International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 11 (2015) pp. 28889-28902

http://www.ripublication.com

Early Stage FPGA Architecture Development By Exploiting Dependence on Logic Density

Prem Lal Paleri
VLSI Design, Department of ECE
VLSI Design, Testing & Security TAG
Amrita Vishwa Vidyapeetham, Coimbatore, India
prem.110190@gmail.com
Ramesh. S R
Assistant Professor(Senior Grade)
Department of ECE, VLSI Design, Testing & Security TAG
Amrita Vishwa Vidyapeetham, Coimbatore, India
srrameshonline@gmail.com

Abstract

FPGA technology owes its rate of advancement to the tremendous amount of research work directed into the FPGA architecture development. There has been a constant striving for FPGA performance improvement from designers through innovative designs. For the purpose of performance enhancement evaluation due to the incorporation of a certain modification, say to the logic block or the routing fabric, a deep insight into the effects of such modification on the performance is needed. At present, this evaluation is carried out through computer aided design simulations which are labor-intensive and computationally-expensive experiments. A more scientific and rational method, based on the insight into the dependency of performance on architectural parameters makes the evaluation of new architectures more rapid and can be done even before a CAD tool is developed. This paper presents a comparison between the results obtained using the conventional CAD flow method as well as using an analytical model which describes relationships between architecture and logic. The model is based fundamentally on Rent's rule. In particular, the model draws a relationship between architectural parameters and the area efficiency of an FPGA. The simplicity of the model's equations renders it an effective tool for FPGAs architects to better comprehend and usher the development of modern FPGA architectures.

Keywords: Analytical Modeling,Field Programmable Gate Arrays,logic density,early stage architecture evaluation.

Introduction

A field-programmable gate array (FPGA) is an integrated circuit (IC) containing a reprogrammable logic array and an interconnecting routing fabric that can implement virtually any digital logic application. The ability to program, erase, and reprogram the reconfigurable logic and flexible routing fabric make FPGAs suitable for rapid testing and prototyping of digital logic designs. Unlike application specific integrated circuits (ASICs), FPGAs do not require the time and cost intensive process of custom designing and manufacturing of masks. Therefore, the utilization of FPGAs leads to a reduction in the design cycle time and upfront costs compared to ASICs.

However, the conveniences of pre-manufactured silicon come at a price. FPGAs have lower performance than ASICs that are custom designed at the transistor level and optimized for clock speed, logic density, and power consumption. The exact performance gap between the two technologies is application specific, but on average FPGAs are more than 40 times larger, 3.2 times slower, and 12 times more power (dynamic) hungry than ASICs. That being said, many digital logic designs do not demand high-end performance. In these cases, FPGAs are a cost-efficient choice for many low to medium volume products that require quick design turn-around times and time-to-market.

Since their introduction, FPGAs have evolved considerably in an attempt to reduce the performance gap and capture more of the higher-end IC market. These advancements can be categorized and attributed to the development in three fields or research: transistor technology, FPGA computer-aided design (CAD) tools, and FPGA architectures. First, advancements in transistor technology have led to a remarkable hike in the number of available transistors and a radical change in the functionality of FPGAs. At their inception, FPGAs could implement just over 1000 gates. Today's FPGAs can implement over a million gates, large enough to implement complete digital system designs.

With the size of present day FPGAs, it is unrealistic for the user to manually program each individual programming bit. Instead, a digital logic circuit is typically described at a high-level of abstraction, usually in the form of a hardware-description language (HDL), that is then compiled using a CAD tool. The CAD tool interprets the HDL and determines a mapping of the appropriate values for the FPGA's programming bits to implement the described digital circuit. Typically multiple mappings exist for the same circuit, some yielding higher performance than others. Much research has gone into incorporating models for metrics such as area, delay, and power into the CAD algorithms. These models provide the algorithm estimated metric values and enables the CAD tool to make informed decisions when generating mappings. This allows the CAD tool to create an optimized mapping based on one or a combination of the metrics listed.

Whatever improvement the FPGA technology has seen in the recent history of its development, almost all can be attributed to the progress made in the field of FPGA architecture research. By architecture we refer to the basic building structure of the fundamental blocks as well as the interconnections between them. For instance, in the case of early FPGA devices, the implementation of logic was carried out using LUTs of size 4, whereas with modern ones more flexibility is available i.e. LUTs are more

complex and fracturable, so that they possess the capacity to be used in a variety of modes.

In the designing phase of a novel FPGA device, each and every enhancement done to the architecture is assessed and evaluated carefully. An experimental procedure is used to carry out such an evaluation process. Experimental CAD tools are used to map a set of benchmarks to the newly modeled architecture, which possesses some enhancement.

The process involving CAD tools mentioned above could be quite sluggish at times. It requires a large number of benchmark circuits to exercise architecture properly. Suppose the number of benchmarks used is not sufficient, architectures tuned for specific circuits can be created rather than architectures meant for a broader span of customers. Researchers in academia generally use about 20 benchmark circuits, whereas in industry, a lot more circuits have to be included. In the experimental approach, each of these benchmark circuits has to be mapped to all potential variants of the architecture under investigation. Modern CAD tools can take a lot of time to compile as its algorithm searches for the optimum mapping. This sluggish process restricts the number of architectures which can explored as alternatives, consequently restricting FPGA companies from exploring novel structures that may result in FPGAs that are more efficient.

Investigation of FPGA architectures can be sped up using analytical models which depict some aspect of architecture. Analytical models relate parameters depicting FPGA architecture to area, delay, or power efficiency of an FPGA. Such analytical models generally assume the guise of easy to understand expressions, and thus looking for competent architectures can be fast and does not require experiments which consume a lot of time.

The models such as mentioned above can be exploited to speed up the architectural investigation process in two ways. Firstly, acquiring an insight into the relationships between architectural parameters makes possible the searching of the design space really quick leading to development of architecture at an early stage. The conventional experimental methods can be used to select exact parameters once an assuring region of design space is isolated. This would considerably speed up the process of FPGA architecture design. Second, as a result of developing such theory, researchers will be encouraged to look for the reasons because of which some architectures perform exceptionally well.

The primary aim of this work is to shape and assess an analytical model that depicts the association between the logic block architecture and the efficiency, in terms of area, of the resulting FPGA. This model must balance complexity and accuracy. In general, simple equations furnish considerable understanding of tradeoffs involved in the architecture than needlessly complicated mathematical expressions. On the other hand, the model must be exact enough that it leads to useful conclusions when investigating new architectures. Another contribution is to predict the extent of resource utilization (LUT, in particular) when circuits are implemented in different FPGA architectures.

Related Work

There have been quite a few numbers of works which have examined the dependency of the performance of FPGA implementation on the architectural parameters of the FPGA. A lot of works are available which concentrate on the routing fabric. A model for non-programmable chip which connects area needed for routing to the number of pins in the logic gates was derived by El-Gamal [2]. Design of several generations of FPGAs have used this model [3].Another significant contribution is of Brown et al., which draws a relationship between the different parameters of FPGA routing architecture and the possibility of that architecture to get routed [4].A more recent work, by Fang and Rose, relates the same parameters to the channel width of an FPGA [5]. Pistorius and Hutton showed bearing of Rent's parameter (Rent's parameter is a quantity to fathom the complexity of the pattern of interconnection inside a circuit [6]) of a circuit on the different parameters of architecture.

There have been many works concerning interconnect and estimation of wire length. Much of our work has its fundamental grounds in Rent's rule, which shows the bearing of the number of available pins in a module of a circuit on the number of fundamental blocks present within a module [6].

An early work by Donath et al. [7] gives a relation between the requirement of area by routing wires and Rent's parameter of a particular circuit.

Stroobandt refines these models so that they could be considered for more realistic architectural assumptions and network topologies [8]. Balachandran and Bhatia use information from the circuit to estimate wire length and interconnect for island-style FPGAs [9].

A model for estimation of post-layout wire length for homogenous as well as heterogeneous architectures of FPGA was presented by Manohararajan [10], which utilizes a look-up table with values of interconnect delay that are pre-recorded, and which are a function of architectural parameters.

A more relevant work when compared to our work is by Gao et al., which presents a bearing of LUT size on area as well as depth of forming N-LUTs in the case of a non-clustered FPGA[11].

Our work draws heavily from the work by Joydip Das et al. [1] which basically derive a model that relates FPGA architectural parameters to the logic size and depth of an implemented FPGA.

Framework

In this section, the assumptions regarding the framework of the architecture and the circuits which have been realized are described. Also the parameters that are used in our model are introduced.

A. Suppositions and ushering principles

Previous works have showed that the models meant for homogenous architectures, are also applicable to heterogeneous architectures as well, with some modifications [6].In our work we consider homogenous architecture, wherein logic cluster or configurable logic block, is made up of N elements of K-input LUTs.

There are three basic philosophies which have guided the development of the model proposed by Das et al. [1]. First, an endeavor has been made to develop the model by drawing relationships analytically, rather than by depending on the experimental techniques or curve-fitting. This makes sure that we capture the quintessence of the programmable logic, rather than constructing a model which is restricted exclusively to a certain experimental CAD tool flow-suite.

Secondly, the model has been developed in such a manner that it is least dependent on the circuit to be implemented on the FPGA. In this aspect our paper is fundamentally different from the previous work of estimation, where in the primary aim was to guess the area, power or speed for a certain specified circuit.

Though we consider all the above mentioned principles, it is however impossible to do away with the specifications of the circuits as a whole. Hence we capture the nature of the circuits using the following parameters: i.) Rent's Parameter ii.) Size of the un-tech mapped circuit. Both the mentioned parameters can be accessed at a quite early phase.

Thirdly, an attempt has been made to compensate for the complexity with precision. The simple equations of the model used by us furnish better insights into the trade-offs in the architecture than unwanted complicated expressions.

Insights of this nature help the designers to have a control on the fine-tuning of the architecture under evaluation.

B. Parameters of the model

The parameters which depict the architecture, circuits and the implementation are given in table I. In the table, the upper-case letters are representatives of the parameters of the architecture, and the lower-case letters stand for the parameters of the circuits, as well as tech parameters which depict the realization of a circuit on a certain specified architecture. Because our model describes the parameters of pre-routing implementation, the parameters which are concerned with detailed routing fabric have been excluded. A detailed description and derivation of the model has been described in [1].

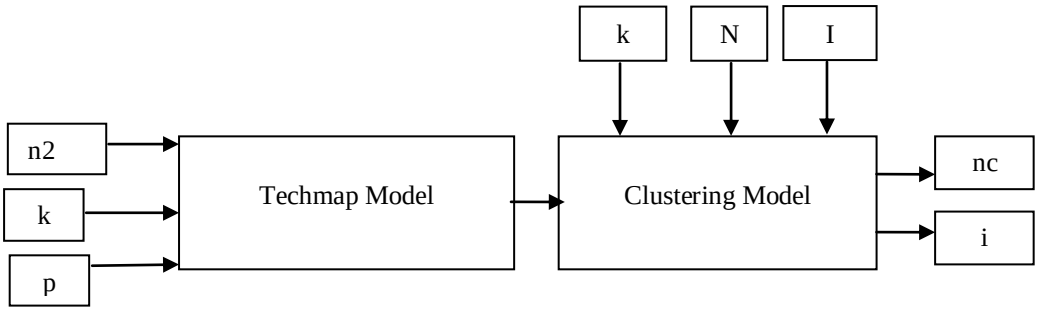

Figure 1: Model of the amount of clusters, number of LUTs and used inputs in each cluster

Table 1: Parameters of the Model

Model Inputs:	
Parameters of architecture:	
N	Number of LUTs per cluster
K	Number of inputs per LUT
I	Number of inputs per cluster
Parameters of circuit:	
n_2	Number of 2-input LUTs in a given circuit
p	Rent Parameter of a circuit

Model Outputs:	
Parameters used for implementation:	
n_k	Number of K-LUTs needed for implementation
n_c	Number of clusters needed for implementation
i	Average number of inputs used in each cluster
f_{avg}	Average fan-out of all nets in the circuit
o	Average number of outputs used in each cluster
d_k	Depth after technology mapping
d_c	Depth after clustering

Methodology

A varied number of classes of FPGA architectures have been proposed in literature including island-style, hierarchical and row-based [12].

Due to the extensive use in industry as well as its relevance in academia, we target our work on island style architecture. Island-style architectures are implemented, commercially on the FPGAs provided by the two largest players, Xilinx and Altera, which are at present, in the vanguard of technological development of FPGAs.

We intend to follow the conventional experimental flow used for FPGA design using MCNC benchmark circuits. This experimental procedure is carried out using CAD tools. Then the analytical modeling method described in [1] is carried out. This model is basically a set of mathematical equations which mirrors the experimental method .The results from both the methods will then be compared to examine whether the time consuming experimental CAD flow can be replaced with the model.

C. FPGA CAD FLOW

Fig. 2 shows a typical FPGA CAD flow which comprises five stages: design entry, technology mapping, packing (clustering), placement and routing.

Design Entry:

The first step in the CAD flow is design entry. At this stage, the user specifies the circuit that will be realized on an FPGA. The methods of design entry can be broadly categorized into two: programming language based techniques and schematic capture based techniques [13]. The schematic capture based techniques generally use a

dataflow paradigm. Programming language based approach typically uses Verilog, VHDL, C and MATLAB-based programs [14].

Technology Mapping:
This phase optimizes the net-list and maps the same into LUTs which forms the fundamental building block of a Basic Logic Element (BLE).There are many existing works which focus on the various aspects of technology mapping. Different tools performing technology mapping target area required for implementation, routability, consumption of power and delay(depth).There have been many studies which focus exclusively on technology mapping alone [15,16].

Clustering (Packing):
The Basic Logic Elements (BLEs) which arrive after performing technology mapping phase are packed closely to form clusters, in the clustering phase. A lot of study has gone into research and towards proposing techniques for clustering to optimize density [17, 18], speed [18, 19], power [20, 21] and routability [21] of FPGA implementation.

Placement:
The placement phase of the CAD flow arranges the clusters from the packing phase on FPGA physical location. There are three primary classes of algorithms for placement which are used in FPGA domain:

a) Partition based min-cut approaches
b) Analytical approach followed by local iterative improvement
c) Simulated annealing based approaches

In order to improve the resulting placement solutions, various works have tried including early predictions for post-routing wirelength, delay and routability into placement algorithms [9, 10].

Routing
Routing phase decides which switches need be turned on so that all the required outputs as well as input pins of the cluster are connected [18].

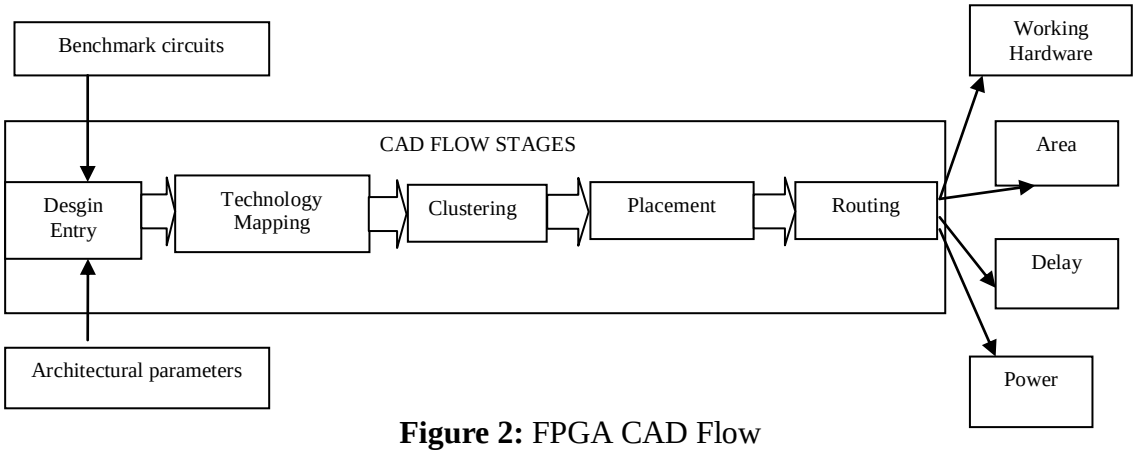

Figure 2: FPGA CAD Flow

Since routing delay is a major contributor towards the overall delay in the FPGA implementation, algorithms which are especially timing driven are preferred over the ones which are time oblivious [18].

D. Model Verification

In this section we evaluate the accuracy of the analytical model. We employed an experimental approach. Experimental data was generated by synthesizing and compiling benchmark circuits using FPGA CAD flow and then compared against the model's estimations.

The benchmark circuits used in this study are from Microelectronics Circuits of North Carolina (MCNC).The names of those circuits, along with their size in 2-LUTs, and Rent's Parameter is shown in the table II.

The Gortian Partitioner [22] has been used to measure the Rent's Parameter of each circuit. Then the partitioner was applied recursively to each and every circuit, and the Rent’s Parameters were obtained after every iteration.

The final value of the Rent's Parameter was calculated as the arithmetic average of the measured Rent's parameter at all but two levels. The Rent's parameter measured during the first two levels are often influenced by I/O restrictions caused by packaging, and are omitted to ensure we capture the circuits 'natural' Rent's parameter. This Rent's parameter as well as the number of 2-input gates in each circuit were then used as inputs to the analytical model implemented in 'C'.

The model predictions were compared to results obtained from an experimental flow as mentioned previously. In this experimental flow, the two-input gates in each circuit were first packed into look-up tables (this process is referred to as technology mapping).We employed the ABC tool for technology mapping which uses the Flowmap algorithm [23].After the circuits get technology mapped we counted the number of lookup-tables required to implement each circuit, and compared this to the results from the analytical model equations presented in publication [1].The comparison is presented in the Results and Analysis section.

After technology mapping, each technology mapped circuit is packed into logic blocks (this process is known as clustering).Here the tool we use is T-Vpack[24] which uses the so called T-Vpack algorithm. After clustering, we examine how many clusters are needed to realize each circuit and compare this to that predicted by equations in the publication by Das et al [1].These comparisons are presented in the section V.

Finally, we employed a multi-level clustering algorithm to cluster the lower-level clusters into higher -level clusters. Since a multi-level clustering algorithm was not immediately available, the T-VPack algorithm was modified to perform recursive clustering for multi-level architectures and was named MT-Vpack.

MT-Vpack shares T-Vpack's core algorithm, a two -step algorithm that 1) selects a seed logic block (LB) for the new cluster, 2)then greedily packs subsequent LBs based on an attraction function. The attraction function is based on two components, a shared net and critical path cost function. The main difference between the two programs is in the calculation of the two functions between the unpacked LBs and the new cluster. Differences between the cost function calculations originate from the additional output

connections which need to be considered when packing multiple output low-level clusters in MT-Vpack in contrast to single output BLEs in T-Vpack.

With the exact quality of MT-Vpack's heuristics undetermined, the use of this clustering tool in our verification may be questionable. However, in the absence of a fully evaluated multi-level clustering tool, MT-Vpack fills our verification needs. Though MT-Vpack is not an optimal clusterer, it is based on the ubiquitous single-level clusterer, T-Vpack, and it's expected to yield reasonable data with similar trends. In verification, the model's tracking accuracy is essential to its use in architecture evaluation. Accurate and absolute value estimates of optimally packed cluster are of secondary importance.

Results and Analysis

In this section we present the results obtained as a result of performing experimental CAD flow as well as analytical modeling approach on the MCNC benchmark circuits. Table II shows the list of benchmarks used.

We have tabulated results obtained using eight randomly selected benchmark circuits among the twenty, though we have got the similar responses from the remaining circuits as well. Table III shows the data collected when the LUT size is set as 4, cluster size is 10 and number of inputs available to each cluster is 22.We observe that the number of LUTs furnished after both the approaches almost coincide, though remains a deviation which could be ignored from the point of view of the purpose served i.e. early evaluation of the architecture. Same observation holds good for both the number of clusters as well as the average inputs used per cluster. Similarly, table IV shows the data obtained with a slight variation in the architecture i.e. LUT size is set as 7 in this case, other parameters remaining the same. Experiments were performed with other LUT sizes as well, but strangely, when the LUT size is set as 7 the analytical model is found to follow the experimental CAD flow results much more closely than any other combination.

The reason for proximity in the results at this LUT size is that, while LUT size is set as 7 after the connections are by design made local, T-VPACK depends on randomly absorbing the connections for all the remaining connections which are left out.

Table 2: List of Benchmark Circuits

Name	Size (No. of 2-input gates)	Rent's parameter
MCNC BENCHMARK CIRCUITS		
s38417	13307	0.5883
ex1010	8020	0.6623
pdc	8408	0.7341
spla	7348	0.6908
frisc	6002	0.6557
elliptic	5464	0.6347
bigkey	2979	0.5442
dsip	2531	0.5693
S298	4268	0.5518
des	2901	0.6141
apex2	3165	0.6659
seq	2939	0.7025
diffeq	2544	0.5777
alu4	2732	0.6473
apex4	2196	0.7160
tseng	1858	0.5836
misex3	2557	0.6871
ex5p	1779	0.7236
i10	1668	0.6378
C6288	1820	0.6378

Table 3: Comparison of Results Obtained From Experimental Flow As Well As Analytical Flow (K=4, I=22, N=10)

Benchmark circuits	No. of LUTs		% error	No. of clusters		% error	Average no. of inputs used per cluster		% error
	Experi-mental	Analytical		Experi-mental	Analytical		Experi-mental	Analytical	
alu4	1151	1464.009	27.2	118	146.4009	24.06	19.016	14.880	21.7
apex2	1331	1016.577	23.6	138	101.577	26.3	19.985	17.74	11.23
apex4	1076	1254.86	16.6	112	125.486	12.04	19.116	17.79	6.9
des	1375	1530.735	11.3	138	153.073	10.9	17.391	14.305	17.74
ex5p	887	1015.72	14.5	91	101.57	11.6	17.87	18.2817	2.3
ex1010	4012	4624.91	15.27	425	462.491	8.8	19.101	19.0897	.05
misex3	3122	3473.59	11.2	398	347.35	12.7	16.56	14.76	10.8
pdc	2901	2823.77	2.6	266	282.377	6.1	18.56	15.78	14.9
seq	1313	1101.432	16.11	137	110.143	19.6	19.197	17.26	10.09

Also, we implemented 4 of the 20 large MCNC circuits on different FPGA boards belonging to different families. We could observe that the resource utilization in terms of number of LUTs is varying with differing families. This variation can definitely be attributed to the architectural diversity of the families used. This suggests that there is always a scope for improvement in the performance metrics of FPGA by assigning the parameters of architecture in an optimal manner. The optimization of this kind is a hot research area wherein optimal algorithms are the most sought after results. Figure 3 shows the obtained results.

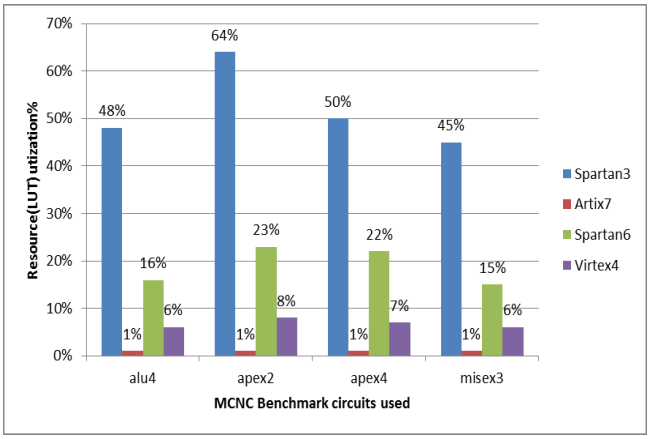

Figure 3: Figure showing resource utilization of resources using different families as well as circuits

Table 4: Comparison of Results Obtained From Experimental Flow As Well As Analytical Flow (K=7, I=22, N=10)

Benchmark circuits	No. of LUTs		% error	No. of clusters		% error	Average no. of inputs used per cluster		% error
	Experi-mental	Analytical		Experi-mental	Analytical		Experi-mental	Analytical	
alu4	799.56	799.934	0.046	79	79.934	.011	21.99	22	0.04
apex2	1014.14	1058.805	4	120	138.88	15.7	22	22	0
apex4	723.60	729.210	0.7	88	92.63	5.26	20.95	22	5
des	820.75	823.347	0.316	80	68.53	14.33	22	21.006	0
ex5p	583.21	590.74	1.2	68	73.995	8.8	22	22	0
ex1010	2695.67	2706.615	0.406	369	375.510	1.76	22	22	0
misex3	796.23	818.197	2.75	95	97.054	2.16	22	22	0
pdc	2820.25	2833.441	0.467	373	391.73	5.02	22	22	0
seq	947.62	950.892	0.345	112	116.026	3.5	22	22	0

Conclusions

This paper basically does a comparison between the results obtained by two methods of performing early stage FPGA architecture evaluation. The two methods are viz. experimental CAD flow and analytical modeling. The experimental method is the conventional way which has been there for quite some time and generally used in the industry, but its time consuming as well as computation intensive. Analytical modeling is a suggested alternative method [1] which can speed up the evaluation process to a great extent. The model also gives insights into the relationship between architecture, circuit and the expected logic density. Insights of this kind will help the designers to go for architecture trade-offs without performing experimental investigations.

However, the model is not devoid of limitations. The model works fine for a sufficiently large number of cluster inputs, but when the number of cluster inputs is small, the model doesn't quite follow the experimental values. An exhaustive analysis to predict the architecture density even with small number of inputs per cluster would be a potential area of research.

References

[1] Das, Joydip ,Andrew Lam, Steven J . E .Wilton, Philip Leong and Wayne Luk, "An Analytical model describing relationships between logic architecture and FPGA density" in *IEEE Trans. in VLSI systems*, vol. 19 NO.12, December 2011

[2] A. A. El Gamal, "Two-dimenional stochastic models for interconnections in master-slice integrated circuits,"*IEEE Trans. Circuits syst.*, vol. 26, no. 4, pp. 127-138, Feb.1981.

[3] J.Rose, "Closing the gap between FPGAs and ASICs, keynote address",*in Proc. Int. Conf. FPT* ,Dec.2006, p. viii.

[4] S. Brown, J. Rose, and Z. Vranesic, "A stochastic model to predict the routabilty of field-programmable gate arrays", *IEEE Trans. CAD Circuits Syst.*, vol. 12, no. 12

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

[6] B. Landman and R. Russo, "On a pin versus block relationship for partitions of logic graphs," *IEEE Trans. Comput.* vol.C-20,pp.1469,1971.

[7] W. Donath, "Placement and average interconnect lengths of computer logic," *IEEE Trans. Circuits Syst*., vol.226, no. 4, pp. 272-277,Apr.1979

[8] D. Stroobandt,*A Priori Wire Length Estimates for Digital Design*.Norwell, MA:Kluwer,2001

[9] S. Balachandran and D. Bhatia, "A priori wirelength estimation and interconnect estimation based on circuits characteristics," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 7, pp.1054-1065, Jul. 2005.

[10] V. Manohararajah, G. Chiu, D. Singh, and S. Brown, "Predicting interconnect delay for physical synthesis in FPGA CAD flow*," IEEE Trans. VLSI Syst.*,vol. 15, no. 8, pp. 895-903, Aug. 2007

[11] 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.

[12] S. D. Brown, J. Rose, and Z. Vranesic, "*Field Programmable Gate Arrays*", Kluwer Academic Publishers, 1992

[13] A. M. Smith, "*Heterogeneous Reconfigurable Architecture Design:An optimisation approach*" PhD thesis,Imperial College, United Kingdom, 2006

[14] A. Nayak, M. Haldar, A.Choudhary and P. Banerjee, "Accurate area and delay estimators for FPGAs" , *Proc. Conf. of Des., Auto. And test*, pp. 862, IEEE computer society

[15] Berkley Logic Synthesis and verification Group, "*ABC: A system for sequential synthesis and verification*",December 2005

[16] R. K. Brayton, G. D. Hatchel, and A. L. Sangiovanni-Vincentelli, "Multilevel logic synthesis", *Proc. Of the IEEE*, 78(2): 264-300, Feb 1990. ISSB 0018-9219

[17] J. F. Beetem, "Rebel:A clustering algorithm for LUT FPGAs", CAD of Integrated Circuits and Systems,*IEEE Trans.* On,17(5):444-451,1998

[18] V. Betz, J. Rose and Marquardt, "*Architecture and CAD for deep-submicron FPGAs*", Kluwer Academic Publishers,1999

[19] J. Cong, J. Peck and Y. Ding, "RASP:A general logic synthesis system for SRAM-based FPGAs", In PGA '96:*Proc. Of the ACM/SIGDA Int'l Symp. On FPGAs*, pp.137-143,1996

[20] K. K. W. Poon, S. J. E. Wilton, and A. Yan, "A Detailed power model for FPGAs", *Design Automation of Electronic Systems.ACM Trans.* On, 10(2):279-302,2005

[21] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in FPGAs.In *Proc. Of the 2002 ACM/SIGDA tenth international Symp. on FPGAs*,FPGA '02,pp.-59-66,2002.

[22] Marcel Gort, "Gortian Partitioner (ece583: Assignment 3). ECE583, 2007

[23] J. Cong and Yuzheng Ding, "FlowMap:An optimal technology mapping algorithm for delay optimization in LUT based FPGAs", *CAD of Integrated Circuits and Systems, IEEE Trans*. On, 13(1):1-12,Jan. 1994

[24] Alexander (Sandy) Marquardt, Vaughn Betz, and Jonathan Rose, "Using cluster-based logiz blocks and timing –driven packing to improve FPGA speed and density", In FPGA '99*: Proc. Of the 1999 ACM/SIGDA seventh international symposium on FPGAs*,pp. 37-46, New York, USA