A System-level Network-on-Chip Simulation Framework Integrated with Low-level Analytical Models

Jinwen Xi, and Peixin Zhong, *Member, IEEE*

Dept. of Electrical and Computer Engineering, Michigan State Univeristy
East Lansing, MI 48824, U.S.A.
{xijinwen, pzhong} @egr.msu.edu

***Abstract*—This paper presents a system-level Network-on-Chip modeling framework that integrates transaction-level model and analytical wire model for design space exploration. It enables the analysis of influence of physical wire properties on the system performance and power dissipation in early design stages. SystemC provides the infrastructure to integrate transaction-level model and low-level models. By utilizing approximate timing, different temporal granularity can be used, leading to fast simulation speed. Six deep-submicron CMOS processes from 180nm to 45nm are used to evaluate the performance/power of NoC. Additionally, temporal and spatial NoC power analysis under different traffic conditions provides an effective basis for power/thermal optimization and design space exploration in early design stages.**

***Index Terms*—NoC, SoC, SystemC, Power Model**

I. INTRODUCTION

With the growing complexity in embedded systems, there is a new trend for heterogeneous System-on-Chip (SoC) architectures consisting of multiple integrated components that communicate with each other at very high speed[1]. New on-chip interconnect architecture (OCA) is necessary because of increasing integration of hundreds of cores and the increasing significance of wire delay along with the decreasing feature size and increasing maximum clock rate. The progressing IC technology makes Network-on-Chip (NoC) a promising scalable solution for SoC[2]. Different from bus-based SoC, functional cores in NoC communicate by sending information packets across the on-chip network instead of directly driving signals across dedicated global wires [3]. The on-chip interconnection is implemented by routers connected with each other via interconnecting links. The distributed nature of NoC provides a scalable solution as compared to shared-medium bus.

As the system scale (number of transistors) keeping exploding under Moore's Law, an SoC design requires a system-level simulation model to verify the functionality, explore the design space, and estimate the cost (area and power) and performance before detailed implementation, in order to avoid expensive design iterations. A decision made at high level will have more impact on the quality of design. As the technology scales down to nanoscale processes, wire is becoming an increasingly critical issue for both power and performance aspects because global wire scales at a much lower rate compared to transistor gate. A system-level modeling framework that integrates low-level design information provides the necessary mechanism to perform system-level design space exploration and design optimization.

By employing high-level read/write function calls to abstract communication operations, transaction-level modeling (TLM)[4] offers system designers not only fast simulation speed, but also easy programming model since they don't need to consider the detailed communication signaling, which is unnecessary, or even unavailable in early design stages. This paper proposes a framework that integrates transaction-level NoC simulation model with analytical interconnecting wire models. The transaction-level simulation framework is built upon SystemC primitive components. It provides interfaces for inserting NoC components' performance/energy models at different levels of abstraction. The wire energy model estimates the links' on-flight leakage and dynamic power by tracking data switching properties during simulation. This simulation model provides detailed NoC run-time information including the temporal/spatial profile of throughput and leakage/switching power.

This paper is organized as following: Section 2 reviews some related works in NoC simulation frameworks and interconnecting wire models. Section 3 describes the proposed NoC modeling framework, including architecture, router modeling, the analytical wire delay and power models and their interface with the TLM simulation model. Section 4 presents some experiment data on six predictive deep sub-micron processes and this paper concludes in section 5.

II. RELATED WORK

In the past years, several research groups have proposed tools to specify, simulate NoCs at different levels of abstraction. Tortosa proposed PROTEO NoC model [5], where VHDL was

utilized to evaluate several features of virtual channels in mesh-based and hierarchical NoC topologies. The accuracy of VHDL model was high, but it suffered from low simulation speed. To accelerate simulation, a VHDL/SystemC hybrid model using a template router to support multiple interconnection networks was presented by Chen[6]. Wang *et al.*[7] introduced a C-based interconnection network simulator, and similarly, an event-driven C++ NoC simulator is proposed in [8]. Different from their cycle-accurate models, our NoC simulation framework is not synchronized by an explicit clock. Instead, approximate time is used. The approximate time allows selection of temporal granularity to achieve desired simulation speed and model accuracy.

Interconnecting wires, especially in deep submicron processes under 180nm, were extensively researched from the timing and energy points of view. Gupta *et al.* [9] presented a high-level interconnect power model for wires of a single core chip. Bokoglu *et al.* [10] proposed the wire delay model considering the optimal repeater sizing and spacing. Ho *et al.*[11] pointed out the power consumption of the delay optimal repeated wire is prohibitively large, and introduced a methodology to save wire power by increasing repeater spacing and decreasing repeater size. Heo *et al.* analyzed the power of the global wires in the on-chip network [3] using first-order RC model. Our work differentiates from theirs by integrating the analytical wire models into the transaction-level simulation framework, allowing evaluation of real applications, and it provides both router and wires' temporal and spatial timing/energy information.

III. NETWORK-ON-CHIP MODELING

A. *Modeling Framework Architecture*

This proposed NoC simulation model targets tile-based topologies. Mesh topology, as shown in Figure 1, is presented in this paper. Such topology contains *M*×*N* routers. Excluding the border ones, each router has four ports to connect with its neighboring ones, and another local port is connected to the processing element (PE), which may be microprocessor, DSP, or ASIC core. If the border routers have wrap-around links to the other end, it is called the torus topology and can also be modeled in the framework.

SystemC is chosen as the modeling language due to its increasing popularity and capability of modeling system at different abstraction levels. Figure 2 shows the overall architecture of the simulation framework. The modeling framework contains models that simulate the functionality of the components. They are annotated with constructs that model the performance and power consumption. Other non-functional properties can be added to the model. The system is composed of an array of PEs and the interconnect (i.e. NoC). In order to simulate the network traffic, PEs are modeled as packet generators and receivers. They can be easily replaced with a real processor model running application code. NoC model is composed of two main functional components: routers and interconnecting links. In SystemC, a component is represented by *sc_module* classes. The traffic patterns, which may be created from real application traces or mathematical models, are generated by member threads of corresponding PE models. A PE has one or multiple threads to emulate a microprocessor's sequential code execution, multi-rate behavior of embedded system, or an ASIC's concurrent computation. The timing is modeled by approximate timing. It is achieved by annotating the parameterized *wait( )* functions. This approach can model the system behavior at different temporal granularity without using a global clock. Alternatively, cycle-based simulation model can be used in any component if desired. The top-level NoC simulation module contains an array of router and link objects, each of which is configured by the topological, architectural and technological parameters, such as the input buffer size and routing algorithm of routers, and the width and length of links, for performance and power estimation. The routers and links are derived from *sc_module* classes. Both router and link classes take parameters transferred from the top-level module to configure their internal structures and functionalities during initialization as Figure 2 shows. Router's interconnection is modeled by binding its output (input) port to the input (output) port of the adjacent ones via the customized channels (links) according to the topology parameters. The operations of receiving and transmitting signals over interconnecting wires are abstracted to *link_read( )* and *link_write( )* methods defined as virtual functions in the port interfaces and implemented in the channel class. The timing information in routers are pre-characterized from low-level designs and annotated to the functional code by inserting parameterized *wait( )* functions.

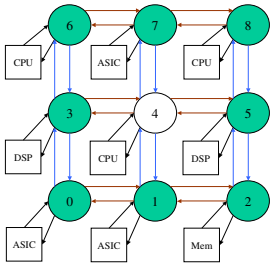


Figure 1. 3×3 mesh Network-on-Chip topology

In our NoC simulation model, the wormhole routing algorithm is implemented. It allows multiple flits (flit is the minimum data unit transferred on links in the network) of a packet to span multiple routers from the source to the destination. Routers employ input buffering to alleviate network performance loss due to traffic contentions, and flow control is utilized to determine at what time a flit is delivered based on the availability of network resources and current traffic status. The input buffer is aggregated with the link component and modeled using *sc_fifo* class. The input buffer depth is configurable during construction. Blocking *read/write* methods guarantee that no read/write operation can take place once the buffer is empty/full.

Routing function decides how each flit of a packet is delivered to the selected port by the router. This NoC model supports deterministic XY routing algorithm. Other routing algorithms can be used by plugging in the corresponding routing functions. In the 5-port router architecture, five concurrently

executing processes model the simultaneous operations on data coming from five input ports. These processes call the same routing function. To prevent the situation that multiple inputs are requesting the same output port at the same time, a round-robin arbitration scheme is employed, and SystemC *sc_mutex* primitive is used to reserve the output port.

Both the router and link objects will call power interface when traffic appears. The power consumption is collected by power monitor for analysis. The power interface will dump the collected power/energy data at each sampling time point and generate the temporal and spatial power profiles for each NoC component.

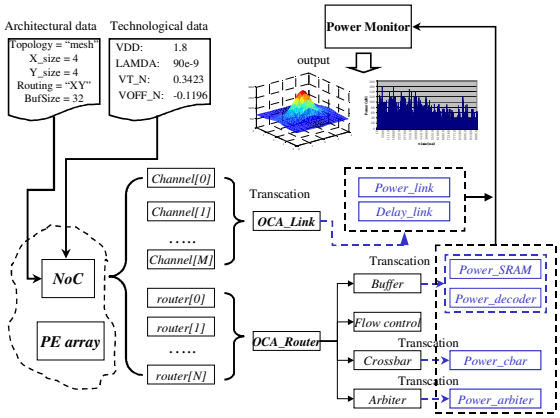


Figure 2. NoC hierarchical model structure

B. *Router's Performance and Power Model*

Router is characterized by its minimum routing latency, which is the time that each flit takes from its arrival at the input buffer to its departure from the output port. In the 5-port router, routing latency is determined by the latencies from its functional components: input buffer, crossbar and arbiter.

$$L_{router} = L_{buffer} + L_{cbar} + L_{arbiter} \quad (1)$$

where L_{buffer}, L_{cbar} and $L_{arbiter}$ are latency of input buffer, crossbar and arbiter respectively. These three parameters could be pre-determined from low-level HDL or circuit simulations, or estimated from the router's architectural parameters during run-time. In the current model, the first method is applied.

To estimate the router's dynamic power behavior, router class has a dedicated power modeling interface as the dotted rectangle associated with "OCA_Router" class in Figure 2 shows. One member function estimates leakage power, and the other models dynamic power. The power model's parameters are initialized during the router's instantiation. All architectural and technological parameters are passed to the model for power estimation. The flit payload that involves in each transaction is transferred to the power model during simulation. The same functional model can be mapped to different power models for design space exploration.

We refer to [7], a well-established dynamic power model for on-chip interconnect components, to derive the router's power model. Input buffer, crossbar and arbiter compose the router's power drain. And an architectural leakage power model is derived according to the methodology proposed in [12], while we use dual-port SRAM cell for the storage element of input FIFO instead of single-port one presented in [12]. Figure 3(a) shows the architecture of an input buffer, while (b) details one SRAM cell. For different SRAM operations, the power consumptions are:

①Read operation:

$$P_{dynamic} = P_{wordline(read)} + 2 \cdot S_{flit} \cdot \left[2P_{precharge} + P_{bitline(read)} + P_{sense}\right] \quad (2)$$

$$P_{leakage} = V_{dd}\{N_{row} \cdot N_{col} \cdot \left[(W_{N1} + 0.5W_{N3})I_{N0} + W_{P2}I_{P0}\right] + (N_{row} - 1) \cdot N_{col} \cdot W_{N5} \cdot I_{N0}\} \quad (3)$$

②Write operation:

$$P_{dynamic} = P_{wordline(write)} + m \cdot \left[P_{bitline(write)} + P_{cell}\right] \quad (4)$$

$$P_{leakage} = V_{dd}\{N_{row} \cdot N_{col} \cdot \left[(W_{N1} + W_{N5})I_{N0} + W_{P2}I_{P0}\right] + (N_{row} - 1) \cdot N_{col} \cdot 0.5W_{N3} \cdot I_{N0}\} \quad (5)$$

③Idle state

$$P_{dynamic} = 0 \quad (6)$$

$$P_{lekage} = V_{dd} \cdot N_{row} \cdot N_{col} \cdot \left[(W_{N1} + 0.5W_{N3} + W_{N5})I_{N0} + W_{P2}I_{P0}\right] \quad (7)$$

where $P_i = \alpha_i C_i V_{dd}^2 f$ is component i's dynamic power. m is the number of switching bits on the word line during a write access. C_i is the switching capacitance and α_i is activity factor. N_{row}, N_{col} are buffer array's row and column sizes. W_i and I_i are transistor i's width and its unit-width subthreshold leakage current respectively.

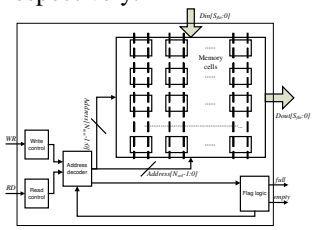
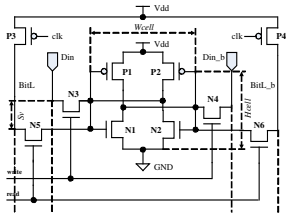


Figure 3(a) Input buffer



(b) Dual-port SRAM cell

The NoC simulation framework is verified by several traffic patterns before inserting interconnecting wire models. Figure 4 depicts the simulation results of NoC's router power with 180*nm* BPTM predictive process model[13]. The NoC is configured to 5×5 mesh with 16-entry input buffer per port, 10-flit packet size and each flit of 64-bit. The traffic pattern is single source, where the center PE injects packets to random destinations. Figure 4(a) and (b) show the spatial distribution of router power in 5ms simulation time with the flit injection rate of 50M/s and 500M/s. At higher injection rate, a more prominent power peak is shown near the source router.

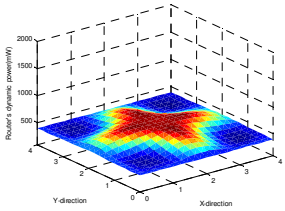


Figure 4(a). T= 20ns/flit

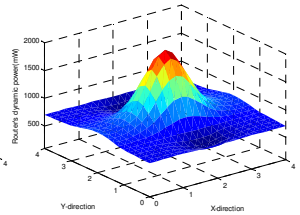


Figure 4(b).T = 2ns/flit

C. *Wire Link Performance and Power Model*

In this section, the analytical latency and power models of interconnecting wires and their interfacing with the NoC

simulation framework are presented. We assume that all interconnecting links are treated as global wires laid out on the top metal layers (M5 and M6 in 180*nm* 6-metal process), and these global wires are buffered and repeated uniformly. The dynamic power results from switching of wires and repeaters, while the leakage power mainly originates from repeaters' subthreshold current.

Figure 5(a) shows the first-order RC model of a wire [11]. The whole global wire with length L is divided into small segments, each with length of L_{seg}, by repeaters. Different repeater sizing and location will result in different wire latency and power, and we assume the repeaters with the same size are distributed uniformly along each interconnecting wire.

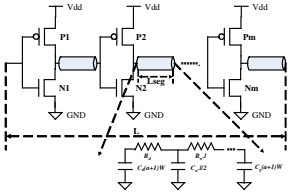
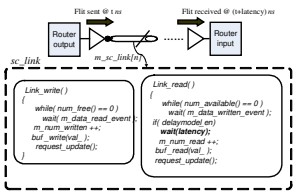


Figure 5(a). 1st-RC model



(b) Wire latency model

In the wire model, wire delay is represented as a function of the repeater sizing r, which is defined as the ratio of the repeater gate capacitance and wire capacitance within a wire segment.

$$r = \frac{w(\beta+1)C_g}{C_w L_{seg}} \quad (8)$$

The wire delay is modeled by two components: RC-delay from metal wires and repeaters:

$$D_R = 0.7 \cdot \frac{L}{L_{seg}} \left[R_d \cdot (\beta+1) \cdot (C_d + C_g) + L_{seg} R_w w (\beta+1) C_g \right] \quad (9)$$

$$D_M = 0.7 \frac{L R_d C_w}{w} + 0.7 L \cdot L_{seg} \frac{R_w C_w}{2} \quad (10)$$

$$D_{wire} = f(L, L_{seg}) = D_R + D_M \quad (11)$$

where R_w and C_w are the unit-length wire resistance and capacitance; C_d and C_g are the unit-length capacitance of drain and gate of minimum size NMOS transistor; and R_d is the resistance of minimum size repeater. β is the size ratio between repeater's PMOS and NMOS transistors.

The wire latency model is integrated with the NoC simulation model as shown in Figure 2. For the S_{flit} -width interconnecting link, latency is the worst-case wire delay among S_{flit} interconnecting wires. The parameterized *wait( )* models wire delay effects in the transaction-level NoC framework, where the data sending/receiving are modeled as *link_read( )* and *link_write( )* functions. The pre-calculated wire latency value is inserted using *wait( )* method in the link class' read/write function calls, and results in the data receiving operation is postponed by "*latency*" amount of time as Figure 5(b) shows.

For a single interconnecting wire, the switching capacitance is:

$$C_{wire} = C_w \cdot L \cdot XOR(bit, bit') \quad (12)$$

where *bit* and *bit'* are current and previous 1-bit data latched on the wire. The other part contributing to wires' dynamic power is the inserted repeaters, whose switching capacitance is:

$$C_{repeater} = (\beta+1) w \cdot C_g \cdot \frac{L}{L_{seg}} \cdot XOR(bit, bit') = r \cdot C_w \cdot L \cdot XOR(bit, bit') \quad (13)$$

From (12) and (13), the power consumption of an interconnecting link with S_{flit} wires depends on the traversing data's switching pattern represented by two consecutive transmitted flits' Hamming distance:

$$P_{link,dyn} = \frac{1}{2} \cdot V_{dd}^2 \cdot f \cdot AF \cdot \sum_{i=0}^{Sflit-1} \left(C_{wire}^i + C_{repeater}^i \right) = \frac{1}{2} \cdot V_{dd}^2 \cdot f \cdot AF \cdot (1+r) C_w L \cdot D_H(x, y) \quad (14)$$

where V_{dd} is link's supply voltage, f is the operating frequency, which is assumed to be the same as router's. AF is activity factor with the assumption value of 50% in the system model. $D_H(x,y)$ is the hamming distance between two successive flits x and y flowing through this link during simulation. The power consumption due to wire-to-wire coupling can be added in future work by inserting the inter-wire capacitance models.

The leakage power model accounts for the repeaters' subthreshold leakage. In the CMOS inverter, either NMOS or PMOS has 50% probability in the off state.

$$P_{link,leakage} = \frac{1}{2} (W_P I_{P0} + W_N I_{N0}) \frac{L}{L_{seg}} \cdot S_{flit} \cdot V_{dd} = \frac{wL \cdot S_{flit}}{2 L_{seg}} (\beta I_{P0} + I_{N0}) \cdot V_{dd} \quad (15)$$

where I_{P0} and I_{N0} are unit-width NMOS and PMOS's leakage current, respectively.

The power model of interconnecting link is implemented by a dedicated class, which is instantiated in each link object during simulation initialization. Each transaction happening on the links will call the power models to estimate the current transient power by tracking the link's switching properties. Similar to the router, the link's power data will be sampled by the power monitor at the specified time interval to generate the temporal link power profile.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

The whole NoC simulation framework and power models of routers and links are coded with SystemC 2.0.1 library and contained in 30 C++ source files. The target process is chosen from BPTM predictive model from 180*nm* to 45*nm* technologies, and the operating frequency is set to 2GHz for all technology nodes. The NoC is 5×5 mesh architecture, and the chip size is assumed to be 10*mm*×10*mm*. The packet size is fixed to be 10-flit, and each flit is 64-bit width.

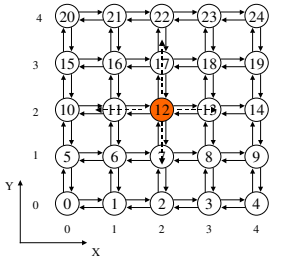


Fig 6 (a) Broadcast mode

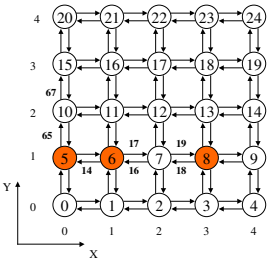
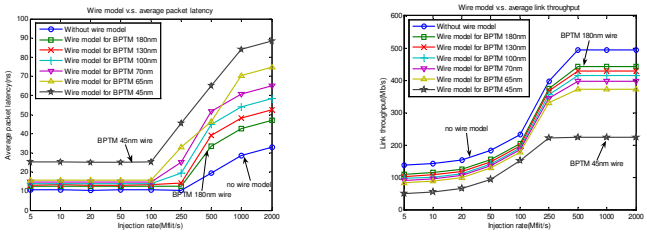


Fig 6(b) Burst mode

The same single source traffic used before is employed to evaluate the influence of inserting wire delay and power models. The router's buffer, crossbar and arbiter have the same latency of 2 clock cycles (1*ns*). First NoC is simulated without wire models, and then wire models of different technologies are enabled to evaluate their impact on the NoC performance and power properties. Based on the assumption of fixed chip size and router number, we use the minimum global wire width of 0.8*um* in BPTM 180*nm* process as the baseline, and keep this wire width same for all the other 5 processes. As a result, the unit-length wire resistance and capacitance scale accordingly, and are shown in Table 1. Figure 7(a) shows the average packet latency under different schemes, and (b) describes the average throughput of interconnecting links. The presented average latency only takes the "pure" network latency of each flit into consideration.



(a) Packet latency (b) Link throughput

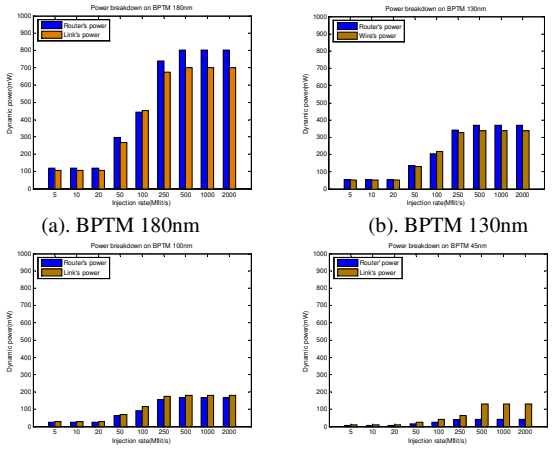
Figure 7. NoC Simulation on constant wire width with different technology nodes

Under the assumptions of fixed operating frequency, chip size and global wire width, the average packet latency increases by 42.51% in the worst case (network is congested) and 18.74% in the best case (network has free bandwidth to accept incoming traffics) after instrumenting wire models under 180*nm* technology. The packet latency increases with the average ratio of 16.82% and 13.51% between each two consecutive technology nodes in the best and worst cases respectively given all the same conditions. And as one of the direct results, an average decrease of ~13.34% on the average link throughput is observed. This shows that the wiring issue is becoming more and more dominant in large-size, multi-core NoCs containing long global wires along with technology downscaling. Table 1 indicates that both unit-length resistance and capacitance for global interconnecting wires increase at an average rate of 42.42% and 14.29% between two neighboring processes. Here we also assume each process keeps its minimum global wire thickness.

TABLE 1. WIRE PARAMETERS FOR DIFFERENT TECHNOLOGIES USING THE SAME GLOBAL WIRE WIDTH(0.8UM)

Technology	$W_{min}(um)$	$R_w(\Omega/mm)$	$C_w(fF/mm)$
BPTM 180nm	0.8	20	400
BPTM 130nm	0.6	26.25	493.33
BPTM 100nm	0.5	33.75	536
BPTM 70nm	0.45	46.13	554.67
BPTM 65nm	0.4	75	592
BPTM 45nm	0.3	114.75	765.33

At 45*nm* processing node, wire delay exceeds the router delay dramatically by ~5X, implying that wiring issue is becoming more crucial in the nanoscale NoCs. This is expectable since wires' scaling is much slower than transistors' in process evolution seen from Table 1, and increasing wire resistance leads to increasing wire delay with the decreasing feature size. Process advance from 180*nm* to 45*nm* results in only 1.91X increase in unit-length wire capacitance, while 5.74X increase in unit-length wire resistance under the same wire width. As a direct result of increased wire delay, the average link throughput reduces due to decreased number of flits flowing through each link in unit amount of time, we observed that the average link throughput reduces 54.88% and 63.89% in congested and non-congested NoC under 45*nm* process with the constant wire width.



(a). BPTM 180nm (b). BPTM 130nm

(c). BPTM 100nm (d). BPTM 45nm

Figure 8. NoC Power breakdown at different processes

Figure 8(a)-(d) describe NoC's power breakdown in four processes under single-source traffic load, where both frequency and voltage are scaled according to the target technology nodes. We can see that the relative weight between router and link. In NoC fabricated in 180*nm* process, the power ratio between link and router is 0.81. This ratio increases to 0.91, 1.12 and 3.13 in 130*nm*, 100*nm* and 45*nm* processes, showing the trend that links are becoming more power-hungry than routers.

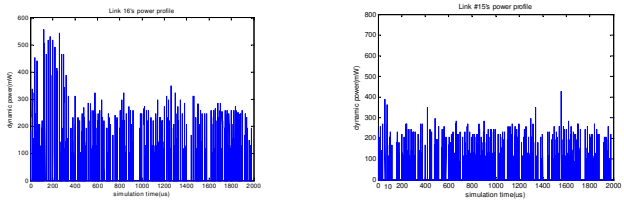


Figure 9(a). Link #16's profile (b). Link #17's profile

To illustrate the temporal and spatial interconnecting power profiles of the proposed simulation framework, a burst-mode traffic representing real applications is applied to the same 5×5 mesh architecture with 180*nm* process. Each PE injects packets with the rate of 10M flits/s randomly, while this rate is increased to 100M flits/s on node #5, #6 and #8 during the burst from 10*us* to 15*us*. And the burst traffic's destination is fixed to node #15 as figure 6(b) shows. Figure 9(a)-(b) describe link #16 and #17's dynamic power temporal profile from 0 to 2ms. When the

burst comes at 10*us*, the involving link's (#16) power increases to ~5.2X in maximum, and gradually returns back after the burst traffic. The link (#17) that doesn't involve in burst, the power keeps almost constant as Figure 9(b) shows.

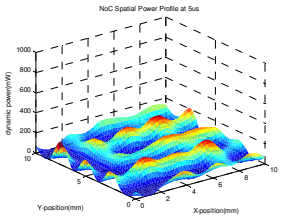


Figure 10(a).Profile at 5*us*

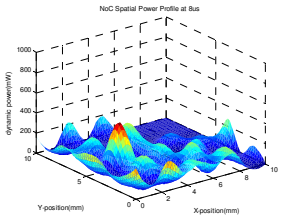


Figure 10(b).Profile 8*us*

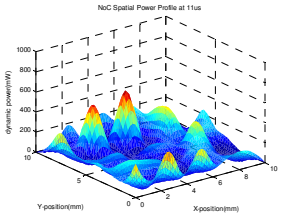


Figure 10(c).Profile at 11*us*

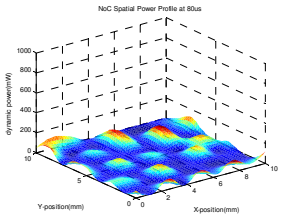


Figure 10(d).Profile at 80*us*

Figure 10(a)-(d) show the spatial power distribution of both routers and links in 5×5 mesh NoC under burst traffic at different simulation time points of 5*us*, 8*us*, 11*us* and 20*us*. The X and Y coordinates represent the physical locations of routers and links. The power of two links connecting the same router pair on the opposite directions is added together since they are assumed to reside in the same physical location. After the NoC reaches burst mode, power consumption of routers and links involving in the burst is increased by ~3X compared with the non-burst nodes as Figure 10(c) shows due to the suddenly increased injection rate. When the burst ends and the traffic returns back to uniform, the power distribution profile will become flat as Figure 10(d) describes. The routers on four edges consume less power than others in that the reduced number of input ports results in less traffic load.

V. CONCLUSION

This paper presents a transaction-level NoC simulation framework integrated with analytical wire delay and energy models. It builds a system-level NoC simulation infrastructure by employing SystemC's high-level primitive components to ease the modeling effort. Interconnecting wires are modeled in terms of latency and power analytically. These models are integrated with the NoC simulation from dedicated interfaces, which are invoked on each transaction of interest. This framework also supports temporal and spatial NoC power profiling that provides designers more insight when an application is simulated. Applying the NoC simulation framework with wire models on 6 predictive deep sub-micron processes shows that the interconnecting wire will take ~5X more latency and 3.45X more power than the router in a predictive 10*mm*×10*mm* NoC chip under 45*nm* process. The temporal and spatial power profiling illustrate the run-time power properties for each NoC component, and enable designers to perform more power-aware and wire-aware design space explorations.

REFERENCES

[1] N. Genko, D. Atienza, G. De Micheli, J.M. Mendias, R. Hermida and F. Catthoor, "A Complete Network-on-Chip Emulation Framework", In *Proceedings of DATE'05*.

[1] S.G. Pestana, A. Radulesu, K. Goossens and O.P. Gangwal, "Cost-Performance Trade-offs in Networks on Chip: A Simulation-Base Approach", In *Proceedings of DATE'04*.

[2] S. Heo and K. Asanovic, "Replacing Global Wires with an On-Chip Network: A Power Analysis",In *Proc.of ISPLED'05*.

[3] L. Cai, D. Gajski, "Transaction Level Modeling: An Overview", In Proceedings of CODES+ISSS'2003.

[4] D.S.Tortosa et al., "VHDL-based Simulation Environment for PROTEO NoC", In *Proc. of HLDVT Workshop*, 2002.

[5] J. Chen et al., "NoCGen: A Template-Based Reuse Methodology for NoC Architecture", In *Proc. ICVLSI*, 2004

[6] H.S. Wang, X. Zhu, L.S. Peh and S. Malik, "ORION: A Power-Performance Simulator for Interconnection Networks", In *Proceedings of MICRO*, 2002

[7] D. Wiklund, S. Sathe, et al., "NoC Simulations for Benchmarking", In *Proc. IWSoC for Real Time Apps.*, 2004

[8] P. Gupta *et al.*, "A High-level Interconnect Power Model for Design Space Exploration", In *Proc. of ICCAD*, 2003

[9] H.B. Bakoglu, Circuits, Interconnections and Packaging for VLSI. Addision-Wesley, 1990

[10] R. Ho et al., "The Future of Wires", *Proceeding of the IEEE*, 89(4):490-504, Apr., 2001

[11] M. Mamidipaka, N. Dutt and M. Abadir, "Leakage Power Estimation in SRAMs", *CECS Technical Report*, 2003

[12] Berkeley Predictive Technology Model and BSIM4. Available at http://www-device.eecs.berkeley.edu/research