the netlist level, test set is at the layout level.

noise is added. The results we get are presented in Figure 9, which should be compared to Figure 6(b). The two plots exhibit a comparable trend and errors of roughly the same magnitude, even though the average modeling error for the layout-level test set is about 3% higher. This means that models developed by only taking netlists into account still show good accuracy even with respect to layout-level power evaluation. The added noise also blurs the accuracy difference between Methodology 1 and Methodology 2, both in maximum error (26% versus 23%) and average error (8.8% versus 8.35%). While Methodology 2 remains marginally more accurate, these results seem to suggest that the unpredictability introduced by the logic synthesis process is somewhat unrelated to that introduced by the placement and routing phase. In other words, even though Methodology 2, thanks to its interpolation of results, can compensate for some of the non-idealities of the logic synthesis process better than Methodology 1, this compensation is less effective when trying to predict the power consumption after placement and routing.

5.4. Experiments with layout-based models and a layout-level test set

In an attempt to check whether more accurate models can be built, we recompute the numerical coefficients starting from a layout-level version of the training set and applying Methodology 2. This model is very close to an ideal reference point, since it is derived from a regression on experimental results which already encompass most of the unpredictable elements of the synthesis flow. However, the time required to build the model coefficients is noticeably longer. In our experience, both logic synthesis and placement steps require a computation time which is not easy to predict, as it largely depends on many factors, such as the switch cardinality and the target operating frequency. However, as a rule of thumb,

FIGURE 10: Modeling inaccuracy (percentage deviation among predicted and actual values of power for the test set) with Methodology 2 characterization policy for the coefficients. Vertical axis represents the occurrence frequency of a given inaccuracy range. Models and test set are at the layout level.

the two steps are about equally time consuming; therefore, the modeling time is approximately doubled.

The error distribution resulting from the usage of the layout-level test set when validating the model coefficients achieved from a layout-level training set is shown in Figure 10.

As can be noticed, and as expected, the average error and the maximum error values both noticeably decrease when compared to Figure 9. However, the decrease is not huge. We attribute the remaining inaccuracies in Figure 10 to the intrinsic unpredictability of the synthesis tools. Even after taking into account all the systematic behaviors in the synthesis flow, the trend is the result of residual instance-to-instance variations due to heuristics in the CAD tools and to degrees of freedom which can only vary in a discrete fashion.

The accuracy improvement guaranteed by a layout-level characterization is associated to a doubling of the runtime overhead, and still does not completely eliminate the presence of some “outlier” instances. The designer may certainly choose to adopt our methodology to characterize devices at the layout level for maximum accuracy. However, we feel that a result that can be derived from our experiments is that, at least at the 0.13 μm technology node, it is still feasible to use accurate netlist-based models in order to save characterization time.

5.5. Experiments with a parabolic model for the dependency on the target synthesis frequency

We conclude our experiments by checking whether a linear model is accurate enough to characterize the dependency of synthesis results on the target synthesis frequency (see Figure 3). We leverage a parabolic model as a potentially more accurate approximation of the actual dependency of model coefficients on the target frequency, then recheck the

TABLE 5: Accuracy of the linear versus parabolic models for the dependency of synthesis results on the target synthesis frequency. Coefficients derived with Methodology 2.

Experiment		Linear approx.	Parabolic approx.
Netlist training set,	Average error	5.19%	6.32%
Netlist test set	Maximum error	14.61%	15.27%
Netlist training set,	Average error	8.23%	6.57%
Layout test set	Maximum error	22.88%	19.94%
Layout training set,	Average error	5.04%	9.23%
Layout test set	Maximum error	16.10%	22.23%

model accuracy on the test set. The results are reported in Table 5.

These results do not seem to indicate a strong bias towards any of the alternatives. The linear approximation seems to cope much better with a netlist-level or layout-level test set when the model is derived from experiments on a training set at the same level, but the parabolic model is quite a bit better at predicting layout-level results starting from netlist-level models. We attribute this behavior to the impact of noise. In other words, although synthesis results do clearly change depending on the target frequency, the choice of a linear or parabolic model to describe this trend does not matter much, since the nonidealities introduced by the synthesis flow induce enough noise to blur the distinction. Overall, the usage of the linear model, which is simpler, seems to be justified.

6. CONCLUSIONS AND FUTURE WORK

We presented a methodology for characterization of NoC switch area and power requirements. The approach we propose is based on thorough parameterization on several architectural, deployment, and runtime variables. This guarantees excellent applicability within an NoC CAD flow for topology mapping and/or design space exploration. The area and power models for the $\times$ pipes case study turn out to be very accurate within the limits allowed by the nonidealities of synthesis tools, even when applied to a whole NoC topology with irregular traffic flows. Our experiments show that, at least at the $0.13\ \mu\text{m}$ node, applying our methodology to netlist-level devices yields an acceptable approximation of the actual behavior even after placement and routing, but that even greater precision can be achieved, if desired, by applying the same technique at the layout level. We also show that another tradeoff among accuracy and modeling effort exists, namely, coefficients can be extracted based on a single (or on just a few) device instances, by normalization of the synthesis report, or on several of them, by an interpolation process.

Future work includes minimizing the characterization effort, testing how well our technique scales to finer process technologies, and creating similar models for other NoC components, such as network interfaces (NIs). At the $0.13\ \mu\text{m}$ node, we observe that the power consumption of NoCs is still largely dominated by dynamic switching activity [23], therefore we do not consider wire loads in our modeling approach for switches. Since this is expected to change in

future technologies, techniques to establish models for NoC links may also become important.

Besides the absolute accuracy of the models, we also plan on quantifying the accuracy of our model when used from within a CAD flow to establish relative cost assessments of alternative NoC topologies; early results in this activity are encouraging, with good agreement between CAD expectations and actual measurements.

ACKNOWLEDGMENTS

This work has been supported by STMicroelectronics and by the Semiconductor Research Corporation (SRC) under contract 1188.

REFERENCES

- [1] P. Meloni, S. Carta, R. Argiolas, L. Raffo, and F. Angiolini, "Area and power modeling methodologies for networks-on-chip," in *Proceedings of the 1st International Conference on Nano-Networks (Nano-Net '06)*, Lausanne, Switzerland, September 2006.
- [2] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference (DAC '01)*, pp. 684–689, Las Vegas, Nev, USA, June 2001.
- [3] L. Benini and G. de Micheli, "Networks on chips: a new SoC paradigm," *Computer*, vol. 35, no. 1, pp. 70–78, 2002.
- [4] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet-switched interconnections," in *Proceedings of Design, Automation and Test in Europe Conference and Exhibition (DATE '00)*, pp. 250–256, Paris, France, March 2000.
- [5] F. Karim, A. Nguyen, S. Dey, and R. Rao, "On-chip communication architecture for OC-768 network processors," in *Proceedings of the 38th Design Automation Conference (DAC '01)*, pp. 678–683, Las Vegas, Nev, USA, June 2001.
- [6] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "QNoC: QoS architecture and design process for network on chip," *Journal of Systems Architecture*, vol. 50, no. 2-3, pp. 105–128, 2004.
- [7] T. Bjerregaard and J. Sparsø, "scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC '05)*, pp. 34–43, New York, NY, USA, March 2005.
- [8] F. Angiolini, P. Meloni, D. Bertozzi, L. Benini, S. Carta, and L. Raffo, "Networks on chips: a synthesis perspective," in *Proceedings of the International Conference on Parallel Computing (ParCo '05)*, pp. 745–752, Malaga, Spain, September 2005.

- [9] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based NoC architectures under performance constraints," in *Proceedings of the Conference on Asia and South Pacific Design Automation (ASP-DAC '03)*, pp. 233–239, Kitakyushu, Japan, January 2003.
- [10] S. Murali and G. de Micheli, "SUNMAP: a tool for automatic topology selection and generation for NoCs," in *Proceedings of the 41st Design Automation Conference (DAC '04)*, pp. 914–919, San Diego, Calif, USA, June 2004.
- [11] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *Proceedings of the 27th Annual International Symposium on Computer Architecture (ISCA '00)*, pp. 83–94, Vancouver, BC, Canada, June 2000.
- [12] W. Ye, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "The design and use of simplepower: a cycle-accurate energy estimation tool," in *Proceedings of the 37th Conference on Design Automation (DAC '00)*, pp. 340–345, Los Angeles, Calif, USA, June 2000.
- [13] A. Bona, V. Zaccaria, and R. Zafalon, "System level power modeling and simulation of high-end industrial network-on-chip," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE '04)*, vol. 3, pp. 318–323, Paris, France, February 2004.
- [14] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff, "Energy characterization of a tiled architecture processor with on-chip networks," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED '03)*, pp. 424–427, Seoul, Korea, August 2003.
- [15] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: a power-performance simulator for interconnection networks," in *Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture (MICRO '02)*, pp. 294–305, IEEE Computer Society Press, Istanbul, Turkey, November 2002.
- [16] H.-S. Wang, L.-S. Peh, and S. Malik, "A technology-aware and energy-oriented topology exploration for on-chip networks," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '05)*, vol. 2, pp. 1238–1243, Munich, Germany, March 2005.
- [17] N. Easley and L.-S. Peh, "High-level power analysis of on-chip networks," in *Proceedings of the 7th International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES '04)*, pp. 104–115, Washington, DC, USA, September 2004.
- [18] J. Chan and S. Parameswaran, "NoCEE: energy macro-model extraction methodology for network on chip routers," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '05)*, pp. 254–259, San Jose, Calif, USA, November 2005.
- [19] G. Palermo and C. Silvano, "PIRATE: a framework for power/performance exploration of network-on-chip architectures," in *Proceedings of the 14th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS '04)*, pp. 521–531, Santorini, Greece, September 2004.
- [20] T. T. Ye, L. Benini, and G. de Micheli, "Analysis of power consumption on switch fabrics in network routers," in *Proceedings of the 39th Design Automation Conference (DAC '02)*, pp. 524–529, New Orleans, La, USA, June 2002.
- [21] C. S. Patel, S. M. Chai, S. Yalamanchili, and D. E. Schimmel, "Power constrained design of multiprocessor interconnection networks," in *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD '97)*, pp. 408–416, Austin, Tex, USA, October 1997.
- [22] H. Zhang, M. Wan, V. George, and J. Rabaey, "Interconnect architecture exploration for low-energy reconfigurable single-chip DSPs," in *Proceedings of the IEEE Computer Society Workshop on VLSI '99 (IWV '99)*, pp. 2–8, Orlando, Fla, USA, April 1999.
- [23] F. Angiolini, P. Meloni, S. Carta, L. Benini, and L. Raffo, "Contrasting a NoC and a traditional interconnect fabric with layout awareness," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '06)*, vol. 1, pp. 124–129, Munich, Germany, March 2006.
- [24] Synopsys Inc., "Design Compiler," <http://www.synopsys.org/>.
- [25] Mentor Graphics, "ModelSim," <http://www.model.com/>.
- [26] Synopsys Inc., "PrimePower," <http://www.synopsys.org/>.
- [27] M. Loghi, F. Angiolini, D. Bertozzi, L. Benini, and R. Zafalon, "Analyzing on-chip communication in a MPSoC environment," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE '04)*, vol. 2, pp. 752–757, Paris, France, February 2004.