Heuristic Performance Optimal and Power Conscious for K-LUT Based FPGA Technology Mapping

ION BUCUR, NICOLAE CUPCEA, ADRIAN SURPATEANU, COSTIN STEFANESCU, FLORIN RADULESCU

Computer Science and Engineering Department
University Politehnica of Bucharest
313 Splaiul Independentei, Bucharest 060042
ROMANIA

{ion.bucur, nicolae.cupcea,adrian.surpateanu, costin.stefanescu, florin.radulescu}@cs.pub.ro

Abstract: - In this paper is presented a new approach for decreasing the power consumption in LUT based FPGA implemented circuits. The attempt is based on reducing logic activity among LUTs. In order to achieve this target it was used the probability approach that estimates the dynamic logic activity of each line in the circuit. Traversing circuits from primary inputs lines to the primary output lines, step by step, stationary probability and transition distribution are computed at the output of each gate (node). Preserving the best depth of the circuits the mapping stage is done searching to hide high transition lines inside LUTs.

Key-Words: - Area optimal mapping, Heuristic mapping for area , LUT-based FPGA, Power estimate driving circuit implementation, Power conscious.

1 Introduction
Power consumption is becoming one of the most important considerations in VLSI design. Increase in both complexity and size of these circuits highlights the importance of the power dissipation. Moreover, the spectacular decrease in chip size and increase in both transistor count and clock operating rate are pointing to the high importance of circuits having low power dissipation. Low power dissipating chips involve low cost of the packaging and cooling. High power often run hot and high temperature tends to exacerbate several silicon failure mechanisms. It’s known that every 10° C increase in operating temperature roughly doubles a component’s failure rate [16]. Therefore, in addition to performance and area optimization a great deal of research has been directed towards issues related to the low power circuit design. FPGAs have emerged as a well-liked technology due to its short turn-around time and low manufacturing costs. However, they are less power efficient than custom ASICs [7], [8]. Our focus in this paper is on searching optimal solutions primarily for depth, and power at the gate level. Our secondary focus is on depth, area and power at the same gate level. It is presented a new technology mapping algorithm based on previous studies and results [3], [4], [5], and [6]. While the second target is much more complex, because there are involved two NP-hard problems, the first target was more fruitful being obtained more convincing results. The paper is organized as follows:

In the remainder part of this section are reviewed the main features of the mapping process for K-LUT-based FPGAs, and considerations on essential resources involved in FPGA mapping. In section 2 there, are presented main power issues in FPGA technology. Relevant previously published work concerning power-aware technology mapping targeting FPGAs are outlined in section 3. Definitions and main model are briefly presented in section 4. Used model for the power estimation, at the gate level, is presented in section 5. Section 6 contains our approach presentation and introduces main features of the PwAwMap algorithm. Experimental studies, results and conclusions are exposed in section 7.

2 Dissipated Power in LUT Based FPGAs
FPGAs circuits are developed in CMOS technology. Power dissipation in CMOS circuits comprises both static (leakage) power and dynamic power. Static power is consumed when a circuit is in a quiescent, idle state. Static power results from leakage current in off transistors, primarily sub-threshold and gate-oxide leakage. Lowering technology grid means lower supply voltages and smaller transistor dimensions. It leads to shorter wire length, less capacity, and an overall reduction in dynamic power. Smaller process geometries also mean shorter transistor channel lengths and thinner gate oxides, producing an increase in static power as technology scales. At the transistor level Virtex-4® and Virtex-5® FPGAs, as an example, employ triple-oxide process technology for leakage mitigation. They are used depending on its speed, power and reliability requirements. Dynamic power, on the other hand, is caused by transitions on signals of a circuit and is...
The programmability and flexibility of FPGAs are making them less power-efficient than custom ASICs when considering the implementation of a given logic circuit. FPGAs consist of three kinds of programmable elements. Configurable logic blocks (CLB), input–output (I/O) blocks, and routing resources. Each configurable logic block contains combinational components such as multiplexers (MUXs), simple gates (e.g., OR and AND), programmable lookup tables (LUTs), and sequential components such as flip-flops. Configurable logic blocks have 4, 5 or 6 input lines. This number of logic input lines is symbolically referred by $K$. Each CLB contains one programmable combinational logic component and two flip-flops. One CLB can implement any $K$-variables logic function or two logic functions of at most $K$-1 variables each but both functions must have at most $K$ variables. The FPGA configuration memory and configuration circuitry consume silicon area, producing longer wire lengths and higher interconnect capacitance. Programmable routing switches, on the other hand, attach to the pre-fabricated metal wire segments in the FPGA interconnect and add to the capacitive load incurred by signals [11]. Most dynamic power in an FPGA is consumed in the programmable routing fabric. About 50% - 70% of total power is dissipated in the inter-connection network [8], [17]. Interconnect comprises a considerable fraction of the FPGA’s transistors and therefore dominates leakage. Signal capacitance is known after the placement and routing processes. While performing placement this capacitance is only estimated. Estimation is made using empirically derived capacitance models. Such models are derived using least-squares regression analysis [11], [17].

**3 Relevant Related Works**

There have been done several works for decreasing the power consumption in circuits mapped with FPGAs. Farrahi and Sarrafzadeh studied the technology mapping problem for lookup table-based FPGAs [9]. The problem is formulated as assigning LUTs to nodes of a circuit so as to minimize the total area. They did show that the decision version of this problem is NP-complete, even for simple classes of inputs such as 3-level circuits. This result is extremely important pointing the difficulty of the problem. The same proof is extended to conclude that the general library-based technology mapping for power minimization is NP-complete [10]. An efficient heuristic algorithm for the low-power design with FPGA is introduced by Wang [21]. The main idea in this paper was to exploit the cut enumeration technique to generate possible mapping solutions for the sub-circuit rooted at each node. However, for the consideration of both run time and memory space, only a fixed-number of solutions were selected and stored by their heuristic. Singh and Marek-Sadowska presented a routability-driven bottom-up clustering technique both for area and power reduction in clustered FPGAs [18]. This technique uses cell connectivity metric to identify seeds for efficient clustering. Anderson and Najm proposed a power-aware technology mapping technique for LUT-based FPGAs which aims to keep nets with high switching activity out of the FPGA routing network and takes an activity-conscious approach to logic replication [1]. The utility of some low-power design methods based on architectural and implementation modifications for FPGA LUT-based systems, are presented in a paper of Sutter and Boemo [20]. The contribution of spurious transitions to the overall consumption is evidenced and main strategies for its reduction are analyzed. Empirical results are presented in order to show the effectiveness of pipelining and sequentialization as low-power design methodologies. Possibilities of power management techniques are quantified. The work of Hsieh et al. discuss optimizing the interconnect power of designs implemented in FPGA platforms [12]. In particular, it is reduced the glitch power on interconnects associated with the output of functional units in a design. The idea is to activate unused flip-flops to block the propagation of glitches, which takes advantage of the abundant flip-flops in modern FPGA structures. Jiang et al. did present a mathematical programming formulation of the integer time budgeting problem for directed acyclic graphs [13]. In the work of Ho et al. is described an approach to estimate the power consumption of a set of hybrid FPGA architectures, island-style fine grained units and domain-specific coarse-grained units [11]. They reported results over a set of floating point benchmark circuits. Mashayekhi et al. present in their paper [15] a method that attempts to reduce the switching activity among LUT blocks. To achieve this, they have introduced the fake register insertion method and combined it with retiming method. Fake registers have been inserted on low-transition wires around high-transition wires to force the synthesis tool to direct those low-transition wires on LUTs' outputs. Retiming is used to move registers that have been located on high-transition wires in order to prevent the synthesis tool to place them on LUTs' outputs.

\[ P_d = 0.5 \sum_i C_i \cdot V_{dd}^2 \cdot d(i) \]  

Where:
- $C_i$ represents the capacitance of a signal, $i$;
- $d(i)$ is referred to as “transition activity” of logic signal $i$ and represents the rate of transitions on signal $i$ (i.e. the number of times that signal $i$ changes its value in unit time);
- $V_{dd}$ is the supply voltage.
Proceedings of the 1st International Conference on Manufacturing Engineering, Quality and Production Systems (Volume I)

5 Dynamic Power Estimation Model

Various approaches to computing dynamic switching activity have been proposed in the literature, and they can be generally considered as either simulation-based approaches or as probability approaches. In [3] and [4], the simulations-based approaches are used to compute the probability that a given logical path is in a logic transition. However, due to the fact that this approach is computationally expensive, an upper bound on the probability of a logic transition is computed using the expression:

\[ \text{level}(u) = \max \{ \text{input}(u) \} \]

The level of a PI node is zero and the level (depth) of a node \( u \) in the network is the largest node level in the network.

6 Algorithm Descriptions

Our approach is using structures, and routines of SIS-1.2 in order to build-up the programmed application and run-time appropriate control functions for minimal depth mapping. The algorithm is based on the K-feasible-cone enumeration method that mainly consists on exhaustively generating all K-feasible cones. The implemented technology mapping procedure operates in three steps. In the first step, the maximum depth mapping is determined for each K-feasible cone \([5, 6] \). In the second step, the mapping of each K-feasible cone is made during a network traverse from primary inputs to primary outputs. In the last step, the primary outputs and the primary inputs are connected to the LUTs in the network. It results that the number of logic changes on an arbitrary line is incremented by \( m \) (1) in (2).

\[ \text{power}(f) = \sum_{i=1}^{N} \text{power}(f_i) \]

(2)
6.1 Generating K-feasible cones

In order to generate power-conscious minimal depth K-LUT mapped network is necessary, in general, the knowledge of an appropriate minimal height K-feasible cone, for each internal node $u$ in the initial network. It is interesting to note that the nodes that are not on a critical path do not need a minimal height K-LUT implementation. Let $N$ be a K-bounded network, and $u$ an arbitrary node of $N$. Then, a K-feasible cone of the node $u$, noted $C(u)$, could be identified by the set:

$$\text{input}(C(u)) = \{v_1, v_2, \cdots, v_m\}, m \leq K$$  \hspace{1cm} (3)

Such a set could be represented as the product (conjunction) of the elements (literals) of the set in (3), $p = v_1v_2 \cdots v_m$. The set of all feasible cones of node $u$, noted $\text{cones}(u)$, can be represented as the sum (reunion) of each of the product (cube) representing the respective cone:

$$\text{cones}(u) = \bigcup_{i} \text{input}(C_i(u))$$  \hspace{1cm} (4)

Representing each K-feasible cone of the node $u$ as a conjunction, in above relation, it becomes:

$$\text{cones}(u) = \bigcup_{i} v_i \cdot v_{i'} \cdots v_{i''}, m \leq k.$$  \hspace{1cm} (5)

Then it holds this Lemma:

**Lemma 1.** Given a node $u$ having as immediate predecessors: $\text{input}(u) = \{v, w, \cdots, z\}$, each predecessor having already computed the set of all K-feasible cones, respective $\text{cones}(v), \text{cones}(w), \cdots, \text{cones}(z)$, than the set $\text{cones}(u)$, of all the K-feasible cones of node $u$, is:

$$\text{cones}(u) \subseteq \left(\bigvee_{v} \text{cones}(v)\right) \cap \left(\bigvee_{w} \text{cones}(w)\right) \cap \cdots \cap (z \cup \text{cones}(z))$$ \hspace{1cm} (6)

Applications of the Lemma are presented in [4]. Computing the sum-of-products (SOP) form of the expression (6), and eliminating (as soon as possible) all the products having more than K literals, one can determines $\text{cones}(u)$, the set of all K-feasible cones of the node $u$. There is only polynomial number of K-feasible cones in the predecessor’s maximum transitive cone of each node $u$ (denoted PMTC$_u$), since the total number of possible combinations of $K$ or fewer nodes is $O(n^K)$, where $n$ is the number of nodes of PMTC$_u$. Results of the generation of all K-feasible cones rooted in every node of a DAG for 10 circuits from the MCNC 91 ATPG benchmark are shown in [4]. A procedure able to compute resourcefully all the K-feasible cones of all nodes in a network, in general, build up the complete solution’s space of K-LUT mapping. One could make use of any optimization criteria and any delay model associated with the edges of $N(V,E)$, the DAG of the gate network $N$. Actually, the implemented algorithm is able to K-LUT map any K-bounded Boolean network. K-LUT based FPGA implementation of multi-level networks can be viewed as subject of fanin K-limited of the gates. Decomposition is the main approach to obtain a K-bounded network. There are used various methods, including Roth-Karp decomposition (Boolean), AND-OR decomposition (algebraic) etc. Making a network K-bounded is considered, in general, a pre-processing step in the K-LUT mapping of FPGAs. In [4] are listed part of our experiments made in order to show that decomposition granularity influences the performance of the mapping process. It was used, after technology independent optimization, AND-OR balanced decomposition of circuits. The number of input lines in each type of gate was the parameter of this experiment. Results in [4] are showing that decomposition made with $K = 2$, are always leading to best results.

6.2 Cost Functions

Once K-feasible cones generation completed, same network is traversed from primary outputs to the primary inputs, starting with the primary outputs having largest delay mapping as it was evaluated during K-feasible cones generation. The right selection among the K-feasible cones of each node is guided using critical path and several appropriate cost functions. The main difficulty lies in the approach we use in order to select a subset of all K-feasible cones to cover the whole circuit. The problem of mapping for depth, of an arbitrary network, can be optimal computed in polynomial time using dynamic programming procedure. Implementing our actual heuristics, for dissipated power minimization, we determined metrics that are slightly similar to those used in the work [1] and [2]. However, our heuristics implements different metrics because we attached several specific data during K-feasible cones generation for each feasible cone. Data that we attached to each feasible cone are related to the number of internal nodes of it (as efficiency measure of the irrespective feasible cone), the count of internal nodes having fan-outs spreading in other feasible cones (marking possible duplicated nodes) etc. The main challenge lies in the approach we use in order to select a subset of all K-feasible cones to cover the whole circuit. The problem of mapping for depth, of an arbitrary network, can be optimal computed in polynomial time using dynamic programming procedure.

Logic replication or duplication is performed implicitly when a LUT is used to implement a K-feasible cone. When a node in a circuit is replicated for depth minimization, a connection from the node to one of its successors is hidden within a LUT. Such hidden connections are no more routed through the FPGA interconnection network and therefore do no more contribute to the interconnect power dissipation. The way it is selected a feasible cone from a set of K-feasible cones of an arbitrary node $u$ is different when are
mapped nodes belonging to two or more transitive cones determined from primary outputs. This way helps avoiding unnecessary node duplication when dissipated power is not an issue. Depth Metric of an arbitrary node \( u \), is computed over one the best depth \( K \)-feasible cone of \( u \):

\[
\text{DepthMetric}(\text{cones}(u)) = 1 + \min (\text{DepthMetric}(v) \mid v \in \text{cones}(u))
\]

This metric is used mainly to quantify the depth criterion. EstimPowerCost is introduced in order to quantify the locally dissipated power. The main target of it is to attract as many high-activity lines as possible inside of LUTs. This cost is computed using the relation:

\[
\text{EstimPowerCost}(\text{cones}(u)) = \min_{C(u) \subseteq \text{cones}(u)} \left\{ \sum_{v \in \text{input}(C(u))} (d(u) \cdot \text{fanout}(u)) + \text{DepthMetric}(\text{cones}(v)) \right\}
\]

Both metrics are, in fact, partially computed during the \( K \)-feasible cones generation step. The algorithm is globally using this parameterized cost:

\[
\text{GlobalCost}(\text{cones}(u)) = w_1 \cdot \text{DepthMetric}(\text{cones}(u)) + w_2 \cdot \text{EstimPowerCost}(\text{cones}(u))
\]

Parameters \( w_1 \) and \( w_2 \) were experimentally determined.

7 Experimental Results and Conclusion

The basis of our approach is the exhaustive generation of all feasible \( K \)-feasible rooted in every node of the network. The speed of this generation is offering enough time margins in order to search among all possible solutions the most appropriate one. It was assumed that all primary inputs have 0.5 switching activities, and all involved capacities have same value. Our implemented algorithm did run for mapping into 5-LUT FPGAs several benchmark circuits, listed in Table 3. To estimate power consumption using (1) it is required the capacitance of each net. Obviously in this stage of designing circuits targeting FPGA mapping, the capacitance of any net it is not known until layout is complete. In actual algorithm implementation, structural properties of the circuit were used in order to have an estimate of the interconnect capacitance. Considering that most of the connections have, on average, same length than the fanout factor was chosen as the main feature making difference between various connections. Since our attempt was to build-up a tool able to evaluate medium-grain different network choices during logic design, the estimated dynamic power for each node \( u \) was simply computed mainly as the product of the transition density the node \( d(u) \) and the fan-out of it:

\[
\text{EstimatedDynamicPower}(u) = d(u) \cdot \text{fanout}(u)
\]

\( \text{PwAwMap} \) is an efficient algorithm being able to compute several low-power optimal options, as can be seen in Table 3. The first option keeps optimum depth and search among power-aware equivalent solutions. The second option is searching, on the base of the user’s explicit option, one of the solutions with optimal depth but performing with improved power consumption.

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Depth</th>
<th>Optimum Depth</th>
<th>Optimal Depth</th>
<th>Optimal Depth &amp; Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>5xp1</td>
<td>3</td>
<td>2.92</td>
<td>2.23</td>
<td>2.56</td>
</tr>
<tr>
<td>9symml</td>
<td>5</td>
<td>3.89</td>
<td>3.17</td>
<td>3.65</td>
</tr>
<tr>
<td>C499</td>
<td>5</td>
<td>10.76</td>
<td>8.82</td>
<td>9.96</td>
</tr>
<tr>
<td>C880</td>
<td>8</td>
<td>16.69</td>
<td>15.17</td>
<td>16.34</td>
</tr>
<tr>
<td>alu2</td>
<td>8</td>
<td>14.05</td>
<td>13.04</td>
<td>12.53</td>
</tr>
<tr>
<td>apex6</td>
<td>4</td>
<td>22.62</td>
<td>20.58</td>
<td>22.68</td>
</tr>
<tr>
<td>apex7</td>
<td>4</td>
<td>10.45</td>
<td>9.87</td>
<td>9.94</td>
</tr>
<tr>
<td>count</td>
<td>3</td>
<td>3.52</td>
<td>3.04</td>
<td>3.50</td>
</tr>
<tr>
<td>des</td>
<td>5</td>
<td>99.67</td>
<td>97.02</td>
<td>98.87</td>
</tr>
<tr>
<td>duke2</td>
<td>4</td>
<td>9.21</td>
<td>9.23</td>
<td>9.25</td>
</tr>
<tr>
<td>misex1</td>
<td>2</td>
<td>2.39</td>
<td>2.23</td>
<td>2.41</td>
</tr>
<tr>
<td>rd84</td>
<td>4</td>
<td>4.98</td>
<td>4.27</td>
<td>4.77</td>
</tr>
<tr>
<td>rot</td>
<td>6</td>
<td>37.68</td>
<td>36.54</td>
<td>36.73</td>
</tr>
<tr>
<td>vg2</td>
<td>4</td>
<td>4.32</td>
<td>3.21</td>
<td>4.23</td>
</tr>
<tr>
<td>z4ml</td>
<td>3</td>
<td>1.49</td>
<td>1.38</td>
<td>1.42</td>
</tr>
</tbody>
</table>

The optimal depth was considered as an incremented optimum depth. For nodes situated on the critical paths of irrespective networks the optimal depth was computed using this relation:

\[
\text{optimalDepth}(u)_{\text{w} \text{C} \text{riticalPath}} = \text{optimumDepth} + \lambda \quad (11)
\]

Values listed in the second column of Table 3 were computed using \( \lambda = 1 \) for nodes belonging to the critical path, while for other nodes the optimal depth values were at most less or equal to the optimal depth of the circuit. Area minimization is extremely important for FPGA synthesis. Since area-optimal technology mapping for \( K \)-LUT-based FPGAs is \( NP-hard \) [13] several methods were developed in our attempt. While maintaining an optimum depth of the network it is searched, among power-aware solutions, those having the minimal area (number of used LUTs). The third solution targets an optimal area and depth while keeping in low margin the dissipated power (illustrated in the third column of Table 3). In Table 3, on average, the detailed experimental results are showing that power-aware mapping for optimal depth the estimated dissipated power is 6.07% less than mapping for optimum depth. So, relaxing mapping conditions for circuits’ depth it is leading to less dissipated power. But,
introducing area minimal constraint it makes mapping, for both optimal depth and area, to be only 2.37% more efficient (concerning the dissipated power) than mapping for optimum depth. Mapping power-aware both for depth and area optimal it seems be more complex and actual used heuristics have to be upgraded because it was searched only a limited part of mapping solutions’ space. It is intended in the future approach to use dynamic programming together with refined heuristics in PwAwMap algorithm.

References: