Wirelength Modeling for Homogeneous and Heterogeneous FPGA Architectural Development

Alastair M. Smith, Joydip Das, Steven J.E. Wilton
Department of Electrical and Computer Engineering
University of British Columbia, Canada
{ alastairs, dasj, stevew } @ece.ubc.ca

ABSTRACT

This paper describes an analytical model that relates the architectural parameters of an FPGA to the average pre-routing wirelength of an FPGA implementation. Both homogeneous and heterogeneous FPGAs are considered. For homogeneous FPGAs, the model relates the lookup-table size, the cluster size, and the number of inputs per cluster to the expected wirelength. For heterogeneous FPGAs, the number and positioning of the embedded blocks, as well as the number of pins on each embedded block is considered. Two applications of the model to FPGA architectural design are also presented.

Categories and Subject Descriptors

B.7.1 [**Types and Design Styles**]: Gate arrays

General Terms

Design, Theory

1. INTRODUCTION

The continuous improvement of integrated circuit technology has had a dramatic impact on the architecture of Field-Programmable Gate Arrays (FPGAs). Recent device generations have seen innovations such as new logic block structures, flexible embedded blocks, and complex interconnect networks. As FPGAs are implemented using even smaller process technologies, their architectures will continue to evolve.

FPGA architectures are usually developed and evaluated using an experimental methodology. FPGA architects gather benchmark circuits and map them to potential architectures using representative computer-aided design (CAD) tools. The resulting implementations are then compared using area, delay, and power models [4, 1, 14]. Although the experimental approach can directly and accurately help search for optimum architectures, it presents a number of challenges. First, FPGA architecture parameters should not be tuned in isolation; an exhaustive architecture study should consider all combinations of parameters, leading to excessive compute times. Second, the experimental methodology requires the development of representative CAD tools for each architecture under consideration. Although parameterized tools, such as VPR [4], can be used to target a range of architectures, these tools still should be "tuned" for architectures, and in any case, must be re-written to investigate radically different architectures. Third, the experimental methodology often provides little insight into *why* certain architectural decisions work well. This insight is critical as architects refine their design or determine new areas of the search space to explore.

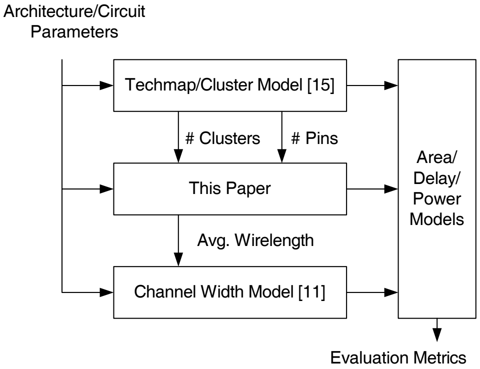

Figure 1: Analytical Model Flow

Recently, there have been efforts to develop a set of analytical models that relate architecture parameters to each other, and to the area, delay, and power of the resulting implementations [12, 16]. Understanding the relationships between architectural parameters enables early-stage architecture development 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.

Two recently published models describe part of the architectural space. In [16], a model relating logic block size, cluster size, and inputs per cluster to the area efficiency of the resulting FPGA was presented. The model in [12] relates the channel width of an FPGA to parameters describ-

FPGA'09, February 22–24, 2009, Monterey, California, USA.

ing the detailed routing architectures. One of the inputs in the channel width model is the expected average pre-routing wirelength of nets after placement. In [12] this is treated as a constant, but it is suggested that for heterogeneous architectures, this may not be the case. In [16], it is suggested that this may also not be true for certain "extreme" architectures not considered in [12].

In this paper, we present *an analytical model that relates FPGA architectural parameters to the average post-placement pre-routing wirelength of an implementation.* We consider both homogeneous and heterogeneous architectures. This is, in effect, the missing link between the models in [16] and [12]. As shown in Figure 1, the model in [16] can be used to predict the number of clusters required to implement a circuit. The model in this paper then uses this quantity to estimate the average wirelength, which can then be used as an input to the model in [12]. The results from all three models can then be used in conjunction with area, delay, and/or power models to evaluate the architecture under consideration.

An understanding of the impact of architectural decisions on the expected wirelength is important. The average wirelength determines the amount of routing that an FPGA architect must include in a device. The size and flexibility of this fabric affects the achievable density, power, and speed of the FPGA to a first order. The model in this paper can be used as a tool for architects to understand the impact of their architectural decisions on these important metrics.

This paper is organized as follows. Section 2 describes the related work. Section 3 then presents our architectural assumptions. A wirelength model for homogeneous FPGAs is presented in Section 4. The model is then generalized for heterogeneous FPGAs in Section 5. Finally, Section 6 shows two examples of how the model can be used during FPGA architecture design.

2. RELATED WORK

Wirelength modeling has received considerable attention. Most wirelength models are based on the empirical observation known as Rent's Rule [17]. Rent's Rule relates the number of terminals T emanating from a partition containing G gates through the expression $T = kG^p$ where k is the average number of terminals per gate and p is the Rent Parameter of a circuit. The Rent Parameter can be thought of as a measure of the interconnect complexity of the circuit.

Most wirelength models have been developed for ASICs [10, 13, 19, 8, 9]. These models can be used directly for FPGA implementations, by treating either a logic cluster or a logic element as a "gate". For example in [18], Feuer's model [13] has been modified and used to model the wirelength requirements of designs mapped to FPGAs. However, this work did not focus on the impact of architecture on wirelength.

El Gamal derives a model that relates the area required for routing to the total number of pins in the logic gates [11]. Although this could be indirectly used to estimate routing area, El Gamal's equation was derived for ASICs and does not take into account the architecture parameters in an FPGA.

Given more information about the circuit, wirelength *estimation* is possible. In [2] a model has been developed for estimating the wirelength and channel width of an FPGA implementation. This is different than our work. Rather than estimating the wirelength of a given circuit, we wish to develop models that relate architectural parameters to expected wirelength across all circuits. The model in [2] is aimed at developers of fast CAD algorithms, or algorithms that need early wirelength estimates during higher-level processing. Our model is aimed at FPGA architects.

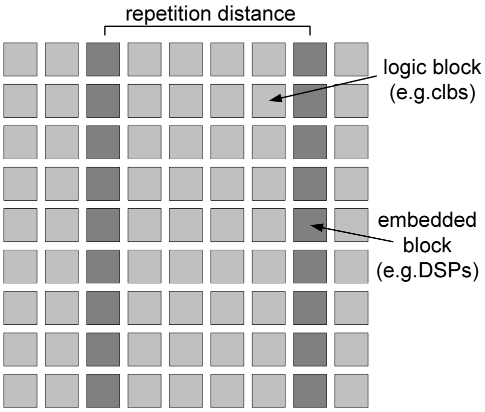

Figure 2: The architectural model used in this paper

Other researchers have developed models that relate FPGA architecture parameters to the expected channel width of an FPGA implementation [12, 5]. These works focus on the "inflation" of wirelength due to the limited flexibility in typical FPGA routing fabric, but do not address the distance between the pins that are to be connected.

Most of the above works focus on homogeneous FPGAs. In this paper, we are concerned with FPGAs which contain embedded blocks such as memories and DSP blocks. Heterogeneous SoCs, which contain cores with different properties, were considered in [23]. The nature of these heterogeneous cores is very different than the heterogeneous blocks in an FPGA, and thus can not be directly applied to our problem.

3. FRAMEWORK

This section first describes the architectural framework we assume and assumptions regarding the circuits being implemented on the FPGA. It then precisely describes our interpretation of wirelength.

3.1 Architectural and Circuit Assumptions

We assume an island-style FPGA in which an array of blocks is connected using prefabricated tracks organized in horizontal and vertical channels [4] as illustrated in Figure 2. Each block is either a logic cluster (also called a configurable logic block, clb) or an embedded macro-block (such as a memory or DSP block). Although we will assume one type of embedded block in the paper, we expect that it would be straightforward to extend our equations for FPGAs with multiple types of embedded blocks.

The parameters that describe this architecture are summarized in the top half of Table 1. Since this paper is concerned with *pre-routing* wirelength, we do not include parameters that describe the detailed routing fabric.

We wish to derive relations between architectural parameters that are as independent of the circuit to be implemented on the FPGA as possible. For example, we would prefer to understand how logic block size affects wirelength, independent of the circuit to be implemented. This is different from

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
H	Repetition distance for embedded blocks
P_h	Pins on embedded block h
Circuit Parameters:	
p	Rent parameter of a given circuit
n_2	Number of 2-LUTs in a given circuit
n_h	Number of embedded block type h in circuit

much prior estimation work, in which the goal is to predict the area, speed, or power for a specific circuit. That being said, it is impossible to completely ignore the impact of the circuit; we describe our circuit using the Rent Parameter, the size of the un-techmapped circuit, and the ratio of blocks that will be mapped to embedded blocks to those that will be mapped to logic blocks. These parameters are summarized in the bottom half of Table 1.

3.2 Pre-Routed Wirelength

The quantity we are concerned with in this paper is the expected average pre-routed wirelength of nets in a circuit to be implemented in the FPGA. For a two-pin net, this is the expected distance between the pins after placement. For a multi-pin net, the best approximation to the amount of wiring is the length of all branches of the Minimum Steiner Tree connecting the pins of the net. To make our models more tractable, we consider approximations to this length in the following derivations.

It is important to note that our definition of wirelength does not include any “inflation” that may occur during routing due to limited flexibility in the routing network. Techniques for estimating the post-routing wirelength given the pre-routing wirelength and architectural parameters (such as F_c and F_s) is a future research direction. Solutions may include techniques similar to those in [12], which finds the expected post-routing channel width of an FPGA implementation.

4. WIRELENGTH MODELING FOR HOMOGENEOUS FPGAS

We first consider modeling the expected pre-routing wirelength of nets in a circuit implemented on a homogeneous FPGA (no embedded memory or DSP blocks). As described in Section 2, previous work has presented equations for the expected wirelength as a function of the number of cells in a circuit. We can apply this to FPGAs by considering each cluster as a single cell, and using the equations from [16] to relate the number of clusters to the architectural parameters K, N, and I.

In the following, we start with the wirelength models of both [13] and [8, 9]. The model of [13] can be written as follows:

$$WL_{2p} = \frac{2\sqrt{2}(3+3p)}{(1+2p)(2+2p)} n_c^{(p-0.5)} \qquad (1)$$

where n_c is the number of clusters (cells) in the circuit and WL_{2p} is the expected wirelength of 2-point nets. Using the results of [16], we can write:

$$n_c = \begin{cases} n_2 \sqrt[p]{\frac{3}{I(1+\frac{1}{f_{avg}})}} & \text{if } I < N^p \frac{K+1-\gamma}{1+\frac{1}{f_{avg}}} \\ \frac{n_2}{N} \sqrt[p]{\frac{3}{K+1-\gamma}} & \text{if } I \geq N^p \frac{K+1-\gamma}{1+\frac{1}{f_{avg}}} \end{cases} \qquad (2)$$

The fanout f_{avg} in Equation 2 can be calculated using a formula from [22]:

$$f_{avg} = \frac{1-(f_{max}+1)^{(p-1)}}{1-(f_{max}+1)^{(p-2)} - \phi(p, f_{max})} - 1 \qquad (3)$$

where:

$$\phi(p, f_{max}) = \sum_{n=1}^{f_{max}} \frac{n^p}{n^2(n+1)}, \qquad (4)$$

and

$$f_{max} = [(i+N)\frac{n_k}{N}(1-p)]^{(1/(3-p))}, \qquad (5)$$

In these equations γ is a constant, n_k is the expected number of k-LUTs and i is the expected number of used cluster inputs (these are given in [16]). f_{max} represents the maximum expected fanout of a net. In the following we refer to this combination as the Feuer/Lam model. The model of [9] can be written as:

$$WL_{2p} = \frac{\frac{p-0.5}{p} - \sqrt{n_c} - \frac{p-0.5}{6\sqrt{n_c}(p+0.5)} + \frac{-p-1+4^{(p-0.5)}}{2p(p+0.5)(p-1)} n_c^p}{1 + \frac{-2p-1+2^{(2p-1)}}{2p(p-1)(2p-3)} n_c^{(p-0.5)} - \frac{p-0.5}{6p\sqrt{n_c}} - \frac{(p-0.5)\sqrt{n_c}}{p-1}} \qquad (6)$$

where n_c can be written using the results of [16] as above. In the following, we refer to this combination as the Davis/Lam model.

The wirelength of nets with more than one sink can be approximated using a technique from [8]:

$$WL = WL_{2p} \frac{4f_{avg}}{3+f_{avg}} \qquad (7)$$

where f_{avg} is from Equation 3. This approximation can be applied to either the Feuer/Lam model or the Davis/Lam model.

Note that our equation describes the expected wirelength of all *inter-cluster* nets. Intra-cluster nets, which are absorbed during the clustering process, are not considered. For our applications, this makes sense, since it is the inter-cluster net that determines the amount of routing required in an FPGA. The model in [12] uses the expected wirelength of the inter-cluster nets as an input. If the average wirelengths of all nets, including intra-cluster nets, is desired, the above equations could be scaled using the method described in [18].

The Rent parameter of each circuit varies significantly depending on architectural parameters. This is due to the nature of the technology-mapping and clustering into appropriate architecture blocks. In this paper, we experimentally determine the Rent parameter for each circuit after technology mapping and placement to use in the wirelength model. This is different from the Rent parameter used in the model for the number of clusters used, as that describes the Rent parameter of the circuit when implemented by 2-input lookup tables before clustering. We note that this is an interesting problem that should be addressed for future architectural modeling techniques.

Figure 3 compares the Feuer/Lam and the Davis/Lam models to experimental results. The experimental results

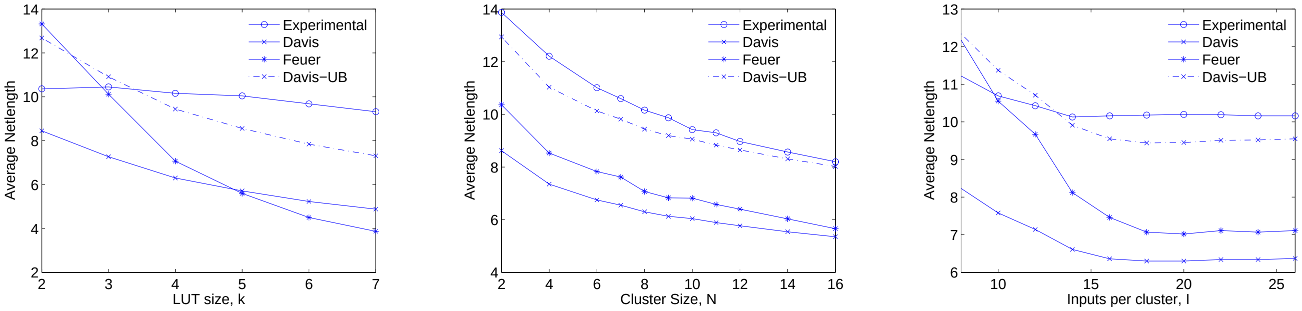

Figure 3: Model Verification for Homogeneous FPGAs

were obtained using 17 large MCNC circuits. We note that out of the twenty commonly used MCNC circuits, three are pad limited, which is not considered in the wirelength modeling techniques described. Each circuit was technology mapped using Flowmap [6], packed into clusters and placed using VPR 5.0 [3]. For each net, the minimum spanning tree was computed, and used as an approximation to the minimum Steiner Tree length. In Figure 3(a), the lookup table size, K, was varied from 2 to 7; as the results show, the Davis/Lam model follows the trends from the experimental results far better than the Feuer/Lam model. The same conclusion can be drawn from Figure 3(b) and 3(c), in which the cluster size N and the inputs per cluster I are varied.

The estimations provided by the Feuer/Lam and Davis/Lam models under estimate the experimental spanning trees due to the nature of the approximations. In [15], a theoretical result was proven that bounds the Steiner length as $> \frac{2}{3}$ of the Spanning tree length. If this is used as an approximation, the model estimation matches the experimental spanning tree length much better. This bound is shown in Figure 3 as the dotted line.

5. WIRELENGTH MODELING FOR HETEROGENEOUS FPGAS

The wirelength models in Section 4 apply to homogeneous FPGAs. Real FPGAs contain embedded blocks such as memories and DSP blocks. Intuitively, the presence of these blocks will tend to increase the average wirelength for a number of reasons.

First, these blocks add placement constraints. In most modern devices, embedded blocks are positioned in columns, as shown in Figure 2. The placement tool must find a solution in which all circuit elements that will be implemented by an embedded block are positioned in an appropriate column. These placement constraints may tend to increase the average wirelength slightly. Intuitively, the amount of this increase depends on the number of embedded blocks, and the number of connections that must be made to these embedded blocks for a given application.

Second, not all applications will tend to use all embedded blocks. When a heterogeneous FPGA is designed, the architect must fix the ratio of the number of embedded blocks to the number of logic blocks. If a given application does not require all embedded blocks, some embedded block sites will go unused. Similarly, if an application needs an excess of embedded blocks, then the designer will have to choose a larger FPGA that would otherwise be necessary, meaning some logic block sites will be left unused. In either case, the presence of these unused sites will tend to "spread out" the placement slightly, leading to larger average wirelengths.

Third, the number of pins in an embedded block may be different than the number of pins in a logic block. An embedded memory, for example, may have wide data input, data output, and address ports. An embedded DSP block may have wide datapath inputs and outputs. The number of pins directly determines the number of "neighbours" of a block; a neighbour of a block is any block connected to this block. In a heterogeneous FPGA, if some blocks have more neighbours, the average distance between the block and the neighbours may tend to be higher, since it is more difficult to place the embedded block close to all its neighbours. The derivations leading up to the equations used in Section 4 assume all blocks have the same number of pins; it is unclear how these equations are affected if some blocks have more pins than other blocks.

Fourth, embedded blocks may have connection patterns that are specific to that kind of block. For example, in [20], it is shown that multiple embedded memory blocks often have their address buses connected together. These block-specific connection patterns may lead to very different wirelength behaviour than that in Section 4.

In the following, we consider each of the first three factors listed above. For each, we experimentally determine the impact on wirelength, and show how the model from Section 4 can be modified to take into account that factor. One of the key challenges is isolating each factor in our experiments. We do not consider the fourth factor, since it is depends entirely on the type of embedded block, and not on the high-level FPGA architecture parameters. Consideration of the fourth factor is an interesting opening for future work.

5.1 Placement Constraints in Heterogeneous FPGAs

As described above, the presence of embedded blocks adds placement constraints that may tend to increase the average wirelength. In this section, we determine to what extent this intuition is true. Since we have not found an accurate way to include placement constraints in our wirelength models, our approach in this section will be experimental.

To measure the impact of embedded blocks, the most straightforward experimental approach would be to gather a suite of benchmark circuits with embedded blocks, map each to an FPGA using experimental CAD tools, and measure the correlation between the wirelength and the number of embedded blocks in each circuit. We have not chosen this approach for two reasons. First, we do not have access to a large enough suite with differing numbers of embedded blocks. Second, even if we did have a such a suite, comparing wirelengths from different circuits could result in misleading conclusions. Different circuits have very different wirelength characteristics, and these different characteristics may overshadow any change in wirelength due to the embedded blocks. Meaningful trends would require averaging results from many (more than 20) benchmark circuits for each different number of embedded blocks. In other words, we would need 20 or 30 circuits with only one embedded block, 20 or 30 more circuits with two embedded blocks, etc. Such a suite of benchmarks would be very difficult to obtain.

Instead, we artificially modify existing MCNC logic circuits by randomly selecting logic clusters and "converting" them to memories (any embedded block type would work, but for clarity, we use memories in the following discussion). In this way, we create a set of benchmark circuits in which each circuit contains one memory, a set of benchmark circuits in which each circuit contains two memories, etc. The base logic circuit is the same for all sets, meaning the circuits in each set are "almost identical" except for the ratio of memories to logic blocks. The circuits in each set have the same Rent parameter, same average fanout, and same connectivity pattern, all of which are known to have a direct impact on wirelength. This technique also ensures that, on average, all "memories" have the same number of pins as the logic clusters, thereby isolating the impact of differing number of pins (this will be considered in Section 5.3). To ensure our results are not affected by the random selection of logic clusters, we repeat the benchmark suite generation 10 times, each using a different seed for the random selection, thereby creating a very large benchmark suite.

Each circuit is then mapped to an FPGA using T-VPACK and VPR 5.0 (which supports embedded memories and DSP blocks). Since we wish to isolate the impact of the placement constraints, we must eliminate changes to mismatches between the number of available memories and memories in the benchmark circuit (this will be investigated in the next section). Therefore, we assume that the number of physical memory blocks in the architecture is the exact same as the number of memory blocks in the benchmark circuit. During the mapping, all clusters that have been "converted" to memory blocks are constrained to lie in memory columns, while all other clusters are constrained to lie in logic sites. We swept the ratio of logic clusters to memories from 0 to 50% and measured the post-placement pre-routing wirelength. A cluster architecture with $N = 10$ and $I = 22$ was assumed.

The wirelength results are shown in Figure 4. Overall, the average wirelength is unaffected, even as the number of memories become large. We believe that this is due to the distributed nature of column-based architecture. Distributing resources evenly across a chip means that the terminals of nets can also be distributed as necessary during the placement phase, since there are always enough connections available in the required locality of a net. In [19] this concept is referred to as the structural distribution of a device.

In Figure 4, we also break down the results into nets that only connect to logic clusters (clb-only), nets that only connect to memories (mem-only), and nets that connect to both (mixed). The shapes of each curve are due to the fanout of each type of net, and the impact of our experimental methodology on the fanout. This relationship is an avenue for future research.

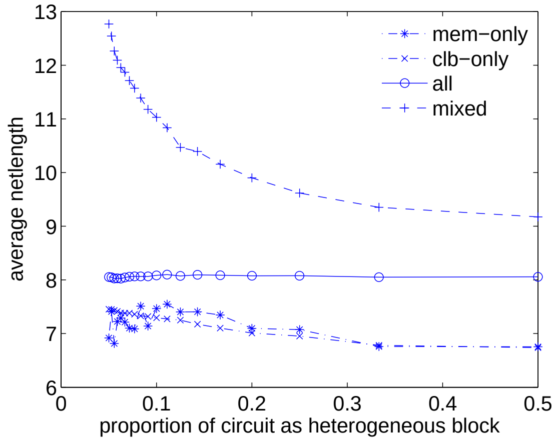

Figure 4: Figure showing the effect of placement constraints on different wire types in a circuit.

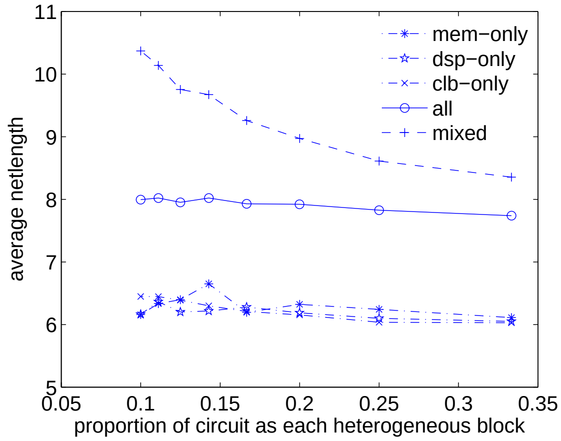

Figure 5: Figure showing the effect of placement constraints on different wire types in a circuit. In this case there are three block types.

We repeated the experiment for two types of embedded blocks (memories and DSPs). As shown in Figure 5, the conclusions from this experiment are the same.

Based on the results of these experiments, we conclude that the placement constraints imposed by the presence of embedded blocks do not have a significant impact on the average wirelength of nets in an FPGA. Thus, we approximate the increase in wirelength due to this factor as 0.

5.2 Dead-Blocks in FPGA Architectures

Dead-blocks are the blocks in an architecture that are left unused. In a heterogeneous FPGA, unused blocks are likely to be common, as it is unlikely that a circuit will use the resources in the precise quantities predetermined by the architecture. As described above, intuitively this will increase the average wirelength. This type of effect has previously been modelled analytically for monolithic circuits [7]. In this section, we derive an analytical model for this effect in heterogeneous circuits and compare to some experimental results.

Assuming that a bounding box of a net in a homogeneous circuit is square with width $\frac{B}{2}$ and height $\frac{B}{2}$, the number of logic blocks within the bounding box is $\frac{B^2}{4}$. If we consider the same net in a heterogeneous FPGA where all architecture blocks have the same size, the net will have a new bounding box of $\frac{B'}{2} \times \frac{B'}{2}$, where the number of *architecture* blocks within the area is $\frac{(B')^2}{4}$. Assuming only one type of heterogeneous block (for clarity, we will refer to it as a memory), occurring once in every H columns, the bounding box in the heterogeneous FPGA will cover $\frac{B'}{2H}$ columns of the memory, which equates to $\frac{(B')^2}{4H}$ sites. By equating the number of logic resources contained within the bounding boxes to be the same, it is possible to derive that the wirelength in the heterogeneous case should increase by the factor in Equation 8, where $l_{c,c}$ represents the relative clb-clb connection length.

$$\frac{B^2}{4} = \frac{(B')^2}{4} - \frac{(B')^2}{4H} \Rightarrow \frac{B'}{B} = l_{c,c} = \frac{1}{\sqrt{1 - \frac{1}{H}}} \tag{8}$$

Considering a circuit that now has a proportion of heterogeneity $c_h = n_h/(n_c + n_h)$ (where n_c is the estimated number of clusters and n_h is defined in Table 1), the net types can be broken down into three categories: clb-clb, memory-memory and mixed nets, where $l_{c,c}$, $l_{m,m}$ and $l_{c,m}$ represent their relative expected wirelengths respectively. Rather than normalising to the homogeneous architecture case, it is more intuitive to normalise each of these terms to the architecture which matches the circuit, since we have empirically shown that wirelength is unaffected in that case. Let us also define the proportion of the architecture grid that is heterogeneous as A_h, where $A_h = 1/H$ when each individual resource occupies the same number of grid cells. The same derivation process as used above can be followed to ascertain the relative netlengths. In the case of clb-clb nets, the number of architecture dead-blocks relative to the ideal circuit proportions is $\frac{A_h - c_h}{1 - c_h}$. The derivation is shown in Equation 9. As an example, in a heterogeneous circuit, when $A_h > c_h$ the relative size of the bounding box will increase, as there is an excess of dead blocks. Similarly, when $A_h < c_h$, the bounding box will decrease as there is an excess of 'active' cells for that net-type within the same bounding area. This leads to the expression for relative netlength of clb-clb connections given in Equation 9. Note that for $c_h = 0$, Equations 8 and 9 are equivalent.

$$\begin{aligned} \frac{B^2}{4} = \frac{(B')^2}{4} - \frac{(B')^2(A_h - c_h)}{4(1 - c_h)} \quad \Rightarrow \\ \frac{B'}{B} = l_{c,c} = \sqrt{\frac{1 - c_h}{1 - A_h}} \end{aligned} \tag{9}$$

By a similar reasoning, the memory nets would increase by the factor in Equation 10, as the proportions of circuit and architecture are interchanged.

$$l_{m,m} = \sqrt{\frac{c_h}{A_h}} \tag{10}$$

We assume that mixed nets only suffer an increase when there is an excess of one particular resource type. This depends on which resource is in excess. When $A_h \leq c_h$, there is an excess of clbs within the bounding box, leading to the same increase as in Equation 9. When $A_h > c_h$ there are too many cells of the heterogeneous resource, leading to the same increase as in Equation 10. Thus, using the same bounding box model, the mixed nets would increase by the factor given in Equation 11.

$$l_{c,m} = \begin{cases} \sqrt{\frac{c_h}{A_h}} & \text{if } A_h \leq c_h \\ \sqrt{\frac{1 - c_h}{1 - A_h}} & \text{if } A_h > c_h \end{cases} \tag{11}$$

A weighted sum of these net types provides the relative change of the expected wirelength,

$$l_d = p_{c,c} l_{c,c} + p_{m,m} l_{m,m} + p_{c,m} l_{c,m} \tag{12}$$

The weights $p_{c,c}$, $p_{m,m}$ and $p_{c,m}$ are based on the proportion of the circuit connections that belong to each net type. In this study, we wish to isolate of the effect of dead-blocks the from the circuit parameters, hence we assume that the connection patterns for all nets have the same statistics. For example if the circuit consists of 25% of a certain block type, then 25% of net terminals in the circuit will be to or from one of these blocks, provided they have an equivalent pin density. Thus, to find the proportions of each type of connection, a binomial distribution can be employed and summed over the all expected net fanouts. Equation 13 gives this quantity for clb-clb nets and Equation 14 gives this quantity for memory-memory nets. The proportion of mixed nets can be obtained simply as the remaining proportion of nets as in Equation 15.

$$p_{c,c} = \sum_{f=1}^{f_{max}} p_f (1 - c_h)^{f+1} \tag{13}$$

$$p_{m,m} = \sum_{f=1}^{f_{max}} p_f (c_h)^{f+1} \tag{14}$$

$$p_{c,m} = 1 - p_{c,c} - p_{m,m} \tag{15}$$

In these equations, p_f represents the proportion of nets with fanout f and is given by Equation 17. It is derived from the expression for the total number of nets of fanout f, N_f according to Equation 16 from [22].

$$N_f = \frac{i n_c \left(f^{p-1} - (p+1)^{p-1}\right)}{f + 1} \tag{16}$$

$$p_f = \frac{N_f}{\sum_{j=1}^{f_{max}} N_{(f=j)}} \tag{17}$$

Figure 6 compares our analytical model to experimental results. Using the technique described in Section 5.1, three different benchmarks sets were created, each set with a different ratio of memories to clusters: (1) no memories, (2) three times as many logic clusters as memories, and (3) seven times as many logic clusters as memories. As in Section 5.1, randomly selected logic clusters in the MCNC circuits were "converted" to memories. The horizontal axis is the repetition distance H, which is a measure of how heterogeneous the architecture is. As the graph shows, the analytical model matches the experimental results well. The model underestimates the average wirelength increase for highly heterogeneous circuits and overestimates for highly homogeneous circuits. We believe this is due to the bounding box model not accounting for changes in the 'vertical' positioning of blocks and because the inherent assumption for mixed-nets, which accounts for expansion due to dead blocks but does not account for contraction of 'active' blocks.

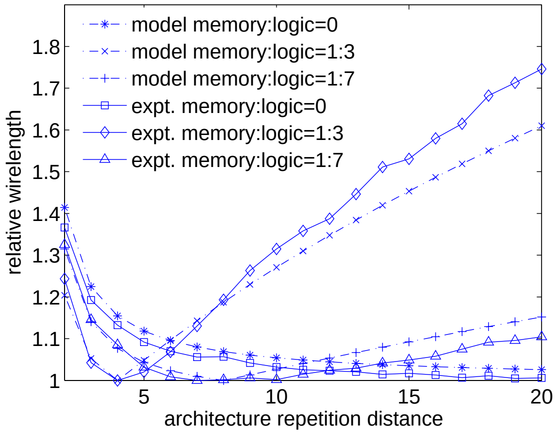

Figure 6: A comparison of relative wirelength estimation techniques considering dead-blocks.

5.3 Module Pins and Wirelength in Heterogeneous FPGAs

In this section, we estimate the expected increase in wirelength due to the fact that embedded blocks typically contain more pins than logic clusters. Intuitively, if a block has more pins, then it has more neighbours. This makes it more difficult for the placement tool to place the block close to its neighbours, possibly leading to an increase in the average wirelength. In this section, we determine how much of an impact this factor has. As in Section 5.1, we have not found a way to calculate this analytically, so we use an experimental approach.

As in Section 5.1, we artificially modify MCNC circuits to create circuits in which one block has more pins than the others. This was done by modifying T-VPACK and using this modified version to create a suite of benchmark sets. Within each set, each circuit contains one block with more pins than all other blocks; the number of pins in the "large" block is the same across all circuits within a set. Since the same base circuits were used to construct each set, the Rent parameters and connection characteristics remained roughly the same, allowing us to isolate the impact of changing the number of pins. Each experiment is performed ten times with randomly chosen iterations of the T-VPack algorithm selected to implement the "large" cluster.

Each benchmark was mapped to a homogeneous FPGA consisting only of logic blocks (the number of pins on each logic block was set to be equal to the number of pins in the "large" cluster). A homogeneous FPGA was used to ensure that there are no artificial placement constraints that might influence our results (although, as shown in Section 5.1, such an impact would be negligible).

We swept the number of pins in the "large" cluster, and measured the average wirelength. The result is shown in Figure 7. Only nets connected to the "large" block are included in Figure 7; the impact on the other nets was negligible. The wirelength of each net is normalized to the length of the same net implemented in the baseline circuit (in which all clusters consist of four 4-LUTs and ten inputs). Since some nets may have pins absorbed into clusters, we only consider nets for which the fanout does not change. This is because fanout is related to wirelength, and may skew any results for low-fanout nets. Another consideration is high fanout nets; since high-fanout nets such as reset signals are likely to connect to most clusters, they will have a large wirelength and have a tendency to hide the effect of wirelength on smaller nets. Hence, we chose to eliminate this effect by only considering nets with a fanout of less than 20. This value was chosen to keep the average fanout equal across the blocks with varying pin densities and equal to the average fanout of all nets in the base-case experiment.

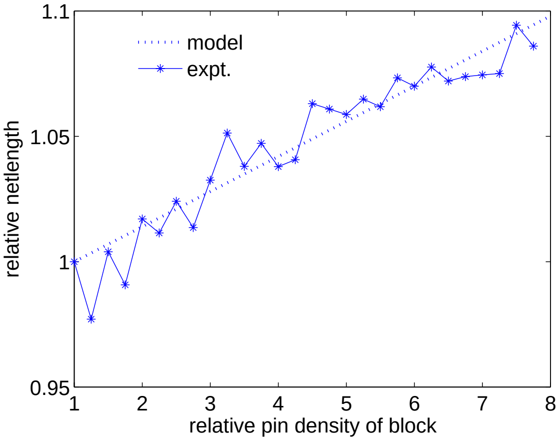

Figure 7: The effect of varying pins of a block on wirelength of the circuit

The horizontal axis in Figure 7 is the ratio of the number of pins in the "large" cluster to the number of pins in the baseline cluster. As the graph shows, the expected wirelength of the nets connected to the cluster does increase as the number of pins increases. The increase appears to level off as the number of pins becomes very large, however, this could be due to the circuit contracting as a result of the number of clusters being marginally reduced.

We assert that the trends from Figure 7 also apply to embedded blocks. Thus, we will use the results from Figure 7 to directly estimate the impact of the number of embedded block pins on the expected wirelength. We have not found a way to analytically model this, however, from the results, we can perform a linear 'best-fit' model to give

$$\text{relative netlength} = 1 + 0.014\,(P_h - 1) \qquad (18)$$

where P_h represents the number of pins on embedded block h relative to the number of pins on a logic cluster.

In terms of the wirelength of the whole circuit, only the nets connected between blocks of differing pin density need to be scaled. This assumption would lead to Equation 19, where l_p represents the relative wirelength of the entire circuit due to the existence of blocks with differing pin densities. $p'_{c,m}$ is calculated in a similar fashion to Section 5.2. In this case the probabilities are calculated as in Equations 13 and 14, where c_h should be replaced by $c_h p_h$, as this represents the proportion of terminals on the device that are connected to the block with higher pin density.

$$l_p = 1 + 0.014 p'_{c,m}\,(n_p - 1) \qquad (19)$$

5.4 Combined Wirelength Model for Heterogeneous FPGAs

Combining the terms in the preceding subsections, we obtain:

$$\begin{aligned} WL &= WL_{homo}\left(1 + 0.014 p'_{c,m}(n_p - 1)\right)\left(p_{c,c} l_{c,c} + \right. \\ &\quad \left. + p_{m,m} l_{m,m} + p_{c,m} l_{c,m}\right) \end{aligned} \tag{20}$$

where WL_{homo} is the expected wirelength in a homogeneous FPGA, derived in Section 4 and $p_{c,c}$, $l_{c,c}$, $p_{m,m}$, $l_{m,m}$, $p_{c,m}$ and $l_{c,m}$ are as derived in Section 5.2.

As described earlier, we do not have enough large heterogeneous benchmark circuits to validate Equation 20 directly. Since circuits vary so widely, point comparison would not be useful. However, since each part of Equation 20 was validated independently, we assert that our model describes the behaviour of the expected wirelength as a function of architecture and circuit parameters.

6. APPLICATIONS OF OUR MODEL

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 homogeneous FPGAs, and the second applies to heterogeneous FPGAs.

6.1 Example 1: Homogeneous Architecture Optimization

In this subsection, we show how the model from Section 4, along with the channel width model from [12], can be used to estimate the number of programming bits in an FPGA as a function of the architectural parameters K, N, and I. Estimating the number of programming bits can lead to a first order approximation of device area, meaning that this study has an interesting significance. The same investigation was performed in [16], however, in that work, it was assumed that the wirelength was constant across all architectures (the same assumption was made in [12]).

We consider three flows:

1. The first flow is purely analytical. We use the model described in [16] to estimate n_c and i as a function of n_2. We then estimate the wirelength using the Davis/Lam model from Section 4. These quantities are then used in conjunction with the channel width model in [12] 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 T-VPack, 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. The third flow is from [16]. This is same as the first flow, except that a constant wirelength of 4.43 is assumed [12].

Figure 8 shows the results for an architecture with $F_s = 9$, $F_{cin} = 20$, $F_{cout} = 4$, and $L = 1$. Figure 8(a) shows the number of programming bits as a function of K, Figure 8(b) shows the number of programming bits as a function of N, and Figure 8(c) shows the number of programming bits as a function of I.

In all cases, the trends observed from Flow 1 (purely analytical) match the trends observed from Flow 2 (purely experimental). This is to be expected, given the accuracy of the models in [12, 16] and Section 4. It is clear that the analytical flow would lead to similar conclusions than would be obtained by the more time-consuming experimental methodology.

Figure 8 also shows that there is a significant difference between the results of Flow 3 and Flow 2. This was observed in [16], and it was suggested that the difference was 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.

6.2 Example 2: Heterogeneous Architecture Optimization

In this subsection we show an example of how the model from Sections 4 and 5 can be used during the development of a domain-specific FPGA with embedded floating point units (FPUs). The speed and density advantage of implementing functionality as hard blocks rather than in FPGA logic is clear. However, the impact on the expected wirelength is less obvious. This could be investigated experimentally, but would require the development of suitable placement and routing tools. In this subsection, we show that we can use our model to provide valuable insight without requiring expensive experimentation. In doing so, we will highlight an important limitation of our model.

In this example, we assume an architecture based on that described in [21]. The embedded FPUs are designed to implement common floating point functions, and are positioned in a vertical column in the fabric. This type of FPU has a relative pin density of 11 and occupies an area of 3x3 clbs. Based on the set of benchmark circuits from Table 2, we assume that our architecture has 8 embedded floating point blocks and 800 clbs.

We used our wirelength model to estimate the average wirelength of each benchmark circuit implemented on our architecture, and the results are shown in Figure 9. For comparison, we also estimated the average wirelength of a circuit that does not use any of the embedded blocks (on our heterogeneous architecture), as well as the average wirelength of a circuit implemented on a homogeneous FPGA without the embedded blocks.

Clearly, the embedded blocks has a significant impact on wirelength. However, these results are primarily due to the mismatch between the logic/embedded block ratio required by the circuit and the logic/embedded block ratio supplied

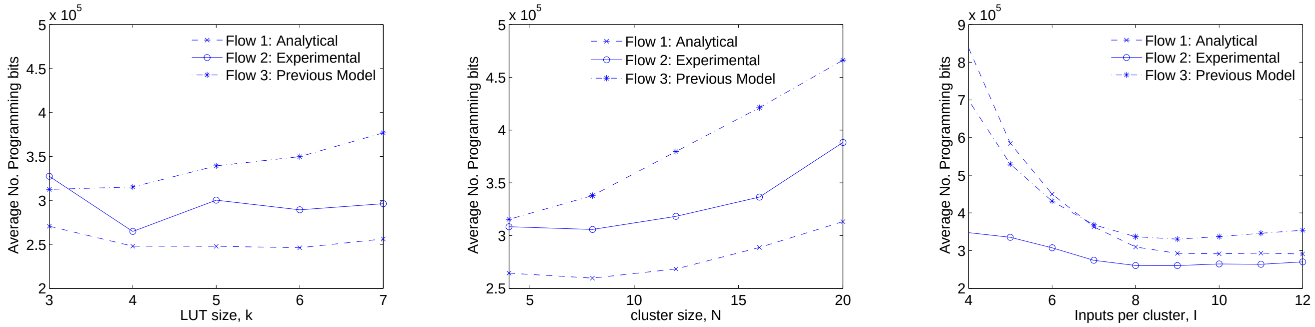

Figure 8: Homogeneous FPGA Architecture Optimization Example

Table 2: Floating point benchmarks

Benchmark:	dscg	bfly	mm3	ode	fir4
No. of clbs	647	790	773	336	180
No. FPUs	8	8	8	8	8

by the architecture. In the *fir4* circuit, for example, the architecture provides many more logic blocks than are required by the circuit. Intuitively, this should not strongly affect the wirelength since a good placement tool would attempt to group the occupied logic blocks tightly around the single column of embedded blocks. The results in Figure 9, however, do not match this intuition; the predicted wirelength for *fir4* is larger than that for all other benchmarks. The reason for this is that the derivation of our model assumed that all embedded blocks are distributed evenly over the fabric. This would tend to "spread out" the logic circuit, thereby increasing the wirelength of all nets. This spreading is more pronounced when the mismatch between the logic/embedded block ratio required by the circuit and the logic/embedded block ratio supplied by the architecture is larger.

This is an important observation and highlights an important limitation of our model. Our model should only be used when the number of embedded blocks is large enough that they can be thought of as being evenly distributed over the fabric. Deriving a model that describes an architecture in which there are only a small number of embedded blocks that are not evenly spread out is an interesting area for future research.

7. CONCLUSIONS

This paper has described a model that relates architecture parameters of an FPGA to the expected average wirelength of circuits implemented on the FPGA. For a homogeneous FPGA, the wirelength is shown to depend primarily on the number of logic blocks and the Rent parameter of the circuit, while for heterogeneous FPGAs, the wirelength also depends on the number, positioning, and number of pins on the embedded blocks.

This wirelength model is a valuable tool for FPGA architects. Understanding the relationship between the architecture, user circuit, and the expected wirelength allows architects to make tradeoffs without requiring expensive experimental investigations. We have shown an example which combines our model with two previous models to relate the area efficiency of an FPGA to basic architecture parameters, as well as an example aimed at understanding the impact of embedding floating point blocks into an FPGA fabric.

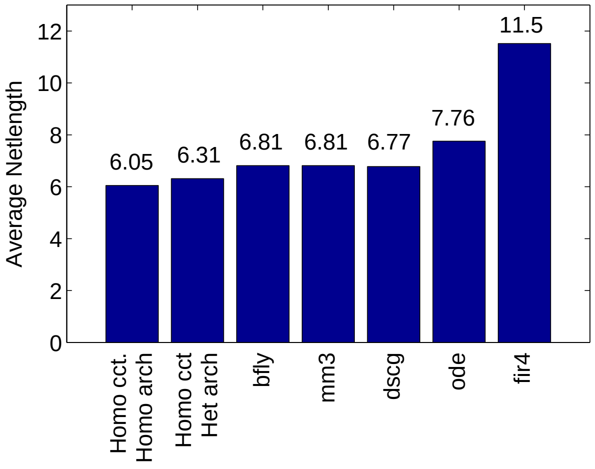

Figure 9: Estimated netlengths for a variety of circuits on an FPGA with floating-point units.

There are a number of limitations of our model. First, our wirelength model assumes that designs are not pad-limited. Pad-limited designs may tend to "spread out" logic somewhat, leading to longer wirelengths (we have observed this experimentally). A method similar to the "dead blocks" model in Section 5.2 may be an appropriate way to model the wirelength of pad-limited designs. A second limitation is that our heterogeneous model assumes a large number of embedded blocks distributed evenly over the array. In Section 6.2, we saw an example of where this could lead to misleading conclusions for architectures with only a small number of embedded blocks. Addressing these limitations is an interesting avenue for future research.

A second interesting area for future research is to analytically model the impact of architectural parameters on the critical path (and hence delay) of a circuit. The average wirelength of nets along the critical path may be less than the average wirelength modeled in this paper, since timing-driven tools will tend to shorten critical path connections at the expense of other connections. Understanding this relationship would be another avenue for future research.

Finally, understanding the relationship between detailed FPGA routing parameters and the average *post-routed* wirelength would be interesting. In a fully flexible architecture, the post-routed wirelength would be similar to the pre-routed wirelength modeled in this paper. However, in typical FPGAs, the detailed routing architecture is far from fully flexible, meaning the post-routed wirelength may be significantly longer than the pre-routed wirelength. Being able to analytically model this effect would be valuable.

8. ACKNOWLEDGMENTS

This work has been funded by the Canadian Commonwealth post-doctoral fellowship programme (Foreign Affairs and International Trade-Government of Canada) and NSERC of Canada.

9. REFERENCES

[1] E. Ahmed and J. Rose. The effect of LUT and cluster size on deep-submicron FPGA performance and density. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 12(3):288–298, Mar. 2004.

[2] 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*, 24(7):1054–1065, Jul. 2005.

[3] V. Betz, W. M. Campbell, T. Fang, P. Jamieson, I. Kuon, J. Luu, A. Marquardt, J. Rose, and A. Ye. VPR 5.0. http://www.eecg.toronto.edu/vpr/.

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

[5] 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 Integrated Circuits and Systems*, 12(12):1827–1838, Dec. 1993.

[6] J. Cong and Y. Ding. Flowmap: An optimal technology mapping algorithm for delay optimization in lookup-table based fpga designs. *IEEE Trans. on Computer Aided Design of Integerated Circuits and Systems*, 13(1):1–12, Jan. 1994.

[7] J. Dambre, D. Stroobandt, and J. V. Campenhout. Toward the accurate prediction of placement wire length distributions in vlsi circuits. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 12(4):339–348, Apr. 2004.

[8] J. Davis, V. De, and J. Meindl. A stochastic wire-length distribution for gigascale integration (GSI). Part I. derivation and validation. *IEEE Trans. on Electron Devices*, 45(3):580–589, Mar. 1998.

[9] J. Davis, V. De, and J. Meindl. A stochastic wire-length distribution for gigascale integration (GSI). Part II. applications to clock frequency, power dissipation, and chip size estimation. *IEEE Trans. on Electron Devices*, 45(3):590–597, Mar. 1998.

[10] W. E. Donath. Wire length distribution for placements of computer logic. *IBM Journal of Research Development*, 2(3):152–155, May 1981.

[11] A. A. El Gamal. Two-dimensional stochastic models for interconnections in master-slice integrated circuits. *IEEE Trans. on Circuits and Systems*, 26(4):127–138, Feb. 1981.

[12] W. Fang and J. Rose. Modeling FPGA routing demand in early-stage architecture development. In *Int'l Symp. on Field-Programmable Gate Arrays*, pages 139–148, Feb. 2008.

[13] M. Feuer. Connectivity of random logic. *IEEE Trans. on Computers*, CAS-26(4):29–33, Jan. 1982.

[14] M. Hutton. FPGA architecture design methodology. In *Int'l Conf. on Field-Programmable Logic and Applications*, page 1, Aug. 2006.

[15] F. K. Hwang. On Steiner minimal trees with rectilinear distance. *SIAM Journal on Applied Mathematics*, 30(1):105–114, Jan. 1976.

[16] A. Lam, S. J. Wilton, P. Leong, and W. Luk. An analytical model describing the relationships between logic architecture and fpga density. In *Int'l Conf. on Field-Programmable Logic and Applications*, Sep. 2008.

[17] B. Landman and R. Russo. On a pin versus block relationship for partitions of logic graphs. *IEEE Trans. on Computers*, C-20(12):1469–1479, Dec. 1971.

[18] J. Pistorius and M. Hutton. Placement rent exponent calculation methods, temporal behaviour and fpga architecture exploration. In *System-Level Interconnect Prediction (SLIP)*, pages 31–38, Apr. 2003.

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

[20] S. Wilton, J. Rose, and Z. Vranesic. The memory/logic interface in fpga's with large embedded memory arrays. *IEEE Trans. on Very-Large Scale Integration Systems*, 7(1), Mar. 1999.

[21] C. W. Yu, J. Lamoureux, S. J. Wilton, P. H. Leong, and W. Luk. The coarse-grained / fine-grained logic interface in fpgas with embedded floating-point arithmetic units. *Southern Conf. on Programmable Logic*, pages 63–68, Mar. 2008.

[22] P. Zarkesh-Ha, J. A. Davis, W. Loh, and J. D. Meindl. Prediction of interconnect fan-out distribution using rent's rule. In *System-Level Interconnect Prediction (SLIP)*, pages 107–112, 2000.

[23] P. Zarkesh-Ha, J. A. Davis, and J. D. Meindl. Prediction of net-length distribution for global interconnects in a heterogeneous system-on-a-chip. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 8(6):649–659, Dec. 2000.