

Optimizing Depth, Area and Dynamic Power of k -LUT Based FPGA Circuits

Ion I. Bucur *, Nicolae Cupcea *, Costin Stefanescu *, Adrian Surpateanu *

* University Politehnica Bucharest, Faculty of Control and Computers,

Computer Science and Engineering Department

E-mail (ion.bucur, nicolae.cupcea, costin.stefanescu, adrian.surpateanu) @cs.pub.ro

Abstract: This paper proposes an efficient algorithm for complex criteria applied to K -LUT based FPGA mapped circuits. This algorithm is developed considering an important design factor – dynamic power consumption - in addition to other design factors that are traditionally considered. To increase performance, it was used a flexible mapping tool based on exhaustive generation of all K -bounded sub-circuits rooted in each node of the circuit. To achieve information about dissipated functional power was implement efficient simulator. Area is of paramount importance in FPGA mapping circuits. Lastly to lower power consumption, we devised several effective techniques designed for reducing area

Keywords: power-aware mapping, optimal area, K -LUT based FPGA, logic activity simulator, functional power.

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 more and more. In many application domains, the condition of longer battery life for the increasing class of portable computing and wireless communication apparatus requires the operation of low-power circuits. 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 (Pedram, 1996). 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 (Jones *et al.* 2007). For now, time to market pressure makes it imperative that design, development, production and testing time be diminished as much as possible. On the other hand, field-programmable gate arrays (FPGAs) are very attractive choices for digital circuit implementation. 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 (Chen, C.-S. *et al.* 1997), and (Chen, D. *et al.* 2006). Our focus in this paper is on searching optimal solutions primarily for depth, and power at the gate level. Our secondary target is on depth, area and dynamic power at the same gate level. It is presented new technology mapping algorithm based on previous studies and results (Bucur, 1999), (Bucur 2007), (Bucur *et al.* 2008), (Bucur *et al.* 2009). Main idea was to build-up an application working at the logic level and able to find power-aware

mapping solutions both for depth and area criteria. 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 Section 2 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. Finally, in Section 7, are presented experimental results, conclusions, and is outlined future work.

2. POWER ISSUES IN FPGA CIRCUITS

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. An off-MOS transistor is an imperfect insulator allowing sub-threshold leakage current to flow between its drain and source terminals. Gate-oxide leakage is caused by tunneling current through the gate terminal of a transistor to its body, drain, and source (Gupta and Anderson, 2007). 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. There are three possible oxide thicknesses, as an example, for each transistor in Xilinx technology. 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 governed by the equation:

$$P_d = 0.5 \sum_i C_i \cdot V_{dd}^2 \cdot d(i) \quad (1)$$

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.

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 (Fig.1). Logic cells named *configurable logic blocks* (CLB), programmable I/O cells named *input-output blocks* (I/O Blocks) blocks, and *programmable interconnect*, routing resources are the main elements of a typical FPGA. 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. Packing LUTs and flip-flops into CLBs, in general, is a critical step in the cluster-based FPGA design flow, since it has a great impact on both timing and routability. While placement and routing is strongly connected with the detailed architecture inside of the chip and mostly managed by the commercial FPGA software, the optimization and mapping can be more influenced by the user.

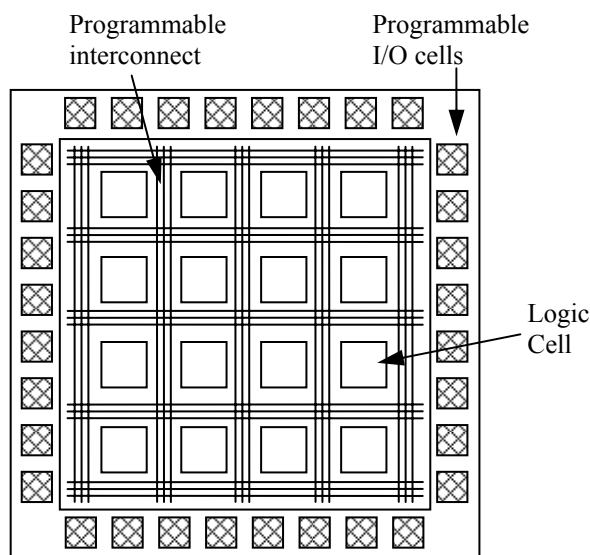


Fig.1. Generic FPGA Architecture.

The place and route optimization intend to lower power in the interconnect fabric. Most of the recent advanced algorithms or heuristics for FPGA mapping approaches area minimization under delay constraints. Even if delay constraints are not specified, an optimum delay, for the considered network is determined and after that step, without modifying the delay, the area is minimized. During the circuit logic optimization stage it's possible to optimize dynamic power by modifying the way the circuit is K -LUT mapped. Hiding, in K -LUTs, lines with high switching activity, dissipated dynamic power by the circuit is lowered altogether. In practice, K is usually small (for example 4-LUTs are widely used in commercial FPGAs), as the area of a K -LUT grows exponentially with large K . It was showed that 4-input, single-output LUT cell yields the smallest FPGA area of any K -LUT cell for a wide range of programming technologies and routing pitches (Rose *et al.* 1990). The I/O blocks can be programmed to become the primary inputs (PIs) or primary outputs (POs) of the circuits on FPGAs.

Routing resources include segmented interconnects and switching blocks. The segmented interconnects connect to the inputs and outputs of logic blocks while the switching blocks link the segments to form long routing tracks to implement routing topology. In a typical FPGA design flow, a circuit is first synthesized and mapped into a netlist of K -LUTs and flip-flops. Then it goes through the following three steps: clustering, placement and routing. The clustering step arranges K -LUTs and flip-flops into CLBs according to the timing and the connectivity of the mapped netlist; the placement places the clustered netlist onto the array of on-chip CLBs; the routing routes all the wires in the netlist with the available routing resources on the device.

Routing tools are using an important portion of the global design effort (Gupta and Anderson, 2007), (Hansen and Thomas, 2005), and (Yang 2005). The FPGA configuration memory and configuration circuitry consumes 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. 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 (Anderson and Najm, 2004) and (Poon *et al.* 2005). 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 (Gupta and Anderson, 2007), (Poon *et al.* 2005).

3. 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 (Farrahi and Sarrafzadeh, 1994b). 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* (Farrahi and Sarrafzadeh, 1994a). A heuristic algorithm for mapping the network onto K -inputs LUTs in polynomial time, aimed at minimizing the power consumption is presented in their paper.

Li *et al.* are presenting in their paper an efficient heuristic to compute both low-power and reduced area mapping solutions (Li *et al.* 2001). The major distinction of their work from previous ones was that while generating a LUT, it is looking ahead at the impact of the mapping solutions of this LUT on the power consumption of the remaining network. Their approach computes min-height K -feasible cuts for non-critical nodes to minimize the power consumption of the mapping solution. Li *et al.* implemented their application, named *PowerMap*, in *C* and tested it on a number of MCNC benchmark circuits. Compared to *FlowMap*, an early very well known delay-optimal mapping application, their algorithm reduces the power consumption by 17.8% and uses 9.4% less LUTs without any depth penalty.

An efficient heuristic algorithm for the low-power design with FPGA is introduced in (Wang *et al.* 2001). 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 the heuristic. To facilitate the selection process, a method that correctly calculates the estimated power consumption for each mapped sub-circuit was developed. The experimental results are showing that their algorithm reduces on average the power consumption by up to 14.18% and the average number of LUTs by up to 6.99% over an existing method.

Singh and Marek-Sadowska presented a routability-driven bottom-up clustering technique both for area and power reduction in clustered FPGAs (Singh and Marek-Sadowska, 2002). This technique uses cell connectivity metric to identify seeds for efficient clustering. It leads to better device utilization, savings in area, and reduction in power consumption. Authors report routing area reduction of 35% over previously published results. Power dissipation simulations using a buffered pass-transistor-based FPGA interconnect model is introduced. Singh and Marek-Sadowska did show that presented clustering technique can reduce the overall device power dissipation by an average of 13%.

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 (Anderson and Najm, 2002). Logic replication is known to be crucial for optimizing depth in technology mapping; an important contribution of their work was to recognize the effect of logic replication on circuit structure and to show its consequences on power.

The utility of some low-power design methods based on architectural and implementation modifications for FPGA LUT-based systems, are presented in the paper of Sutter and Boemo (Sutter and Boemo 2007). 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.

Gupta and Anderson, presents several very interesting results from power-aware placement and routing optimization (Gupta and Anderson 2007). A set of industrial designs were placed and routed using both the traditional place and route flow, as well as the power flow in ISE 9.2i Design Tools. The designs were augmented with built-in automatic input vector generation by attaching a linear feed-back shift register-based (*LFSR-based*) pseudo-random vector generator to the primary inputs. This feature permitted to perform board-level measurements of dynamic power without requiring a large number of externally applied waveforms. The industrial designs were mapped into Spartan-3, Virtex-4, and Virtex-5 devices. Results showed that dynamic power was reduced as much as 14% for Spartan-3 FPGAs, 11% for Virtex-4 FPGAs, and 12% for Virtex-5 FPGAs. On average, across all designs, dynamic power was reduced by 12% for Spartan-3 FPGAs, 5% for Virtex-4 FPGAs, and 7% for Virtex-5 FPGAs. For all families, on average, the speed performance hit was between 3% and 4%, which was appreciated as being small and acceptable in power-conscious designs.

The work of Hsieh *et al.* discuss optimizing the interconnect power of designs implemented in FPGA platforms (Hsieh *et al.* 2008). 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 (Jiang *et al.* 2008). In particular, it is formally proved that their constraint matrix has a special property that enables a polynomial-time algorithm to solve the problem optimally with a guaranteed integral solution. Their theory can be directly applied to solving a scheduling problem in behavioral synthesis with objective of minimizing the system power consumption. Given a set of scheduling constraints and a collection of convex power-delay tradeoff curves for each type of operation, their scheduler can intelligently schedule the operations to appropriate clock cycles and simultaneously select the module implementations that lead to low-power solutions. Experiments demonstrate that their proposed technique can produce near-optimal results (within 6% of the optimum by the ILP formulation), with 40x + speedup.

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 (Ho *et al.* 2008). They reported

results over a set of floating point benchmark circuits. The power performance of a hybrid FPGA is compared to the Xilinx Virtex II 3000 which has the same architecture as the hybrid FPGA but no coarse-grained unit. On average, floating point applications implemented on hybrid FPGA can reduce dynamic energy consumption by a factor of 14 compared to the Virtex II FPGA.

Mashayekhi *et al.* present in their paper a method that attempts to reduce the switching activity among LUT blocks (Mashayekhi *et al.* 2008). 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. As of benchmarking, two of ISCAS89 benchmark library circuits were employed. Experimental results have shown that this scheme will decrease the off-block switching of the implemented circuits up to 25%.

The paper of Jang *et al.* describes a powerful simulator and several complementary algorithms for power-aware logic optimization (Jang *et al.* 2009b). The proposed simulator draws on new ideas in logic representation and is geared for speed, e.g. it can simulate a 1 M-node sequential design using 1 000 bit patterns for 100 cycles in about 10 seconds on a typical one-core CPU. When applied to large industrial designs in a highly-optimized industrial flow the techniques described in this paper led to a 19.6% reduction in switching activity without a substantial increase in runtime or degradation of other metrics.

4. PROBLEM FORMULATION

Combinational part of a general logic network N can be represented as a direct acyclic graph (DAG) noted as $N(V, E)$ where V is the set of nodes and E is the set of directed edges. Each node in $N(V, E)$ represents a logical gate (possible complex), and a direct edge (u, v) exists if the output of gate u is an input of the gate v . The set of direct predecessors of gate v is expressed as $input(v)$ and the set of direct predecessors of a graph $H \subseteq N(V, E)$, is similar expressed as $input(H)$. The set of direct predecessors of G is the set of all primary inputs of the network. Primary input (PI) nodes and primary output (PO) nodes in a network are nodes that have no incoming edge, respectively are nodes that have no outgoing edge. Flip-flop outputs are considered as pseudo-PIs and flip-flop inputs are considered as pseudo-POs and no distinction is made in terms of notation. Let u be a gate in N and we are interested to compute generic K -feasible cone of node u (denoted C_u) in direct acyclic graph $N(V, E)$. Cone C_u is rooted in u and is included in the predecessor's transitive cone of the node u (denoted $PMTC_u$) and having no more than K direct predecessors ($|input(C_u)| \leq K$). Given a cone C_u and a node $v \in C_u$, any path connecting the node v and the node u , lies entirely in C_u . A logic network N , modelled by the $N(V, E)$, such that for each node u (in $N(V, E)$):

$$|input(u)| \leq K,$$

is K -bounded. The level of a node u is computed, in general, is computed using the expression:

$$level(u) = 1 + maximum_level\{input(u)\} \quad (2)$$

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

5. DYNAMIC POWER EVALUATION

Dynamic power has two main components: switching power and short-circuit power. Both components can only occur when a signal transition happens at the CMOS gate output. In this work the focus is on switching power. Switching power evaluation is essentially concerned with switching activity estimation and load capacitance estimation. Gates (including buffers) and wires contribute capacitance in the FPGA's circuits. It is known that in a combinational circuit, switching at a node has correlations with its own past values and with its neighbors in the circuit. Temporal correlation is due to the fact that switching in a node is dependent on its last value and its function. Spatial correlation among nodes arises of the basic logical connections. Various approaches to computing switching activity have been proposed in the literature, and they can be, generally, considered as either:

- Probabilistic models, or
- Characterization through board measurement, or
- Simulation-based approaches.

The approach of switching activity estimation is based on probabilistic models. Such strategies are also known in literature as *non-simulative* switching activity techniques (Bucur *et al.* 2009). The probabilistic techniques use knowledge about input statistics to estimate the switching activity of internal nodes (Li *et al.* 2001). Probabilistic models supporting signal probability calculus were introduced several years ago, when it was developed the *random testability*. Net signal probability was studied and various methods were established in order to compute an exact value or an estimate of it (Chen, D. *et al.* 2006). Najm introduced the concept of transition density. It is propagated throughout the circuit using Boolean difference algorithm (Najm 1993). Basically are used the equilibrium probability (the stationary probability that the signal has the value 1) $p(w)$, and transition density, $d(w)$ i.e. the number of times that the signal changes its value per time unit, of each net w in the circuit. This approach is using the Boolean difference of each net w :

$$\begin{aligned} \frac{\partial w}{\partial x_i} &= w(x_0, x_1, \dots, x_i = 0, x_{i+1}, \dots, x_n) \\ &\oplus w(x_0, x_1, \dots, x_i = 1, x_{i+1}, \dots, x_n) \end{aligned} \quad (3)$$

The probability of the Boolean difference $p(\frac{\partial w}{\partial x_i})$ is the probability that a transition at input variable x_i causes a transition at the output w . The transition density at the output w of a gate can be calculated as in (4).

$$d(w) = \sum_{i=1}^m p(\frac{\partial w}{\partial x_i}) d(x_i) \quad (4)$$

This theorem makes possible computing transition density of any node in the logic network $N(V, E)$ when the transition densities at the primary inputs are known. Proof of this theorem can be found in (Najm 1993). Once the transition densities of every node (gate) in $N(V, E)$ have been determined the power consumption of a given circuit can be calculated as in (1). It has to be remarked that this approach is computing an upper bound of the stationary probability in networks when there are *re-convergent fanouts* (Anderson and Najm, 2004). This model also assumes that the switching behavior of input signals are not interrelated, which is usually not true. Using this model is also hard to capture spurious power activity. This model is oriented towards the signal transitions necessities to perform the required logic functions arising between two consecutive clock ticks. Such transitions also are named in literature *functional transitions* (Chen D. et al. 2006).

Characterization through board measurement approach is very well described in (Xu and Kurdahi 1997). Using an emulation board embedded with a Virtex®, Xilinx FPGA, for power measurement the authors calculated the average switching activity for logic elements using a power estimation formula published by Xilinx:

$$P_{\text{int}} = V_{\text{core}} \cdot K_p \cdot f_{\text{Max}} \cdot N_{LC} \cdot \text{To}_{LC} \quad (5)$$

Symbols in (5) are defined as follows:

P_{int} being the internal power consumption caused by charging and discharging the capacitance of each switched element;

- V_{core} is the core voltage;
- K_p is a technology-dependent constant;
- f_{Max} is the maximum clock speed;
- N_{LC} is the number of used logic elements;
- To_{LC} is the average switching activity of all logic elements.

Switching activity determination based on simulation is using several sequences of random generated input vectors applied on the primary inputs and cycle-accurate gate-level simulation can be carried out for the whole circuit. Joint with back-annotated delay information obtainable after placement and routing, this kind of estimation is most precise for switching activity computation because it can also reflect behaviour due to glitches (D. Chen et al. 2006). Many works are using this model (Gupta and Anderson, 2007), (Sutter and Boemo, 2007). It was appreciated that the main difficulty of this approach stays, mainly, in its prohibitive runtime (Chen, D. et al. 2006).

In this work, simulation-based approach was used to determine dynamic switching activity in 15 combinational circuits of the MCNC benchmark. These benchmark circuits are first optimized using *rugged script* in SIS application, an interactive system for the synthesis of sequential circuits (SIS-1.2 1994). It was used modified fault simulator from ATPG section of SIS-1.2 and used for simulation-based logic activity analysis. This simulator has new added capabilities for capturing the number of logic transitions on each net

during simulation, as well as the proportion of time each net spends in the high and low logic states. Simulation with zero logic delays can be done for combinational or synchronous sequential circuits. However it was restricted our research only on combinational circuits but, we intend, in the near future, to extend our research. Circuits are simulated using almost 10 000 randomly generated input vectors. Each time an internal line is incrementing its logic activity (the number of changes from 1 to 0 or from 0 to 1), it is evaluated the following expression:

$$\left| \frac{n}{N} - \frac{n+1}{N+m} \right| \leq \varepsilon \quad (6)$$

In (6) was noted with N the number of simulated vectors after that the number of logic changes, on an arbitrary line w , became n . Let be $N + m$ the number of simulated vectors when the number of logic changes, on line w , is incremented ($n + 1$). Parameter ε is introduced at beginning of the simulation and it represents the precision of simulated-based logic activity determination. The simulator is checking when all internal lines and primary output lines of the circuit fulfill condition (6). The power results that were computed are based on identical switching activities (0.5) for each primary input. But the simulator is able to determine switching activities of network's internal lines for arbitrary switching activities of the primary inputs of any network. As technology scales down, power consumption in interconnects becomes the dominant source in sub-micron FPGAs. In K -LUT based FPGAs circuits the essential dynamic power consumption is caused by transitions that take place at the inputs and outputs of LUTs. It results that the sum of the dynamic power should be minimized over all the LUTs in the mapped network. As a consequence, power estimation for FPGAs has to consider routing interconnect equivalent capacitance. Interconnect estimation becomes increasingly accurate as the design enters lower design levels. In K -LUT based FPGAs circuits the essential dynamic power consumption is caused by transitions that take place at the inputs and outputs of LUTs. It results that the sum of the dynamic power should be minimized over all the LUTs in the mapped network. Nets having the greatest transition density have to be, if possible, hidden in LUTs.

6. ALGORITHM DESCRIPTION

Our algorithm has been implemented in the C language atop the Berkeley SIS framework (SIS-1.2, 1994). The approach is using structures, and routines of SIS-1.2 in order to built-up the programmed application and tune-up appropriate cost functions for *minimal depth mapping power-aware*. The algorithm is based on exhaustive K -feasible cones generation of each arbitrary node u in the mapped network. The implemented technology mapping procedure operates in three steps:

1. In the first step are generated, for each node in the network, the set of all K -feasible cones. K -feasible cones generation is made during a network traverse from primary inputs to primary outputs and compute edge-delay for each feasible cone (Bucur 2007).

2. In the second step are computed specific cost functions of each *K-feasible* cone of each node in the network (Bucur 2009).
3. In the third step, using the set of cost functions values, is determined the *power-aware* minimum depth mapped network.

6.1 Generating *K-feasible* Cones

In order to generate power-aware 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 useful to note that the nodes that are not on a critical path do not need a minimal height *K-LUT* implementation. The generation of all *K-feasible* cones rooted in every node of a node in network has to be considered in the context of network model. 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:

$$\text{input}(C(u)) = \{v_1, v_2, \dots, v_m\}, m \leq K \quad (7)$$

Such a set could be represented as the product (conjunction) of the elements (literals) of the set in (7):

$$p = v_1 v_2 \dots v_m \quad (8)$$

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:

$$\text{cones}(u) = \bigcup_i \text{input}(C_i(u)) \quad (9)$$

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

$$\text{cones}(u) = \bigcup_i v_{i_1} \cdot v_{i_2} \cdot \dots \cdot v_{i_m}, m \leq k. \quad (10)$$

Then it holds this Lemma:

Lemma1. Given a node u having as immediate predecessors: $\text{input}(u) = \{v, w, \dots, z\}$, each predecessor having already computed the set of all *K-feasible* cones, respective $\text{cones}(v)$, $\text{cones}(w)$, ... $\text{cones}(z)$, than the set $\text{cones}(u)$, of all the *K-feasible* cones of node u , is:

$$\text{cones}(u) \subseteq \{(v \cup \text{cones}(v)) \cap (w \cup \text{cones}(w)) \cap \dots \cap (z \cup \text{cones}(z))\} \quad (11)$$

Applications of the Lemma1 are presented in [6]. It was established that this algorithm computes all possible mapping solution for each node (Bucur 1999).

Computing the sum-of-products (SOP) form of the expression (11), 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 . It is not difficult to see that 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 in PMTC_u . In practice,

Table 1. Estimating computing effort in *K-feasible* cone generation

Circuit	Node Count	Depth	Time (seconds)	Max Cone Count	Max Node Count
C432	179	24	0.04	28	7
C499	206	13	0.08	45	15
C880	354	25	0.12	114	11
C1355	518	25	3.50	285	18
C1908	617	27	0.45	147	11
C2670	901	26	0.85	176	17
C3540	1270	41	1.05	327	15
C5315	2120	37	2.22	177	32
C6288	2353	120	10.44	284	21
C7552	2648	30	4.67	672	27

however, most of these combinations do not form cones, since the network connections determinate the cones. 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 Table 1. These circuits were chosen because contain mostly two input gates and are among the largest of the benchmark. Circuit's internal nodes count and depth, were computed after removing (*sweep*) inverters, buffers, and constant nodes. It was followed by simple two inputs AND-OR decomposition of those gates having more than two inputs. The largest circuit in Table 1, namely C6288, having 2120 internal nodes and depth 120, require less than 11 seconds for an exhaustive 5-feasible cone generation rooted in each internal node of this circuit. Time generation, listed in Table 1, was measured on Intel Dual 2 Duo Core T9300. The exhausting generation of all *K-feasible* cones of each internal node of a network leads to a simpler and smoother approach of the power-aware and minimum depth *K-LUT* network's mapping. The 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. Multiple-level circuits, that are interconnections of single-output combinational complex gates, are considered in this work. Multiple-level logic optimization is, usually, divided into two stages (Jiang *et al.* 2009a).

First, the logic is optimized while neglecting the implementation constraints on the logic gates and assuming loose models for their area and performance. Second stage is related to the used technology or gate library. *K-LUT* based FPGA implementation of multi-level networks can be viewed as subject of fan-in *K*-limitation of the used gates. Decomposition is the main approach to obtain a *K-bounded* network. It were used various methods, including Roth-Karp decomposition (Boolean), AND-OR decomposition (algebraic) etc.

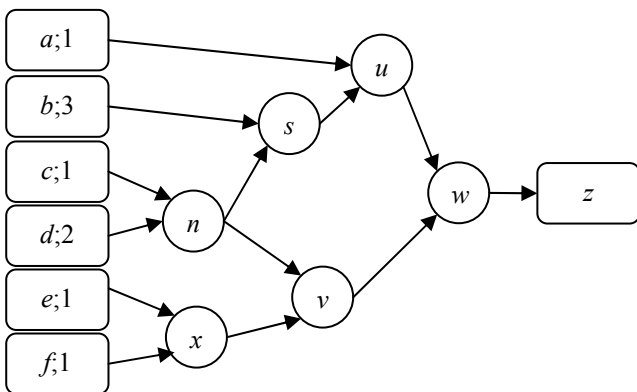
Making a network *K-bounded* is considered, in general, a pre-processing step in the *K-LUT* mapping of FPGAs. In Table 2 are listed part of our experiments made in order to show that decomposition granularity influences the performance of the mapping process. It was used, after

Table 2. Decomposition factor and the influence of it on the minimum mapped depth

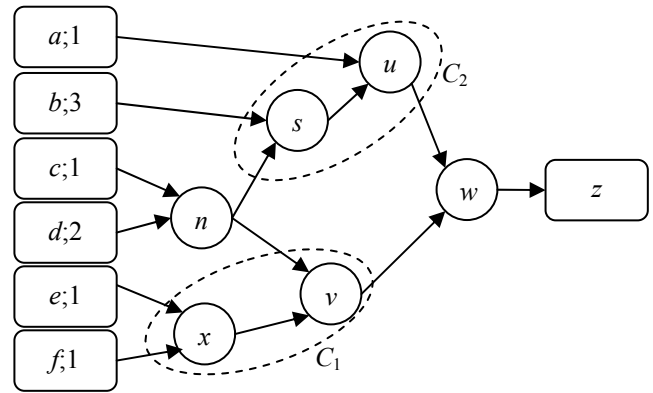
Circuit	Initial depth	Circuit's decomposition parameter and mapping minimum depth			
		$D = 2$	$D = 3$	$D = 4$	$D = 5$
5xp1	2	2	2	2	2
9sym	3	3	3	3	3
C499	10	4	4	4	5
C5315	34	8	8	8	8
C880	19	7	8	9	9
alu2	6	5	6	6	5
alu4	7	5	6	7	7
apex2	11	5	6	6	6
apex4	9	5	6	7	6
apex6	5	4	5	5	5
apex7	5	4	5	5	5
b9	4	3	4	4	4
bw	5	1	1	1	1
clip	4	3	4	4	4
count	4	3	4	4	4
des	8	5	5	6	6
duke2	10	4	5	4	5
e64	9	3	3	3	3
f51m	4	3	4	4	4
misex1	2	2	2	2	2
misex2	5	2	2	2	2
rd73	2	2	2	2	2
rd84	3	3	3	3	3
rot	14	6	7	7	8
sao2	5	4	4	4	4
vg2	4	3	3	4	4

technology independent optimization, AND - OR balanced decomposition of circuits with parameterized number of input lines in each type of gate. Technology independent optimization is made using an approach similar to the optimization made in SIS-1.2 with the *script.rugged* (SIS-1.2, 1994).

Results in Table 2 are showing that making decomposition with $D = 2$ always results best results. This value of the decomposition parameter bring best results

**Fig. 2.** Initial circuit for minimum depth K -LUT mapping ($K=3$).

The main advantage of using presented approach, using 3-LUT, will be pointed out using the simple circuit in Fig. 1. Primary inputs nodes (a , b , c , d , e , and f) contain an additional information concerning the individual delay of each one, in Fig. 1.

**Fig. 3.** Initial circuit for minimum depth K -LUT mapping ($K=3$).

There are two critical paths in this circuit: (b, s, u, w) and (d, n, s, u, w) . Primary inputs nodes (a , b , c , d , e , and f) contain an additional information concerning the individual delay of each one. Both critical paths contain the node u .

The set of all the K -feasible cones of each of the circuit's internal nodes are:

$$\text{cones}(x) = \{(e, f; 2)\},$$

$$\text{cones}(n) = \{(c, d; 3)\},$$

$$\text{cones}(s) = \{b, n; 4\}, (b, c, d; 4)\},$$

$$\text{cones}(v) = \{(n, x; 4), (c, d, x; 3), (n, e, f; 4)\},$$

$$\text{cones}(u) = \{(a, s; 5), (a, b, n; 4)\},$$

$$\text{cones}(w) = \{(u, v; 5), (a, s, v; 5), (u, n, x; 5)\};$$

In the above cone sets, each cone contains a number indicating the minimal height (delay) of the respective cone when the inputs of the cone have attained their minimal height. The node w has three cones, all having the same height. From the three cones of the node w it will be considered, because it has the fewest literals, the first one: $(u, v; 5)$. It results that the node u and the node v height's implementation must not exceed 4. Node u has only one such implementation (node u belongs to both critical paths), namely $(a, b, n; 4)$, while node v has three such possible implementations: $(n, x; 4)$, $(c, d, x; 3)$, and $(n, e, f; 4)$. Note, also, that node v does not belong to any critical path. Since node n was required for the node u implementation and node e and node f are primary inputs nodes, from the three implementations of the node v , only the third one, namely $(n, e, f; 4)$ implementation, will give the optimal area minimum depth 3-LUT mapping of the circuit. In Fig. 2 are marked with two elliptical surfaces the way nodes are chosen to collapse. This optimal area minimum depth circuit implementation will have only four 3-LUTs. In fact, the cone

having the minimum height implementation of the node v will imply a mapping solution having the same depth, but having five 3-LUTs. In Fig. 3 is presented the optimal area and minimal depth, 3-LUT mapping of the considered circuit. It is remarkable that the chosen implementation for the node v is a cone having the minimum possible height for this node (minimum height of the node v is 3).

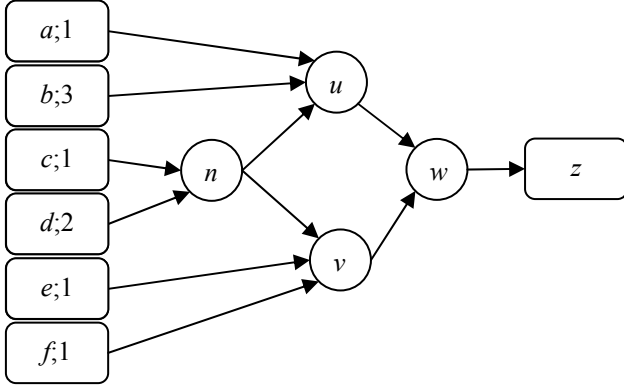


Fig. 4. Mapped circuit for minimum depth K -LUT mapping ($K=3$).

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 used metrics as it were used in previous work (Anderson and Najm 2004) and (Jang *et al.* 2009b). 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. 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 K -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 K -LUT (Anderson and Najm 2002). 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 dynamic 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 :

$$\text{DepthMetric}(\text{cones}(u)) = 1 + \min(\text{DepthMetric}(v \mid v \in \text{cones}(u))) \quad (12)$$

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 following relation:

$$\begin{aligned} \text{EstimPowerCost}(\text{cones}(u)) = \\ \min_{C(u) \in \text{cones}(u)} \left\{ \sum_{v \in \text{Input}(C(u))} (d(u) \cdot \text{fanout}(u) + \right. \end{aligned} \quad (13)$$

$$\left. \text{EstimPowerCost}(\text{cones}(v))) \right\}$$

EstimAreaCost of an arbitrary node u , is used to estimate the area involved for each cone $C(u)$ belonging to the set of K -feasible cones of u , $\text{cones}(u)$:

$$\begin{aligned} \text{EstimAreaCost}(u) = \\ \min_{C(u) \in \text{cones}(u)} (\text{Area}(C(u)) \\ + \sum_{v \in \text{Input}(C(u))} \frac{\text{Area}(v)}{\text{fanout}(v)}) \end{aligned} \quad (14)$$

All three metrics are, in fact, partially computed during the K -feasible cones generation step.

Globally the algorithm is using this parameterized cost:

$$\begin{aligned} \text{GlobalCost}(u) = \\ w_1 \cdot \text{DepthMetric}(u) + \\ w_2 \cdot \text{EstimPowerCost}(u) + \\ w_3 \cdot \text{EstimAreaCost}(u) \end{aligned} \quad (15)$$

Parameters w_1 , w_2 and w_3 , in (15) were experimentally determined. Choosing appropriated values of w_1 , w_2 and w_3 , one can obtain a balanced mapping process.

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. In fact, switching activities, of each input line, are input parameters of our simulator.

Our implemented algorithm did run for mapping into 5-LUT FPGAs several benchmark circuits. Obtained results are presented in Table 3.

Mappers targeting LUT based-FPGAs evolved from the early ones (Bucur 1999) to those needing reduced runtime and enhanced quality of results. Our main idea was to hide into

LUTs nodes having high estimated dynamic consumption power. Main support of this approach is the fact that capacitance inside a LUT is very small and the power consumption will be reduced.

Table 3. Experimental results of PwAwMap mapping tool for FPGA K-LUT.

Circuit	Depth	Estimated Dynamic Power		
		Optimum Depth	Optimal Depth	Optimal Depth & Area
5xp1	3	2,92	2,23	2,36
9symml	5	3,89	3,17	3,55
C499	5	10,76	8,82	9,16
C880	8	16,69	15,17	15,82
alu2	8	14,05	13,04	13,23
apex6	4	22,62	20,58	20,86
apex7	4	10,45	9,87	9,91
count	3	3,52	3,04	3,38
des	5	99,67	97,02	97,27
duke2	4	9,21	9,23	9,29
misex1	2	2,39	2,23	2,29
rd84	4	4,98	4,27	4,42
rot	6	37,68	36,54	36,66
vg2	4	4,32	3,21	3,29
z4ml	3	1,49	1,38	1,42
		244,64	229,8	232,91

To estimate power consumption using (1) it is necessary to know the capacitance of each net or an estimate of it. Obviously, in this stage of designing circuits targeting FPGA mapping, the capacitance of any net it is not known until placement and layout is complete. Most of the power-conscious mappers (Anderson and Najm 2002) and placement and routing power-aware applications (Lamoureux and Wilton 2003) are using switching activity information of the netlist in order to estimate dynamic power dissipation. There are works that are not using nets capacitances or an estimate of them but are concentrating their efforts only toward globally reducing switching activity in the considered network (Jang *et al.* 2009b). In our 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 it results that the fanout factor could be 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 mapping 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:

$$EstimatedDynamicPower(u) = d(u) \cdot fanout(u) \quad (16)$$

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. 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:

$$optimalDepth(u)_{u \in CriticalPath} = optimumDepth + \lambda \quad (17)$$

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 (Farrahi and Sarrafzadeh, 1994a) 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). On average, in Table 3, the detailed experimental results are showing that power-aware mapping for optimal depth, the estimated dissipated power is 7.07% less than mapping for optimum depth. 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 4.80% more efficient (concerning the dissipated power) than mapping for optimum depth. Mapping power-aware both for depth and area optimal it appears 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 development of our research to use dynamic programming together with refined heuristics in PwAwMap algorithm.

REFEERNCES

- Anderson, J.H. and Najm, F.N. (2002). Power-aware technology mapping for LUT-based FPGAs. *IEEE International Conference on Field-Programmable Technology*, pp. 211-218, Hong Kong.
- Anderson, J.H., and Najm, F.N. (2004). Power Estimation Techniques for FPGAs. *IEEE Transactions on VLSI*, Vol. 12, No. 10, pp. 1015-1027.
- Bucur, I. (1999). An Optimal Mapping for delay Optimization of Lookup Table-Based FPGAs. *Proc. of the 12th International Conference on Control Systems and Computer Science*, pp. 127-132.
- Bucur, I. (2007). Performance mapping of k-LUT based FPGAs. *Univ. Politehnica of Bucharest, Scientific Bulletin, Series: C*, Vol. 69, No. 2, pp.49-60.
- Bucur, I., Fagarasan, I., Popescu, C., Boiangiu, C.-A., and Culea, G. (2008). On K-LUT Based FPGA Optimum Delay and Optimal Area Mapping. *Proc. of WSEAS International Conference on Math. and Comput. Methods in Science and Engineering*, 2008, pp.137-142.
- Bucur, I., Stefanescu, C., Surpateanu, A., and Cupcea, N. (2009). Power-Aware and Optimal Depth Mapping of LUT Based FPGA Circuits. *Proceedings of the 17th ICCSCS-17*, May'09, Bucharest, Romania, pp. 117-124.
- Chen, C.-S., Hwang, T.T., and Liu, C.L. (1997). Low Power FPGA Design – A Re-engineering Approach. *Proc. of the 35th DAC*, pp. 656 – 661.

- Chen, D., Cong, J., and Pan, P. (2006). FPGA Design Automation A Survey. *Foundations and Trends® in Electronic Design Automation*, Vol.1, No.3, pp. 195-330.
- Farrahi, A.H. and Sarrafzadeh, M. (1994a). FPGA technology mapping for power minimization," R.W. Hartenstein, and M.Z. Servit (Editors), *Field-Programmable Logic Architectures, Synthesis and Applications*, Springer, Lect. Notes in Comp. Science, Germany, pp.66-77.
- Farrahi, A.H., and Sarrafzadeh, M. (1994b). Complexity of the look-up table minimization problem for FPGA technology mapping," *IEEE Tran. on CAD of IC and Systems*, Vol.13, No. 11, pp.1319 – 1332.
- Gupta, S. and Anderson, J. (2007). Optimizing FPGA Power with ISE Design Tools. *Xcell Journal*, Second Quarter.
- Hansen, L., and Thomas, T. (2005). Complete FPGA and CPLD Power Analysis, *Xcell Journal*, Second Quarter.
- Ho, C.H., Leong, P.H.W., Luk, W., and Wilton, S.J.E. (2008). Rapid Estimation of Power Consumption for Hybrid FPGAs, *Intl. Conf. on FPGA and Apps.*, pp. 227-232.
- Hsieh, C.-T., Cong, J., Zhang, Z., and Chang, S.-C. (2008). Behavioral Synthesis with Activating Unused Flip-Flops for Reducing Glitch Power in FPGA. *Proc. ASP- DAC*, pp. 10-15.
- Jang, S., Chung, B., Chan, K., and Mishchenko, A. (2009a). WireMap: FPGA Technology Mapping for Improved Routability and Enhanced LUT Merging. *ACM Transactions on Reconfigurable Technology and Systems*, Vol.2, No.2, Article 14, June'09.
- Jang, S., Chung, B., Chan, K., Mishchenko, A., and Brayton, R. (2009b). A power Optimization Toolbox for Large Synthesis and Mapping. *Proc. IWLS'09*, pp. 1-8.
- Jiang, W., Zhang, Z., Potkonjak, M., and Cong, J. (2008). Scheduling with Integer Time Budgeting for Low-Power Optimization. *Proc. ASP-DAC'08*, pp. 22-27.
- Jones, P.H., Cho, Y.C., and Lockwood, J.W. (2007). Dynamically Optimizing FPGA Applications by Monitoring Temperature and Workloads. *Proc. 20th Intl. Conf. on VLSI Design and 6th Intl. Conf. Embedded Systems*, Bangalore, India, Jan 6-10, pp. 391-400.
- Lamoureux, J. and Wilton, S. (2003). On the Interaction Between Power-Aware FPGA CAD and Algorithms. *Proc. IEEE/ACM ICCAD'03*, pp. 701-708.
- Li, H., Mak, W.-K., and Katkoori, S. (2001). LUT-Based FPGA Technology Mapping for Power Minimization with Optimal Depth. *Proc. IEEE CS Workshop on VLSI*, Orlando, FL, April 19-20, pp. 123-128.
- Mashayekhi, M., Jeddi, Z., and Amini, E. (2008). Power Optimization of LUT based FPGA circuits. *Proc. 11th IEEE Intl Conf. On Optimization of Electrical and Electronic Equipment, Optim Brasov*, Romania, pp. 37-40.
- Najm, F.N. (1993). Transition Density: A New Measure of Activity in Digital Circuits. *IEEE Trans. on CAD of IC and Systems*, Vol.12, No. 2, pp. 310-323.
- Pedram, M. (1996). Power minimization in IC design: Principles and Applications," *ACM TODAES*, Vol. 1, No.1, pp. 3-56.
- Poon, K., Wilton, S., and Yan, A. (2005). A Detailed Power Model for Field-Programmable Gate Arrays," *ACM TODAES*, Vol. 10, Issue 2, pp 279-302.
- Rose, J., Francis, R., Lewis, D., and Chow, P. (1990). Architecture of Field Programmable Gate Arrays: The effect of Logic Block Functionality on Area Efficiency. *IEEE J. of Solid State Circuits*, Vol. 25, No. 5, pp. 1217-1225.
- Singh, A. and Marek-Sadowska, M. (2002). Efficient Circuit Clustering for Area and Power Reduction in FPGAs. *International Symposium on FPGAs*, 2002, pp. 59 – 66.
- SIS-1.2. (1994). <http://embedded.eecs.berkeley.edu/pubs/downloads/sis/index.htm>.
- Sutter, G., Boemo, E. (2007). Experiments in Low Power FPGA Design. *Latin American Applied Research*, Vol. 37, No. 1, pp. 99-104.
- Sasao, T., Mishchenko, A. (2009). LUTMIN: FPGA logic synthesis with MUX-based and cascade realizations, *Proc. IWLS'09*, pp. 310-316.
- Wang, Z.-H., Liu, E.-C., Lai, J., and Wang, T.-C. (2001). Power minimization in LUT-based FPGA technology mapping," *Proc. ASP-DAC'01*, pp. 635-640.
- Xu, M. and Kurdahi, F. J. (1997). ChipEst-FPGA: a tool for chip level area and timing estimation of lookup table based FPGAs for high level applications. *Proc. ASP-DAC '97*, pp. 435-440.
- Yang, A. (2005). Design Techniques to Reduce Power Consumption," *Xcell Journal*, Third Quarter.