AN ANALYTICAL MODEL DESCRIBING THE RELATIONSHIPS BETWEEN LOGIC ARCHITECTURE AND FPGA DENSITY

Andrew Lam[1]*, Steven J.E. Wilton*[1]*, Philip Leong*[2]*, Wayne Luk*[3]

[1]Elec. and Comp. Engineering
University of British Columbia
Vancouver, B.C., Canada
{andrewl,stevew}@ece.ubc.ca

[2] Computer Science and Engineering
Chinese University of Hong Kong
Hong Kong
phwl@cse.cuhk.edu.hk

[3]Dept. of Computing
Imperial College
London, U.K.
wl@doc.ic.ac.uk

ABSTRACT

This paper describes an analytical model, based principally on Rent's Rule, that relates logic architectural parameters to the area efficiency of an FPGA. In particular, the model relates the lookup-table size, the cluster size, and the number of inputs per cluster to the amount of logic that can be packed into each lookup-table and cluster, and the number of used inputs per cluster. Comparison to experimental results show that our models are accurate. This accuracy combined with the simple form of the equations make them a powerful tool for FPGA architects to better understand and guide the development of future FPGA architectures.

1. INTRODUCTION

Field-Programmable Gate Arrays (FPGAs) have evolved considerably since their introduction. A dramatic increase in the number of available transistors has motivated research in both academia and industry, leading to new logic block structures, flexible embedded blocks, and complex interconnect networks.

FPGA architectural enhancements are often developed in a somewhat ad-hoc manner. Expert FPGA architects perform experiments in which benchmark circuits are mapped using representative computer-aided design (CAD) tools, and the resulting density, speed, and/or power dissipation are estimated [1, 2, 3]. Based on the results of these experiments, architects use their intuition and experience to design new architectures, and then evaluate these architectures using another set of experiments. This is repeated numerous times, until a suitable architecture is found. This process occurs both within FPGA companies as new generations of devices are developed, and within academia, as new architectural innovations are investigated.

During this process, there is virtually no body of theory that architects can use to guide their investigations. Previous studies have produced empirical "rules of thumb" (such as the best logic element size), however, these rules of thumb usually provide no insight into the underlying tradeoffs between flexibility and density, speed, and power dissipation.

Such insight, however, would be extremely valuable for two reasons. First, understanding the relationships between architectural parameters enables *early-stage architecture development* [4] in which the design space can be searched quickly using analytical models. Once a promising region of the architecture space has been identified, traditional experimental methods can be used to choose precise architectural parameters. This would significantly accelerate the FPGA architecture design process. It may also allow the study of a wider variety of "interesting" architectures since experimental CAD tools need not be developed for each architecture under consideration. Second, the development of such theory will encourage researchers to understand *why* certain architectures work well, and may eventually provide bounds on the capabilities and efficiencies of programmable logic.

This paper is a step towards such a body of theory. Specifically, this paper presents an *analytical model that describes the relationship between logic block and cluster parameters and the area-efficiency of the resulting FPGA*. The inputs of the model are the lookup-table size, cluster size, and inputs per cluster. The outputs are (1) the expected number of two-input logic gates that can be packed into each lookup-table, (2) the expected number of lookup-tables that can be packed into each cluster, and (3) the expected number of inputs of each cluster that are used. The first two outputs can be used to deduce the density of an FPGA implementation, while the third output can be used as an input to the channel width model presented in [4]. Together with [4], our model can be used to quickly evaluate a wide variety of lookup-table/cluster architectures, without requiring time-consuming empirical experiments.

This paper is organized as follows. Related work and preliminary background are described in Sections 2 and 3, respectively. The model itself is described in Section 4, and it is validated against experimental results in Section 5. Section 6 gives an example of how the model can be used as a tool in FPGA architectural investigation.

978-1-4244-1961-6/08/$25.00 ©2008 IEEE.

2. RELATED WORK

Several previous publications have described analytically-derived relationships between FPGA architectural parameters. Much of the work focuses on the routing fabric. El Gamal derives a model that relates that relates the area required for routing to the total number of pins in the logic gates [5]. Although this work was originally designed to describe a non-programmable chip, the model has been used in the design of numerous generations of FPGAs [6]. Work by Brown et al relates various parameters that describe an FPGA routing architecture to the routability of that architecture [7], and more recently Fang and Rose related the same architectural parameters to the channel width of an FPGA [4]. Pistorius and Hutton relates the Rent parameter (the Rent parameter is a measure of the "complexity" of the interconnect pattern in a circuit [8]) of a circuit to various architectural parameters [9].

There has also been much work on interconnect prediction. Early work by Donath [10] and others related the area requirements of routing wires to the Rent parameter of a circuit. Later work by Stroobandt refined the models to consider more realistic network topologies and architectural assumptions [11]. More recent work has produced more accurate estimates of routing area using more information from the circuit to be implemented [12].

The previous work closest to ours is by Gao et al, who relates lookup-table size to area in a non-clustered FPGA [13]. Compared to that work, we consider clustered architectures (which are more representative of real FPGAs) and use cluster architectural parameters in our equations.

3. PRELIMINARIES

3.1. Guiding Principles

Three principles guided us in the development of this model.

First, we endeavored to develop the model by deriving relations analytically, without relying on curve-fitting or experimental techniques. This ensures that we are capturing the "essence" of programmable logic, and not creating a model that is limited to a particular CAD flow or tool suite. As will be discussed in Section 4, all aspects of our model were derived analytically[1] except for one parameter, γ.

Second, since we wish to derive relations between architectural parameters, we attempted to create a model that is independent of the circuit to be implemented on the FPGA. For example, we would prefer a relation between block size and cluster size that is independent of the circuit to be implemented. This is different from much prior *estimation* work, in which the goal is to predict the area, speed, or power for a given circuit [12]. That being said, it is impossible to completely ignore the impact of the circuit; we describe our circuit using the Rent Parameter and size of the un-techmapped circuit.

Third, we attempted to balance the complexity of our equations with their accuracy. We feel that simple equations provide significantly more insight into architectural tradeoffs than expressions which are unnecessarily complex.

[1]Technically, Rent's Rule is an empirical expression, yet we use it as part of our analytical derivation. Rent's Rule is well accepted, so we do not feel it violates the spirit of this guiding principle.

Table 1. Model Parameters

Architectural Parameters:	
K	Number of inputs per lookup table
N	Number of lookup tables per cluster
I	Number of inputs per cluster
Circuit Parameters:	
p	Rent parameter of a given circuit
n_2	Number of 2-LUTs in a given circuit
Implementation Parameters:	
n_k	Number of K-LUTs needed to implement a given circuit
n_c	Number of clusters needed to implement a given circuit
c	Average number of LUTs packed into each cluster ($c = n_2/n_c$)
i	Average number of inputs used in each cluster
o	Average number of outputs used in each cluster
f	Average fanout of all nets in the circuit
γ	Average number of inputs *not* used in each LUT

3.2. Model Parameters

Table 1 summarizes the parameters used to describe the architecture, circuit, and implementation. In general, uppercase letters are used for architectural parameters, and lowercase letters are used for circuit parameters and parameters that describe the implementation of the circuit on a given architecture.

4. MODEL DERIVATION

Although our goal is to model architectures, our model mirrors the CAD flow used when mapping circuits to FPGAs. The first part of our model mirrors the process of technology mapping, in which 2-input gates are mapped to K-LUTs. This part of the model can be used to compute the number of K-LUTs needed to implement a circuit as a function of K. The second part of the model mirrors clustering, in which K-LUTs are packed into clusters with a pre-defined capacity and a pre-defined number of unique inputs [1]. The inputs of this part of the model are the number of K-LUTs in a circuit as well as the cluster parameters N (the capacity of each cluster) and I, the maximum number of available input pins on each cluster, and the outputs are two quantities that describe the clustering: the expected number of LUTs that can be packed into each cluster, and the expected number of inputs to each cluster that are used. In the following, we focus on each part of the model separately.

4.1. Number of K-LUTs to implement a circuit

We first model the process of technology mapping. Consider an un-techmapped circuit consisting of n_2 two-input LUTs. During technology mapping, these n_2 gates will be mapped into a smaller number of LUTs, which we will denote n_k. In this section, we seek a closed form expression for n_k.

Consider a portion of the *un*-techmapped-circuit consisting of x two-input gates ($1 < x \leq n_2$). Denote the number of signals that connect across the boundary of this region as y. Since each gate has three pins (two inputs and one output), we can use Rent's Rule [8] to write:

$$y = 3x^p \tag{1}$$

where p is the Rent parameter of the circuit. Now suppose this same region is mapped to z K-LUTs using a technology mapping algorithm. The number of signals that connect across the boundary of this region is still y. The number of pins used in each K-LUT is $1 + K - \gamma$ (the first two terms correspond to the output and K inputs, and the final term, γ, will be described below). We can then write

$$y = (K + 1 - \gamma)z^p \tag{2}$$

Since y is the same in both equations, we can eliminate y to get

$$\frac{z}{x} = \sqrt[p]{\frac{3}{K + 1 - \gamma}}. \tag{3}$$

In other words, every z K-LUTs can implement x 2-LUTs in the original circuit. Intuitively, this ratio is a measure of how much logic can be packed into each lookup-table.

Finally, using Equation 3, we can write:

$$n_k = n_2 \sqrt[p]{\frac{3}{K + 1 - \gamma}} \tag{4}$$

The term γ in Equations 2 to 4 arises because not all K inputs are always used in a K-LUT. γ is the expected number of inputs to a K-LUT that are *not* used. We have not found a way to accurately model this analytically, however, experimentally we have found that γ (as a function of K) is extremely consistent across all benchmark circuits we considered. Table 2 shows our measured values of γ; the derivation of a closed form for this expression is an interesting topic of future work.

Table 2. γ values from 20 MCNC benchmarks

K	2	3	4	5	6	7
γ	0.000	0.261	0.466	0.701	0.996	1.232

4.2. Number of clusters needed to implement a circuit

We next model the process of clustering. Consider a technology mapped circuit consisting of n_k K-LUTs. During clustering, these n_k LUTs will be packed into a smaller number of clusters, which we will denote n_c. In this subsection, we seek closed-form expressions for n_c. In the next subsection, we derive an expression for the expected number of inputs used per cluster, i.

Each cluster can contain up to N LUTs and have up to I unique inputs. In an architecture with a large value of I and small value of N, it is likely that most clusters will be completely filled, while in architectures with a small value of I, some clusters may not be completely filled, because of the limitation on the number of unique inputs. Since we wish our model to apply to both types of architectures, we consider each case separately below. We also define an equation for the boundary between the two cases[2].

4.2.1. I-Limited Clustering

We first consider architectures in which I is small, and the expected number of LUTs packed into each cluster is dictated by the number of physical pins on each cluster. In this case, the expected number of LUTs packed into each cluster, $c = n_k/n_c$, will be smaller than the capacity of the cluster N. To estimate c, and hence n_c, we employ Rent's Rule as follows. Consider the same region of the technology-mapped circuit from Section 4.1 which contains z K-LUTs and has y signals that connect outside the region. When the same region is mapped to clusters, we can write

$$y = (i + o)v^p \tag{5}$$

where v is the number of clusters needed for this region ($v \leq z$), i is the average number of used inputs per cluster, and o is the average number of used outputs per cluster. The latter quantity can be written as $o = i/f$ where f is the average fanout of the circuit (this term will be computed below). Thus, we can write:

$$y = i(1 + \frac{1}{f})v^p \tag{6}$$

Eliminating y from Equations 2 and 6 and solving for n_c gives:

$$n_c = n_k \sqrt[p]{\frac{K + 1 - \gamma}{I(1 + \frac{1}{f})}}, \tag{7}$$

and

$$c = \sqrt[p]{\frac{I(1 + \frac{1}{f})}{K + 1 - \gamma}} \tag{8}$$

[2]Note that in real FPGA implementations, regardless of N and I, some clusters will be full and some will not be. We could model this by considering a *distribution* of packing capacities rather than a single expected value. We have chosen to model a single expected value, since we feel it provides simpler equations, and thus more insight into the underlying tradeoffs between the architectural parameters.

The fanout f in Equations 7 and 8 can be calculated using a formula from [14]:

$$f = \frac{1-(f_{max}+1)^{(p-1)}}{1-(f_{max}+1)^{(p-2)}-\phi(p,f_{max})} - 1 \quad (9)$$

where:

$$\phi(p, f_{max}) = \sum_{n=1}^{f_{max}} \frac{n^p}{n^2(n+1)}, \quad (10)$$

and f_{max} is the maximum fan-out written as:

$$f_{max} = [(i+N)\frac{n_k}{N}(1-p)]^{(1/(3-p))}. \quad (11)$$

In Equation 11 we approximated the number of outputs as N and the number of clusters as n_k/N; experimentally, we have found that the fanout is only a weak function of f_{max}, and thus these approximations lead to only a small error.

4.2.2. N-Limited Clustering

This case is trivial. Since cluster input pins are plentiful, clusters can be filled to capacity. Hence, $c = N$ and

$$n_c = \frac{n_k}{N} \quad (12)$$

4.2.3. Boundary Condition

For architectures in which $c < N$, clustering is I-limited, otherwise it is N-limited. Using Equation 8, we can write the following condition that indicates that clustering is I-limited:

$$\sqrt[p]{\frac{I(1+\frac{1}{f})}{K+1-\gamma}} < N \quad (13)$$

This can be rearranged to produce:

$$I < N^p \frac{K+1-\gamma}{1+\frac{1}{f}} \quad (14)$$

For all values of I in which Inequality 14 holds, clustering is I-limited.

4.3. Average Number of Used Inputs

Again, we consider I-limited and N-limited architectures separately. The boundary condition between the two types of architectures is the same as in the previous section.

4.3.1. I-Limited Clustering

In these architectures, we would expect all cluster input pins to be used. Thus, we can write

$$i = I \quad (15)$$

4.3.2. N-Limited Clustering

By applying Rent's Rule to a single cluster, we obtain:

$$i + o = (K+1-\gamma)N^p, \quad (16)$$

since a cluster has N K-LUTs that each has $(K+1-\gamma)$ I/O pins and $i+o$ number of exterior pins.

By substituting $o = i/f$ into (16) and solving for i, we obtain

$$i = \frac{(K+1-\gamma)N^p}{1+\frac{1}{f}}. \quad (17)$$

5. MODEL VERIFICATION

To evaluate the accuracy of our model, we compare the model predictions to experimental results.

Figure 1(a) illustrates the accuracy of our technology mapping model. The experimental results were obtained by averaging the results for twenty large MCNC benchmark circuits. Results for two different technology mapping algorithms are shown. The analytical results were obtained from Equation 4 using the average Rent parameter from all benchmark circuits. As the graph shows, the analytical results track the experimental results very closely.

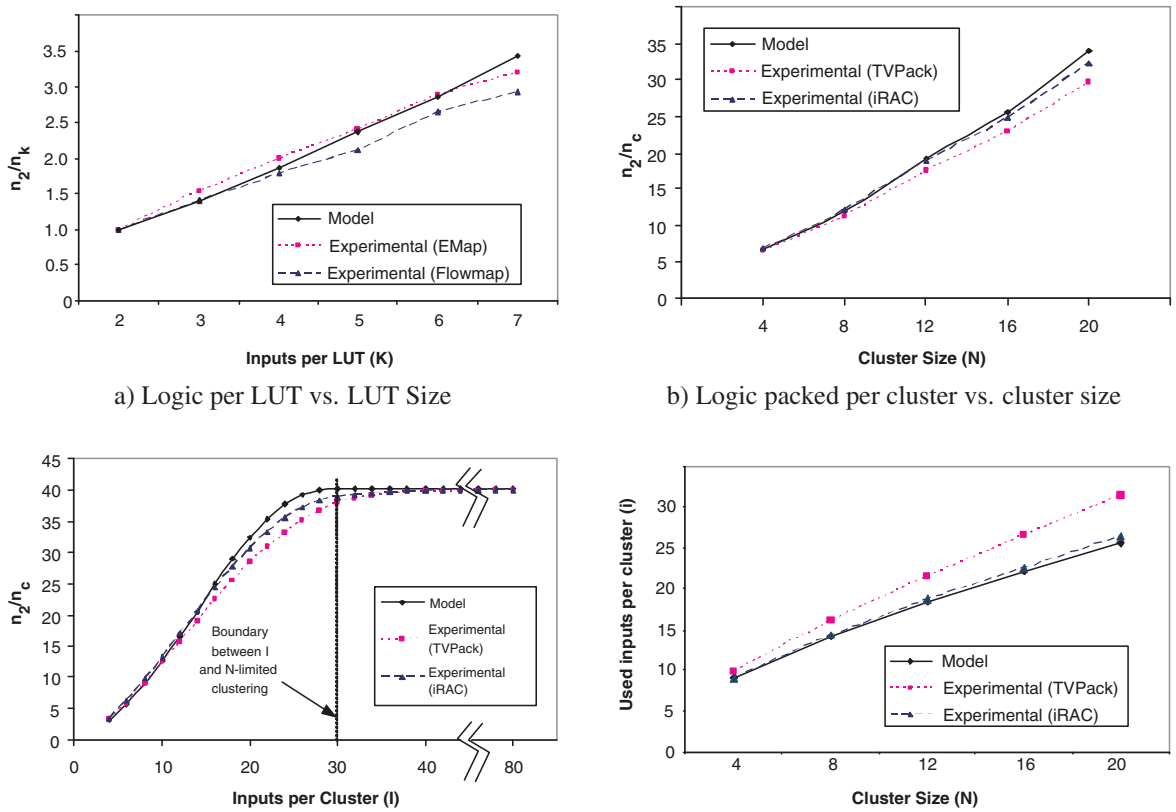
Figures 1(b) through 1(d) illustrate the accuracy of our clustering model. Figure 1(b) shows the ratio $\frac{n_2}{n_c}$ as a function of cluster size. In all cases, we set $K = 4$, and I to be $0.88N + 3.2$ from [4] (this ensures that our clustering is I-limited, which is the interesting case for this graph). The analytical results were obtained using Equation 7 and 4, while the experimental results were obtained using two separate clustering algorithms, T-Vpack [1] and our implementation of iRAC [15]. Again, the analytical results are very consistent with both sets of experimental results.

Figure 1(c) shows the same ratio as a function of the number of input pins per cluster, I. In all cases, $K = 4$ and $N = 20$. The boundary between I-limited and N-limited architectures is also shown. The graph shows that our model tracks the experimental results well in both regions.

Finally, Figure 1(d) shows the average number of used inputs per cluster as a function of cluster size. Although our model tracks both sets of experimental results, it matches the iRAC results more closely. This is expected, since iRAC explicitly tries to minimize the use of cluster pins.

6. EXAMPLE APPLICATION OF OUR MODEL

One of the purposes of our model is to allow for early architectural evaluation. In this section, we show how the model, along with the channel width model from [4], can be used to estimate the number of programming bits in an FPGA as a function of the architectural parameters K, N, and I. We will investigate whether an analytical flow employing our



a) Logic per LUT vs. LUT Size

b) Logic packed per cluster vs. cluster size

c) Logic packed per cluster vs. cluster inputs

d) Used Inputs per Cluster

Fig. 1. Model Validation

model leads to similar conclusions that would be obtained by a more time-consuming experimental methodology.

We consider three flows:

1. The first flow is purely analytical. We use the model described in Section 4, to estimate $\frac{n_2}{n_c}$ and i. These quantities are then used in conjunction with the channel width model in [4] to determine the amount of routing needed for a given architecture. We then use equations that model the number of programming bits in a clustered logic block, connection block, switch block. These equations are similar to those used within VPR 5.0, and assume an architecture with uni-directional, single-driver wires. Multiplexers are assumed to be implemented using a two-tiered structure with one-hot select lines (as in VPR 5.0). Finally, these area estimates are used to determine the number of programming bits required for each of twenty large benchmark circuits. Note that this flow is purely analytical and does not require experimental CAD tools.

2. The second flow is purely experimental. We map each of the twenty benchmark circuits to 4-input LUTs using Flowmap, clusters containing four LUTs and 10 inputs using TVPack, and then place and route the circuits using VPR 5.0. We use the model within VPR 5.0 to count the number of programming bits for each benchmark circuit.

3. As an intermediate between the first two flows, we use the model in Section 4 to estimate $\frac{n_2}{n_c}$ and i, but use VPR 5.0 to determine the actual channel width (instead of using the model from [4]). This flow is included to provide insight into each portion of the analytical model.

Figure 2 shows the results for an architecture with $F_s = 9$, $F_{cin} = 20$, $F_{cout} = 4$, and $L = 1$. Figure 2(a) shows the number of programming bits as a function of K, Figure 2(b) shows the number of programming bits as a function of N, and Figure 2(c) shows the number of programming bits as a function of I.

In all cases, the results from Flow 2 (purely experimental) and Flow 3 (analytical technology mapping and clustering but experimental routing) match closely. This is to be expected, given the accuracy illustrated in Figure 1.

When the model from [4] is used (Flow 1), the analytical estimates still track the experimental results, however, not as

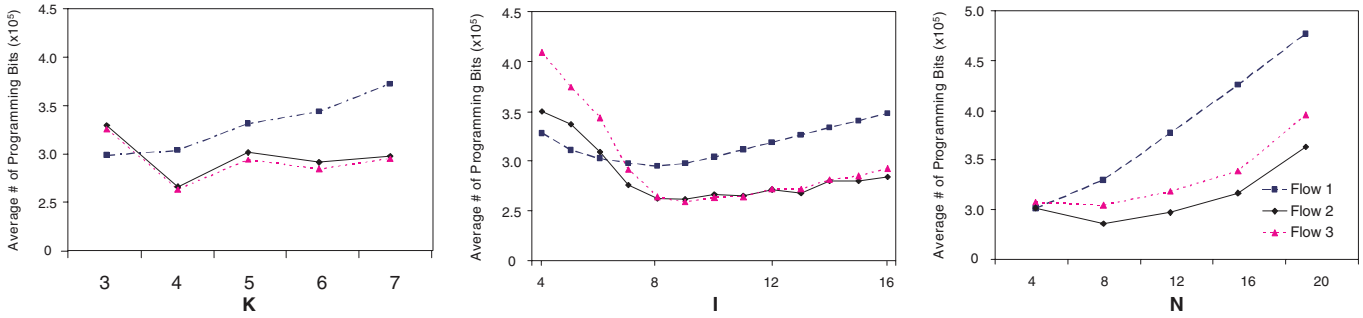


Fig. 2. Routing Bit Model Verification

closely. The largest differences arise for architectures with a small number of cluster inputs. These architectures contain far fewer inputs per cluster than the equations in [4] were intended to model. We have found that for such "extreme" architectures, the average wirelength is 25% larger than the architectures considered in [4]. This is primarily because as I decreases, the sharing of LUT inputs within a cluster is encouraged. This tends to decrease the average fanout, which increases the average wirelength.

This is an important observation – although the model presented in this paper correctly models architectures with small values of I, a complete analytical flow that accurately models these architectures does not yet exist. Addressing this limitation is an interesting area for future research.

7. CONCLUSION

In this paper, we have presented an analytical model that relates lookup-table size, cluster size, and the number of inputs per cluster to amount of logic that can be packed into each lookup-table and cluster, and the number of used inputs per cluster. Comparing the model predictions with experimental results has shown that our model is accurate.

We have shown that the model can be used during early architecture evaluation, in which potential architectures are considered before custom CAD tools are created. However, the model has the potential to be much more powerful than that. By combining our model with that from [4], we have a means of relating routing architecture parameters with lookup-table and cluster size. Recent FPGAs are employing larger lookup-tables, and it seems reasonable to expect that the logic architecture of future FPGAs may change further. Using a model such as ours provides the ability to understand the impact that potential logic block architectures may have on the routing fabric, and more importantly, *why* the routing fabric is impacted in a certain way. For FPGA architects, this could potentially be much more useful than experimental results that evaluate a single architecture (or even a sweep of a single architectural parameter).

A C-implementation of our model can be downloaded from `http://www.ece.ubc.ca/~stevew`.

8. REFERENCES

[1] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-submicron FPGAs*. Kluwer Academic Publishers, 1999.

[2] M. Hutton, "FPGA architecture design methodology," in *Int'l Conf. on Field-Programmable Logic and Applications*, Aug. 2006, p. 1.

[3] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Trans. on VLSI Systems*, vol. 12, no. 3, pp. 288–298, Mar. 2004.

[4] W. Fang and J. Rose, "Modeling FPGA routing demand in early-stage architecture development," in *Int'l Symp. on Field-Programmable Gate Arrays*, Feb. 2008.

[5] A. A. El Gamal, "Two-dimensional stochastic models for interconnections in master-slice integrated circuits," *IEEE Trans. on Circuits and Systems*, vol. 26, no. 4, pp. 127–138, Feb. 1981.

[6] J. Rose, "Closing the gap between FPGAs and ASICs, Keynote Address," in *Int'l Conference on Field-Programmable Technology*, Dec. 2006.

[7] S. Brown, J. Rose, and Z. Vranesic, "A stochastic model to predict the routability of field-programmable gate arrays," *IEEE Trans. on Computer-Aided Design of Circuits and Systems*, vol. 12, no. 12, pp. 1827–1838, Dec. 1993.

[8] B. Landman and R. Russo, "On a pin vs. block relationship for partitions of logic graphs," *IEEE Trans. on Computers*, vol. C-20, pp. 1469–1479, 1971.

[9] J. Pistorius and M. Hutton, "Placement rent exponent calculation methods, temporaral behavior and FPGA architectural evaluation," in *SLIP Workshop*, Apr. 2003, pp. 31–38.

[10] W. Donath, "Placement and average interconnect lengths of computer logic," *IEEE Trans. on Circuits and Systems*, vol. 26, no. 4, pp. 272–277, 1979.

[11] D. Stroobandt, *A Priori Wire Length Estimates for Digital Design*. Kluwer Academic Publishers, 2001.

[12] S. Balachandran and D. Bhatia, "A priori wirelength estimation and interconnect estimation based on circuit characteristics," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 7, pp. 1054–1065, July 2005.

[13] H. Gao, Y. Yang, X. Ma, and G. Dong, "Analysis of the effect of LUT size on FPGA area and delay using theoretical derivations," in *Int'l Symposium on Quality Electronic Design*, Mar. 2005, pp. 370–374.

[14] P. Zarkesh-Ha, J. A. Davis, W. Loh, and J. D. Meindl, "Prediction of interconnect fan-out distribution using rent's rule," in *Proceedings of the 2000 international workshop on System-level interconnect prediction*. New York, NY, USA: ACM, 2000, pp. 107–112.

[15] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in FPGAs," in *Int'l Symposium on FPGAs*, Feb. 2002, pp. 59–66.