On-Chip Networks Modeling and Simulation

Qi Zhu
Zhengya Zhang
Alessandro Pinto
Alberto L. Sangiovanni-Vincentelli

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2006-126
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-126.html

October 7, 2006

On-Chip Networks Modeling and Simulation

Qi Zhu, Zhengya Zhang, Alessandro Pinto
Alberto Sangiovanni-Vincentelli

ABSTRACT

We present an interconnect model library for the synthesis and design exploration of on-chip communication networks. This library can provide the energy and delay estimation interface to synthesis tools. We start from the definition of interconnect models at different abstract levels, and then estimate the energy consumption and delay of them. To help the synthesis which is oriented to the optimization of energy consumption, the library provides an interface to get the minimal energy within a given bandwidth bound. Furthermore, an quadratic approximation of the relation between energy and wire is proposed to make automatic synthesis easier and more quickly. Some examples on how the entire framework can be used are shown.

1. MOTIVATION

Today's on-chip systems have two peculiar characteristics: complexity and heterogeneity. Complexity comes from the technology scaling and increasing cost of the mask-set, which is the dominant portion of the non-recurrent engineering (NRE) costs. The prohibitive NRE cost forces system integrators to design reusable platforms that is able to support a wide range of applications. Given the heterogeneous nature of today's applications, the platform has to accommodate a variety of functionalities that call for the integration of an increasing number of intellectual properties (IP) on the same die. Integration has been made possible by technology scaling, which decreases the feature size and allows millions of transistors to fit on the same amount of silicon area. Chips with a hundred IP's are becoming realities in today's consumer electronic products.

As stated in [8], local wires will scale with technology scaling while global wires will not. It means that while synthesis of a single IP can still be done using the traditional register transfer level (RTL) design flow (at least for the next few years), IP's integration will become more and more problematic.

The computer aided design community has focused on networks-on-chip as a new design paradigm to overcome the increasing chip complexity. Recently Benini *et al.* have proposed a new paradigm for network on-chip (NOC) design [5]. The on-chip communication problem is based on an approach similar to the micro-network stack model [13]. The authors discussed the design problems and possible solutions for each level of the stack from the application level to the physical level through the topology and protocol level. For the topology selection problem the current standard solution is the use of a single bus, but this may turn out quite inefficient from a energy consumption viewpoint. Hence, the authors of [5] pointed out the energy-saving benefits of using a packet-switching architecture. However, they focused on providing some examples of known topologies and did not discuss the problem of selecting an optimal topology. In [10] the authors proposed a methodology centered on the simulation and analysis of traces. The resulting communication architecture is an interconnection of known and well-characterized communication structures like buses. In [6] the interconnection structure between computation blocks is fixed (in a grid) and predictable. Information is routed in the communication network by means of dedicated switches. In [9] the authors proposed an algorithm to optimize the chip floorplan in order to minimize the point-to-point communication cost among IP's. Starting from the task graph of a given application, the authors proposed an algorithm to implement the communication architecture as a set of point-to-point dedicated links whose cost is minimized by choosing an appropriate floorplan.

Pinto et. al [11] proposed a constraint driven communication synthesis (CDCS) which follows an approach that is inherently different from all the previous ones. CDCS aims to derive a communication architecture as the union of heterogeneous subnetworks that all together satisfy the original communication constraints given by the designer. Each subnetwork is realized by composing elements that are instanced from the communication library. While some constraints may end-up being implemented as point-to-point dedicated channels, others are *merged* together and realized as a single shared communication medium like a bus. The final communication architecture is automatically synthesized by solving a constrained optimization problem.

The goal of our work is to provide a library of interconnect elements to help the synthesis process in CDCS. Since the optimization algorithm searches the best implementation of a given specification in design space, we need to associate the interconnect elements with the performance/cost manifold. An analytical wire model for the computation of delay and energy consumption of an on-chip wire (and parallel wires as well) was given. Different from some traditional schemes which only aim to minimize the delay, CDCS also focuses on optimizing the energy consumption within a given bandwidth requirement. Therefore, the library provides both the interface to get the minimal delay and the interface to get the minimal energy within a given bandwidth requirement. Synthesis tool can utilize these interfaces to estimate the performance and cost of different implementations. Furthermore, the relation between energy and wire length is studied, which will facilitate the auotmatic synthesis progress more.

Our library can be integrated to a SystemC [1] netlist that simulates the network function and performance. The result of the communication synthesis is a network composed of point-to-point links connected through routers.

The paper is organized as follows: section 2 explains the general methodology, section 3 introduces our interconnect models at different abstract levels, section 4 presents the energy-driven optimization, section 5 shows the synthesis results by utilizing our library , and finally section 6 draws a conclusion.

2. METHODOLOGY

We build our library hierarchically, starting from a detailed elementary wire model to an abstract bus model. The methodology of our whole library is shown as following Figure 1.

As in the figure, the elementary wire model is the base of other elements. It deals with the circuit parameters and gives the estimation of delay and energy consumption. The entire wire model is constructed by combining elementary wires through buffers, also it can be seen as the result of inserting buffers to an elementary wire. Bus model is a set of parallel entire wires; each of them will transfer "1" or "0" at a certain time and some interline capacitance may occur when adjacent wires have the opposite value. Furthermore, we can construct higher level models above bus, e.g., router.

All the elements on different abstract levels have the interfaces which can provide the delay and energy estimation for some given parameters. They can also provide the estimation of minimal delay and minimal energy under a bandwidth constrain. Synthesis tools can utilize these interfaces to optimize the on-chip interconnect.

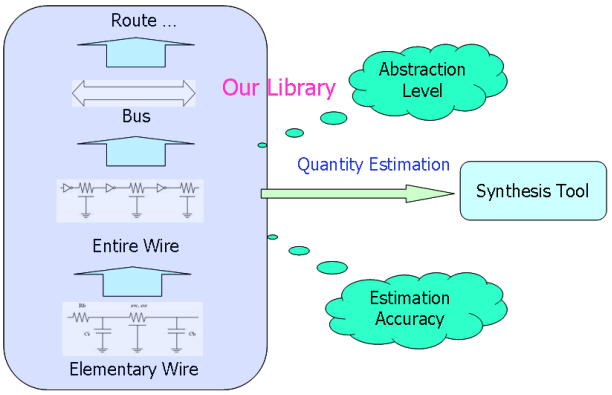

Figure 1: Methodology of Our Library

3. INTERCONNECT MODELS

Long interconnect wires present bandwidth bottlenecks for on-chip networks. The most widely used method to reduce propagation delay through a long wire is to insert buffers and break the long wire into smaller segments [4, 12, 2, 3], as shown in Figure 2. Inserting intermediate buffers may lead to high energy consumption. The trade-off between delay and energy consumption can be explored and optimization could be carried out based on certain metrics, which will be discussed in next section. In our work, we focus on energy consumption instead of power consumption to separate the effect of frequency.

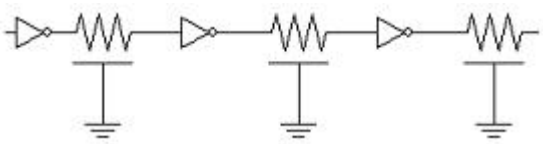

Figure 2: Buffered wire

With buffers inserted, the wire is divided into a number of shorter segments. We start constructing our models from these segments. We use an elementary wire model to characterize delay and energy consumption of a wire segment. Assume an elementary wire of length l_{elet}, resistance r_w, and capacitance c_w. Using a distributed wire model, the propagation delay through the wire $t_{p,w} = 0.38 r_w c_w$. The elementary wire is terminated by one buffer at the front and one buffer at the end. Figure 3 shows the schematic of one such elementary wire. Let buffer intrinsic resistance be R_b, buffer gate capacitance C_b and the intrinsic capacitance C_i. Assume $C_i = C_b$.

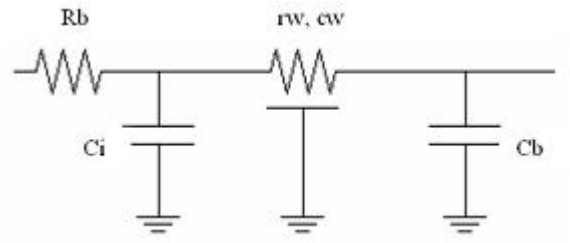

Figure 3: Elementary wire model

Using Elmore Delay formula [7], the delay through the elementary wire is,

$$t_{elet} = 0.69 R_b C_b + 0.69 R_b c_w + 0.69 (R_b + r_w) C_b + 0.38 r_w c_w$$

C_b is proportional to the size of the buffer and R_b is inversely proportional to the size of the buffer. Suppose the minimum buffer has a resistance of R_d and capacitance of C_d. Define $t_{p0} = 0.69 R_d C_d$, which is a constant. We can approximate $R_b \approx \frac{R_d}{s}$ and $C_b \approx s C_d$.

$$\begin{aligned} t_{elet} &= 0.69 \frac{R_d}{s} s C_d + 0.69 \frac{R_d}{s} c_w + 0.69 (\frac{R_d}{s} + r_w) s C_d + 0.38 r_w c_w \\ &= 2 t_{p0} + 0.69 \frac{R_d}{s} c_w + 0.69 r_w s C_d + 0.38 r_w c_w \end{aligned} \quad (1)$$

Wire resistance r_w and capacitance c_w are related to the length of the wire. The longer the wire, the higher the capacitance and resistance. Quantitatively, r_w can be calculated as

$$r_w = R_{square} \frac{l_{elet}}{w_{elet}}$$

w_{elet} is the width of the elementary wire. Moving to a higher metal layer, wire width increases, thus reducing wire resistance. c_w is a sum of area capacitance, fringe capacitance, and interwire capacitance. It is given by,

$$\begin{aligned} c_w &= C_{area} + C_{fringe} + C_{interwire} \\ &= C_a l_{elet} w_{elet} + 2 C_f l_{elet} + n C_i l_{elet} \end{aligned}$$

C_a, C_f, and C_i are unit area, fringe, and interwire capacitances. They are commonly found in data sheets for a given technology and process. Actual interwire capacitance is also data and configuration dependent. If a wire is isolated by itself, its interwire capacitance is zero. This is rarely the case in practice. In a bus configuration, for example, a wire is surrounded by two neighboring wires, one on each side. The switching characteristics on the wire and its neighbors either contribute or reduce the interwire capacitance due to Miller Effect. In the worst case, for instance, the wire undergoes high-to-low switching, and two neighbors both carry on low-to-high switching. The resulting interwire capacitance will be four times as large. In the best case, the wire and two neighbors carry on switching in the same direction, reducing the interwire capacitance to zero.

From equation 1, we can minimize propagation delay t_{elet} with respect to buffer size s.

$$\begin{aligned} \frac{\partial t_{p0}}{\partial s} &= -0.69 \frac{R_d}{s^2} c_w + 0.69 r_w C_d = 0 \\ s_{opt} &= \sqrt{\frac{R_d c_w}{r_w C_d}} = \sqrt{\frac{R_d C_w l_{elet}}{R_w l_{elet} C_d}} = \sqrt{\frac{R_d C_w}{R_w C_d}} \end{aligned} \quad (2)$$

C_w and R_w are unit capacitance and resistance of the wire. Notice that the optimal buffer size s is not dependent on wire length. Rather it is related to metal layer and technology.

We can also find the impact on energy consumption when buffers are introduced. The average energy consumption per transition is give by,

$$E_{elet} = \alpha (2 C_b + c_w) V_{dd}^2 = 2 \alpha s C_d V_{dd}^2 + \alpha c_w V_{dd}^2$$

Where α is the activity factor. When buffers are used, extra energy is dissipated. With larger buffers, energy dissipation will be higher.

Using the elementary wire model as an abstraction, interconnect wires can be modeled as a concatenation of elementary wires. Consider equal division of an interconnect wire. We made this assumption because it is usually true in implementation and equal distance will give a short delay. The number of elementary wire segments is give by $m = \left\lceil \frac{L_{total}}{l_{elet}} \right\rceil$ or $\frac{L_{total}}{l_{elet}}$ for long wires. We assume the latter one for convenience. The total delay through the long wire will be

$$\begin{aligned} t_{total} &= m t_{elet} = \frac{L_{total}}{l_{elet}} (2 t_{p0} + 0.69 \frac{R_d}{s} c_w + 0.69 r_w s C_d + 0.38 r_w c_w) \\ &= L_{total} (\frac{2 t_{p0}}{l_{elet}} + 0.69 \frac{R_d C_w}{s} + 0.69 R_w s C_d + 0.38 R_w C_w l_{elet}) \end{aligned} \quad (3)$$

Result from equation 3 is intuitive. When no buffer is inserted, $l_{elet} = L_{total}$ and only the last term in the equation remains, so $t_{total} = 0.38 R_w C_w L_{total}^2$. As more buffers are added, l_{elet} decreases, reducing wire delay but also

introducing buffer delays and delays from coupling between wire and buffers. We can minimize t_{total} with respect to l_{elet} to find the optimal elementary wire length.

$$\begin{aligned} \frac{\partial t_{total}}{\partial l_{elet}} &= L_{total}(-\frac{2t_{p0}}{l_{elet}^2} + 0.38R_wC_w) = 0 \\ l_{opt,elet} &= \sqrt{\frac{2t_{p0}}{0.38R_wC_w}} \end{aligned} \quad (4)$$

Optimal elementary wire length is independent on total wire length. It is only related to metal layer and technology. Using the optimal buffer size found in 2 and optimal elementary wire length found in 4, we can calculate the optimal interconnect wire delay.

$$\begin{aligned} t_{opt,total} &= L_{total}(\sqrt{0.38R_wC_w2t_{p0}} + 0.69\sqrt{R_wC_wR_dC_d} \\ &+ 0.69\sqrt{RwC_wR_dC_d} + 0.38\sqrt{\frac{2R_wC_wt_{p0}}{0.38}}) \\ &= 2.82L_{total}\sqrt{R_wC_wR_dC_d} \end{aligned}$$

Optimal delay depends on wire length linearly. This makes the buffer insertion strategy especially attractive to long wires. We then derive the total energy consumption per transition on the wire.

$$\begin{aligned} E_{total} &= mE_{elet} = \frac{L_{total}}{l_{elet}}(2\alpha sC_dV_{dd}^2 + \alpha c_wV_{dd}^2) \\ &= L_{total}(\frac{2\alpha sC_dV_{dd}^2}{l_{elet}} + \alpha C_wV_{dd}^2) \end{aligned} \quad (5)$$

Equation 5 suggests that as we introduce more buffers and larger buffers, the first term increases and energy consumption goes up. To minimize energy consumption, the intuitive way is not to use any buffers at all, but this will lead to a long delay. In real application, we can optimize the energy consumption within some range. There is generally a trade-off between delay and energy consumption.

4. ENERGY DRIVEN OPTIMIZATION

The optimal energy consumption can be obtained at a given bandwidth, which is the reciprocal of the delay. Given a delay requirement, we can use Equation 5 and 3 to optimize energy consumption. Since there is no closed form solution for the energy optimization problem, we will simply search the available buffer sizes and elementary wire lengths. The search is carried out under the delay constraint. We can enumerate the buffer sizes since we can assume it is a discrete number. By analyzing the equations, range of enumeration can be set. For example, the buffer size for the optimal delay will be the upper bound. Because if the buffer size is bigger than this value,we can reduce the size to achieve lower energy consumption and delay at the same time, which means it will not be the size for optimal energy consumption. Following figure 4 depicts the raw constraints for the ranges of elementary wire length and buffer size, which may give an optimal energy consumption. We only need to search the elementary wire lengths which are larger than the optimal delay elementary wire length, and the buffer sizes which are smaller than the optimal delay buffer size. We can shrink these ranges by more constraints to fast the search process.

Intuitively we can achieve a lower energy consumption with a loose delay requirement. Figure 5 depicts the relationship between energy consumption and delay bound. X-axis is the normalized delay , i.e., the delay bound over optimal delay; Y-axis is the normalized energy, i.e., optimal energy within the delay bound over the energy for the optimal delay.

This figure was derived based on 0.25μm technology and 0.18μm technology, using a total wire length of 20mm. We see that the energy consumption can be reduced to a great extent when the delay bound is relaxed. For example, in 0.25μm technology, 110% delay requires only 72.5% energy. For applications with loose delay constraints, we can optimize for energy consumption, rather than for minimal delay. Furthermore, when technology scales down, we get more energy savings from optimizing

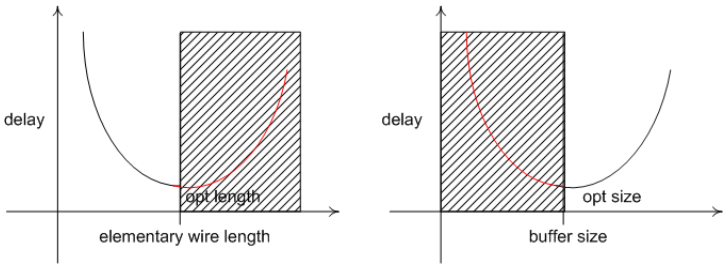

Figure 4: Constraints of Elementary Wire Length and Buffer Size

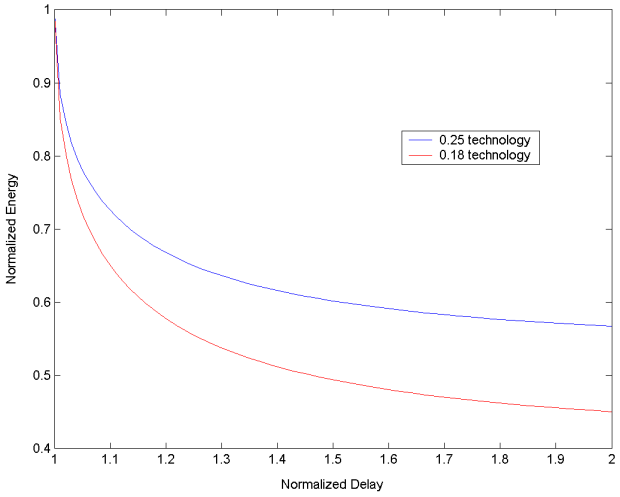

Figure 5: Normalized Energy VS Delay

energy within a certain delay bound. In 0.18μ technology, for example, 110% delay only requires 65% energy, even less than that in 0.25μm technology. This is mainly because the effect of leakage energy becomes increasingly prominent in submicron processes. It will increase the effect of first part in Equation 5, so we can get more benefits in energy optimization. And this observation makes our energy optimization strategy more advantageous in newer processes, which has a newer technology.

Above analysis shows the effect of energy optimization and our library provides the interfaces to do this optimization. We can utilize these interfaces to help synthesis. And more thing can be done; following we will discuss how to make the automatic synthesis easier by providing the function between energy consumption and total wire length. This will be used in the automatic topology synthesis algorithm which optimizes for the energy. For a given delay bound or a given bandwidth bound, we can first compute the optimal energy consumption versus different total wire lengths. Figure 6 is an example, in which the bandwidth requirement is 100M bit/s and the total length varies between 0.1mm and 100mm. For shorter wire length, the relationship between energy and length is linear while for longer wire length, this relationship becomes quadratic. The reason is that when the wire length goes up, larger buffers and shorter elementary wire lengths are needed, so the first term of Equation 5 will have a larger effect, resulting in a non-linear relationship. When the total wire length exceeds a certain value, it cannot meet the bandwidth requirement no matter what buffer size and elementary wire length are chosen. By observing curve shapes under different bandwidth requirements, we chose a quadratic function $f = cx^2$ to approximate the relationship between energy consumption and total wire length. One reason of choosing quadratic function is that it can help the synthesis algorithm more, and another reason is that this approximation is close to the reality when the total wire

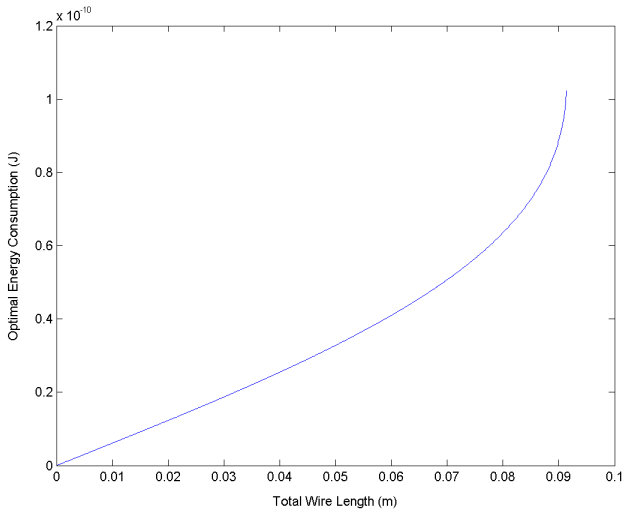

Figure 6: Optimal Energy versus Total Wire Length

length goes up, which is more valuable to be dealt with. The coefficient *c* is related to bandwidth requirement *b*. By using function fitting scheme, we get *c* versus bandwidth relation, as shown in Figure 7. The 0.25 *um* technology was used, and the bandwidth varies from 10M bit/s to 500M bit/s. We find that when the bandwidth is below a certain value, e.g. 80M

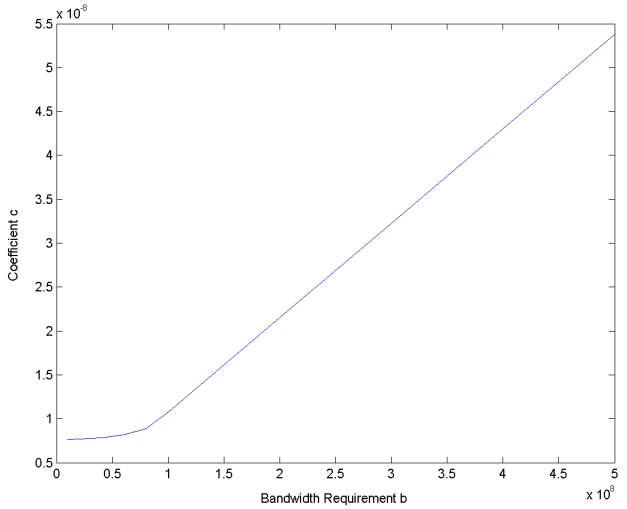

Figure 7: Coefficient c versus Bandwidth Requirement b

bit/s in Figure 7, the relationship between *c* and bandwidth *b* is concave. Past that value, the function is almost linear. This property is useful for synthesis, which will be explained later. We use a linear function to approximate the relation between *c* and *b*. Finally, we can approximate the relation between optimal energy and total wire length using the following equation,

$$E_{total} = CbL_{total}^2$$

where *C* is a constant, and *b* is bandwidth requirement. This equation is more accurate for longer wire and higher bandwidth requirement.

5. SYNTHESIS

First we will briefly introduce the SystemC library we built. All the elements are constructed hierarchically. The elementary wire model is the base of other models, and the higher abstract models are constructed based on the lower ones. All elements have the interfaces to provide delay and energy estimation, also the interfaces to get optimal delay and optimal energy. To help the automatic synthesis, the function between energy and wire length is also provided. Furthermore, synthesis tools can configure the models by giving many parameters, such as the technology parameters, type of optimization, wire length and so on.

For the CDCS synthesis tool, the models used are a point-to-point(PtP) bus model and a router model. PtP bus can be configured through parameters, such as those ones mentioned above, and the router is constructed by providing a routing table and the FIFO length. Following Figure 8 shows the whole synthesis methodology.

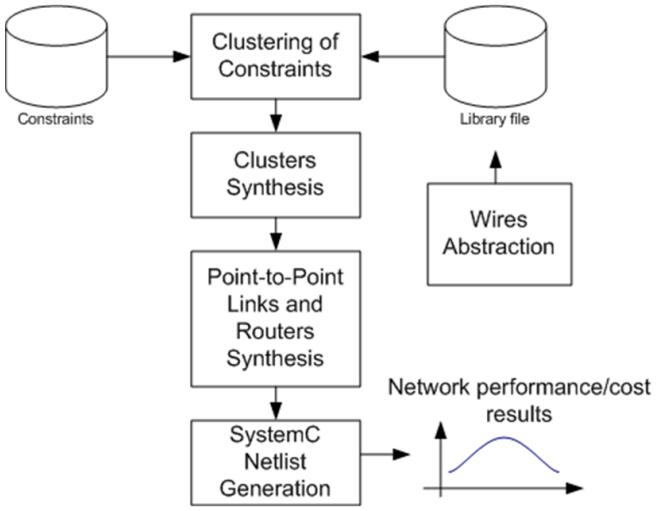

Figure 8: Synthesis Methodology

The synthesis results are shown as follows. Figure 9 depicts the result of automatic synthesis, which utilizes the function between energy and wire length. The left graph is the interconnect constraints, and the right graph is the topology synthesis result.

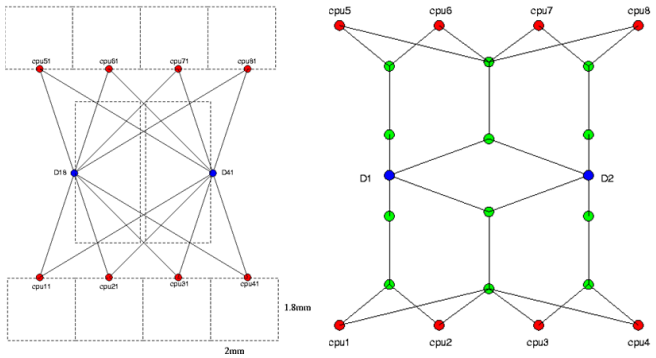

Figure 9: Synthesis Topology

Figure 10 shows the comparison of energy optimization and delay optimization. We do the computation based on the result of topology synthesis mentioned above. From the result, we can see that energy optimization scheme can greatly reduce the energy consumption.

6. CONCLUSION

We developed an analytical model for delay and energy estimation of on-chip wires. Also we developed two algorithms for delay and energy

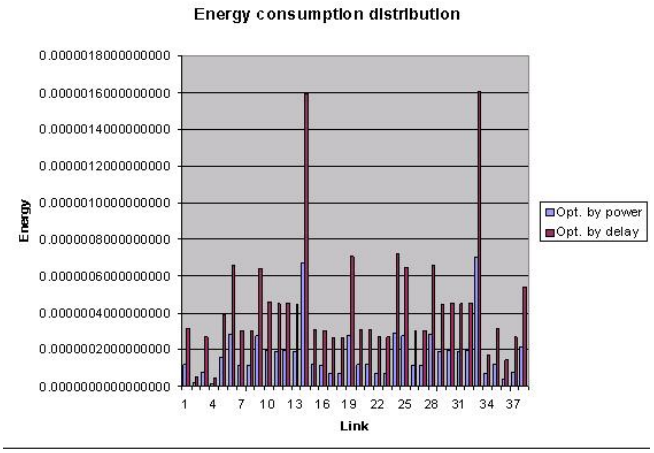

Figure 10: Energy Optimization VS Delay Optimization in Synthesis

optimization for given length and bandwidth constraints. Based on these, we built a library of communication components for on-chip networks synthesis and simulation.

Future work may include adding the effect of on-chip routing and limited buffer availability; building estimation models for more complex structures, e.g., building a more complex bus model; and comparing the results with final implementations.

7. REFERENCES

[1] *SystemC 2.0*. available at http://www.systemc.org.

[2] C. J. Alpert and A. Devgan. Wire segmenting for improved buffer insertion. In *Proc. of the Design Automation Conf.*, pages 588–593, 1997.

[3] C. J. Alpert, A. Devgan, and S. T. Quay. Buffer insertion for noise and delay optimization. In *Proc. of the Design Automation Conf.*, pages 362–367. ACM Press, 1998.

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

[5] L. Benini and G. De-Michely. Networks on-chips: A new soc paradigm. *IEEE Computer*, January 2002.

[6] William J. Dally and Brian Towles. Route packets, not wires. In *Proc. of the Design Automation Conf.*, pages 684–689, 2001.

[7] W. C. Elmore. The transient response of damped linear networks with particular regard to wide-band amplifiers. *J.Appl.Phys.*, 19(1):55–63, 1948.

[8] R. Ho, K. Mai, and M. Horowitz. The Future of Wires. *Proc. of the IEEE*, 89(4):490–504, April 2001.

[9] Radu Marculescu Jingcao Hu, Yangdong Deng. System-level point-to-point communication synthesis using floorplanning information. In *ASP-DAC/VLSI*, January 2002.

[10] K. Lahiri, A. Raghunathan, and S. Dey. Efficient exploration of the soc communication architecture design space. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 424–430, 2000.

[11] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli. Constraint-Driven Communication Synthesis. In *Proc. of the Design Automation Conf.*, pages 783–788. IEEE, June 2002.

[12] L. P. P. P. van Ginneken. Buffer placement in distributed rc-tree networks for minimal elmore delay. In *Proc. Intl. Symposium on Circuits and Systems*, pages 865–868, May 1990.

[13] J. Walrand and P. Varaija. *High Performance Communication Networks*. Morgan Kaufmann,San Francisco, 2000.