fGREP - Fast Generic Routing Demand Estimation for Placed FPGA Circuits

Parivallal Kannan, Shankar Balachandran, and Dinesh Bhatia

Center for Integrated Circuits and Systems
Erik Jonsson School of Engineering and Computer Science
University of Texas at Dallas
PO Box 830688, Richardson, TX 75083, USA
{parik, shankars, dinesh}@utdallas.edu

Abstract. Interconnection planning is becoming an important design issue for large FPGA based designs and ASICs. One of the most important issues for planning interconnection is the ability to predict the routability of a given design. In this paper, we introduce a new methodology, fGREP, for *ultra-fast* estimation of routing demands for placed circuits on FPGAs. Our method uses logic block fanout as a measure of available routing alternatives for routing a net. Experimental results on a large set of benchmark examples show that our predictions closely match with the detailed routing results of a well known router, namely VPR[1]. fGREP is simultaneously able to predict the peak routing demand (channel width) and the routing demands for every routing channel. It is currently used for post-placement estimation of routing demands, but can be used during the placement process also. fGREP can be used with any standard FPGA place and route flow.

1 Introduction

Routing of nets for FPGAs is a hard and very time consuming task. In commercial CAD tools, the majority of the design time is spent performing routing of nets. With changing size and complexity of FPGA devices, CAD tools are faced with challenges related to good convergence of mapped designs in an acceptable time frame. Most complex FPGAs have more than a million gates and the complexity of the designs is constantly rising. With such enormous sizes, the problem related to FPGA based design are no different from those in the custom and semi-custom ASIC arena, i.e. confidence of routability, good performance, reasonable total mapping time and more. Prediction of wiring requirements and managing overall CAD to ensure wireability of a circuit is one problem that demands the most attention. In this paper, we have studied the problem of interconnection prediction for large FPGAs. Although our method demands a fully placed circuit as an input to our estimation techniques, it can be very easily embedded within a placement tool for optimizing routability based cost during placement iterations. The execution time overheads of our method are very low. Thus our method can be used post-placement to ensure routability of the placed circuit prior to actual routing.

G. Brebner and R. Woods (Eds.): FPL 2001, LNCS 2147, pp. 37–47, 2001.

2 Prior Work

Interconnection prediction and related problems are being actively investigated for various design styles and technologies. Most of the recent literature has addressed interconnection prediction keeping Rent's rule[2] in consideration. Rent's rule establishes an empirical relationship between the *number of pins* N_p and the *number of logic blocks* N_g in a logic design. It shows that a log-log plot between the two parameters form a straight line and empirically yield a relationship

$$N_p = K_p {N_g}^{\beta} \tag{1}$$

Here, β is the Rent's constant, and K_p is a proportionality constant. K_p is also the average number of interconnections per block. Van Marck et. al. [3] used Rent's rule to describe local variations in interconnect complexity. Sadowska et. al. [4] used Van Marck's results and modified the VPR placement cost function (originally linear wirelength based) to account for interconnection complexity. Wei [5] has used Rent's rule to arrive at a statistical model for predicting routability for hierarchical FPGAs prior to placement.

El Gamal [6] proposed a stochastic model for estimating the channel densities in mask programmable gate arrays. The model assumes a normal distribution of interconnection within channels. Brown [7] et. al. extended this model by taking into account the FPGA routing architecture and various flexibilities associated with programmable switching elements. Both the models predict routing resource requirements in the post placement stage of design. Although our work addresses the prediction of routing resources in the post placement stage, it avoids complex evaluation of conditional probabilities, thus making the entire evaluation very fast. Wood [8] et. al. used boolean satisfiability with BDDs to estimate routability for FPGAs. The representation of routing channels as a satisfiability problem makes the entire process very cumbersome and difficult to evaluate. The time spent in routability estimation sometimes exceeds the time required to perform incremental routing. This makes the model very impractical for even small size problems.

Some other works that indirectly address the routing resource prediction or evaluation include the congestion minimization techniques due to Wang and Sarrafzadeh[9], simultaneous place and route by Nag and Rutenbar[10], and wireability analysis for gate arrays by Sastry and Parker[11].

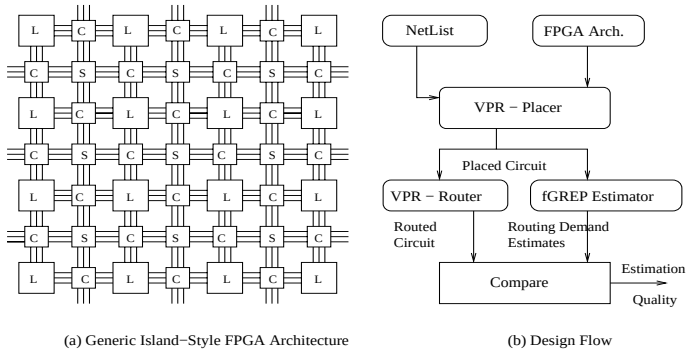
In general, routing estimation methods should be very fast, have high accuracy, be capable of predicting both global and local routing requirements, and conform as much as possible to actual routed results from standard routers. In the following sections we explain fGREP, our new routing demand estimation methodology, and show that the estimates produced by fGREP are very close to the actual *detailed routes* produced by VPR's router.

3 Preliminaries

3.1 FPGA Physical Design Tool: VPR

VPR is a well known FPGA physical design tool suite, capable of targeting a broad range of FPGA architectures. The placer in VPR is simulated annealing based and optimizes a cost function containing a wirelength estimate among other things. VPR's router is based on the PathFinder [12] negotiated congestion algorithm. The router has to be given a track width, W as an input. In the absence of W, the router has no way of predicting the track width and so starts with $W = 12$, tries to route the circuit and performs a binary search on W, trying to find the optimum value for W. This repeated routing attempts make the routing process time consuming.

3.2 FPGA Architecture and Design Flow



(a) Generic Island−Style FPGA Architecture (b) Design Flow

Fig. 1. *Island Style FPGA Architecture and fGREP Design Flow with VPR*

Fig 1(a) shows a conventional island style FPGA architecture commonly used for research purposes. The logic blocks are marked L, the connection boxes as C and the switch boxes as S. The routing matrix is composed of the routing channels and the switch boxes. Each routing channel consists of a number of routing tracks. The number of tracks in a channel is called the track width and is commonly represented by W. fGREP produces estimates on a channel by channel basis and hence considers channels as routing elements. Hence the terms routing element and routing channel are used interchangeably.

Fig 1(b) portrays the design flow of fGREP using VPR. VPR's placer produces a placement solution for the given netlist and FPGA architecture. The placement solution forms an input to both fGREP and VPR's router. VPR's router produces a detailed route for the optimum value of track width W_{vpr}, which is found by the binary search method as explained earlier. fGREP's output is a routing demand value $D_i \in \Re$ for each channel E_i. The maximum value of D_i is the estimated peak routing demand or

the track width W_{est}. This is directly compared with W_{vpr}. The individual D_i values are compared with the channel occupancy values of VPR's detailed route, to bring out the local estimation accuracy of fGREP. It is to be noted that fGREP is completely independent of the placer and router, and the same design flow can be used with the VPR's placer and router replaced with any other placer and router.

4 Routing Estimation Model

4.1 Routing Demand

The routing estimation model of fGREP is based on the concept of routing flexibility over all the routing elements. First we define routing flexibility and routing demand in their general sense. In section 4.3, we define them in the context of fGREP.

Routing flexibility of a net N_i is defined as the number of different routes possible for the net. Let P^i be the set of all possible routes for net N_i. Each net exacts a certain routing demand on the routing elements used by its paths. Let $P^i_k \subseteq P^i$ be those paths that use the routing element E_k. Formally, the routing demand on E_k due to the net N_i is defined as,

$$D^i_k = \frac{|P^i_k|}{|P^i|} \tag{2}$$

Similarly, all the nets in the given circuit exact routing demands on all the routing elements in the device. Hence the total routing demand due to all the nets on a routing element E_k is defined as,

$$D_k = \sum_{i=1}^{\#nets} D^i_k \tag{3}$$

4.2 Bounding Box Considerations

For a variety of reasons it is preferable to limit the range of influence of a net. Usually the route of a net is limited to within the bounding box of the net. The bounding box of a net is the smallest possible rectangle covering all the terminals of the net. This is done to reduce both wirelength and routing resource utilization. [9] shows that a placement with minimum wirelength has minimum total congestion. Also, minimizing the wirelength is preferred as it has a direct impact on performance. Taking these facts into consideration, we limit the range of all nets to their bounding boxes. Hence Eqn 2 is rewritten as,

$$D^i_k = \begin{cases} 0 & \text{if } E_k \text{ outside the bounding box of net } N_i \\ \frac{|P^i_k|}{|P^i|} & \text{otherwise} \end{cases} \tag{4}$$

This concept is similar to and is drawn from earlier works [13, 14]. In these works, a list of possible paths for each net is enumerated and the routing demands on the individual routing elements are calculated. In practice, the number of paths enumerated is usually far less than the total number of paths possible. This affects the accuracy of

the routing demands. The main difference in our work is that we theoretically calculate the routing demands on all routing elements, instead of actually trying to enumerate the list of possible paths. Our formulation produces accurate routing demands at far lesser runtimes. We explain our method in the following sections.

4.3 fGREP Routing Demand

The routing fabric of an FPGA can be represented by a graph $G(V, E)$. The vertices V represent the channels on the FPGA and the edges E represent the switch-boxes connecting them. There exists an edge (v_i, v_j) if there is a switch-box connecting the two channels i and j.

Consider a vertex v_i on the routing graph. The level l_{ij} of any other vertex v_j is defined as the level of v_j in the breadth first search (BFS) tree with v_i as the root. The level-set L_{ik} is defined as all the vertices in level k on the BFS tree with v_i as the root.

$$L_{ik} = \{v_j \in V | l_{ij} = k\} \tag{5}$$

Definition 1. *We define a path p_{ij} from the root v_i to a vertex v_j as a set of connected vertices $\{v_i, p_{ij}^2, p_{ij}^3, \cdots, p_{ij}^k, \cdots, v_j \ \ | 2 \leq k \leq l_{ij} - 1, p_{ij}^k \in L_{ik}\}$.*

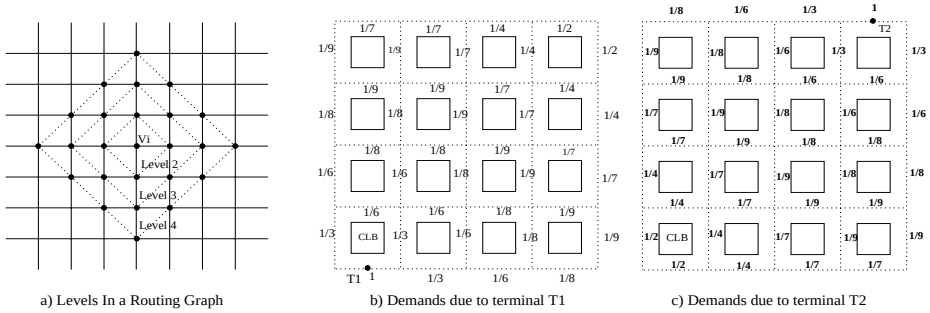
Such paths represent the shortest distance paths (non-returning paths) on the FPGA routing fabric. Since these paths also use the smallest number of routing elements, they are commonly produced by most routers.

Theorem 1. *If all possible unique paths, of large lengths, originating from a vertex v_i and expanding outwards, were to be enumerated, then the ratio of the number of paths that contain any vertex v_j to the total number of paths will be at least $\frac{1}{|L_{ik}|}$, where k is the level of v_j.*

Proof. Let v_i be the root vertex. Consider all the vertices at some level k from v_i. The level-set at k is L_{ik}. The number of vertices in L_{ik} is the number of alternatives available for paths coming from levels greater than k. Let Φ_{k+1} be the total number of paths at level $k+1$, all of them proceeding towards v_i. Since the paths are all equally distributed about the periphery of the level k, they are also equally distributed over the vertices in L_{ik}. Hence each vertex in L_{ik} will be used by $\frac{\Phi_{k+1}}{|L_{ik}|}$ paths. Hence the ratio of the number of paths using a vertex in L_{ik}, to the total number of paths is $1/|L_{ik}|$.

Since $|L_{ik}|$ represents the number of alternatives that may be available for a net at the level k, the routing demand on each routing element in the level-set L_{ik} is $1/|L_{ik}|$.

Consider Fig 2(a). It shows a hypothetical routing graph, on which v_i is the current root vertex. The vertices at the same level from v_i are all shown to be connected by dotted lines. For example, vertices marked Level 2 are all at a distance of 1 from the root vertex and those vertices form the level-set L_{i2}. The number of alternatives available for paths going out of v_i is 4 here. Thus the demand on the nodes at level 2 is $1/|L_{i2}| = 1/4$.



a) Levels In a Routing Graph b) Demands due to terminal T1 c) Demands due to terminal T2

Fig. 2. *Levels (a) and Routing Demands With Bounding Box Constraint (b), (c)*

4.4 Routing Demand with Bounding Box

We now explain the effect of the bounding box on the routing demands. Consider a two terminal net with terminals t_1, t_2 represented by vertices v_1 and v_2 respectively on the routing graph. For reasons detailed in Section 4.2, we calculate the routing demands due to the terminals on the routing elements that are contained in the bounding box only.

Figures 2(b,c) show the terminals and the bounding box of the two terminal net, on the FPGA layout. The routing demands on the channels due to each of the terminals is also shown. The routing demands are obtained by considering only those channels inside the bounding box. The number of channels $|L_k|$ in each level k are calculated using breadth first search and the demands for all the channels in that level are assigned $1/|L_k|$. For all other channels outside the bounding box, the demands are assigned zero.

4.5 Demands due to a Net : Interaction of Multiple Terminals

This section deals with the interaction of multiple terminals and the effect of such interaction on the demand values of the routing elements. By the method detailed above, we get as many demand values for each routing element as there are terminals in the net. Since we are considering the demands due to the whole net, we have to compose the demands in such a way that it captures the concept of a net, as opposed to different independent terminals. We do that by assigning the contribution of the nearest terminal as the routing demand of the net. This has the effect of creating regions of influence around each terminal, where it contributes to the routing demand. For those routing elements that are equidistant from more than one terminal, the higher of the routing demands due to the terminals is assigned. This is done because the higher routing demand due to some terminal will mask out any lower demand due to any other terminal, when considering the net as a whole.

In Fig 3, we have shown the interaction of the terminals of a two terminal net. The routing demands of each channel are marked next to the channels. The entries in regular typeface are those due to terminal t_1 and those in boldface are due to terminal t_2. Note the entries with circles on them. These elements are equidistant from both the terminals. We assign the maximum of the demands due to two terminals for them, which in this case happens to have the same value of $1/9$.

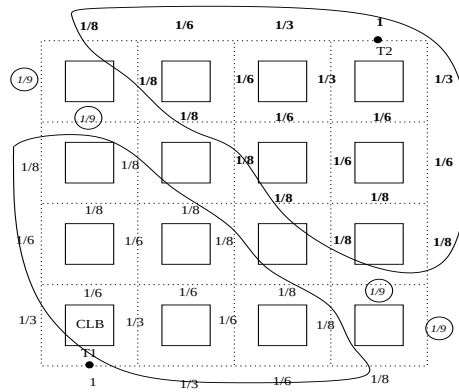


Fig. 3. *Interaction of Multiple Terminals*

The expression for the routing demand on a routing element r_i due to a net n_j is

$$D_i^j = \begin{cases} 0; \text{ if } r_i \text{ outside the bounding box of net } n_j \\ \max(\text{demand due to nearest terminals}); \text{ otherwise} \end{cases} \quad (6)$$

4.6 Interaction of Multiple Nets

The demand on a routing element r_i due to all the nets is the sum of demands due to every individual net.

$$D_i = \sum_{j=1}^{\#nets} D_i^j \quad (7)$$

This value is calculated for all the routing elements in the FPGA layout.

5 Algorithm and Complexity Analysis

The algorithm for calculating the routing demands for all channels is listed in Figure 4. The innermost loop performing the BFS is of the order $|E|$, the number of routing elements in the routing graph. The combined complexity of the outer two loops is of the order $|T|$, where T is the set of all terminals in the netlist. Thus the overall complexity is of the order $O(|E| \cdot |T|)$.

6 Experimentation

We have implemented the fGREP system for a generic FPGA architecture as defined in [1]. The implementation was done in C and was executed on a standard Pentium 800MHz system running Linux. To evaluate the quality of the estimation, we use the FPGA physical design suite VPR. For a fair comparison, VPR was also run on the same

```
Procedure fGREP(netlist N, target architecture A, placement):
    Build routing graph G(V, E) from target architecture A;
    Read Netlist and placement information;
    Create and clear global_demand for all routing elements;
    for each net n_i ∈ N do
        Calculate the bounding box of the net;
        Create and clear demand_for_net and level_for_net;
        for each terminal t_ij of net n_i
            Set t_ij as the root for BFS;
            Mark all routing elements in the bounding box as unvisited;
            while unvisited nodes in the bounding box exist
                Find the number of children C in the next level;
                Calculate demand due to t_ij as 1/C;
                for each child
                    if level_for_net(child) > bfslevel(child)
                        Update demand_for_net and level_for_net;
                    end if;
                end for; –end of one bfs level
            end while; –end of bfs search for one terminal
        end for; –end of all terminals
        Add demand_for_net to global_demand;
    end for;–end of all nets
    return global_demand;
End Procedure fGREP
```

Fig. 4. Routing Estimation Procedure

machine. Most of the runtime options to VPR were set to their default values. The only changes made were to set the number of I/O pads per row(column) to 1 ($io\_rat = 1$) in the FPGA architecture description file, 4lut_sanitized.arch and set the router option to search for routes only inside the bounding box ($bb\_fac = 0$). The FPGA architecture has 4-input LUTs, switch-box flexibility $F_s = 3$, and connection-box flexibility $F_c = 1$.

VPR is first run in $place\_only$ mode for all the circuits to get the placement. VPR places the circuits in the smallest possible rectangular area, subject to pad constraints. The fGREP estimator then uses VPR's placement information and produces estimates for all the routing channels. Then, VPR is run in the $route\_only$ mode to produce actual global and detailed routes. VPR's router performs a binary search on the track widths, starting with $W_{vpr} = 12$, to find a feasible route and the lowest possible track width. The fGREP estimates and the VPR's routed results are compared on a channel-by-channel basis and the mean and standard deviation of the differences, over all the channels are calculated. In order to make make fair comparison with VPR, the router is again executed with the $route\_only$ option and the optimal value of W_{vpr} found by VPR earlier, and the runtimes are noted down as T_{vpr}. This ensures that the runtimes reported in our experiments, as stated in next section, are from a single run of the VPR router and not multiple runs due to binary search for finding smallest feasible W_{vpr}.

	fGREP	*VPR Detailed*	*VPR Global*
Total #of Tracks	201.87	217	147
% Diff in Tracks	6.97%	0	32.26%
Total Runtime	478s	2852s	-

Table 1. *Comparison of fGREP with VPR - Summary of Results*

We used the 20 largest circuits from the standard ISCAS-89 benchmark set. The benchmarks and their characteristics are tabulated in the first 4 columns in Table 2. The benchmarks range in size from 1263 to 8384 CLBs and 1072 to 8444 nets. Extensive experimentation results for all the ISCAS-89 benchmarks are available in [15].

7 Results

The results are tabulated in Table 2. The column headed by $N \times N$ is the dimension of one side of the placement. W_{est}, W_{vpr} and W_{gvpr} are respectively the peak channel width predicted by fGREP, optimum peak channel width found by VPR's detailed router and optimum peak channel width found by VPR's global router. T_{est} is the runtime for fGREP in seconds. T_{vpr} is the runtime for VPR's detailed router to route on a device with a maximum channel width of W_{vpr}. The runtimes for the global router were similar to T_{vpr} and are not tabulated. The column headed by $Mean$ lists the mean of the difference between routing demand of fGREP and that of VPR's detailed router, over all the channels. The column headed by σ lists the standard deviation of the differences. It should be noted that the peak demands W_{vpr} and W_{est} are local values that define the required channel widths based on actual routing and estimated values. The $Mean$ and σ highlight a more global picture across the complete routing architecture. In vast majority of cases, the local values as obtained by the VPR router closely match the estimated values through out the FPGA. This is also supported by very small values for the $Mean$ and σ.

It can be seen that fGREP's estimates are very close to the actual routed results of the VPR's router. For most of the circuits, the difference is of the order of one track, for the maximum channel width W. fGREP is 5 to 20 times faster than the detailed router. Note that this comparison is done between the runtimes of fGREP and that of running VPR with a specific maximum channel width. Under normal circumstances, the maximum channel width is not available to VPR and the runtimes are many times higher. Table 1 summarizes these observations. In Table 1, the percentage difference in tracks is with respect to VPR's detailed router results.

8 Conclusion and Future Work

In this paper we have described a post placement routing demand estimation method for field-programmable gate arrays. Our method avoids costly computations that are associated with other techniques that use stochastic [7], [6], [11], and satisfiability based [8]

techniques. In most cases our method is correctly able to predict the peak (local) routing demand (also called as channel width) and the routing demand for each and every channel in the FPGA. The computation is very fast and hence can provide a quick check on the feasibility or non-feasibility of the routing. As shown in the Table 2, the difference between the actual peak demand and the estimated demand is not more than one in majority of the cases. The $Mean$ and σ gives even greater confidence where we are assured that our prediction is pretty much how the overall routing would be performed. The difference of 6.97 % in track estimation as reported in Table 1 amounts to no more than a difference of one track in prediction. It is also interesting to note that the channel width reported by VPR running in global routing mode (W_{gvpr}) is far less than actual results, even though high execution times are spent in finding the global routes.

As future work we are extending this work to generate a placement method that will assure routable designs with very high degree of confidence. Extensions to various FPGA architectures is a natural extension of this work.

Circuit	*#Cells*	*#Nets*	*#Pads*	*NxN*	W_{est}	W_{vpr}	W_{gvpr}	T_{est}	T_{vpr}	*Mean*	σ
alu4	1523	1536	22	40	9.974	11	7	4.956	38	1.447	1.086
apex2	1879	1916	41	44	10.539	12	8	4.751	59	1.652	1.276
apex4	1263	1271	28	36	11.956	13	9	2.560	43	1.879	1.396
bigkey	1708	1935	426	107	7.466	9	6	46.748	101	0.587	0.746
clma	8384	8444	144	92	11.059	13	8	118.280	549	1.697	1.247
des	1592	1847	501	126	8.072	8	6	26.840	76	0.593	0.652
diffeq	1498	1560	103	39	8.087	8	6	2.352	23	1.114	0.813
dsip	1371	1598	426	107	7.558	7	6	55.887	181	0.429	0.688
elliptic	3605	3734	245	62	10.461	11	7	19.430	237	1.381	1.041
ex1010	4599	4608	20	68	10.728	12	8	32.923	131	1.609	1.196
ex5p	1065	1072	71	33	12.974	14	10	1.855	35	1.853	1.461
frisc	3557	3575	136	60	11.762	14	9	15.307	150	1.662	1.269
misex3	1398	1411	28	38	10.018	11	6	3.097	45	1.508	1.143
pdc	4576	4591	56	68	16.162	16	11	35.321	460	2.132	1.523
s298	1932	1934	10	44	7.590	8	6	9.317	60	1.143	0.814
s38417	6407	6434	135	81	8.899	8	6	28.148	203	1.104	0.823
s38584.1	6448	6484	342	86	8.942	9	6	44.173	248	1.136	0.851
seq	1751	1791	76	42	10.384	12	8	4.073	53	1.680	1.249
spla	3691	3706	62	61	11.865	15	9	19.888	132	1.973	1.438
tseng	1048	1098	174	44	7.375	6	5	2.280	28	0.948	0.732

Table 2. *fGREP Results for the 20 Biggest ISCAS-89 Circuits*

References

[1] Vaughn Betz and Jonathan Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA research," in *Field-Programmable Logic and Applications*. Sep 1997, pp. 213–222, Springer-Verlag, Berlin.

[2] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Addison Wesley, Reading, MA, 1990.

[3] H. Van Marck, D. Stroobandt, and J. Van Campenhout, "Toward an Extension of Rent's Rule for Describing Local Variations in Interconnection Complexity," in *Proceedings of the 4th International Conference for Young Computer Scientists*, 1995, pp. 136–141.

[4] G. Parthasarathy, M. Marek-Sadaowska, and A. Mukherjee, "Interconnect Complexity-aware FPGA Placement Using Rent's Rule," in *To appear in, Proc. Intl. Workshop on System Level Interconnect Prediction (SLIP)*, April 2001.

[5] Wei Li, "Routability Prediction for Hierarchical FPGAs," in *Proc. Great Lakes Symposium on VLSI*, 1999.

[6] Abbas A. El Gamal, "Two-Dimensional Stochastic Model for Interconnections in Master Slice Integrated Circuits," *IEEE Trans. CAS.*, Feb 1981.

[7] S.Brown, J.Rose, and Z.G.Vranesic, "A Stochastic Model to Predict the Routability of Field Programmable Gate Arrays," *IEEE Transactions on CAD*, pp. 1827–1838, Dec 1993.

[8] R.G. Wood and R.A. Rutenbar, "FPGA Routing and Routability Estimation via Boolean Satisfiability," in *ACM International Symposium on FPGAs FPGA98*. June 1998, ACM.

[9] M. Wang and M. Sarrafzadeh, "Congestion Minimization During Placement," in *Proceedings of the 1999 International Symposium on Physical Design (ISPD)*, 1999.

[10] Sudip K. Nag and R.A. Rutenbar, "Performance-driven Simultaneous Placement and Routing for FPGAs," *IEEE Transactions on CAD*, June 1998.

[11] S. Sastry and A.C. Parker, "Stochastic Models for Wireability Analysis of Gate Arrays," *IEEE Trans. on CAD*, Jan 1986.

[12] Larry McMurchie and Carl Ebeling, "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs," in *ACM Symp. on FPGAs, FPGA95*. ACM, 1995, pp. 111–117.

[13] S. Brown, J. Rose, and Z.G. Vranesic, "A Detailed Router for Field Programmable Gate Arrays," *IEEE Transactions on CAD*, May 1992.

[14] G. Lemieux and S. Brown, "A Detailed Router for Allocating Wire Segments in FPGAs," in *ACM Physical Design Workshop*, April 1993, pp. 215–226.

[15] Parivallal Kannan, Shankar Balachandran, and Dinesh Bhatia, "fGREP Results for ISCAS-89 Benchmarks," Tech. Rep., CICS, University of Texas at Dallas, 2001, http://www.eac.utdallas.edu/pubs/rep1_01.pdf.