# An Analytical Model Relating FPGA Architecture to Logic Density and Depth

Joydip Das, Andrew Lam, Steven J. E. Wilton, *Senior Member, IEEE*, Philip H. W. Leong, *Senior Member, IEEE*, and Wayne Luk, *Fellow, IEEE*

***Abstract*—This paper presents an analytical model that relates FPGA architectural parameters to the logic size and depth of an FPGA implementation. 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, the number of used inputs per cluster, and the depth of the circuit after technology mapping and clustering. Comparison to experimental results shows that our model has good accuracy. We illustrate how the model can be used in FPGA architectural investigations to complement the experimental approach. The model's 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.**

***Index Terms*—Analytical modeling, critical path delay, early stage architecture evaluation, field-programmable gate array architectures, logic density.**

## I. Introduction

FIELD-PROGRAMMABLE gate arrays (FPGAs) have evolved considerably since their introduction. Originally used primarily for prototyping and small glue logic replacement, FPGAs are now used to implement entire systems containing memory, embedded processors, and other embedded functionality.

Much of the improvement in FPGA technology is a result of improvements in FPGA *architecture*. The architecture of an FPGA refers to the structure and interconnection of the configurable elements inside the device. In early FPGAs, for example, logic was implemented using 4-input lookup tables (LUTs), while in modern FPGAs, more complex fracturable LUTs which can be used in many different modes are employed [1]. These new architectures are designed to provide higher density, lower power consumption and/or faster circuit implementations. FPGA companies expend tremendous effort (and money) evaluating architectural enhancements for every generation of their devices.

Manuscript received March 09, 2010; revised June 09, 2010; accepted September 14, 2010. Date of publication October 25, 2010; date of current version October 28, 2011. This work was supported in part by Altera Corporation, in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada, and in part by the United Kingdom Engineering and Physical Sciences Research Council (EPSRC).

J. Das, A. Lam, S. J. E. Wilton are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC SV6T 1Z4, Canada (e-mail: dasj@ece.ubc.ca).

P. H. W. Leong is with the School of Electrical and Information Engineering, The University of Sydney, NSW 2006, Australia.

W. Luk is with the Department of Computing, Imperial College, London SW7 2RH, U.K.

Digital Object Identifier 10.1109/TVLSI.2010.2079339

During the design of a new FPGA, each architectural enhancement has to be evaluated to determine whether it should be incorporated in the new device. This evaluation is typically done using an experimental approach [2]. The new architecture is modelled (usually using software) and experimental computer-aided design (CAD) tools are used to map a set of benchmarks to the new architecture. Detailed area, delay, and power models are then used to evaluate the resulting implementation of each benchmark on the new architecture [2]–[4]. Based on the results, the architectural enhancement may be deemed worthwhile, in which case it may be incorporated into the device. Often, the results suggest modifications to the enhancement, and these are then evaluated using the same experimental techniques. This is often repeated numerous times until a suitable architecture is found. This process occurs both within FPGA companies and in academia.

The above experimental process can be slow. To properly exercise an architecture, many benchmark circuits are required. If the choice of benchmark circuits is insufficient, it is possible to create an architecture that is tuned for specific circuits rather than one suitable for a wide range of customers. In academia, researchers typically use roughly twenty benchmark circuits [2], but in industry, many more are employed. In the experimental approach, each of these circuits must be mapped to all potential variants of the architecture under investigation; each mapping can take several hours using modern CAD tools. This slow progress limits the number of alternative architectures that can be considered, and thus limits the ability of FPGA companies to explore new structures that may lead to more efficient FPGAs.

Architectural investigation can be accelerated using analytical models that describe some aspects of an architecture. Analytical models relate parameters describing an FPGA architecture to area, delay, or power efficiency. These usually take the form of simple expressions, and thus searching for efficient architectures can be fast and the need for time-consuming experiments is reduced.

Such models can be used to accelerate the architectural investigation process in two ways. First, understanding the relationships between architectural parameters enables *early-stage architecture development* [5] 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 are not needed 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 and logic depth of circuits implemented on the resulting FPGA*. The inputs of the model are the lookup-table (LUT) 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, (3) the expected number of inputs of each cluster that are used, (4) the expected number of LUTs along the critical path of a circuit implementation, and (5) the expected number of clusters along the critical path of a circuit implementation. 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 [5]. The final two outputs can be used as inputs to the delay models presented in [6] and [7]. Together with the channel width model and the delay model, 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 II and III, respectively. The model itself is described in Section IV, and the detailed derivation is presented in Section V. The model is validated against experimental results in Section VI. Section VII shows how our model can be used to evaluate the impacts of changing architectural parameters on area and speed. Section VIII gives examples of how the model can be used as a tool in FPGA architectural investigation.

An early version of the area model appears in [8] and an early version of the depth model appears in [9]. In this paper, we extend the discussion in both previous conference papers, and show how the model can be used together during architectural exploration to gain additional insight on the impact of the architectural parameters.

## II. Related Work

Several previous publications have examined the relationship between FPGA architectural parameters and consequent performance of FPGA implementations. Much of the work focuses on routing fabric. El Gamal derives a model for non-programmable chip that relates the area required for routing to the total number of pins in logic gates [10]. This model has been used in design of numerous generations of FPGAs [11]. Brown *et al.* relates various FPGA routing architecture parameters to the routability of that architecture [12], and more recently Fang and Rose relates the same architectural parameters to the channel width of an FPGA [5]. Pistorius and Hutton relates the Rent parameter (the Rent parameter is a measure of the complexity of the interconnect pattern in a circuit [13]) of a circuit to various architectural parameters [14].

There has also been much work related to interconnect and wirelength estimation. Much of this work is based on Rent's rule that relates the number of pins per module in a circuit to the number of blocks in a module [13]. Among the work for non-programmable chips, early work by Donath [15] and others relates the area requirements of routing wires to the Rent parameter of a circuit. Later work by Stroobandt refines these models to consider more realistic network topologies and architectural assumptions [16]. More recent work by Balachandran and Bhatia uses circuit information to estimate interconnect and wirelength for island-style FPGAs [17]. Smith *et al.* presents a model to estimate post-placement wirelength for both homogeneous and hetergenous FPGA architectures [18]. There has also been work providing early stage delay values for FPGAs by Manohararajah [19], which uses a lookup table with pre-recorded values of interconnect delays as a function of architecture parameters.

The previous work closest to ours is by Gao *et al.*, who relates LUT size to area as well as depth of forming $N$-LUTs for a non-clustered FPGA [20]. We present a more complete model that considers cluster-based architectures (which are more representative of real FPGAs), and we model a wider range of architectural parameters.

## III. Framework

In this section, we first describe the assumptions in regards with the architectural framework as well as the circuits that are being implemented. We then present the parameters that we use in our model.

### A. Assumptions and Guiding Principles

We consider homogeneous clustered architectures, where a logic cluster, also known as a configurable logic block (CLB), is formed by $N$ elements of $K$-input LUTs. Previous work has demonstrated that the models for homogeneous architectures can be extended to heterogenous architectures [18].

Three principles guide us in the development of our model.

First, we endeavor 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.

Second, we wish to derive a model that is as independent of the circuit (to be implemented on FPGA) as possible. For example, we would prefer a relation between LUT-size and post-techmapping depth, which is independent of a given circuit. This makes our paper different from prior *estimation* work, in which the goal is to predict the area, speed, or power for a given circuit [17]. That being said, it is impossible to completely ignore the impact of specific circuits; hence we describe our circuit by using three parameters, (1) Rent parameter, (2) size of the un-techmapped circuit and (3) depth of the un-techmapped circuit. All of these three parameters are available during *early stage evaluation*.

Third, we attempt to balance complexity with accuracy. The simple equations of our model will provide more insight into architectural trade-offs than unnecessarily complex expressions. Such insights will help designers to effectively fine-tune an architecture under evaluation.

### B. Model Parameters

Table I lists the parameters used to describe the architecture, circuit and implementation. In general, upper-case letters represent architectural parameters, and lower-case letters represent the circuit parameters as well as the parameters, which describe

TABLE I
MODEL PARAMETERS

| | |
|---|---|
| Model Inputs: | |
| 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 |
| $d_2$ | Maximum depth of 2-LUT netlist of a given circuit |
| Other Input Parameters: | |
| $\gamma$ | Average number of inputs *not* used in each LUT |
| Model Outputs: | |
| 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 in each cluster ($c = n_k/n_c$) |
| $i$ | Average number of inputs used in each cluster |
| $o$ | Average number of outputs used in each cluster |
| $f_{avg}$ | Average fanout of all nets in the circuit |
| $d_k$ | Maximum post-techmapping depth for a circuit |
| $d_c$ | Maximum post-clustering depth for a circuit |

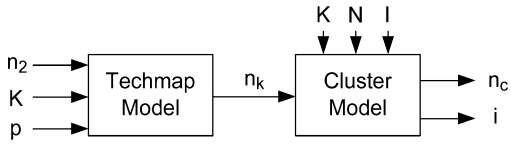



Fig. 1. Model for amount of logic in each LUT and cluster and used inputs in each cluster.

the implementation of the circuit on a given architecture. Except for $\gamma$, all implementation parameters are derived analytically. Since our model describes the pre-routing implementation parameters, we do not include the architecture parameters that describe the detailed routing fabric.

## IV. MODEL OVERVIEW

Our model consists of two parts. The first part, which we call the *area model*, relates the number of logic elements and clusters required to implement a circuit on an FPGA to parameters that describe the architecture of the logic blocks and clusters. The second part, which we call the *depth model*, relates the depth of a circuit implemented on the FPGA to the same architectural parameters. This section gives an overview of both parts; Section V will provide the detailed derivation.

Our derivation mirrors the CAD flow used in mapping circuits to FPGAs. Each part of the model consists of two *phases*. The first phase mirrors the process of technology mapping [21], and the second phase mirrors the process of clustering [2], [22]. As we shall describe below, each phase consists of one or two closed-form equations.

### A. Area Model

Fig. 1 shows an overview of the area model. The inputs to the model are the architectural parameters $K$, which represents the number of inputs to each lookup table, $N$, which is the number of lookup tables per cluster, and $I$, the number of unique inputs per cluster, as well as the circuit parameters $n_2$ and $p$ which represent the number of two-input gates in the circuit and the Rent parameter of the circuit, respectively. The outputs of the area model are the number of $K$-LUTs required to implement the circuit, $n_k$, the number of clusters required to implement the circuit, $n_c$, and the average number of inputs to each cluster that are used.

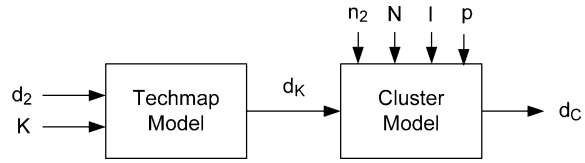



Fig. 2. Model for depth estimation.

*1) Technology Mapping Phase:* Consider the implementation of a circuit consisting of $n_2$ two-input gates. During technology mapping, these $n_2$ gates will be mapped into a $n_k$ of $K$-input LUTs, where $n_k \leq n_2$. We expect that the ratio $n_2/n_k$ will be higher for the larger value of $K$. Intuitively, the ratio will also depend on the Rent parameter of the circuit, $p$. The first phase of the area model is a closed form expression for $n_k$ as a function of $K$, $n_2$, and $p$.

*2) Clustering Phase:* After technology mapping, the set of $n_k$ LUTs are packed into $n_c$ clusters, also called Configurable Logic Blocks (CLBs) or Logic Array Blocks (LABs). Each cluster can contain up to $N$ LUTs, and can have up to $I$ distinct inputs. Clearly, the ratio $n_k/n_c$ will be higher for higher values of $N$ and $I$; the second part of the model provides a closed-form expression for this relation.

In addition, the second part of the model contains an expression that relates $i$, the average number of *used* inputs to each cluster. Clearly, in all cases, $i \leq I$. The reason for providing this relation is that it has been shown that the routing requirements of an FPGA are directly related to $i$ [5]. In addition, this quantity will be required as an input to the Fang model [5] in Section VIII.

In the detailed derivation of Section V, we consider two types of architectures. In architectures with a low value of $N$ and a high value of $I$, all $N$ slots within each cluster can typically be filled. In this type of architecture, referred to as an $N$-limited architecture, the ratio of $n_k/n_c$ is simply $N$. However, in such an architecture, it is likely that the number of used inputs is less than $I$. On the other hand, in an architecture with a lower value of $I$ and a higher value of $N$, it is likely that not all $N$ slots in a cluster can typically be filled, due to limitations on the number of inputs to each cluster. We call such an architecture an $I$-limited architecture. Since we wish our model to be flexible, in the next section we derive expressions for $n_c$ and $i$ for both $N$- and $I$-limited architectures.

### B. Depth Model

Fig. 2 shows an overview of the depth model. The inputs to the model are the same architectural parameters used in the area model, as well as the number of two-input gates along the critical path of the circuit $d_2$. The outputs are the number of $K$-LUTs along the critical path of the implemented circuit $d_k$ and the number of clusters along the critical path $d_c$.

*1) Technology Mapping Phase:* Consider the implementation of a circuit with a critical path depth of $d_2$. During technology mapping, gates are mapped to LUTs; the depth of the circuit in LUTs will be $d_k$ where $d_k \leq d_2$. As with the area model, we expect that the ratio $d_2/d_k$ will be higher for the

larger value of $K$. The first phase of the depth model is a closed form expression for $d_k$ as a function of $K$ and $d_2$.

*2) Clustering Phase:* When LUTs are packed into clusters, some connections will be encapsulated into clusters. This gives rise to a distinction between two types of connections: *intra-cluster* connections which connect LUTs only within a single cluster, and *inter-cluster* connections, which connect LUTs in more than one cluster. We denote the number of clusters along the critical path as $d_c$, where $d_c \leq d_k$. The ratio $d_k/d_c$ clearly depends on the architectural parameters $N$ and $I$ (larger clusters means fewer inter-cluster nets). We will show that it also depends on the size of the circuit $n_2$. The second phase of the delay model presents equations that describe this relation. Again, we consider both $I$- and $N$-limited architectures.

## V. Model Derivation

### A. Area Equations

*1) Number of $K$-LUTs to Implement a Circuit, $n_k$:* We will start with an un-techmapped circuit consisting of $n_2$ two-input gates. Consider a portion of the $un$-techmapped circuit consisting of $x$ two-input gates, where $1 < x \leq n_2$. 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 [13] to write

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

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 pins used in each $K$-LUT is $K + 1 - \gamma$, where the first two terms represent input and output parameters respectively and $\gamma$ is described later in this subsection. Since the number of signals that connect across the boundary of this region is still $y$, Rent's Rule gives us:

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

Since we are considering the same region in (1) and (2), $y$ is same for these two equations, we get

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

Intuitively, the ratio $z/x$ is a measure of how much logic can be mapped into each LUT.

Since every $z$ number of $K$-LUTs can implement $x$ number of 2-input gates in the original netlist of the circuit, using (3), we can write

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

During the techmapping process, all $K$ inputs are not always used in a $K$-LUT. The term $\gamma$ in (2) to (4) represent the expected number of inputs to a LUT that are *not* used. However, we have found that the experimental results of $\gamma$ as a function of $K$ is extremely consistent across all benchmark circuits that we considered, and that there is a linear relationship between $K$ and $\gamma$ ($\gamma = (1/4)K - (1/2)$). This relation is based on experimental results over a set of benchmark circuits, and we are yet to find an analytical closed form expression for $\gamma$. The derivation of a closed form for $\gamma$ is an interesting topic of future work.

*2) Number of Clusters Needed to Implement a Circuit, $n_c$:* As explained in Section IV-A, in predicting the implementation parameters, $n_c$ and $i$, we present separate equations for $N$-limited clustering and $I$-limited clustering.

*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 input-output pins on each cluster. In this case, the average number of LUTs packed into each cluster $c$ ($c = n_k/n_c$), will be smaller than the capacity of the cluster $N$. To estimate $c$, and hence $n_c$, we employ Rent's Rule [13] as follows.

Consider the same region of the technology-mapped circuit from (2), which contains $z$ $K$-LUTs and has $y$ signals that cross the region. When this region is mapped to clusters, we can write

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

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 $o = i/f_{\text{avg}}$, where $f_{\text{avg}}$ is the average fanout of the circuit and will be computed below. Using this expression for $o$, we can write

$$y = i \cdot \left(1 + \frac{1}{f_{\text{avg}}}\right) \cdot v^p. \tag{6}$$

We consider a large region to derive the equations and the Rent parameter of a circuit is assumed to be constant with respect to the changes in architectural parameters. We can make this assumption if the changes in the Rent parameters contribute negligibly to the values of the implementation parameters. In Section VI, we will demonstrate that this is indeed the case for the MCNC as well as the large benchmark circuits.

Eliminating $y$ from (2) and (6) gives us:

$$i \cdot \left(1 + \frac{1}{f_{\text{avg}}}\right) \cdot v^p = (K + 1 - \gamma) \cdot z^p \tag{7}$$

where $v$ and $z$ are respectively the number of clusters and the number of $K$-LUTs required for the region that we consider. Solving for the total number of clusters required for the circuit, $n_c$ gives us

$$n_c = n_k \cdot \sqrt[p]{\frac{K + 1 - \gamma}{I \cdot \left(1 + \frac{1}{f_{\text{avg}}}\right)}} \tag{8}$$

and

$$c = \sqrt[p]{\frac{I \cdot \left(1 + \frac{1}{f_{\text{avg}}}\right)}{K + 1 - \gamma}}. \tag{9}$$

Since we expect all cluster input pins to be used in an $I$-limited architecture, we approximate the average used inputs per cluster, $i$ as the available input pins per cluster, $I$ in (8) and (9). The average fanout, $f_{\text{avg}}$ in (8) and (9) can be calculated using a formula from [23]

$$f_{\text{avg}} = \frac{1 - (f_{\max} + 1)^{(p-1)}}{1 - (f_{\max} + 1)^{(p-2)} - \phi(p, f_{\max})} - 1 \tag{10}$$

where $f_{\max}$ is the maximum fan-out of the circuit, implemented on FPGA, described as

$$f_{\max} = \left[(I+N)\cdot\frac{n_k}{N}\cdot(1-p)\right]^{(1/(3-p))} \quad (11)$$

and,

$$\phi(p, f_{\max}) = \sum_{n=1}^{f_{\max}} \frac{n^p}{n^2\cdot(n+1)}. \quad (12)$$

In (11), we approximated the number of outputs as $N$ and the number of clusters as $n_k/N$. Experimentally, we find that $f_{\text{avg}}$ is only a weak function of $f_{\max}$, and these approximations lead to only a small error.

*$N$-limited clustering*. This case is trivial. Since unique input pins to clusters are plentiful, clusters can be filled to capacity. Hence, $c = N$ and we can write,

$$n_c = \frac{n_k}{N}. \quad (13)$$

*3) Average Number of Used Inputs, $i$:* For the average number of used inputs, we again present separate equations for $N$-limited and $I$-limited architectures.

*$I$-limited clustering*. In these architectures, we would expect all cluster input pins to be used. Thus, we can write:

$$i = I. \quad (14)$$

*$N$-limited clustering*. A cluster with $N$ number of $K$-LUTs has $(K+1-\gamma)$ I/O pins and $i+o$ exterior pins. By applying Rent's Rule [13] to a single cluster, we then have

$$i + o = (K+1-\gamma)\cdot N^p. \quad (15)$$

Substituting $o$ by $o = i/f_{\text{avg}}$, we obtain the expected number of average used inputs $i$ for $N$-limited architecture

$$i = \frac{(K+1-\gamma)\cdot N^p}{1+\frac{1}{f_{\text{avg}}}}. \quad (16)$$

*4) Boundary Condition for $N$- and $I$-Limited Architectures:* For the $I$-limited architecture and consequent clustering, $c < N$, where $c$ is the expected number of LUTs packed into each cluster ($c = n_k/n_c$). For the $N$-limited architecture and consequent clustering, $c = N$. Using (9), we can write the following condition for the $I$-limited case

$$\sqrt[p]{\frac{I\left(1+\frac{1}{f_{\text{avg}}}\right)}{K+1-\gamma}} < N. \quad (17)$$

This can be rearranged to produce

$$I < N^p\cdot\frac{K+1-\gamma}{1+\frac{1}{f_{\text{avg}}}}. \quad (18)$$

Clustering is $I$-limited if Inequality (18) holds.

### B. Delay Equations

*1) Post Technology Mapping Depth:* In this section, we describe a relation between the LUT size $K$, and the expected depth of a circuit after technology mapping. The inputs to this part of the model are the LUT size $K$ and the depth of the un-techmapped circuit $d_2$. The output of this part is the depth of the circuit after it is mapped into $K$-input LUTs. We represent this depth parameter by $d_k$.

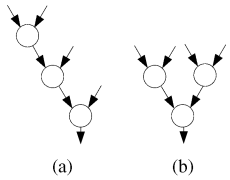



Fig. 3. Two possible mappings for $K = 4$.

Consider the portion of the original circuit covered by a single LUT during technology mapping. Most technology mappers attempt to minimize the depth of the resulting implementation. However, the actual pattern of nodes covered by a single LUT depends on the structure of the original netlist of the circuit. For mapping two-input nodes of a small example into a 4-input LUT, Fig. 3 shows two possible scenarios. The depth after techmapping for these two scenarios will be 3 and 2 respectively. For a $K$ input LUT, such extremes can be generalized as $K-1$ and $\log_2(K)$. For a large netlist, we would expect the "average" depth to be somewhere between these two extremes.

From Section V-A, typically not all $K$ inputs to a $K$-input LUT are actually used and the expected number of *not*-used inputs in a $K$-LUT is represented by the parameter $\gamma$. Incorporating $\gamma$, depth values for the two possible extreme techmapping solutions can be defined by $K-1-\gamma$ and $\log_2(K-\gamma)$. We assume that the average of these two extrema can capture the reduction of depth from $d_2$ to $d_k$, which gives us the depth of the tehnology mapped netlist $d_k$ as:

$$d_k = \frac{2\cdot d_2}{K-1-\gamma+\log_2(K-\gamma)} \quad (19)$$

In Section VI, we will show that this simple expression matches the experimental results well.

*2) Post Clustering Depth:* Logic elements (LEs) are usually grouped into tightly connected *clusters*. Connections within a cluster are fast, while connections between clusters are realitvely slow. In this subsection, we derive a relation between the FPGA cluster architecture and the depth of the circuits after they are mapped to clusters.

We derive this relation in two steps. First, we derive the expected proportion of all connections in a circuit that are made local after clustering and denote it as $s_{ckt}$. Intuitively, as cluster size is increased, more connections can be made local and hence $s_{ckt}$ is also increased. Second, we determine the expected proportion of connections *along the critical path* that are made local after clustering, which we will denote by $s_{cp}$. These two steps allow us to compute the expected number of inter-cluster and intra-cluster connections along the critical path of a given circuit.

Note that each connection in a circuit corresponds to one sink in a multi-sink net, and represents one input to an LE. Thus, in this paper, we count connections by counting the number of input pins of an LE, and *not* the output pins. An LE with $K-\gamma$

used inputs and one used output contributes $K-\gamma$ connections to the total connection count.

*Proportion of connections made local.* Most clustering algorithms operate incrementally; that is, they choose a seed and iteratively add related LEs until the cluster is full [2]. Each time an LE is added to the cluster, additional connections are typically made local. These local connections can be one of two types: (1) those that are made local due to the optimization algorithm, and (2) those that are made local "by chance". We will explain and consider each of these separately.

Consider a cluster consisting of a single LE with $K-\gamma$ used inputs. In such a cluster, the only way a net can be made local (becomes completely absorbed by the cluster) is if the output of the LE feeds directly back to one of its own inputs. Experimentally we have observed that this rarely happens, so we can approximate the number of local connections in this case as 0. Now consider adding additional LEs to the cluster. A timing-driven cluster algorithm would attempt to pack as many LEs along the critical path into a cluster as possible. This often leads to packings as shown in Fig. 5, in which each LE receives an input from an LE that is already in the cluster. We will again use the notation $c$ here to represent the average number of LEs (LUTs) in a cluster $n_k/n_c$. Following this construction, if there are $c$ LEs in the cluster, then the cluster has a total of $c(K-\gamma)$ connections, of which $c-1$ are local. Thus, we have the number of connections made local by design $n_{ckt_d}$ as

$$n_{ckt_d} = c - 1. \tag{20}$$

Of the remaining $c(K-\gamma)-(c-1)$ connections in the cluster, some will be global and some will be local. We assume that, apart from the $c-1$ connections described above, each connection in the implementation is equally likely to be local. If there are $n_k$ logic elements in the circuit, and if $c$ of these are packed into each cluster, the chances that the logic elements for each connection is within the same cluster is $c/n_k$. This construction gives us the total number of *additional* connections made local as

$$n_{ckt_c} = \frac{c}{n_k}[c(K-\gamma) - c + 1]. \tag{21}$$

Combining (20) and (21) and simplifying leads to an expected number of local connections $n_{ckt}$ as

$$n_{ckt} = (c-1) + \frac{c}{n_k}[c(K-\gamma) - c + 1]. \tag{22}$$

and since there are $c(K-\gamma)$ total connections in each cluster, the expected proportion of the connections made local after clustering $s_{ckt}$ can be expressed by:

$$s_{ckt} = \frac{(c-1) + \frac{c}{n_k}[c(K-\gamma) - c + 1]}{c(K-\gamma)} \tag{23}$$

where $c$ can be written as a function of the architectural parameters $N$ and $I$ and the circuit Rent parameter $p$ using the following results from Section V-A

$$c = \begin{cases} N & \text{if } I \geq N^p \frac{K+1-\gamma}{1+\frac{1}{f_{\text{avg}}}} \\ \sqrt[p]{\frac{I\left(1+\frac{1}{f_{\text{avg}}}\right)}{K+1-\gamma}} & \text{if } I < N^p \frac{K+1-\gamma}{1+\frac{1}{f_{\text{avg}}}} \end{cases} \tag{24}$$

where the average fanout $f_{\text{avg}}$ is given by (10).

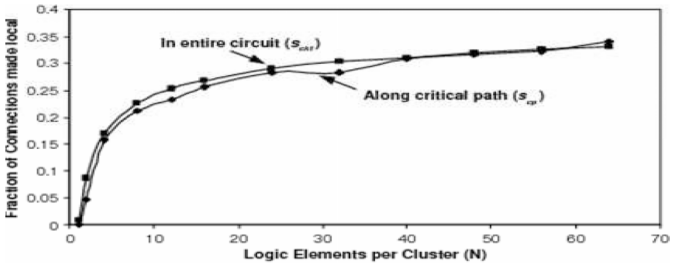



Fig. 4. Comparison of $s_{ckt}$ and $s_{\text{cp}}$.

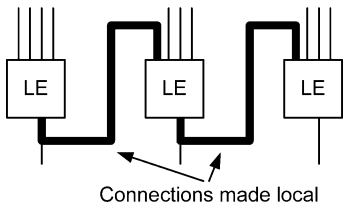



Fig. 5. Cluster with three lookup-tables.

*Connections Along the Critical Path.* The previous subsection computed the expected number of connections that are made local in a circuit. In this section, we seek $s_{\text{cp}}$, which is the expected number of connections *along the critical path* that are made local in the same circuit. Intuitively, a good packer will attempt to make more paths along the critical path local, compared to other paths, so we would expect $s_{\text{cp}} > s_{ckt}$.

We investigated this relation experimentally using two clustering tools: T-VPACK [2] and a replica of iRAC [22]. As shown in Fig. 4 (which was obtained using T-VPACK), the values of $s_{\text{cp}}$ and $s_{ckt}$ are roughly the same for all values of $N$. The results from iRAC were similar. Based on these results, our model assumes $s_{\text{cp}} = s_{ckt}$.

The results of Fig. 4 may appear counter-intuitive. We would expect the clustering algorithm to give preference to paths that are critical. However, as packing proceeds, the criticality of paths are changed. Even if the criticality of a net is recalculated frequently, the problem of optimizing the wrong path in early stages of clustering will still exist. This suggests that T-VPACK and iRAC are *not* optimizing the critical path well and are optimizing all paths roughly equally.

This suggests an interesting topic of future work: to find a better way to predict, ahead of time, which paths are actually going to be critical. The packing stage can then pack the logic blocks along the critical path more efficiently. In such cases, $s_{\text{cp}}$ is expected to be higher than $s_{ckt}$. Modeling the ratio of $s_{\text{cp}}$ and $s_{ckt}$ for such packing tools may be an interesting area of future research. Once such model is available, it can be readily plugged into our proposed model for $d_c$.

*Overall model for post-clustering depth.* To summarize, the number of clusters on the critical path and hence the post-clustering inter-cluster depth, $d_c$ can be written as

$$\begin{aligned} d_c &= d_k \cdot (1 - s_{\text{cp}}) \\ &= d_k \cdot \left[1 - \frac{(c-1) + \frac{c}{n_k}[c(K-\gamma) - c + 1]}{c(K-\gamma)}\right] \end{aligned} \tag{25}$$

where $c$ is given by (24).

TABLE II
MCNC BENCHMARK CIRCUITS

| Circuit Name | $n_2$ | $d_2$ | Inputs | Outputs | Rent Parameter |
|---|---|---|---|---|---|
| ex5p | 1779 | 15 | 8 | 63 | 0.738 |
| misex3 | 2557 | 13 | 14 | 14 | 0.714 |
| apex4 | 2196 | 12 | 9 | 19 | 0.738 |
| alu4 | 2732 | 14 | 14 | 8 | 0.662 |
| tseng | 1861 | 43 | 52 | 122 | 0.524 |
| seq | 2939 | 14 | 41 | 35 | 0.721 |
| apex2 | 3165 | 17 | 39 | 3 | 0.743 |
| diffeq | 2556 | 39 | 64 | 39 | 0.554 |
| dsip | 2531 | 10 | 229 | 197 | 0.527 |
| des | 2901 | 14 | 252 | 243 | 0.646 |
| s298 | 4272 | 32 | 4 | 6 | 0.560 |
| bigkey | 2979 | 10 | 263 | 197 | 0.517 |
| spla | 7438 | 19 | 16 | 46 | 0.726 |
| frisc | 6023 | 67 | 20 | 116 | 0.644 |
| elliptic | 5474 | 52 | 131 | 114 | 0.593 |
| pdc | 8408 | 19 | 16 | 40 | 0.748 |
| ex1010 | 8020 | 17 | 10 | 10 | 0.749 |
| s38584.1 | 12491 | 25 | 39 | 304 | 0.632 |
| s38417 | 13656 | 25 | 29 | 105 | 0.591 |
| clma | 14253 | 40 | 383 | 82 | 0.726 |

TABLE III
QUIP BENCHMARK CIRCUITS

| Circuit Name | $n_2$ | $d_2$ | In/Out | Rent |
|---|---|---|---|---|
| oc_aes_core_inv | 25724 | 33 | 260/129 | 0.670 |
| oc_aes_core | 18178 | 25 | 259/129 | 0.673 |
| oc_des_des3perf | 78872 | 16 | 234/64 | 0.634 |
| oc_video_compression_systems_ jpeg_syn | 78245 | 65 | 20/27 | 0.614 |

TABLE IV
MCNC BENCHMARK CIRCUITS, USED TO MEASURE THE VALUES OF $\gamma$ FOR MODEL VALIDATION

| Circuit Name | $n_2$ | Inputs | Outputs |
|---|---|---|---|
| C6288 | 1820 | 32 | 32 |
| C7552 | 1781 | 207 | 107 |
| i10 | 1668 | 257 | 224 |
| apex3 | 1452 | 54 | 50 |
| parker1986 | 1137 | 48 | 8 |

Within each cluster, the critical path is expected to pass through $d_k/d_c$ lookup tables. If we have estimates of the intra-cluster delay $t_{\text{intra}}$ and the inter-cluster delay $t_{\text{inter}}$, then the total critical path delay can be estimated as

$$d_c \left[ t_{\text{inter}} + \frac{d_k}{d_c} t_{\text{intra}} \right] = d_c \cdot t_{\text{inter}} + d_k \cdot t_{\text{intra}}. \tag{26}$$

## VI. MODEL VALIDATION

To evaluate the accuracy of different components of our analytical model, we compare the model predictions to the experimental results, obtained using CAD tools. The first two subsections present the evaluation results for area equations and delay equations separately for twenty large MCNC [24] benchmark circuits, listed in Table II. The third subsection examines the effects of the Rent parameter on the validation results. The last subsection presents the validation results for four large QUIP [25] benchmark circuits, listed in Table III.

To validate our model, we need the values of unused LUT inputs $\gamma$, a closed form for which is not yet available. For validation, the set of $\gamma$ values have been measured using five MCNC benchmark circuits that are different from the evaluation circuits, listed in Table II. These five MCNC benchmark circuits and the corresponding values of $\gamma$ are respectively listed in Table IV and V.

TABLE V
$\gamma$ VALUES FROM FIVE MCNC BENCHMARKS

| $K$ | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| $\gamma$ | 0.000 | 0.279 | 0.427 | 0.898 | 1.278 | 1.648 |

### A. Validation of Area Equations

Fig. 6(a) through (d) illustrate the accuracy of our model's area equations. The experimental results are obtained by averaging the results for twenty large MCNC benchmark circuits. For some data points, we have included the maximum and minimum values that we obtain from experimental results. The values enclosed by the brackets ( ) are from one of the experimental flows (Flowmap or T-VPack) and the values enclosed by the brackets [ ] are from the other experimental flow (EMAP or iRAC).

We measure the values of $n_2$ and $d_2$ by running T-VPACK on two-input LUTs representations of the corresponding circuits. We incorporate the depth calculation module into the original version of T-VPACK to measure $d_2$.

Fig. 6(a) illustrates the accuracy of our model in predicting the number of $K$-LUTs required to implement a given circuit. Results for two different technology mapping algorithms are presented. The analytical results are obtained from (4) using the Rent parameter for each benchmark circuit. We use an inhouse tool *bcgen* to measure Rent parameters. This tool uses recursive bipartitioning [14] method to calculate the Rent parameters. The graph shows that the analytical results track the experimental ones very closely.

Fig. 6(b) illustrates the accuracy of the analytical model in predicting the ratio $n_2/n_c$ as a function of cluster size $N$. In all cases, we set $K$ = 4. $I$ is set to be $0.88N$ + 3.2 from [5] to ensure that our clustering is $I$-limited, which is the interesting case for this graph. The analytical results are obtained using (4) and (8), while the experimental results are obtained using two separate clustering algorithms, T-VPACK and our implementation of iRAC. Again, the analytical results are very consistent with both sets of experimental results. We observe similar consistencies for other LUT sizes.

Figure 6(c) shows the same ratio as a function of the number of input pins per cluster $I$. In all cases, we use $K = 4$ and $N = 20$. The boundary between $I$-limited and $N$-limited clustering is also shown. This boundary is determined by using (18). The graph shows that our model tracks the experimental results well in both regions.

Finally, Fig. 6(d) shows the average number of used inputs per cluster $i$ as a function of $N$. We set the value of $I$ to be $(K/2) \cdot (N + 1)$, making it an $N$-limited architecture. Since iRAC explicitly tries to minimize the use of cluster pins [22], our model matches the iRAC results more closely.

### B. Validation of Delay Equations

To evaluate the accuracy of our model in predicting depth equations, we again compare the model predictions to the results that we obtain from the academic CAD tools. Again, we use the twenty large MCNC benchmark circuits, listed in Table II [24]. Although the value of $N$ is typically between 4 and 16 in current generation FPGAs, we validate our depth model for much higher values of $N$ to examine its applicability in future generations of FPGAs.

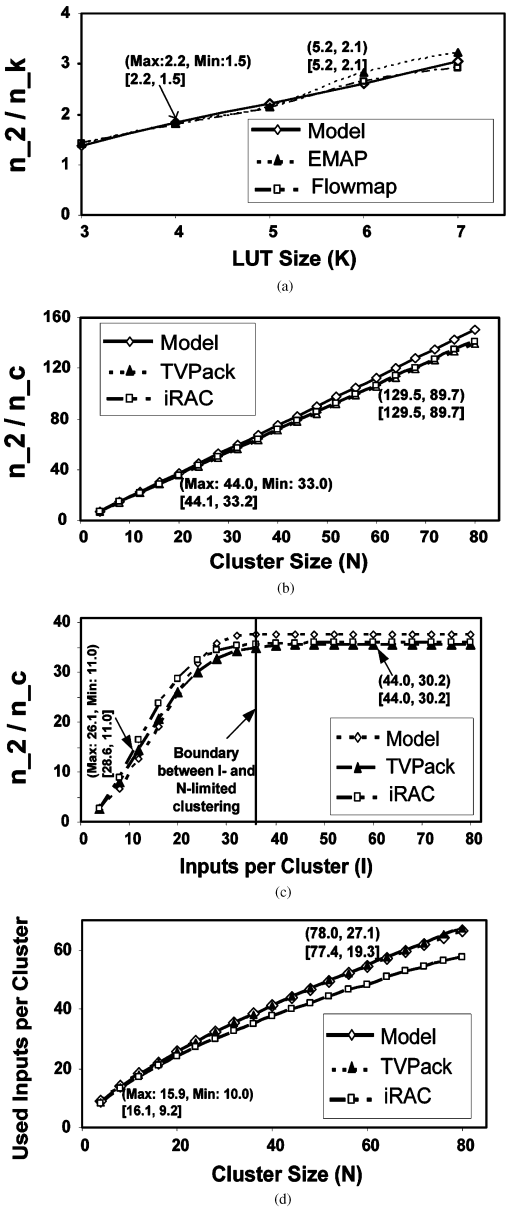



Fig. 6. Model validation for area equations. (a) Logic per LUT versus LUT size. (b) Logic packed per cluster versus cluster size. (c) Logic packed per cluster versus cluster inputs. (d) Used inputs per cluster versus cluster size.

First we discuss $d_k$. Measured results for $d_k$ are gathered by recording the maximum depth for the benchmark circuits after they are technology mapped using Flow-Map [21]. Analytical results are obtained by using the measured $d_2$ and (19). The $d_2$ values are measured from the 2-input netlist of the benchmark circuits.

Fig. 7 shows a plot of the measured versus estimated depth for each of the twenty benchmark circuits. We have shown two representative data sets, one for $K = 4$ and one for $K = 6$. The solid line with unit slope in Fig. 7 represents the points where the predicted $d_k$ values are equal to the measured $d_k$ values. Since data for some of the benchmark circuits overlap with each other, data points for all twenty benchmark circuits (for each value of $K$) are not visible in this graph. Due to close proximity of data values, some of the benchmarks overlap with each other both for $K = 4$ and $K = 6$. We fit lines to the data-points

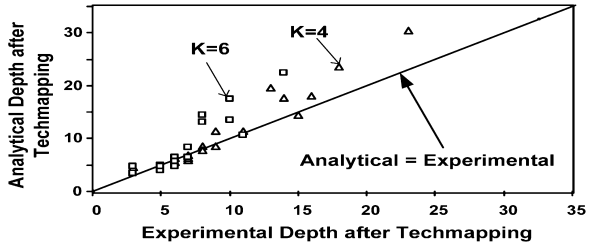



Fig. 7. Model verification for $d_k$ (circuit by circuit).

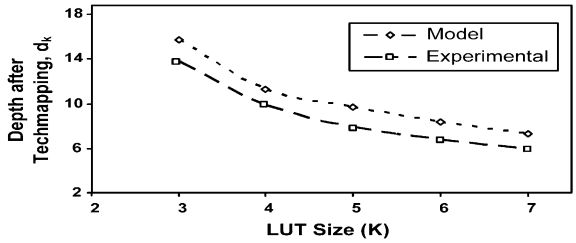



Fig. 8. Model verification for $d_k$ (For different LUT sizes).

TABLE VI
STANDARD DEVIATION OF ESTIMATING $d_k$ FOR 20 CIRCUITS

| LUT-Size: | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|
| Std. Dev.: | 3.45 | 2.15 | 2.87 | 2.57 | 2.35 |

TABLE VII
ABSOLUTE AND PERCENTAGE ABSOLUTE DIFFERENCE BETWEEN EXPERIMENTAL AND ESTIMATED VALUES OF $d_k$

| LUT-Size: | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|
| Absolute Diff.: | 2.39 | 1.83 | 2.21 | 2.06 | 2.15 |
| % Absolute Diff.: | 15.24 | 16.25 | 22.81 | 24.68 | 29.33 |

for $K = 4$ and $K = 6$. The slope for these two lines are 1.4 and 1.6 respectively with $r^2$ value of 0.94 and 0.83 respectively, where $r$ is the correlation coefficient. This graph shows that the prediction loses some accuracy for higher values of depth. Fig. 8 plots the maximum post-techmapping depth for different LUT sizes. Each point in this graph represents the arithmetic mean of the depth values across the benchmark suite. As these two graphs show, the analytical results track the experimental results very closely.

We also examine the standard deviation of the differences between experimental and estimated values for different values of $K$, results for which are presented in Table VI. The absolute error between the experimental and estimated values (averaged for 20 circuits) are presented in Table VII.

Fig. 9 illustrates the accuracy of our model in estimating $s_{ckt}$ for three representative values of $K$. As before, we collected experimental results using two packing tools, T-VPACK [2] and our implementation of iRAC [22]. As the graphs show, our model captures the experimental trends of both. However, for small clusters, our model overestimates the local connections, while for large clusters, our model underestimates. All of the cases in Fig. 9 are for $N$-limited clustering, where $I = (K/2) \cdot (N + 1)$.

The discrepancies in Fig. 9 can be partially explained as follows. First, consider a small cluster with $N = 2$, $K = 4$ and $I = 6$. As shown in Fig. 5, our model assumes that the clustering algorithm will always find a second LE that can use the output

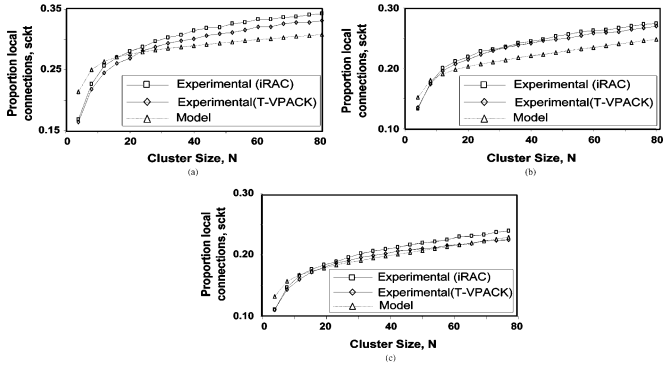



Fig. 9. Verification of equation for $s_{ckt}$. (a) K $= 4$, (b) K $= 6$, (c) K $= 7$.

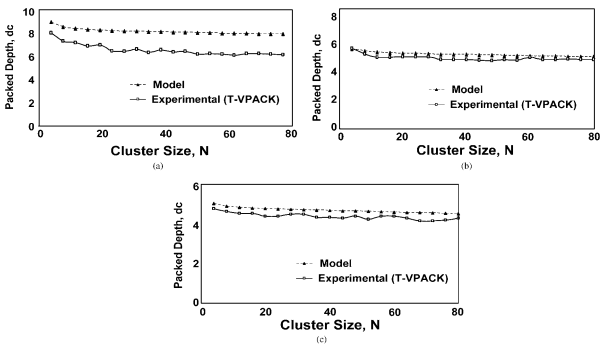



Fig. 10. Verification of equation for $d_c$. (a) K $= 4$. (b) K $= 6$. (c) K $= 7$.

of the first LE. If the clustering algorithm chooses an LE with four inputs as the seed, the second LE will use the output from the seed and at most two more unique inputs. It seems likely that, often, the clustering algorithm will be unable to find such an LE, so it would instead choose an LE that shared the appropriate number of inputs, but not the output from the first LE. Our model then overestimates the local connections.

For large clusters, the situation is different. In such cases, it is possible that LEs may receive more than one input from a local LE (so adding a LE creates more than one new local connection). As a result, the number of connections made local by design will be more than $c-1$ that is assumed in (20) and the experimental results will be higher than the predicted values.

In all cases, however, the slopes for the results from our model are comparable to those for the experimental results, especially for higher values of $K$. An interesting observation is that for a LUT size of 7, results from T-VPACK almost coincide with the results from our model. It makes us believe that at this LUT-size, after making connections local *by design*, T-VPACK relies on random absorption of connections for the remaining connections.

Finally, Fig. 10 compares our model for post-clustering depth $d_c$ to experimental results obtained using T-VPACK. Again, our results track the experimental values well.

### C. Effects of Rent Parameter

The validation of area equations approximates the Rent parameter for a circuit to remain the same for implementations with different architectural parameters, such as $N$. We have examined the effect of this assumption by using two sets of Rent parameters for each circuit:

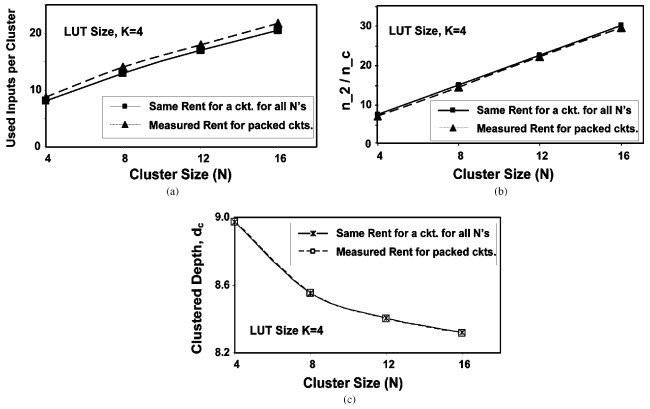



Fig. 11. Effects of rent parameter on area and delay equations. (a) Used inputs per cluster versus cluster size. (b) Logic packed per cluster versus cluster size. (c) Post-clustering depth versus cluster size.

1) In the first case, for each circuit, we have used a fixed value of Rent parameter $p$. We do not change this value with the changes in architectural parameters, such as $N, K$ and $I$. We have listed these values of $p$ for the MCNC and the QUIP benchmark circuits in Tables II and III respectively. This set of Rent parameters have also been used to validate our model outputs.
2) The second case measures the Rent parameters for the packed circuits using our inhouse tool $bcgen$ that uses recursive bipartitioning method. These measured values of $p$ may change with the architectural parameters.

The differences between the maximum and the minimum values of the Rent parameters for the latter range between 3% to 15% for all circuits except for s298, dsip and bigkey. We generate two sets of model outputs by using the above two sets of Rent parameter separately. Fig. 11 shows the corresponding results. Fig. 11(a) and (b) illustrates that the effects of Rent parameter on the implementation parameters for area equations can be assumed to be negligible for early stage architecture evaluation. Our delay equations depend on the Rent parameter through the area parameters, $n_c$ and $i$. Consequently, for the estimated depth values, we will have negligible effects for the changes in Rent parameter, as illustrated in Fig. 11(c).

### D. Validation for Quip Benchmark Circuits

We further validate our model using four additional benchmark circuits that are much larger than the MCNC benchmark circuits. The QUIP benchmark circuits are a number of VHDL and Verilog benchmark circuits, provided by Altera Corporation [25]. Details of the QUIP benchmark circuits can be found in the *quip_benchmark.pdf* document, available with the QUIP suite [25]. Table III lists the four circuits that we use, which are the larger circuits in the QUIP suite. We also want to use another large QUIP benchmark circuit $uoft\_raytracer$, but could not synthesize it to flip-flops and LUTs.

Fig. 12 illustrates the accuracy of our area model for QUIP benchmarks. The experimental results represent the average of the results for the QUIP circuits. Figs. 13 and 14 illustrate the accuracy of our depth model. Figs. 12, 13 and 14 respectively correspond to Fig. 6, 8 and 10(a) for the MCNC circuits.

For these large benchmark circuits, we find that the model results follow the trends of the experimental results fairly well.

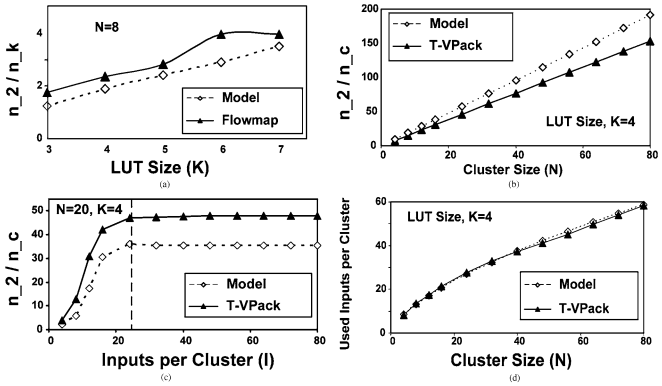



Fig. 12. Model validation for area equations: QUIP benchmark circuits (a) Logic per LUT versus LUT size. (b) Logic packed per cluster versus cluster size. (c) Logic packed per cluster versus cluster inputs. (d) Used inputs per cluster versus cluster size.

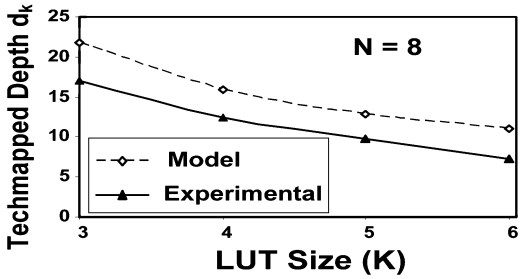



Fig. 13. QUIP benchmark circuits: model verification for $d_k$.

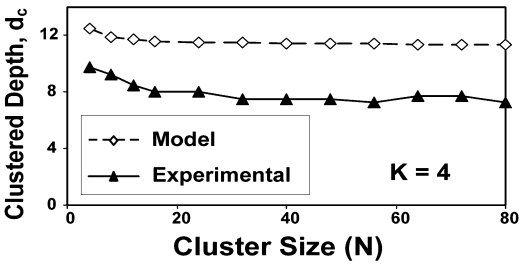



Fig. 14. QUIP benchmark circuits: model verification for $d_c$.

Furthermore, the validation results for the QUIP benchmark circuits agree with the conclusions that we have drawn for the MCNC benchmark circuits. As an example, Fig. 6(c) and 12(c) represent logic packed per cluster $(n_2/n_c)$ with respect to inputs per cluster $I$ for the MCNC circuits and the QUIP circuits respectively. In both cases, the trends of the model results are similar. More importantly, the boundary between $I$- and $N$-limited regions is very similar for both sets of benchmarks. This demonstrates our model's applicability for a wide range of benchmark circuits.

Table VIII presents the numerical comparison of our model's accuracy for (a) QUIP benchmark circuits, (b) MCNC benchmark circuits and (c) the combination of QUIP and MCNC benchmark circuits, for two representative sets of values for $N$ and $K$. We present the results for the total logic packed per cluster $(n_2/n_c)$, the total used inputs per cluster $(i)$ and the depth after clustering $(d_c)$. The values are averaged over the corresponding number of circuits.

TABLE VIII
COMPARISON OF ACCURACY

| | QUIP | | MCNC | | QUIP + MCNC | |
|---|---|---|---|---|---|---|
| | Model | Exp. | Model | Exp. | Model | Exp. |
| Cluster Size $N$: 8; LUT Size $K$: 4 | | | | | | |
| $n_2/n_c$ | 15.4 | 19.0 | 14.8 | 14.2 | 14.9 | 15.0 |
| $i$ | 13.2 | 13.4 | 14.2 | 13.6 | 14.0 | 13.5 |
| $d_c$ | 11.9 | 9.3 | 8.5 | 7.4 | 9.0 | 7.7 |
| Cluster Size $N$: 24; LUT Size $K$: 6 | | | | | | |
| $n_2/n_c$ | 86.9 | 133.6 | 60.3 | 56.9 | 64.8 | 69.7 |
| $i$ | 40.5 | 36.4 | 35.3 | 38.5 | 36.2 | 38.1 |
| $d_c$ | 9.3 | 5.8 | 8.0 | 8.0 | 8.3 | 5.1 |

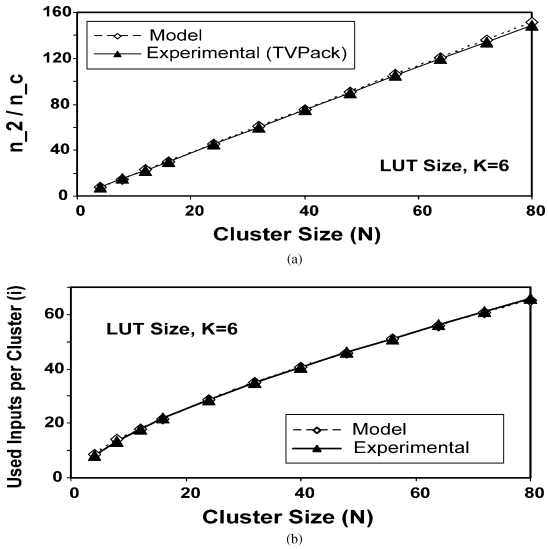



Fig. 15. Effects of higher LUT size: QUIP + MCNC benchmark circuits. (a) Logic packed per cluster versus cluster size. (b) Used inputs per cluster versus cluster size.

### E. Area Model and Higher LUT Sizes

In the earlier subsections, we have presented the validation results for LUT size of 4. We observe that our model performs well for higher LUT sizes, as illustrated in Fig. 15 for $K = 6$. The presented values are the aggregates over 24 circuits from the MCNC and the QUIP suites. Due to space, we omit further results.

## VII. IMPACT OF ARCHITECTURAL AND CIRCUIT PARAMETERS

The model derived in Section V is intended to be used during FPGA architectural design. There are at least two ways the model can be used. During the very initial "back of the envelope" design, intuition about what architectural parameters affect the amount of logic and the speed of the logic would be valuable. This intuition would allow designers to come up with a reasonable first cut at an architecture. Later stages of architectural development involve "parameter sweeps" which involve estimating the area and delay of an architecture for various values of architectural parameters. Usually, parameter sweeps are done with the aid of a representative CAD tool (such as VPR), however, initial sweeps can be performed without such CAD tools by combining our model with detailed area and delay models. In this section we develop a number of "rules of thumb" that can guide initial development, and in the next section, we will show how the model can be used to perform initial parameter sweeps.

The impact of an architectural parameter on area or delay can be obtained experimentally by mapping example circuits to architectures with different values for the parameter under investigation, and measuring the absolute differences between the results obtained for consecutive values of the parameter. There are two problems with this approach. First, it requires representative CAD tools and a large number of benchmark circuits. Second, experimental results using benchmark circuits often display experimental "noise", caused by second or third order effects, and often are a result of pathological mapping results of the benchmark circuits.

Instead, the closed-form nature of our equations allows us to calculate the derivative of the behaviour around any point, and use this to understand the impact of the parameter. As an example, consider the impact of the LUT size on the values of $n_k/n_2$ and $d_k/d_2$. Clearly, both quantities decrease as $K$ increases, but the rate at which each decreases as well as the impact of other parameters on this decrease are not clear. Differentiating Equations (4) and (19) with respect to $K$, we get

$$\frac{\partial(n_k/n_2)}{\partial K} = -\frac{\sqrt[p]{4}}{p(K+2)^{\frac{1+p}{p}}} \tag{27}$$

and

$$\frac{\partial(d_k/d_2)}{\partial K} = -\frac{3}{2}\left[\frac{1+\frac{4}{(\ln 2)(3K+2))}}{\left[\frac{3K}{4}-\frac{1}{2}+\log_2\left(\frac{3K}{4}+\frac{1}{2}\right)\right]^2}\right]. \tag{28}$$

In deriving these, we replaced $\gamma$ with a linear approximation ($\gamma = (1/4)K - (1/2)$). We omit the derivation of these expressions for space. We can calculate the relative change in $n_k/n_2$ due to a change $\Delta K$ around point $K$ as follows:

$$\frac{f'(K)}{f(K)} \cdot \Delta K \tag{29}$$

where $f(K) = n_k/n_2$ and $f'(K) = \partial(n_k/n_2)/\partial K$. Similarly, we can find the relative change in $d_k/d_2$.

Fig. 16(a) shows this quantity as a function of $K$ for $p = 0.67$. The graph says that increasing $K$ by $\Delta K$ causes a $0.3\Delta K$ reduction in $n_k/n_2$ for $K = 4$ and the reduction in $n_k/n_2$ slightly changes for higher values of $K$. Fig. 16(a) also shows that for $K = 4$, increasing $K$ by $\Delta K$ causes a $0.25\Delta K$ reduction in $d_k/d_2$. For larger values of $K$, the impact on $d_k/d_2$ is smaller. Taken together, these graphs show that for the values of $K > 4$, the parameter $K$ has a much stronger impact on the total number of LUTs required to implement a circuit than the number of LUTs along the circuit's critical path.

The impact of parameter $N$ can be obtained similarly. For $N$-limited clustering, by differentiating Equations (13) and (25) with respect to $N$, we obtain

$$\frac{\partial(n_c/n_k)}{\partial N} = -\frac{1}{N^2} \tag{30}$$

and

$$\frac{\partial(d_c/d_k)}{\partial N} = -\frac{1}{K-\gamma}\left[\frac{1}{N^2}+\frac{K-\gamma-1}{n_k}\right]. \tag{31}$$

We can use the last two equations with (29) to calculate the relative change in $n_c/n_k$ and in $d_c/d_k$ due to a change $\Delta K$ around point $K$.

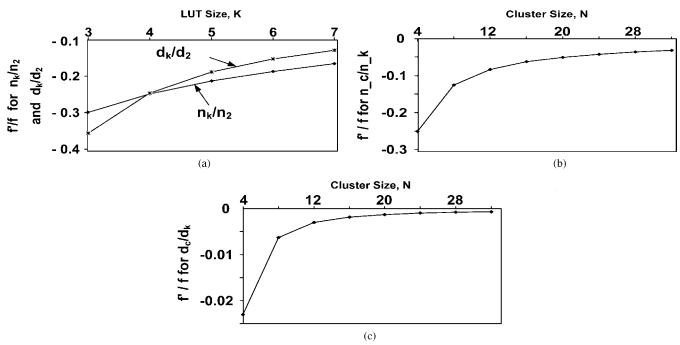



Fig. 16. Impact of architectural and circuit parameters. (a) Impact of LUT size on the number of logic blocks and the post-techmapping depth. (b) Impact of cluster size on the number of clusters. (c) Impact of cluster size on the post-clustering depth.

Fig. 16(b) and (c) plot these quantities as a function of $N$ for $p = 0.67$. In Fig. 16(c), we assume $n_k$ to be 3 000. (We observe that $d_c/d_k$ is a weak function of $n_k$). We also observe that, for smaller values of $N$, the $\Delta N$ change will have a prominent impact on both $n_c/n_k$ and $d_c/d_k$. However for higher values of $N$, the change in $N$, $\Delta N$ will not have any significant impact on the number of clusters $n_c$ or the post-clustering depth $d_c$. Fig. 16(b) and (c) also show the relative impact of the cluster size $N$ on the density and the speed, where $\Delta N$ affects density (function of $n_c/n_k$) more than speed (function of $d_c/d_k$) by an order of magnitude.

## VIII. Using the Model for Architectural Investigation

One of the purposes of our model is to allow for early architectural evaluation. In this section, we present two examples of how our model can be used in FPGA architectural development. The first example applies to area investigation, and the second applies to delay investigation.

### A. Area Investigation

In this subsection, we show how the model from Section V-A, along with the channel width model from [5], can be used to estimate the number of programming bits in an FPGA as a function of the architectural parameters $K$, $N$, and $I$. A similar investigation was performed in [18].

We consider three flows:

1) The first flow is purely analytical. We use our model to estimate $n_c$ and $i$ as a function of $n_2$ for twenty large benchmark circuits. We then estimate the wirelength using the model from [18]. These quantities are used in conjunction with the channel width model in [5] 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. Finally, these area estimates are used to determine the number of programming bits required for each of twenty large benchmark circuits.
2) The second flow is purely experimental. We map each of the twenty benchmark circuits to 4-input LUTs using Flowmap ($K = 4$), cluster using T-VPack ($N = 4, I = 10$), and then place and route the circuits using VPR 5.0.

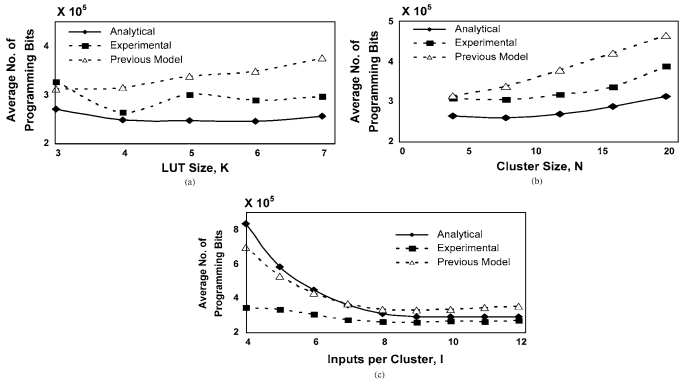



Fig. 17. FPGA area optimization example. (a) K-sweep. (b) N-sweep. (c) I-sweep.

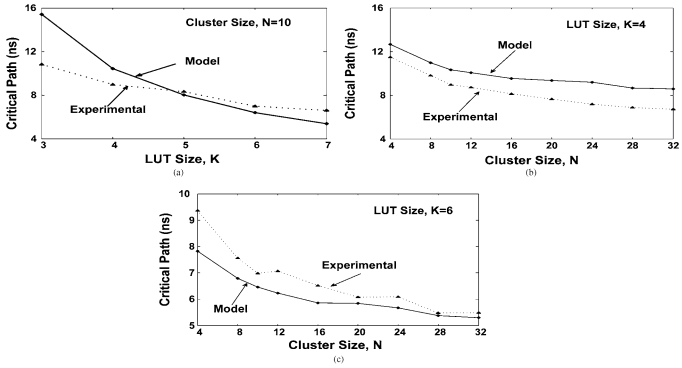



Fig. 18. FPGA delay optimization example. (a) K-Sweep ($\mathrm{N} = 10$). (b) N-Sweep ($\mathrm{K} = 4$). (c) N-Sweep ($\mathrm{K} = 6$).

We use the model within VPR 5.0 to count the number of programming bits for each benchmark circuit.

3) The third flow is same as the first flow, except that a constant wirelength of 4.43 is assumed for the benchmark circuits as assumed in [5].

Fig. 17 shows the results for an architecture with $F_s = 9$, $F_{\mathrm{cin}} = 20$, $F_{\mathrm{cout}} = 4$, and $L = 1$. Fig. 17(a), (b) and (c) show the number of programming bits as a function of $K$, $N$ and $I$ respectively. In all cases, the trends observed from Flow 1 (purely analytical) follow the trends from Flow 2 (purely experimental). The analytical flow leads to similar conclusions that would be obtained by the more time-consuming experimental methodology. This is to be expected, given the accuracy of the models in [5], [18] and Section V-A.

Fig. 17 also shows a significant difference between the results of Flow 3 and Flow 2. We have observed it in [8] and suggested that the difference is due to the inaccurate wirelength assumption. The comparison between Flow 3 and Flow 1 (which differ only in their wirelength assumption) shows that this is indeed the case. The fact that, in most cases, Flow 1 matches the experimental results much better than Flow 3 indicates that an accurate wirelength model is important.

### *B. Critical Path Delay Investigation*

In this subsection, we show how the model from Section V-B, along with the wirelength model from [18], can be used to estimate the speed of an FPGA as a function of the architectural parameters $K$ and $N$. We will investigate whether our model leads to similar conclusions that would be obtained by a more time-consuming experimental methodology.

We consider two flows. The first flow is purely experimental. For each of the twenty MCNC circuits of Table II, we use Flowmap [21] to technology map to LUTs, T-VPACK to map to clusters, then VPR to place and route. We vary $K$ and $N$. A routing fabric with $F_s = 3$, $F_{\mathrm{cin}} = 0.25$, $F_{\mathrm{cout}} = 1.0$, and segment length of 1 is assumed. For reasons described below, a very wide channel width (200 tracks per channel) is used in these experiments. The critical path is measured after routing.

The second flow is analytical. We use the model derived in Section V-B to find the post techmapping depth $d_k$ and the post-clustering depth $d_c$ for each of the circuits in Table II. These results are then used in (26) to estimate the expected critical path of each circuit. To use (26), we also need an estimate of $t_{\mathrm{intra}}$ and $t_{\mathrm{inter}}$. Since we do not yet have an analytical model for these quantities, we estimate them using experimental results from VPR 5.0 as follows.

For $t_{\mathrm{intra}}$, we add the intra-cluster routing delay and the LE delay obtained from architecture files included with VPR 5.0. Both of these quantities are functions of $N$ and $K$.

Estimating $t_{\mathrm{inter}}$ is more challenging. We start with the wirelength model from [18] to estimate the average wirelength for each of the circuits that we use in the experimental flow. We then tabulate the relationship between wirelength and delay of a net using the delay table constructed during the first phase of VPR's timing-driven placement step [2]. In this way, we obtain an estimate for $t_{\mathrm{inter}}$ as a function of the architecture parameters $K$ and $N$. This quantity, however, underestimates the true value of $t_{\mathrm{inter}}$ for two reasons. First, the delay estimates used during placement do not account for congestion. To minimize this effect, we assume a very wide channel width when gathering our experimental results. Second, we have observed that wires along the critical path are typically longer than the "average wirelength". This appears counter-intuitive. We would expect a timing-driven placement algorithm to place cells so that wires along the critical path are shorter than the average. However, the critical path after placement often is not the same path as the critical path before placement. In fact, those nets that were deemed "not critical" before placement tend to be longer than average, and paths using these longer segments are more likely to become critical. To account for this, we assume that the wires along the critical path are a factor of $\beta$ slower than the average wire. Experimentally, we have found that $\beta = 2$ works well, and we use this scaling factor to compute $t_{\mathrm{inter}}$. An analytical method for computing $t_{\mathrm{inter}}$, especially for FPGAs with a narrow channel width, is an open problem, and would be an interesting topic for future research.

Fig. 18(a) shows the predicted and measured critical path delay for various values of $K$. We only show data for $N = 10$ due to space. Each point represents the geometric mean over twenty circuits. For both analytical and experimental flows, the critical path delay decreases with increasing LUT size and the rate of decrease becomes smaller with increasing LUT size. However, for smaller values of lookup-table size, our model overestimates the experimental delay and for higher values of LUT size, it slightly underestimates the experimental values.

The discrepancies of Fig. 18(a) can be partially explained as follows. Equation (26) shows that the critical path delay is

dependent on the values of the post-techmapping depth $d_k$ and the post-clustering depth $d_c$. Section VI.B shows that, for high values of post-techmapping depth $d_k$, our model overestimates the experimental values (of $d_k$), which is the case for low values of LUT size. Consequently, the model overestimates the post-clustering depth $d_c$ as well as the critical path delay in such cases. For higher LUT sizes, this limitation of techmapping depth model becomes less prominent.

Critical path delay from (26) is also dependent on intra- and inter-cluster delay values. In determining the inter-cluster delay, we used a scaling factor $\beta$ to relate the wires along the critical path to the average wire. We find that the product of (a) $\beta$ and (b) the average wirelength from the model in [18], still underestimates the wirelength along the critical path to some extent. This explains why the model slightly underestimates the experimental values for higher LUT sizes.

We find similar discrepancies in Fig. 18(b) and (c). These figures show the predicted and measured critical path delay for various values of cluster size $N$ for two values of LUT size $K$. Critical path delay is averaged (gemetric) over twenty circuits. We again find that the model overestimates the delay for lower LUT size and underestimates the delay for higher LUT size. However, in both analytical and experimental cases, the critical path delay decreases with increasing value of $N$, and the rate of change decreases for higher values of $N$. In summary, for both analytical and experimental flows in Figs. 17 and in 18, we can reach similar conclusions in regards with the effects of the architectural parameters (cluster size, LUT size) on area and delay. However, with the analytical flow, we are able to come to the conclusions *without running time consuming experiments*.

## IX. Conclusion

This paper presents an analytical model that relates the lookup-table size, cluster size and inputs per cluster to the area and depth of a circuit implementation. We show that our model can be used during early-stage architecture evaluation, in which potential architectures are considered before custom CAD tools are created. Comparing the model predictions with experimental results, we find that our model is sufficiently accurate for this purpose.

This model helps designers to understand the relation between architecture, circuit and the expected density and depth. Such an understanding will help designers to make architectural trade-offs without experimental investigations. This model can provide the bounds for architectural parameters with respect to density and speed. Recent FPGAs employ larger lookup-tables and clusters. Using a model such as ours provides the ability to understand the relative impacts of such changes on area and speed and, more importantly, the insights about why the area and speed are impacted in a certain way. For FPGA architects, this could be potentially much more useful than experimental results that evaluate a single architecture (or evan a sweep of a single architecture). As a side effect, our depth model also gives insights that may be used to enhance various stages of a typical CAD flow.

However, our model has limitations. For small values of inputs per cluster $I$, our area model correctly models architectures. However, when incorporated with a channel-width model, it may not follow the experimental values for the number of programming bits. A complete analytical flow to predict the density for these architectures with small $I$ is an interesting area for future research. Our depth model cannot accurately predict the proportion of local connections for higher cluster sizes. This is primarily due to the simplified assumptions regarding the number of connections shared during clustering. The depth model also overestimates the post-techmapping depth values for high values of depth. More detailed modeling to resolve these issues is an interesting direction for future research. However, since simple sets of equations can provide designers with more insights about the effects of architectural trade-offs, such detailed modeling should carefully balance complexity and accuracy. Modeling post-routing critical path delay as a function of architecture parameters (including parameters that describe the routing) will significantly speed up the evaluation of new architectures. Such critical path delay model may use the depth model from this paper as a building block.

Finally, in FPGA domain, several architectural parameters exhibit *discrete* effects. As an example, for a circuit that predominantly consists of 4-input multiplexer functions, a 5-LUT implementation may be much more efficient than a 4-LUT implementation, whereas a 6-LUT implementation may not provide any additional benefit. Using our model, designers can make fast analysis to short list a region around such discrete parameters, before running expensive experiments only on these selected regions. However, while addressing some design questions, our model may not provide useful insight due to the presence of such discrete effects. We plan to identify such discrete effects and the consequent limitations of our model, which would better inform FPGA architects and designers about the scope of our model.

## References

[1] M. Hutton, J. Schleicher, D. M. Lewis, B. Pedersen, R. Yuan, S. Kaptanoglu, G. Baeckler, B. Ratchev, K. Padalia, M. Bourgeault, A. Lee, H. Kim, and R. Saini, "Improving FPGA performance and area using an adaptive logic module," in *Proc. Int. Conf. FPL*, 2004, pp. 135–144.

[2] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-submicron FPGAs*. Norwell, MA: Kluwer, 1999.

[3] M. Hutton, "FPGA architecture design methodology," in *Proc. Int. Conf. FPL*, 2006, p. 1.

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

[5] W. Fang and J. Rose, "Modeling FPGA routing demand in early-stage architecture development," in *Proc. Int. Symp. FPGAs*, 2008.

[6] A. M. Smith, G. A. Constantinides, S. J. E. Wilton, and P. Y. K. Cheung, "Concurrently optimizing FPGA architecture parameters and transistor sizing: Implications for FPGA design," in *Proc. Int. Conf. FPT*, 2009, pp. 54–61.

[7] E. Hung, S. J. E. Wilton, H. Yu, T. C. P. Chau, and P. H. W. Leong, "An analytical FPGA delay path model," in *Proc. Int. Conf. FPT*, 2009, pp. 96–103.

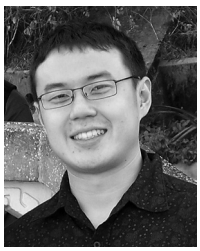
[8] A. Lam, S. J. Wilton, P. Leong, and W. Luk, "An analytical model describing the relationship between logic architecture and FPGA density," in *Proc. Int. Conf. FPL*, 2008, pp. 221–226.

[9] J. Das, S. J. Wilton, P. Leong, and W. Luk, "Modeling post-techmapping and post-clustering FPGA circuit depth," in *Proc. Int. Conf. FPL*, 2009, pp. 205–211.

[10] A. A. El Gamal, "Two-dimensional stochastic models for interconnections in master-slice integrated circuits," *IEEE Trans. Circuits Syst.*, vol. 26, no. 4, pp. 127–138, Feb. 1981.
[11] J. Rose, "Closing the gap between FPGAs and ASICs, keynote address," in *Proc. Int. Conf. FPT*, Dec. 2006, p. viii.
[12] S. Brown, J. Rose, and Z. Vranesic, "A stochastic model to predict the routability of field-programmable gate arrays," *IEEE Trans. CAD Circuits Syst.*, vol. 12, no. 12, pp. 1827–1838, Dec. 1993.
[13] B. Landman and R. Russo, "On a pin versus block relationship for partitions of logic graphs," *IEEE Trans. Comput.*, vol. C-20, pp. 1469–1479, 1971.
[14] J. Pistorius and M. Hutton, "Placement rent exponent calculation methods, temporal behaviour and FPGA architecture evaluation," in *Proc. Int. Workshop SLIP*, 2003, pp. 31–38.
[15] W. Donath, "Placement and average interconnect lengths of computer logic," *IEEE Trans. Circuits Syst.*, vol. 26, no. 4, pp. 272–277, Apr. 1979.
[16] D. Stroobandt, *A Priori Wire Length Estimates for Digital Design*. Norwell, MA: Kluwer, 2001.
[17] S. Balachandran and D. Bhatia, "A priori wirelength estimation and interconnect estimation based on circuit characteristics," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 7, pp. 1054–1065, Jul. 2005.
[18] A. M. Smith, J. Das, and S. J. Wilton, "Wirelength modeling for homogeneous and heterogeneous FPGA architectural development," in *Proc. Int. Symp. FPGAs*, 2009, pp. 181–190.
[19] V. Manohararajah, G. Chiu, D. Singh, and S. Brown, "Predicting interconnect delay for physical synthesis in a FPGA CAD flow," *IEEE Trans. VLSI Syst.*, vol. 15, no. 8, pp. 895–903, Aug. 2007.
[20] 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 *Proc. Int. Symp. QED*, 2005, pp. 370–374.
[21] J. Cong and Y. Ding, "Flowmap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 1, pp. 1–12, Jan. 1994.
[22] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in FPGAs," in *Proc. Int. Symp. FPGAs*, 2002, pp. 59–66.
[23] P. Zarkesh-Ha, J. A. Davis, W. Loh, and J. D. Meindl, "Prediction of interconnect fan-out distribution using Rent's rule," in *Proc. Int. Workshop SLIP*, 2000, pp. 107–112.
[24] S. Yang, Logic Synthesis and Optimization Benchmarks User Guide Version 3.0 Technical Report MCNC, 1991.
[25] *Quartus II University Interface Program (QUIP) Toolkit*. San Jose, CA: Altera corporation [Online]. Available: https://www.altera.com

**Joydip Das** received the B.Sc. degree from the Electrical Engineering and Electronics Department at Bangladesh University of Engineering and Technology, Bangladesh, in 1996 and the M.S. degree from the Electrical and Computer Engineering Department at the University of Louisiana at Lafayette, in 2006. He is currently pursuing the Ph.D. degree in the Electrical and Computer Engineering Department at the University of British Columbia, Vancouver, BC, Canada.

His research interests include FPGA logic architecture, analytical modeling, and low-power VLSI design.



**Andrew Lam** received the B.A.Sc. degree from the Engineering Science Program at University of Toronto, Toronto, ON, Canada, in 2006 and the M.A.Sc. degree from the Electrical and Computer Engineering Department at the University of British Columbia, Vancouver, BC, Canada, in 2010.

His research interests include field-programmable gate array (FPGA) logic architecture, computer-aided design (CAD), analytical modeling, and computer architecture.

**Steven J. E. Wilton** (SM'03) received the M.A.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 1992 and 1997, respectively.

In 1997, he joined the Department of Electrical and Computer Engineering at the University of British Columbia, where he is now a Professor and Associate Head. During 2003 and 2004, he was a Visiting Professor in the Department of Computing at Imperial College, London, U.K, and at the Interuniversity MicroElectronics Center (IMEC) in Leuven, Belgium. He is a co-founder of Veridae Systems, which supplies post-silicon validation architectures and tools. His research focuses on the architecture of FPGAs, and the CAD tools that target these devices. In 2005, he was the Program Chair for the ACM International Symposium on Field-Programmable Gate Arrays and the program co-chair for the International Conference on Field Programmable Logic and Applications.

Dr. Wilton received best paper awards at the International Conference on Field-Programmable Technology in 2003, 2005, and 2007 and at the International Conference on Field-Programmable Logic and Applications in 2001, 2004, 2007, and 2008. In 1998, he won the Douglas Colton Medal for Research Excellence for his research into FPGA memory architectures. He is currently an Associate Editor of the ACM Transactions on Reconfigurable Technology and Systems.

**Philip H. W. Leong** (SM'02) received the B.Sc., B.E., and Ph.D. degrees from the University of Sydney, Sydney, Australia.

In 1993 he was a consultant to ST Microelectronics in Milan, Italy working on advanced flash memory-based integrated circuit design. From 1997–2009 he was with the Chinese University of Hong Kong. He is currently an Associate Professor in the School of Electrical and Information Engineering at the University of Sydney. He is also a Visiting Professor at Imperial College, London and the Chief Technology Consultant to Cluster Technology. He is the author of more than 100 technical papers and 4 patents.

Dr. Leong was the co-founder and program co-chair of the International Conference on Field Programmable Technology (FPT); program co-chair of the International Conference on Field Programmable Logic and Applications (FPL) and is an Associate Editor for the ACM *Transactions on Reconfigurable Technology and Systems.* He was the recipient of the 2005 FPT Conference Best Paper as well as the 2007 and 2008 FPL conference Stamatis Vassiliadis Outstanding Paper awards.

**Wayne Luk** (F'09) received the M.A., M.Sc. and D.Phil. degrees in engineering and computing science from Oxford University, Oxford, U.K.

Currently Professor of Computer Engineering at Imperial College, he founded and leads the Computer Systems Section and the Custom Computing Group in Department of Computing, and was Visiting Professor at Stanford University and Queen's University Belfast. He is a member of the Program Committee of many international conferences such as FCCM, FPGA and DATE. He has been an author or editor for 6 books and 4 special journal issues.

Dr. Luk had ten papers that received awards from the ASAP, FPL, FPT, SAMOS, SPL and ERSA conferences, and he also won a Research Excellence Award from Imperial College in 2006. He is a Fellow of the BCS, and is founding Editor-in-Chief for ACM *Transactions on Reconfigurable Technology and Systems*.