Improved On-Chip Router Analytical Power and Area Modeling

Andrew B. Kahng[†‡], Bill Lin[‡] and Kambiz Samadi[‡]

[†]CSE and [‡]ECE Departments, UC San Diego, La Jolla, CA

***Abstract*—Over the course of this decade, uniprocessor chips have given way to multi-core chips which have become the primary building blocks of today's computer systems. The presence of multiple cores on a chip shifts the focus from computation to communication as a key bottleneck to achieving performance improvements. As industry moves towards many-core chips, networks-on-chip (NoCs) are emerging as the scalable fabric for interconnecting the cores. With power now the first-order design constraint, early-stage estimation of NoC power has become crucially important. Existing power models (e.g., ORION 2.0 [12], Xpipes [7], etc.) are based on certain router microarchitecture and circuit implementation. Therefore, when validated against different NoC prototypes - different router implementations – we saw significant deviation (up to 40% on average) that can lead to erroneous NoC design choices. This has prompted our development of a new, accurate architecture- and circuit implementation-independent router power and area modeling methodology with complete portability across existing NoC component libraries. Also, validation against a range of implemented router designs confirms substantial improvement in accuracy over existing models.**

I. Introduction

As diminishing returns in performance of uniprocessor architectures have led to multi-core chips, networks-on-chip (NoCs) are emerging as the scalable fabric for interconnecting the cores. When the number of on-chip cores increases, the need for scalable and high-bandwidth communication fabric becomes more evident [8], [16]. However, with increasing demand for network bandwidth, interconnection network power has become increasingly substantial, e.g., 28% and 36% of the total chip power in the Intel 80-core [10] and Raw [21] processors, respectively. This requires designers to work within a tight power budget. To aid designers in early stages of the design, architectural power models have been proposed for rough power estimations (see Section II).

Architectural power estimation is used to (1) verify that power budgets are approximately met by the different parts of the design and the entire design, and (2) evaluate the effect of various high-level optimizations, which have been shown to have much more significant impact on power than low-level optimizations. Therefore, a number of recent research efforts (e.g., [12], [17], [22]) have focused on developing architectural router power models. However, these models are derived from a mix of template circuit implementations and technology trends, and hence can not guarantee accuracy within an architecture-specific flow. The main advantages of these models are flexibility and applicability in early design space explorations when the NoC component library is not available. In addition, existing router power models [5], [18] do not consider all relevant microarchitectural parameters, which limits accuracy and applicability.

The above shortcomings of the existing models has led us to explore new directions to improve the efficiency of these models for early-stage design space exploration. To accomplish our goal, we start from an existing RTL description of a router with any given architecture. We then create a library of fully-synthesizable router RTLs with different microarchitectural parameters. Using an industrial implementation flow we take the RTL descriptions through physical design steps and compute corresponding power and area values. We apply a machine learning technique, i.e., non-parametric regression, using the generated power and area data sets to develop accurate architectural-level router power and area models. The highlight of our modeling methodology is the decoupling of the router microarchitecture and underlying circuit implementation from the modeling effort. One might question the runtime complexity of the approach; however, as technology advances every year, designers find themselves with increasing access to multiple, parallel computing resources in the form of multi-processor workstations, multi-core microprocessors, load sharing facilities (LSF), etc. [20]. These ample computing resources can be efficiently utilized to enhance modeling accuracy and applicability. The contributions of our work are as follows.

1) We are the first to consider decoupling of router microarchitecture and underlying circuit implementations from the modeling effort. This enables a modeling methodology in which both router microarchitecture and underlying circuit implementations are transparent to the system-level designer.
2) We use nonparametric regression to develop accurate router microarchitectural power and area models using power and area data sets gathered from layout information.
3) We propose a modeling methodology for on-chip router power and area that can be used within any existing NoC component library.

The remainder of this paper is organized as follows. In Section II we contrast against prior related work. Section III describes our flow implementation for our experiments and the details about our design of experiments. Section IV describes our modeling methodology, while Subsections IV-B and IV-C derive the corresponding power and area models, respectively. In Section V we validate our proposed models against layout data, and show the impact of the new models on achievable NoC configurations. Finally, Section VI concludes the paper.

II. Related Work

Power modeling can be carried out at different levels of fidelity, trading off modeling time with accuracy, ranging from real-chip power measurements [11], to pre- and post-layout transistor-level simulations [29], to RTL power estimation tools [30] to early-stage architectural power models [4], [12], [23]. Low-level power estimation tools, even RTL power estimation, require complete RTL code to be available, and simulate slowly, on the order of hours, while an architectural power model takes on the order of seconds [12]. Circuit-level power estimation tools, though providing excellent accuracy, has even longer simulation times, and require even more substantive development effort. These shortcomings have led to a plethora of architectural power models, such as the widely-used Wattch [4] and SimplePower [23] for uniprocessors.

Power models have also been proposed for a variety of on-chip networks. These models can be divided into two categories: (1) models derived from a mix of circuit templates, and (2) models derived from pre- and post-layout simulation results. In the first category, Patel *et al.* [19] first proposed an analytical power model for interconnection networks, derived from transistor count. Since the model does not consider the architectural parameters, it cannot be used to explore tradeoffs in router microarchitecture design. ORION 2.0 [12], a set of architectural NoC power models, is an enhanced version of its predecessor, ORION [22], which provides parameterized power and area models derived from a mix of circuit templates and technology trends. ORION models can be more efficiently deployed for early-stage design space exploration as they take into account the router microarchitectural parameters.

In the second category, power models are based on either pre-layout [3], [5], [15], [18] or post-layout [1], [2], [17] simulation results. Bona *et al.* [3] presented a methodology for automatically

978-1-4244-5767-0/10/$26.00 2010 IEEE

generating energy models for bus-based and crossbar-based communication fabrics. Lee *et al.* [15] proposed a high-level power model based on the number of flits passing through a router, and used parametric regression to derive the model. In [5] and [18], cycle-accurate architectural power models are proposed, however, limited dependence of these models on the microarchitectural parameters degrades their applicability for efficient design space exploration framework. Chan *et al.* [6] used the parametric models proposed in [5] to drive their NoC synthesis cost functions. On the other hand, Banerjee *et al.* [1] presented an accurate power characterization of a range of NoC routers using standard ASIC tool flow, but failed to present any analytical models. In [2], a RTL-level NoC power model was developed by first extracting the SPICE level netlist from the layout and then integrating the characterized values into the RTL description. More recently, Meloni *et al.* [17] proposed an architectural regression analysis for router power based on energy numbers obtained from post-layout simulations.

The existing models assume a specific architecture and underlying circuit implementation in their modeling efforts, in one way or the other. For example, [12] assumes certain architecture (i.e., VC router) and a set of circuit implementations or [17], even though claims that the proposed models are architecture-specific, it still requires the modeler to have understanding about the underlying architecture. In addition, most of the current work fail to consider the contributing microarchitectural as well as implementation parameters in their models. In contrast, we develop an architecture- / circuit implementation-independent router power and area modeling methodology. We efficiently utilize the accuracy of post-layout simulation results and use nonparametric regression to develop closed-form analytical power and area models. Finally, our models include all the router microarchitectural parameters to enhance its usability at the system-level and efficiency for design space explorations.

III. Implementation Flow and Design of Experiments

A. Implementation Flow and Tools

Figure 1 shows our implementation flow which includes traditional synthesis, placement and routing (SP&R) flow plus power estimation and model generation steps that we have scripted for "push-button" use in our experiments. The steps in Figure 1 represent the major physical design steps. At each step we require that the design must meet the timing requirements before it can pass on to the next step.

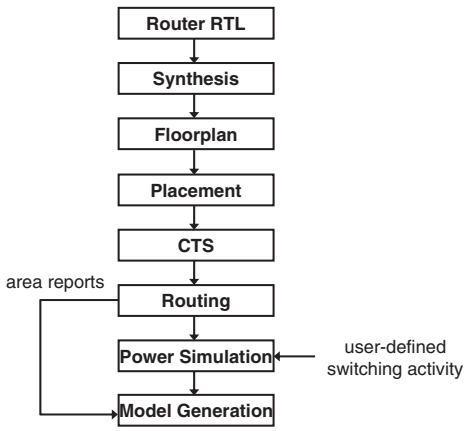

Fig. 1. Implementation flow.

In the flow, we first generate a library of fully-synthesizable router RTLs using *Netmaker* [27]. We generate the library by choosing a router microarchitecture and sweeping the corresponding microarchitectural parameters space. Then we synthesize RTL codes with worst-case timing libraries. During synthesis process we can set certain optimization objectives among which are: (1) target frequency, (2) target area, and (3) target power. Choosing different combinations of these objectives yields widely different quality of results. Hence, to mimic a typical industrial timing-driven flow, where certain performance constraints must be satisfied, we impose the target frequency as the primary objective while area and power minimization are considered as secondary objectives [17]. In addition to optimization parameters, there are some 'inherent noise' in IC implementation tools that may be aggravated with respect to tightness of the timing constraint (i.e., target frequency) [13]. Therefore, we implement the designs at a slightly slower frequency than the maximum achievable. To obtain near-maximum clock frequency we run several synthesis simulations with different target frequencies, and pick the clock frequency beyond which area is increased significantly without timing improvement.

Using synthesized netlist we implement the designs through place & route (P&R) steps. We set the row utilization and aspect ratio of all the designs to 65% and 1, respectively, at the floorplan stage. Since our goal is to develop microarchitectural router power and area models we do not perturb the row utilization and aspect ratio values after they are fixed. At the end of routing stage we extract the design power/area and associate them with the corresponding microarchitectural values.

We use *Netmaker v0.82* [27] to generate a library of synthesizable router RTLs. We perform our experiments using libraries in TSMC 130nm, 90nm, and 65nm technologies. Target clock frequencies for the above technologies are 200 MHz, 300 MHz, and 400 MHz, respectively. We use *Synopsys Design Compiler v2009.06-SP2* to synthesize the RTL designs and *Cadence SOC Encounter v7.1* [31] to execute the P&R flow. Finally, *MARS3.0* tool suite is used for model generation [28].

B. Design of Experiments

As with any other model characterization task, it is not trivial to obtain a meaningful power data set, as large number of variables determine the ultimate results. Hence, we only choose the microarchitectural parameters that are of interest at the system-level. In addition, as explained in Section III-A, we fix certain implementation-related objectives / parameters across the entire design of experiments. In our experiments, we use *Netmaker* to generate a library of fully-synthesizable parameterized router RTLs. We pick a baseline virtual channel (VC) router in which VC allocation and switch allocation are performed sequentially in one clock cycle. In a VC router the microarchitectural parameters are:

- fw: flit-width is the width of link or channel between routers and is specified in bits
- n_{vc}: number of virtual channels
- n_{port}: number of input and output ports (e.g., 5 for mesh topology: N, S, E, W, and tile connection)
- l_{buf}: buffer length which is the number of flit buffers per virtual channel (i.e., per input port)

TABLE I
List of router microarchitectural parameters used in our design of experiment.

Parameter	Variables
fw	{16, 24, 32, 64}-bits
n_{vc}	{2, 3, 5, 7}
n_{port}	{3, 5, 7, 9}
l_{buf}	{2, 3, 5, 7}-flit buffers

Using automation scripts we vary the above parameters and generate corresponding RTL for each combination of parameters. Table I shows the router microarchitectural parameters and the values they take on in our design of experiments. We then implement the RTL codes in three technology nodes using TSMC 130nm G, 90nm G, and 65nm GP libraries. As explained in Section III-A for synthesis step we use a target frequency slightly slower than

the near-maximum clock frequency. We also fix row utilization and aspect ratio values to 65% and 1, respectively. According to Table I there are 256 configurations (i.e., RTL descriptions) per technology node. This implies a total of 768 synthesis and 768 place and route (P&R) runs across all three technology nodes. Due to relatively simple router logic, each synthesis simulation takes up to 3 minutes, and each place and route simulation takes up to 15 minutes on a 2.0GHz Intel Xeon processor. Hence, using 10 CPUs we can perform the entire design of experiments in less than a day. Note that having access to LSF and/or other multiprocessor computing environments (as is the case in large companies) will significantly reduce the setup time to generate the necessary power and area data sets for our models. We also note that the router configuration space is quite large. For example, if we assume that buffer length can vary from 1 to 40 flits, number of ports can vary from 2 to 16 (for high radix routers), flit-width can vary from 8 to 128 bits in 8-bit increments, and number of VCs can vary from 1 to 10, then there are approximately 100000 possible router configurations, which are impractical to explore exhaustively. Our models, which are developed using only a small subset (of configurations), have very small overhead in terms of number of experiments.

IV. Modeling Methodology

Approximating a function of several to many variables using only the dependent variable space is a well-known problem with applications in many disciplines. The goal is to model the dependence of a target variable y on several predictor variables $x_1, \cdots, x_n$ given R realizations $\{y_i, x_{1i}, \cdots, x_{ni}\}_1^R$. The system that generates the data is presumed to be described by [9]

$$y = f(x_1, \cdots, x_n) + \epsilon \tag{1}$$

over some domain $(x_1, \cdots, x_n) \in \mathcal{D} \subset \mathcal{R}^n$ containing the data. Function f captures the joint predictive relationship of y on $x_1, \cdots, x_n$, and the additive stochastic noise component ϵ, usually reflects the dependence of y on quantities other than $x_1, \cdots, x_n$ that neither controlled nor observed. Hence, the aim of the regression analysis is to construct a function $\hat{f}(x_1, \cdots, x_n)$ that can accurately approximate $f(x_1, \cdots, x_n)$ over the domain $\mathcal{D}$ of interest. There are two main regression analysis methods: (1) global parametric, and (2) nonparametric. The former approach has limited flexibility, and can produce accurate approximations only if the assumed underlying function $\hat{f}$ is close to f. In the latter approach, $\hat{f}$ does not take a predetermined form, but is constructed according to information derived from the data. Multivariate adaptive regression splines (MARS) is a nonparametric regression technique which is an extension of linear models that automatically models nonlinearities and interactions, and is used in our methodology.

A. Problem Formulation

Given a router microarchitecture $\mathcal{X}^{\mu arch}$, circuit implementation $\mathcal{C}^{circ}$, we apply MARS to construct router power and area models, $p = \hat{f}(x_1^{\mu arch}, \cdots, x_n^{\mu arch}, c_1^{circ}, \cdots, c_n^{circ})$, and $a = \hat{g}(x_1^{\mu arch}, \cdots, x_n^{\mu arch}, c_1^{circ}, \cdots, c_n^{circ})$, respectively. Variables $x_1^{\mu arch}, \cdots, x_n^{\mu arch}$, and $c_1^{circ}, \cdots, c_n^{circ}$ denote router microarchitectural and implementation-related parameters, respectively. The general MARS model can be represented as [24]

$$\hat{y} = c_0 + \sum_{i=1}^{I} c_i \prod_{j=1}^{J} b_{ij}(x_{ij}) \tag{2}$$

where $\hat{y}$ is the target variable (i.e., router power and area in our problem), c_0 is a constant, c_i are fitting coefficients, and $b_{ij}(x_{ij})$ is the truncated power basis function with x_{ij} being the microarchitectural parameter used in the i^{th} term of the j^{th} product. I is the number of basis functions and J limits the order of interactions. The basis functions $b_{ij}(x_{ij})$ are defined as

$$b_{ij}^{-}(x^{\mu arch} - t_{ij}) = [-(x^{\mu arch} - t_{ij})]_+^q \tag{3}$$
$$= \begin{cases} (t_{ij} - x^{\mu arch})^q & x^{\mu arch} < t_{ij} \\ 0 & \text{otherwise} \end{cases}$$
$$b_{ij}^{-}(x^{\mu arch} - t_{ij}) = [+(x^{\mu arch} - t_{ij})]_+^q \tag{4}$$
$$= \begin{cases} (x^{\mu arch} - t_{ij})^q & x > t_{ij} \\ 0 & \text{otherwise} \end{cases}$$

where q (≥ 0) is the power to which the splines are raised to adjust the degree of $\hat{y}$ smoothness, and t_{ij} is called a knot. When $q = 0$ simple linear splines are applied.

The optimal MARS model is built in two passes. (1) Forward pass: MARS starts with just an intercept, and then repeatedly adds basis function in pairs to the model. Total number of basis functions is an input to the modeling. Backward pass: during the forward pass MARS usually builds an overfit model; to build a model with better generalization ability, the backward pass prunes the model using a generalized cross-validation (GCV) scheme

$$GCV(K) = \frac{1}{n} \frac{\sum_{k=1}^{n} (y_k - \hat{y})^2}{[1 - \frac{C(M)}{n}]^2} \tag{5}$$

where n is the number of observations in the data set, K is the number of non-constant terms, and $C(M)$ is a complexity penalty function to avoid overfitting.

To generate the required data set for power and area modeling we use a library of fully-synthesizable router RTLs and implement them through a complete physical design flow to get placed and routed designs. W then perform switching activity-aware power estimation to accurately estimate power values. Area values can be extracted once we have the placed and routed design. Router microarchitecture impacts its power as it determines the underlying circuit components and their activity. The area of a router is also determined by the chosen microarchitecture and the circuit implementations. We assume that across all possible microarchitectural parameter values, circuit implementation (i.e., flow, optimization objectives, etc.) are fixed. Hence, our proposed models accurately capture the impact of microarchitectural parameters on router power and area within a CAD-specific flow.

B. Power Modeling

In this subsection, we describe an architectural-level parameterized power model for network routers given router microarchitecture $\chi^{\mu arch}$ and underlying circuit implementations C^{circ}. We derive models for both dynamic and leakage power. Dynamic power is due to charging and discharging of switching capacitances, and leakage power is due to subthreshold and gate leakage currents. Dynamic power consumption in CMOS circuits is formulated as $p_{dyn} = \frac{1}{2}\alpha \times c_{sw} \times v_{dd}^2 \times f_{clk}$, with f_{clk} the clock frequency, α the switching activity, c_{sw} the switching capacitance, and v_{dd} the supply voltage. Dynamic power can be further divided into two categories: (1) switching power, and (2) internal power. The former (Equation (6)) is the dynamic power consumed by the charging and discharging of the output load external to the cells, while the latter (Equation (7)) is the dynamic power dissipated within cells.

$$p_{switching} = \frac{1}{2} \times c_{load} \times v_{dd}^2 \times f_{clk} \tag{6}$$

$$p_{internal} = \frac{1}{2} \times c_{int} \times v_{dd}^2 \times f_{clk} + v_{dd} \times i_{sc} \tag{7}$$

where, c_{load} and c_{int} are capacitive and internal loads, respectively. Short-circuit current, i_{sc}, is the current between NMOS and PMOS when they are both on. Short-circuit current is noticeable only when the input slews are extremely large. Also, we note that accurate power estimation requires accurate switching activity estimation,

hence we report power with different switching activity values. Finally, we extract switching and internal power data sets and then normalize the power values with respect to switching activity, supply voltage, and clock frequency. Thus, the goal is to model dependence of $\frac{p_{switching}}{\alpha \times v_{dd}^2 \times f_{clk}}$ and $\frac{p_{internal}}{\alpha \times v_{dd}^2 \times f_{clk}}$ on microarchitectural parameters. The modeling task becomes one of finding $\hat{c}_{load}$ and $\hat{c}_{int}$ as functions of the microarchitectural parameters: $\hat{c}_{load} = \hat{f}_1(x_1^{\mu arch}, \cdots, x_n^{\mu arch})$, and $\hat{c}_{int} = \hat{f}_2(x_1^{\mu arch}, \cdots, x_n^{\mu arch})$.

As technology scales to deep sub-micron processes, leakage power becomes increasingly important as compared to dynamic power. There is thus a growing need to characterize and optimize network leakage power as well. The largest percentage of static power results from source-to-drain subthreshold leakage current. However, from 65nm and beyond gate leakage gains importance and becomes a significant portion of the leakage power. This is even more visible for high-performance applications where gate oxides are much thinner (i.e., ∼1.5nm in 65nm HP library). We consider both subthreshold and gate leakage currents by using the average leakage current from the Liberty. Leakage power is dependent on the supply voltage and temperature, which are defined by the library used (i.e., through the characterization process). Leakage power can be shown as, $p_{leak} = v_{dd} \times i_{leak}$, where i_{leak} is the leakage current which includes both subthreshold and gate leakage currents. To model leakage power, we first normalize the leakage power values with respect to the supply voltage, and then estimate the leakage current as a function of microarchitectural parameters, $i_{leak} = \hat{f}_3(x_1^{\mu arch}, \cdots, x_n^{\mu arch})$. Figure 2 shows our proposed router power model for the target router described above in 65nm. Table I shows all the microarchitectural parameters in our models (in Section III-B).

Basis Functions
b_1 = max(0, n_{port} − 3); b_2 = max(0, n_{vc} − 2)×b_1;
b_3 = max(0, l_{buf} − 2)×b_2; b_4 = max(0, fw − 16)×b_3;
b_5 = max(0, l_{buf} − 2); b_6 = max(0, fw −16);
b_7 = max(0, n_{vc} − 2); b_8 = max(0, n_{port} − 5)×b_6;
b_9 = max(0, 5 − n_{port})×b_6; b_{10} = max(0, n_{port} − 5)×b_5;
b_{11} = max(0, 5 − n_{port})×b_5; b_{12} = max(0, n_{vc} − 2)×b_{11};
b_{13} = max(0, fw − 16)×b_{12}; b_{14} = max(0, fw − 16)×b_7;
b_{15} = max(0, fw − 16)×b_5; b_{16} = max(0, n_{port} − 7);
b_{17} = max(0, 7 − n_{port}); b_{18} = max(0, n_{vc} − 2)×b_{10};
b_{19} = max(0, n_{port} − 5)×b_7; b_{21} = max(0, n_{port} − 3)×b_{15};
b_{22} = max(, n_{port} − 5)×b_{14}; b_{23} = max(0, 5 − n_{port})×b_{14};
b_{24} = max(0, n_{vc} − 2)×b_{15}; b_{25} = max(0, n_{vc} − 2)×b_{17};
Power Model
$p = \alpha \times (1.714 + 0.861 \times b_1 + 0.199 \times b_2 + 0.180 \times b_3 +$
$0.002 \times b_4 + 0.741 \times b_5 + 0.055 \times b_6 + 0.690 \times b_7 +$
$0.017 \times b_8 - 0.007 \times b_9 + 0.233 \times b_{10} - 0.106 \times b_{11} +$
$0.120 \times b_{12} + 0.002 \times b_{13} + 0.019 \times b_{14} + 0.012 \times b_{15} +$
$0.382 \times b_{16} - 0.078 \times b_{18} + 0.224 \times b_{19} + 0.004 \times b_{21} +$
$0.004 \times b_{22} - 0.003 \times b_{23} + 0.004 \times b_{24} + 0.050 \times b_{25})$
$\times v_{dd}^2 \times f_{clk}$

Fig. 2. Power model for our target router in 65nm.

Another useful contribution of our method is to identify the importance of each of the microarchitectural parameters in the overall router power. To calculate variable importance, we use MARS to refit the model after dropping all terms involving the variable in question and calculating the reduction in goodness of fit (i.e., microarchitectural parameters are independent). The least important variable is the one with the smallest impact on the model quality; similarly, the most important variable is the one that, when omitted, degrades the model fit the most.

C. Area Modeling

With the increase in number of cores on a single design, the area occupied by the communication components such as links and network routers increases (e.g., 19% of total area in the Intel 80-core chip [10]). As area is an important economic incentive in IC (integrated circuit) design, it needs to be estimated early in the design flow to enable efficient design space exploration. In this section we present an accurate model for router area. For each router configuration, we extract the logic area from the corresponding placed and routed netlist. We then use nonparametric regression to develop router area model. The general form of the area model can be shown as:

$$a = f(x_1^{\mu arch}, \cdots, x_n^{\mu arch}, c_1^{circ}, \cdots, c_n^{circ}) \quad (8)$$

where $x_1^{\mu arch}, \cdots, x_n^{\mu arch}$ and $c_1^{circ}, \cdots, c_n^{circ}$ are router microarchitectural and circuit implementation parameters, respectively. The circuit implementation parameters are defined by (1) design library, and (2) the specific choice of implementation parameters (e.g., target frequency, aspect ratio, row utilization, etc.). To factor out the impact of floorplanning on area we use sum of standard cells areas as the router area. The sum of standard cell areas depends on logical depth of different router components. Figure 3 shows our proposed router area model for our target router in 65nm.

Basis Functions
b_1 = max(0, n_{port} − 3); b_2 = max(0, n_{vc} − 2)×b_1;
b_3 = max(0, l_{buf} − 2)×b_2; b_4 = max(0, fw − 16)×b_2;
b_5 = max(0, l_{buf} − 2); b_6 = max(0, fw − 16)×b_5;
b_7 = max(0, n_{vc} − 2); b_8 = max(0, n_{port} − 5)×b_7;
b_9 = max(0, 5 − n_{port})×b_7; b_{10} = max(0, n_{port} − 7)×b_6;
b_{11} = max(0, 7 − n_{port})×b_6; b_{12} = max(0, n_{vc} − 2)×b_6;
b_{13} = max(0, fw − 16); b_{14} = max(0, n_{port} − 7)×b_5;
b_{15} = max(0, 7 − n_{port})×b_5; b_{16} = max(0, n_{port} − 5)×b_{13};
b_{17} = max(0, 5 − n_{port})×b_{13}; b_{18} = max(0, l_{buf} − 2)×b_9;
b_{19} = max(0, n_{vc} − 5)×b_{10}; b_{20} = max(0, 5 − n_{vc})×b_{10};
b_{21} = max(0, n_{port} − 7); b_{23} = max(0, n_{vc} − 5)×b_1;
b_{25} = max(0, n_{vc} − 2)×b_{13};
Area Model
$a = 0.008 + 0.006 \times b_1 + 0.001 \times b_2 + 0.0004 \times b_3 +$
$2.351e-5 \times b_4 + 0.006 \times b_5 + 0.0001 \times b_6 +$
$0.006 \times b_7 + 0.002 \times b_8 + 2.736e-5 \times b_{10} -$
$1.819e-5 \times b_{11} + 4.480e-5 \times b_{12} + 0.0004 \times b_{13} +$
$0.001 \times b_{14} - 0.0002 \times b_{15} + 0.001 \times b_{16} -$
$0.001 \times b_{17} + 0.001 \times b_{18} + 9.684e-5 \times b_{20} -$
$3.100e-5 \times b_{21} + 1.073e-5 \times b_{22} - 3.299e-6 \times b_{23} +$
$7.777e-5 \times b_{24} - 5.265e-5 \times b_{25};$

Fig. 3. Area model for our target router in 65nm.

V. Validation and Significance Assessment

We now validate the router power and area models described in earlier sections. We compare four different models: (1) our proposed new model (New), (2) a model derived using synthesis data sets, i.e., with the same methodology as in (1) (Synth.), (3) a model derived using parametric regression (Reg.), and (4) ORION 2.0. In (1) and (2) we use nonparametric regression method to develop the models. In (2) switching power is unrealistic due to the use of inaccurate wireload models during synthesis. Also, during P&R stages certain cells are up or down sized accordingly which results in further differences in internal and leakage power between (1) and (2). The relatively large error of "Synth." models is because we use a different target variable space for model generation (i.e., post-synthesis power values), but evaluate the model at a new target variable space (i.e., post-layout power values). If we evaluate the "Synth." models with post-synthesis power and area values their accuracy matches that of the new models. To develop "Reg." model we extract corresponding power and area data sets from the P&R tool for a baseline VC router. The drawback of parametric regression methods (i.e., linear, quadratic regression, etc.) is their limited accuracy since there are no procedures in

TABLE II
SENSITIVITY OF THE PROPOSED MODELS WITH RESPECT TO DIFFERENT TRAINING / TESTING SET SIZES.

Metric	Power Model					Area Model				
	s_{tr} = 1/2	s_{tr} = 1/3	s_{tr} = 1/5	s_{tr} = 1/10	s_{tr} = 64	s_{tr} = 1/2	s_{tr} = 1/3	s_{tr} = 1/5	s_{tr} = 1/10	s_{tr} = 64
minimum % error	0.006	0.006	0.007	0.01	0.006	0.0004	0.038	0.042	0.054	0.044
maximum % error	12.415	49.226	81.112	109.224	77.321	14.105	43.099	78.236	111.384	61.236
average % error	1.662	4.012	7.997	27.177	21.230	1.814	3.700	8.568	25.163	16.417

TABLE III
COMPARISON OF DIFFERENT POWER AND AREA MODEL ACCURACY WITH RESPECT TO IMPLEMENTATION DATA.

Metric		Power Model				Area Model			
		New	Synth.	Reg.	ORION2.0	New	Synth.	Reg.	ORION2.0
minimum % error	130nm	**0.0112**	30.453	7.659	9.526	**0.0011**	0.062	29.884	10.121
	90nm	**0.0081**	28.361	7.236	6.865	**0.0015**	0.064	27.821	8.229
	65nm	**0.0067**	28.854	6.921	7.730	**0.0010**	0.056	29.115	9.111
maximum % error	130nm	**62.053**	47.735	96.512	103.214	**60.723**	30.231	107.772	104.118
	90nm	**60.073**	47.232	92.312	85.345	**60.151**	29.787	109.236	88.331
	65nm	**59.413**	48.343	108.369	81.810	**61.844**	30.914	111.278	86.228
average % error	130nm	**6.012**	42.610	23.463	41.326	**5.961**	13.143	26.332	38.117
	90nm	**5.654**	40.197	25.110	30.224	**5.019**	12.225	27.113	32.566
	65nm	**5.817**	37.494	24.431	32.780	**5.411**	11.410	26.226	33.298

the modeling methodology to help the modeler understand the interactions between the predictors (i.e., independent variables). To model router power, using parametric regression, we need to understand the specific circuit implementation used for each of router building blocks (i.e., FIFO buffers, crossbar switch, and VC / switch allocation).

In the baseline VC router, FIFO buffers are implemented as flip-flop registers, and a multiplexer tree crossbar implementation along with combined VC and switch allocation is used. For FIFO buffers, power is linearly dependent on the buffer length (l_{buf}), flit-width (fw), number of ports (n_{port}), and number of VCs (n_{vc}). As either buffer length or flit-width increases, we expect FIFO dynamic and leakage power to increase linearly. This is because buffer length and flit-width linearly increase the number of flip-flops required. If we increase the number of VCs, buffer dynamic power will not change since the number of flits arriving at each input port is the same. However, we expect leakage power to increase linearly because in VC routers there are n_{vc} queues in each input port. Finally, the number of router ports linearly changes FIFO dynamic and leakage power because addition of a new port will add a new buffer set, i.e., with the same buffer length and flit-width.

In a $n_{port} \times n_{port}$ multiplexer tree crossbar implementation, power is linearly (quadratically) dependent on flit-width (number of ports), respectively. Buffer length and the number of VCs have no impact on crossbar power. Flit-width linearly increases power since it linearly increases the number of parallel multiplexer needed to implement the fw-bit n_{port}:1 multiplexers. As we increase the number of ports, we expect crossbar dynamic and leakage power to increase quadratically since a $n_{port} \times n_{port}$ crossbar allows arbitrary one-to-one connections between n_{port} input ports and n_{port} output ports.[1] In our baseline VC router VC allocation is modeled as VC "selection" instead, as was first proposed in [14]. In this approach, the pipeline first goes through switch allocation, before selecting an available VC, effectively simplifying VC allocation to reading from a queue of available VCs. Power consumed thus becomes largely invariant to actual number of VCs, however, power is quadratically dependent on the number of ports. This is because the request width for each arbiter, in allocation stage, increases linearly with respect to the number of ports, and also the number of such arbiters is proportional to the number of ports.

We use MATLAB [26] to develop the parametric regression-based models using 64 (out of 256) data points for training and the rest of the sample set for testing. For dynamic power we normalize the dynamic power values with respect to switching activity α, supply voltage v_{dd}, and clock frequency f_{clk} and define the switching capacitance as in Equation (9). For leakage power, we normalize the leakage power values with respect to supply voltage, and define the leakage current as in Equation (10). Dynamic and leakage power models are then constructed by plugging back the normalizing parameters into the corresponding equations.

$$c_{switching} = a_1 \times (l_{buf} \times fw \times n_{port}^2) + b_1 \times (n_{port}^2 \times fw) + c_1 \times (n_{port}^2) \quad (9)$$

$$i_{leak} = a_2 \times (l_{buf} \times fw \times n_{vc} \times n_{port}^2) + b_2 \times (n_{port}^2 \times fw) + c_2 \times (n_{port}^2 \times n_{vc}) \quad (10)$$

where a_1, a_2, b_1, b_2, c_1, and c_2 are fitting coefficients. In both Equations (9) and (10) the first, second, and third terms are due to FIFO buffers, crossbar, and VC / switch allocation, respectively. We do not account for the contribution of the control logic to the overall power, and this could degrade the quality of the fit compared against the implementation data. In parametric regression-based techniques the fitting accuracy cannot be enhanced unless the underlying functional form is close approximation of the original function. We also run ORION 2.0 with similar microarchitectural parameters as those of the corresponding implemented router designs. Table III shows a comparison of the aforementioned models against the implemented router designs. We notice the significant accuracy improvement of our new models.

Table IV shows relative variable importance using post-synthesis and post-layout power data sets. As described in Section IV-B, we use *MARS3.0* to find the relative importance of each of the microarchitectural parameters in the overall power estimation. We observe that n_{vc} and n_{port} are the dominant parameters with respect to the post-synthesis and post-layout data sets, respectively. This shows the impact of missing layout information at the post-synthesis stage. After synthesis, the RTL design is only mapped to a standard-cell library and does not include detailed wiring information. This results in underestimation of the crossbar power. In a multiplexer tree crossbar, power is due to (1) multiplexers and (2) the interconnection grid between n_{port} input and n_{port} output

[1]Note that we have extracted the power data set from after the P&R stage which includes accurate interconnect switching power. Hence, crossbar power quadratically increases with respect to the number of ports due to n_{port}^2 increase in interconnect length.

ports. When the interconnection power is ignored, crossbar power is modeled as $O(n_{port} \log_2 n_{port})$. This changes to a quadratic dependence after layout information becomes available. In our current router microarchitecture, this is the major difference between post-synthesis and post-layout data; however, absolute power values of all microarchitectural blocks differ between post-synthesis and post-layout power data sets.

To assess the robustness of our approach we used five different scenarios to train and test our models. Table II shows minimum, maximum, and average error of our new models based on these scenarios. In the first four scenarios we train our models using a fraction s_{tr} of the available data points, and validate them on the rest of the points. This is to assess the sensitivity of the model to the sampling size. However, in the fifth case, to verify whether the model is general enough we use only 64 (out of 256) data points to train the model, but validate it across all 256 available data points. We observe that as the sampling size decreases, the accuracy of the model also degrades. However, we can obviously trade off model accuracy for additional setup time. In addition, Figures 4, 5, 6, and 7 show the significant accuracy improvement of our new model relative to ORION 2.0 when compared against router implementation data.

TABLE IV
RELATIVE VARIABLE IMPORTANCE USING POWER DATA SETS.

Parameter	Variable Importance (%)	
	Post-Synthesis	Post-Layout
n_{port}	92.98	100.00
n_{vc}	100.00	95.44
l_{buf}	88.41	73.99
fw	67.03	64.81

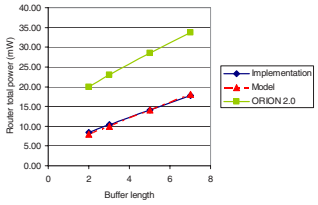

Fig. 4. Router total power with respect to buffer length.

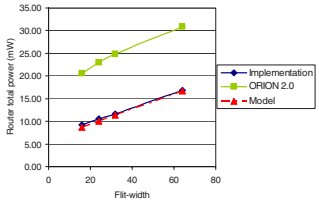

Fig. 5. Router total power with respect to flit-width.

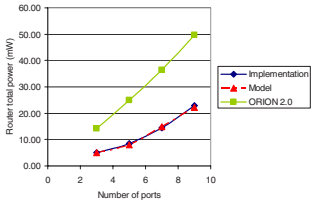

Fig. 6. Router total power with respect to number of ports.

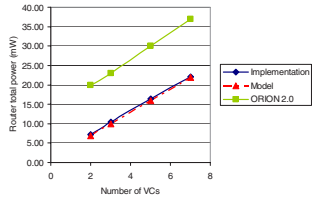

Fig. 7. Router total power with respect to number of virtual channels.

VI. Conclusions

Accurate estimation of power and area of interconnection network routers in early phases of the design process can drive effective NoC design space exploration. Existing router power and area models (e.g., ORION 2.0 [12], Xpipes [7], etc.) are based on certain architectures and circuit implementations. Therefore, they cannot guarantee maximum accuracy within an architecture-specific CAD flow. In this paper, we have proposed an efficient router power and area modeling methodology in which the underlying architecture and circuit implementation are decoupled from the modeling effort. In addition, our models show a significant accuracy improvement over the existing models, i.e., within 6% (on average) of the implementation data. This enables system-level designers to efficiently perform design space exploration without having to know the underlying router microarchitecture and circuit implementation details. We have also presented a reproducible methodology for developing our models given a NoC component library.

VII. Acknowledgments

Partial support for this research was provided by the MARCO Gigascale Systems Research Center, the National Science Foundation, the Semiconductor Research Corporation, and the UC MICRO program / Qualcomm.

References

[1] A. Banerjee, R. Mullins and S. Moore, "A Power and Energy Exploration of Network-on-Chip Architectures", *Proc. NOCS*, 2007, pp. 163–172.
[2] N. Banerjee, P. Vellanki and K. S. Chatha, "A Power and Performance Model for Network-on-Chip Architectures", *Proc. DATE*, 2004, pp. 1250–1255.
[3] A. Bona, V. Zaccaria, and R. Zafalon, "System Level Power Modeling and Simulation of High-End Industrial Network-on-Chip", *Proc. DATE*, 2004, pp. 318–323.
[4] D. Brooks, V. Tiwari and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations", *Proc. ISCA*, 2000, pp. 83–94.
[5] J. Chan and S. Parameswaran, "NoCEE: Energy Macro-Model Extraction Methodology for Network on Chip Routers", *Proc. ICCAD*, 2005, pp. 254-259.
[6] J. Chan and S. Parameswaran, "NoCOUT: NoC Topology Generation With Mixed Packet-Switched and Point-to-Point Networks", *Proc. ASPDAC*, 2008, pp. 265-270.
[7] M. Dall'Osso, G. Biccari, L. Giovannini, D. Bertozzi and L. Benini, "Xpipes: A Latency Insensitive Parameterized Network-on-Chip Architecture for Multiprocessor SoCs", *Proc. ICCD*, 2003, pp. 536–539.
[8] W. J. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks", *Proc. DAC*, 2001, pp. 684–689.
[9] J. H. Friedman, "Multivariate Adaptive Regression Splines", *Annals of Statistics*, 19(1), 19991, pp. 1–66.
[10] Y. Hoskote, S. Vangal, A. Singh, N. Borkar and S. Borkar, "A 5-GHz Mesh Interconnect for a Teraflops Processor", *IEEE MICRO*, 2007, pp. 51–61.
[11] C. Isci and M. Martonosi, "Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data", *Proc. MICRO*, 2003, pp. 93–104.
[12] A. B. Kahng, B. Li, L.-S. Peh and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration", *Proc. DATE*, 2009, pp. 423–428.
[13] A. B. Kahng and S. Mantik, "Measurement of Inherent Noise in EDA Tool", *Proc. ISQED*, 2002, pp. 206–211.
[14] A. Kumar, P. Kundu, A. Singh, L.-S. Peh and N. K. Jha, " A 4.6Tbits/s 3.6GHz Single-cycle NoC Router with a Novel Switch Allocator in 65nm CMOS", *Proc. ICCD*, 2007, pp. 63–70.
[15] S. E. Lee and N. Bagherzadeh, "A High Level Power Model for Network-on-Chip (NoC) Router", *INTEGRATION, the VLSI journal*, 2009, pp. 1–7.
[16] G. de Micheli and L. Benini, "Networks On Chip: A New Paradigm for Systems on Chip Design", *Proc. DATE*, 2002, pp. 2–6.
[17] P. Meloni, I. Loi, F. Angiolini, S. Carta, M. Barbaro, L. Raffo and L. Benini, "Area and Power Modeling for Networks-on-Chip With Layout Awareness", *VLSI Design*, 2007, pp. 1–12.
[18] G. Palermo and C. Silvano, "PIRATE: A Framework for Power/Performance Exploration of Network-on-Chip Architectures", *Proc. PATMOS*, 2004, pp. 521-531.
[19] C. S. Patel, S. M. Chai, S. Yalamanchili and D. E. Schimmel, "Power Constrained Design of Multiprocessor Interconnection Networks", *Proc. ICCD*, 1997, pp. 408–416.
[20] A. G. Spix and R. R. Varada, "Monte Carlo Techniques for Physical Synthesis Design Convergence Exploration", *Proc. DAC*, 2009, *to appear*.
[21] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, A. Agarwal, "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs", *IEEE MICRO*, 22(2), 2002, pp. 25–35.
[22] H. Wang, X. Zhu, L.-S. Peh and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Networks", *Proc. MICRO*, 2002, pp. 294–395.
[23] W. Ye, N. Vijaykrishnan, M. Kandemir and M. J. Irwin, "The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool", *Proc. DAC*, 2000, pp. 340–345.
[24] Y. Zhou and H. Leung, "Predicting Object-Oriented Software Maintainability Using Multivariate Adapitive Regression Splines", *Journal of Systems and Software*, 80, 2007, pp. 1349–1361.
[25] *International Technology Roadmap for Semiconductors*, 2007 Edition, http://public.itrs.net/.
[26] *MATLAB*, http://www.mathworks.com/
[27] *Netmaker*, http://www-dyn.cl.cam.ac.uk/ rdm34/wiki/index.php?title=Main_Page.
[28] *MARS User Guide*, http://www.salfordsystems.com/mars.php.
[29] http://www.cadence.com/us/pages/default.aspx.
[30] http://www.synopsys.com/products/power/power ds.html.
[31] *Cadence SOC Encounter User's Manual.* http://www.cadence.com/.