FPGA Power and Timing Optimization: Architecture, Process, and CAD
Chun Zhang, Lerong Cheng, Lingli Wang* and Jiarong Tong

Abstract—Field programmable gate arrays (FPGAs) allow the same silicon implementation to be programmed or re-programmed for a variety of applications. It provides low NRE (non-recurring engineering) cost and short time to market. As CMOS technology continues to scale down to nanometer, increased power consumption and worsened process variation become crucial constraints for FPGAs. The survey reviews the process and architecture evaluation and CAD algorithms to optimize power and delay for FPGAs.

I. INTRODUCTION

Since being invented in 1984, FPGAs have become one of the most popular implementation media for digital circuits and have grown into two billion US dollars per year industry [1]. The continuous scaling of silicon-based integration technology enables the ever growing number of transistors on a single chip to implement circuits and systems. For Application Specific Integrated Circuits (ASICs), the same chip can be used only for one circuit or system. This leads to a high NRE cost, measured in over one million US dollars per application for the current leading silicon technology and expected to grow with silicon scaling. On the other hand, programmable logic fabrics such as FPGAs where the same chip can be programmed or re-programmed into a variety of circuits and systems, have a much lower NRE cost and also a much shorter time to market. Therefore, FPGAs have started to gain more and more market shares from ASICs.

However, in order to achieve programmability, FPGA pays the penalty with decrease of performance, power, and area. Due to the large number of transistors required for field programmability and the low utilization rate of FPGA resources (typically 62.5% [2]), there are a 100X energy difference, a 4.3X delay difference, and a 40X area difference between FPGA designs and their ASIC counterparts [3], [4]. As the process advances to nanometer technologies, power consumption is a crucial design constraint for nano-scale VLSI circuits. The power problem is more significant for FPGAs than ASICs because FPGA has higher power consumptions. There are lots of existing works on optimizing timing and power for FPGAs. In this survey, we review the two major types of techniques for FPGA power and delay optimization: 1) process and architecture evaluation; 2) computer-aided design (CAD) algorithms.

This survey is organized as follows: Section II first introduces some preliminaries of FPGAs; then Section III reviews existing process and architecture evaluation techniques; Section IV presents FPGA CAD algorithms; finally Section V concludes this survey.

II. PRELIMINARIES

A. Conventional FPGA Architecture

The most popular architecture for FPGAs is cluster-based island style FPGA architecture [5]. A cluster-based logic block (see Figure 1) includes several fully or partially connected Basic Logic Elements (BLES). Each BLE includes one K-input lookup table (LUT) and one flip-flop (DFF). The logic blocks are surrounded by routing channels consisting of wire segments. The input and output pins of a logic block can be connected to the wire segments in routing channels via a connection block (see Figure 1 (b)). A routing switch block is located at the intersection of a horizontal channel and a vertical channel. Figure 1 (c) shows a subset switch block [6], where the incoming track can be connected to the outgoing tracks with the same track index. The connections in a switch block (represented by the dashed lines in Figure 1 (c)) are programmable routing switches. The routing switches can be implemented as bi-directional tri-state buffers (Figure 1 (d)), uni-directional tri-state buffers, or pass-transistors. interconnect segment is a wire segment driven by a tri-state buffer. Usually, the LUTs are referred to as logic circuit elements, interconnect within logic blocks is referred to as local interconnect, and interconnect outside logic blocks is referred to as global interconnect.

B. FPGA Power and Delay Analysis Framework

Power and delay analysis framework is necessary for FPGA power and delay optimization. A popular FPGA timing evaluation tool is the static timing analysis engine in the timing driven Versatile Placement and Routing (VPR) tool [5]. VPR provides the critical path delay report after performing placement and routing for an FPGA.

In the past decade, there are several research work on power analysis for FPGAs [7]–[10]. fpgaEVA-LP [10] is one of the most popular frameworks. Figure 2 illustrates the overall analysis for fpgaEVA-LP. For a given circuit, SIS [11] is used to perform the technology independent logic optimization and use Flowmap [12] in RASP [13] to conduct the technology-mapping. Then physical design in VPR [5], including timing-driven packing, placement and routing is performed to generate the BC-netlist (Basic Circuit Netlist) back-annotated with post-layout resistance and capacitance. With the BC-netlist, cycle accurate power simulation is performed to estimate the chip level power.
Fig. 1: (a) Island style routing architecture; (b) Connection block; (c) Switch block; (d) Routing switches.

III. PROCESS AND ARCHITECTURE EVALUATION

A. Low Power FPGA Circuits

Due to the low utilization rate, the leakage power consumption is significant for FPGAs than that for ASICs. As a result, the problem of leakage power reduction was first studied in the literature. Region based power gating for FPGA logic blocks [14] and fine-grained power-gating (PG) for FPGA interconnects [15] were proposed. In power-gating circuits, a gating transistor is placed between power supply and circuit elements to shut down the power supply for the unused circuit elements. A new type of routing multiplexer and an input control method were developed [16] to reduce leakage of unused routing multiplexers, and a circuitry combining gate biasing, body biasing and multi-threshold techniques was designed [17] to reduce interconnect leakage.

Besides leakage power reduction, dynamic power reduction is also studied. A dual-Vdd FPGA architecture was studied in [8], [18]–[22]. The dual-Vdd FPGA circuits are shown in Figure 3 and 4. In these circuits, two transistors are placed between supply voltage and circuit elements to choose Vdd level. In dual-Vdd FPGA, most non-critical path elements are assigned low Vdd to save dynamic power. Moreover, for the low Vdd circuit elements to drive high Vdd circuit, level shifter (LS) is needed. Moreover, [22] introduced dual-Vdd circuit without level converters, which further reduces power and area introduced by level converters. Notice that here, dual-Vdd circuits can achieve not only Vdd selection but also power gating. Figure 5 [22] compares power consumption of different FPGA circuits. It can be seen that dual-Vdd technique significantly reduces FPGA power consumption.

Fig. 2: FPGA power analysis framework.

Fig. 3: (a) Configurable level shifter; (b) Vdd-programmable routing switch; (c) Vdd-programmable connection block. (SR stands for SRAM cell and LC stands for level shifter.)

Fig. 4: (a) Vdd-programmable switch; (b) SRAM-efficient Vdd-programmable routing switch; (c) SRAM-efficient Vdd-programmable connection block.
process, there are more than six parameters to be evaluated, much higher impact on circuit power and performance. To benchmark set, logic was optimized and circuit was mapped between 4 and 10 can produce the best area-delay product achieves the smallest area [23] and an LUT size of 5 or 6 leads to the best performance [24]. Later on, the cluster-based island style FPGA was studied to optimize area-delay product and it showed that LUT sizes ranging from 4 to 6 and cluster sizes between 4 and 10 can produce the best area-delay product [25]. Besides area and delay, FPGA architecture evaluation considering energy was studied recently [7]–[10]. An FPGA architecture evaluation flow considering area, delay, and power was introduced in [10], as shown in Figure 6. For a given benchmark set, logic was optimized and circuit was mapped to a given LUT size. Then time driven packing (TV-Pack) is used to pack the mapped circuit to a given cluster size. After packing, VPR is used to place and route the circuit. Finally, cycle-accurate power simulator [7] is used to estimate the chip level power consumption. People may apply the above flow for different FPGA architectures to obtain the timing and power. Then the optimum architecture can be chosen by evaluating the timing and power values. [26] further evaluated FPGA architectures with field programmable dual-Vdd and power gating, and considering area, delay, and energy. It was shown that in 0.35µm technology, an LUT size of 3 consumes the smallest energy [9]. In 100nm technology, an LUT size of 4 consumes the smallest energy [10].

C. Process and Architecture Co-Optimization

Compared to architecture and circuit design, process parameters, such as Vdd, gate channel length, dopant density, have much higher impact on circuit power and performance. To further reduce power and delay, process and architecture co-optimization is studied [27]–[29]. Combining architecture and process, there are more than six parameters to be evaluated, which leads to huge amount of combinations. In this case, the architecture evaluation flow discussed above is too slow to perform such evaluation. [27]–[29] present a timing efficient power and delay evaluation flow called Ptrace. The basic idea is to perform traditional architecture evaluation flow (as in Figure 6) under one set of process parameters to generate the trace information and reused the trace information for other process parameters to save the simulation time. The Ptrace is shown in Figure 7. It is shown that for 65nm technology, applying architecture and process co-optimization reduces energy delay product by 55%.

IV. CAD Algorithms

A. Deterministic CAD Algorithms

Compared to ASICs, the programmability of FPGAs offers a unique opportunity to perform post-silicon design optimization. Low power and high performance CAD algorithms for FPGAs have been studied for a decade.

By nature, the dynamic power consumption is defined by equation (1), where $f$ is the circuit frequency, $V_{dd}$ is the circuit working voltage, and $C_i$ is the load capacitance of circuit
node $i$. The transition density, $E_i$, defines the frequency signal switching (either from high to low or from low to high) of each node. It is easy to observe that by reducing $E_i$, the total power consumption can be reduced.

$$P_{\text{dyn}} = \frac{1}{2} f V_{dd}^2 \sum_{i=1}^{n} C_i E_i$$  \hspace{1cm} (1)

The interaction of a suit of power-aware FPGA CAD algorithms without changing the existing FPGAs was studied in [30]. Based on that fact that inter-CLB/LUT interconnects have much larger capacitance than intra-CLB/LUT resources, this work aims at absorbing those high transition density nodes into same CLB/LUT in different CAD tasks like technology mapping, placement, etc. In that case, they’re associated with lower capacitance value and lead to reduction in power consumption.

Take the placement problem as an example. The original timing driven placement algorithm incorporates both wiring cost and timing cost into consideration, which are defined by equation (2) and (3).

$$\text{Wiring Cost} = \sum_{i=1}^{N_{\text{nets}}} q(i) \cdot [bb_x(i) + bb_y(i)]$$ \hspace{1cm} (2)

$$\text{Timing Cost} = \sum_{\forall i,j \in \text{circuit}} \text{Delay}(i,j) \cdot \text{Criticality}(i,j)^{CE}$$ \hspace{1cm} (3)

The wiring cost is included to punish the appearance of long nets (i.e., nets with large bounding box), while the timing cost helps placing timing criticality components nearby to enhance the circuit performance. To make the placer power aware, a third cost component, the power cost, is also as well introduced. Equation (4) formulates the power cost function. By multiplying the bounding box and its transition density for each net, it estimates the power consumption of different nets.

$$\text{Power Cost} = \sum_{i=1}^{N_{\text{nets}}} q(i) \cdot [bb_x(i) + bb_y(i)] \cdot D_i$$ \hspace{1cm} (4)

Finally, all three cost components are incorporated into the overall placement cost function, which is given by equation (5).

$$\text{Cost} = \alpha \cdot \frac{\Delta \text{Timing Cost}}{\text{Previous Timing Cost}} + \beta \cdot \frac{\Delta \text{Wiring Cost}}{\text{Previous Wiring Cost}} + (1 - \alpha - \beta) \cdot \frac{\Delta \text{Power Cost}}{\text{Previous Power Cost}}$$  \hspace{1cm} (5)

The coefficients $\alpha$ and $\beta$ adjust the importance of each cost component, thus making the trade-off among area, performance and power consumption of the final placement result. Following the same idea of reducing $\sum C_i \cdot E_i$, similar power cost is added to technology mapping and routing algorithms as well.

Later on, A configuration inversion method to reduce the leakage power of multiplexers without any additional hardware [31] was investigated. As low power FPGA circuits, such as power gating, body bias and dual-Vdd are introduced, there are more flexibility for CAD algorithms to reduce power and delay. Power-driven partition algorithm for mapping applications to FPGAs with different Vdd-levels [32] was proposed. [20], [22] propose linear programming based Vdd assignment algorithms to save power.

### B. Statistical CAD Algorithms

As technology scales down, process variation becomes more and more significant. To handle process variation, statistical CAD algorithms are presented: [33] presents a statistical FPGA placement algorithm to improve timing yield. [34] proposes a physical synthesis flow to consider both process variation and pre-routing interconnect uncertainty. [35] introduces a statistical dual-Vdd assignment algorithm to improve both timing and leakage yield. Leveraging the programmability of FPGAs, [36] provides a chipwise placement flow to improve FPGA performance, as shown in Figure 8. The basic idea of this flow is to perform different placement for different chips according to the chips’ variation information. For a given set of FPGA chips, a set of sample chips are tested to characterize process variation. Based on such information, the potential delay improvement of chipwise placement is estimated. If the improvement is large, it is worthwhile to perform placement for each chip. In this case, the variation map for each chip is generated by synthesizing test circuits for each chip. On the other hand, when the improvement is not significant, the conventional design flow, which uses the same placement and route for all chips, will be applied. It is shown that chipwise FPGA placement improves circuit performance by up to 12.1%.

![Fig. 8: Design flow of variation aware chipwise placement.](image)

### V. CONCLUSION

This survey has reviewed the existing research works on optimizing process, architecture, and CAD algorithms to reduce power and improve performance for FPGAs. Although the optimization techniques have improved dramatically in the past decade, as CMOS technology level scales down, the fundamental questions, such as power, performance, variation, and reliability become more and more significant. It requires further studies to solve these problems.