Chapter 10
Low-Power Techniques for FPGAs

Nikil Mehta and André DeHon

Abstract Field-programmable gate arrays (FPGAs) are reconfigurable devices that can be programmed after fabrication to implement any digital logic. As such, they are flexible, easy to modify in-field, and cheaper to use than manufacturing a customized application-specific integrated circuit (ASIC). However, this programmability comes at a cost in terms of area, performance, and perhaps most importantly power. As currently manufactured, FPGAs are significantly less power efficient than ASICs. Fortunately, in the last decade concentrated attention to power consumption has identified many approaches to power reduction. This chapter surveys the techniques and progress made to improve FPGA power efficiency.

10.1 Introduction

As integrated circuit design becomes more complex, time-consuming, and expensive, FPGAs offer several advantages over ASICs: faster time to market, lower non-recurring engineering costs, early access to advanced technology, and post-fabrication programmability that allows task specialization and field upgrades. However, the flexibility and programmability of FPGAs come at a cost in terms of area, performance, and power. We will see in Section 10.2.2 that the power dissipation gap between FPGAs and ASICs has been estimated to be 7–14× on average [30]. This gap becomes increasingly important as energy, power, and power density become dominating concerns for electronic systems. Understanding and reducing this near order of magnitude gap has been the subject of intense research by the FPGA community.

N. Mehta (✉)
Department of Computer Science, California Institute of Technology, Pasadena, CA, USA
e-mail: nikil@caltech.edu

While many of the low-power design techniques developed for ASICs mentioned in prior chapters can be directly applied to FPGAs, there are a number of FPGA centric optimizations that yield additional power savings. These techniques target specific characteristics of FPGAs across all levels of design. By combining and exploiting these techniques we can narrow the power gap.

This chapter will outline the challenges in decreasing FPGA power dissipation (Section 10.1) and progress made in FPGA power reduction across all levels of design, from top to bottom: CAD (Section 10.3), architecture (Section 10.4), circuits (Section 10.5), and devices (Section 10.6). Section 10.7 summarizes the key developments and future challenges in FPGA power reduction.

10.2 FPGAs Power Dissipation

In the next two sections we provide a brief overview of FPGA architecture, sources of power dissipation (both qualitatively and quantitatively), and the power gap between FPGAs and ASICs.

10.2.1 FPGA Overview

A conventional, island-style FPGA can be viewed as an array of configurable logic blocks (CLBs) connected by programmable interconnect (Fig. 10.1). Desired logic functions are programmed into look up tables (LUTs) (Fig. 10.2a) inside the CLBs and wired together by programming switches (Fig. 10.2b) in connection boxes and switch boxes in the interconnect. Any circuit can be realized; however, to support this programmability the FPGA must provision significantly more resources than those required by a custom designed and fabricated implementation. This resource overhead costs area, performance, and power.

As with ASICs, power dissipation in an FPGA can be separated into static and dynamic power (assuming short circuit power is negligible); we can express power

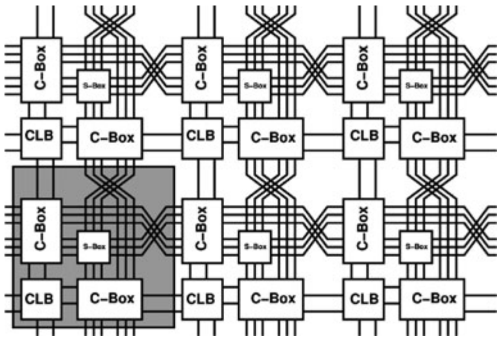


Fig. 10.1 Island-style FPGA architecture

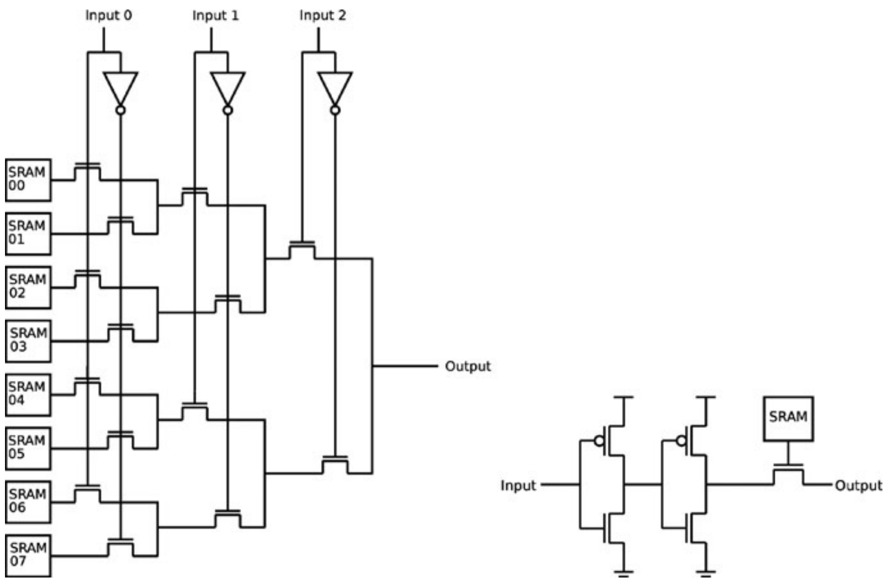


Fig. 10.2 FPGA logic and switch circuits. (**a**) 3-input LUT (**b**) Switch

as the sum of the canonical power equations (Chapter 2, Equations (2.2) and (2.10)) over all N transistors in the chip:

$$P_{\text{dynamic, chip}} = (V_{\text{dd}})^2 f \left(\sum_{i=0}^{N} \alpha_i C_i \right) \tag{10.1}$$

$$P_{\text{static, chip}} = V_{\text{dd}} \left(\sum_{i=0}^{N} I_{\text{leak},i} \right) \tag{10.2}$$

where α is switching probability, C is switched capacitance, f is frequency, I_{leak} is average leakage current, and V_{dd} is the power supply. An important element missing from this model is the amount of power dissipated due to spurious logic transitions, or glitches. The probability α_i here assumes that node i is switching at most once per cycle, and that every switching event is necessary to perform the desired logic function. In reality, because combinational paths are often unbalanced, internal nodes in a combinational block may switch several times per cycle, depending on the inputs to the block. We will revisit glitch power in Section 10.3.6.

The main sources of power dissipation in an FPGA are the logic, interconnect, clock network, and configuration memory. Modern FPGAs also consume non-negligible power in embedded blocks (memories, multipliers, etc.) and in I/O drivers. Several studies have estimated the power dissipated in each of these elements to provide a breakdown of both static and dynamic power dissipation [31, 48, 52]. Each study arrives at similar breakdowns for power dissipation; Fig. 10.3 shows

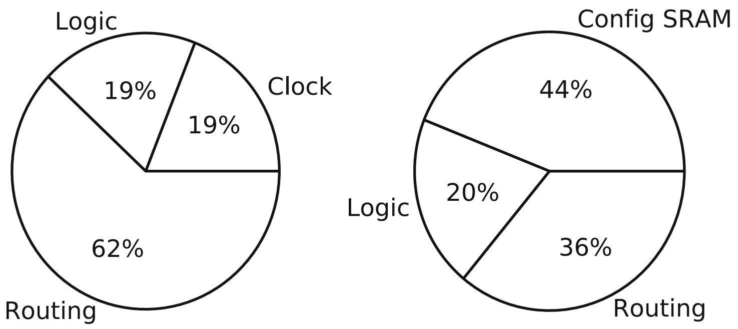


Fig. 10.3 Measured power breakdown for 90-nm Xilinx Spartan-3 [52]. (**a**) Dynamic Power (**b**) Static Power

the most recently published results for a 90-nm Xilinx Spartan-3 [52]. Dynamic and static power charts are shown; in this particular device dynamic power is typically an order of magnitude larger than static power. However, for more recent, high performance devices static power is expected to dominate total power. In Fig. 10.3a we see that interconnect dominates dynamic power, contributing 62%. Clock and logic power are the next two largest contributers and are roughly equivalent at 19%. For static power (Fig. 10.3b) configuration bits contribute the largest amount of leakage at 44%, as every switch requires a configuration bit and every k-input LUT requires 2^k bits. Routing and logic transistors also leak significantly. If we assign the static power of configuration bits to their sources (either interconnect or logic), interconnect static power will dominate. Hence, interconnect is the primary source of both dynamic and static power dissipation.

While interconnect can often be a major driver of power in ASICs, the need for active elements to buffer and switch interconnect in FPGAs makes it much clearer how much energy goes into the interconnect. This effect is exacerbated by the need to provision adequate programmable interconnect resources to support a wide range of designs. In the next section we will examine the cost of this programmability in how FPGA power compares to ASIC power.

10.2.2 FPGA/ASIC Power Gap

While the configurability, time-to-market, and low NRE advantages of FPGAs are compelling, many power-sensitive designs are still drawn to ASICs. When solving identical problems, ASICs consume less power.

The main source of power dissipation in both dynamic and static cases is the added capacitance associated with programmability. This additional capacitance comes both from used and potential switches in the interconnect, and from longer wires that result from the area overhead for programmable switches and configuration memories. The area required to implement an FPGA LUT and flip-flop using standard cells will be similar to the FPGA's LUT and flip-flop area [30]. However, from Fig. 10.1 we see that to connect two logic elements, instead of using and driving a single wire as in the ASIC case, in the FPGA we may need to route signals through several interconnect switches, each of which drives the input capacitance of several other switches.

For the static power case, if we assume that all transistors have identical characteristics (i.e. no variation), then I_{leak} is the same for all devices (Chapter 2, Equation (2.10)), and consequently P_{static} (Equation (10.2)) depends solely on N, or area. The main source of area overhead in an FPGA is again the programmable interconnect switches along with their configuration bits, which are not needed in an ASIC.

Kuon and Rose [30] begin to quantify the FPGA/ASIC power gap by synthesizing a set of benchmark circuits for both a 90-nm FPGA and a 90-nm ASIC process using a standard, commercial EDA flow without any specific power optimizations. They compared area, delay, and dynamic power of the ASIC to that of the FPGA in several different configurations: logic only and logic with different combinations of modern embedded structures (memory and DSPs). Table 10.1 summarizes their results, showing that on average a logic only FPGA without any optimizations utilizes 14× the dynamic power of a process equivalent ASIC. With non-configurable embedded memories and DSPs this gap decreases to 7×. Embedded elements do not contain programmable interconnect, saving capacitance and hence dynamic power. When comparing static power, they found that the gap is roughly correlated (correlation coefficient of 0.8) to the area overhead of $18 - 32\times$, with embedded blocks again reducing power overhead.

Table 10.1 FPGA/ASIC gap [30]

	Ratio (FPGA/ASIC)			
Metric	Logic only	Logic & DSP	Logic & memory	Logic, memory & DSP
Area	35	25	33	18
Delay	3.4	3.5	3.5	3.0
Dynamic power	14	12	14	7.1

The Kuon and Rose study carefully guarantees that the FPGA and ASIC are implemented in the same technology and solving the same problem. This provides a useful baseline for comparison. FPGAs, however, can typically exploit two advantages that are not expressed in this baseline. First, FPGAs may be able to use a smaller feature size technology that has lower capacitance. For example, state-of-the-art FPGAs are now manufactured in 40-nm technologies, while many ASICs are still using 130-nm technology because of the cost and difficulty of deep submicron design. Second, an FPGA may be able to solve a more specialized problem than the ASIC. For some problems it is possible to configure the FPGA and solve a specific instance of a problem rather than all instances. For example, ASIC filters must generally support programmable coefficients, while the FPGA implementation can be specialized to the particular coefficients needed for a special programming task. This results in substantial area reduction for the FPGA with commensurate power reduction [14] [23, chapter 22]. A 30% power advantage due to a smaller feature size technology coupled with an 80% power reduction due to specialization to a specific instance of the problem reduces FPGA power to $(1-0.3)\times(1-0.8)=0.14$ of the same-problem and same-technology power and could close a $7\times$ raw power gap.

While FPGAs consume more power than equivalent ASICs for identical technologies and solution implementations, FPGAs can consume significantly less power than conventional CPUs when performing identical tasks in identical technologies. CPUs put substantial energy into instruction switching, cycling of large memories, and control circuitry aimed at latency reduction. Their wordwide architecture also prevents them from matching their computation to the particular needs of a problem. For example, Abnous et al. compare energy consumption on a low-power embedded ARM processor to an FPGA for digital-signal processing functions, showing that the processor consumes over $15\times$ the energy of the FPGA [2]. A generalized comparison between the FPGA/CPU as in the Kuon FPGA/ASIC study is still future work.

10.3 CAD

FPGA CAD optimizations have the potential to reduce costs without demanding any changes in the FPGA itself. In that sense, any gains they offer are immediately applicable without redesigning the FPGA or designing and fabricating a new family of chips. Since they do not change the physical chip, any tradeoffs arising at the mapping phase are elective, allowing the consumer to select the tradeoff that is appropriate to the problem. Consequently, there is a wealth of recent researches aimed at optimizing FPGA CAD algorithms to synthesize and map circuits that dissipate less power. The next several sections summarize these efforts in the FPGA CAD from start to finish: technology mapping, clustering, place, and route. Most of these techniques focus on reducing dynamic energy; however, one (Section 10.3.3) reduces static power.

Standard, technology-independent CAD transformations are used for both ASICs and FPGAs to reduce area, delay, and energy. While the FPGA and ASIC algorithms differ mainly in their relative cost models, in some cases unique FPGA features create both additional challenges and opportunities for optimization. Originally, many existing FPGA CAD mapping algorithms were designed to optimize area and delay, not to explicitly reduce energy consumption. Fortunately, many of these algorithms are already based around a cost function, and this function can be adapted to target energy by adding terms accounting for the energy impact of choices and transformations. A common theme in many of these optimizations is to exploit implementation freedom to reduce the capacitance associated with high activity (α) nodes.

10.3.1 Technology Mapping

FPGA designs typically begin with a netlist of gates and registers created by hand or more commonly compiled from a high-level language (e.g., VHDL, Verilog).

The technology mapping step of the FPGA CAD flow transforms this netlist into an FPGA targeted netlist of k-input LUTs and flip-flops. Algorithms partition the gate netlist into pieces and determine how to best “cover” sets of gates with LUTs. Technology mappers are most often concerned with minimizing area by reducing the total number of mapped LUTs and with minimizing delay by reducing logic depth (i.e., the maximum number of LUTs in series).

The goal of low-power technology mapping is to reduce dynamic power (Equation (10.1)) by either minimizing the switching activity of connections between LUTs (α_i) or reducing the total number of these connections and therefore the overall capacitance (C_i). Because connections between LUTs are highly capacitive due to the large number of switches attached to a wire segment (Fig. 10.1), it would be beneficial to both activate these segments as infrequently as possible and to reduce the total number of segments.

Previous work has reduced interconnect switching activity by identifying the highest activity connections between gates and mapping in such a way that these nodes are internal to a LUT [17, 33, 40]. The ability to hide internal nodes in compute blocks is an unique opportunity that arises in the FPGA that is not present in ASIC optimization. Figure 10.4 demonstrates an example of two ways of performing technology mapping on a set of gates with nets annotated with switching activity factors. Both mappings produce the same number of LUTs (area) and the same logic depth (delay). However, the power optimized mapping hides the $\alpha = 0.90$ node within a LUT, reducing the mapped circuit switching activity and power.

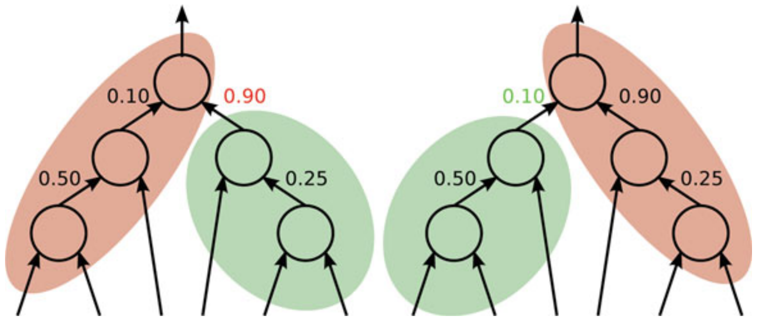


Fig. 10.4 Low-power FPGA technology mapping. (**a**) Unoptimized for power; (**b**) Optimized for power

Other work has reduced the total number of interconnect segments by limiting duplication [4], a common technique used in technology mapping. Figure 10.5 demonstrates an example of mapping a set of gates to 4-LUTs with and without duplication. Gates X and Y must be covered with two separate LUTs because LUTs only have one output. Gate B fans out to both X and Y, and without duplication must be covered with a third LUT. However, if we duplicate B we can place one B gate in the X LUT and the other in the Y LUT, reducing the total number of LUTs by one. Duplication is necessary for depth optimal mapping and can further aid in hiding high activity nodes within LUTs. Unfortunately, duplication can sometimes result

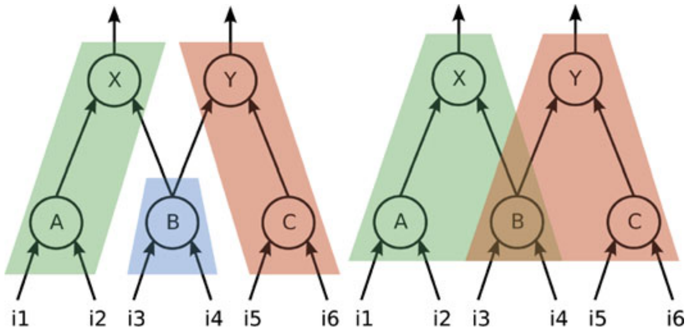


Fig. 10.5 Duplication in technology mapping [4]

in a net energy increase because of increased routing capacitance. In our example, without duplication signals i1–i6 must only connect to six LUT input pins. With duplication, however, inputs i3 and i4 must connect to two additional input pins, for a total of 8 pins. Since these signals are arriving from highly capacitive general purpose interconnect, dynamic energy may increase. This can be avoided by only performing duplication when it results in a net power reduction.

In general, low-power technology mapping has been shown to yield approximately $7.6-17\%$ power savings using both switching activity reduction and limited duplication [33, 40].

10.3.2 Clustering

After producing a netlist of LUTs and flip-flops from technology mapping, these elements must be clustered together to produce a netlist of CLBs. Modern FPGAs typically have between 4 and 10 LUTs per CLB, with each LUT having the option to register its output. Connections between LUTs are highly capacitive when those LUTs are located in different CLBs as signals must travel through global interconnect switches (Fig. 10.1). However, when LUTs are located within the same CLB, connections use local interconnect that has lower capacitance, which is more power efficient. Figure 10.6 shows an example of how inter-CLB connections are organized.

Conventional clustering algorithms attempt to either minimize area by reducing the total number of CLBs by packing them as full as possible, or to minimize delay by packing LUTs on the critical path within shared clusters. A commonly used clustering algorithm for FPGAs, T-VPack [8], uses a greedy approach that selects the least-cost LUTs to pack into a CLB. The cost of a LUT is determined by its criticality and the number of connections shared with LUTs in a cluster; LUTs on the critical path or LUTs that share many connections are prioritized.

Low-power clustering operates on a similar principle as low-power technology mapping. In technology mapping the goal is to cover high activity gate-to-gate connections within the same LUT to hide the switching power of these nodes. In

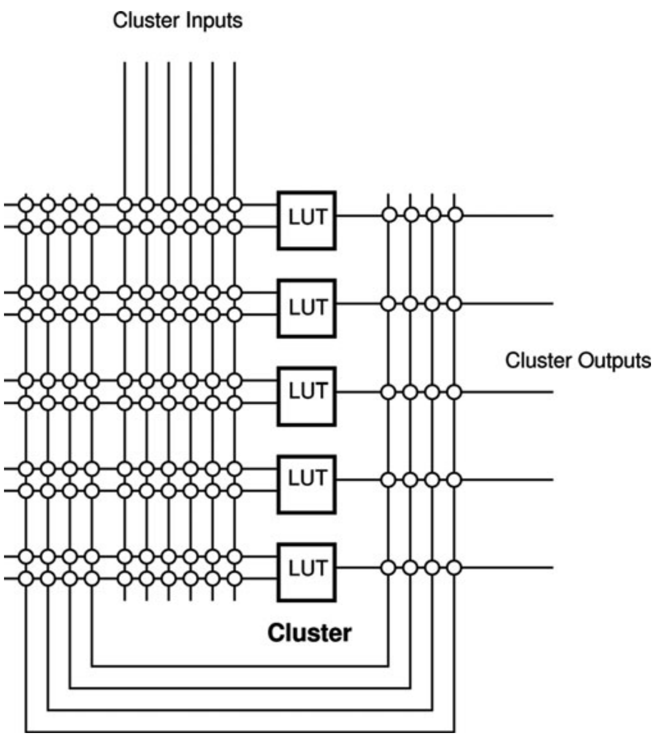


Fig. 10.6 CLB Local interconnect

low-power clustering, the goal is to place high activity LUT to LUT connections within the same CLB, utilizing the more energy efficient local interconnect. This can be accomplished by modifying the cost function of T-VPack to assign minimum cost to the LUTs with the highest activity inputs and outputs [33]. Lamoureux et al. demonstrated that for a cluster size of 4, an average energy reduction of 12.6% is achievable.

An alternative technique to reduce power through improved clustering is to depopulate CLBs, or remove LUTs from fully occupied CLBs and place them elsewhere. Depopulation has been shown to reduce routing resource demand, decreasing both average wirelength and switch utilization [15]. Approximately 13% total power can be saved through depopulation-based clustering [49].

10.3.3 LUT Input Transformations

Several leakage current paths exist in pass transistor multiplexer-based LUTs (Fig. 10.2a). The magnitude of leakage current flowing through these paths depends

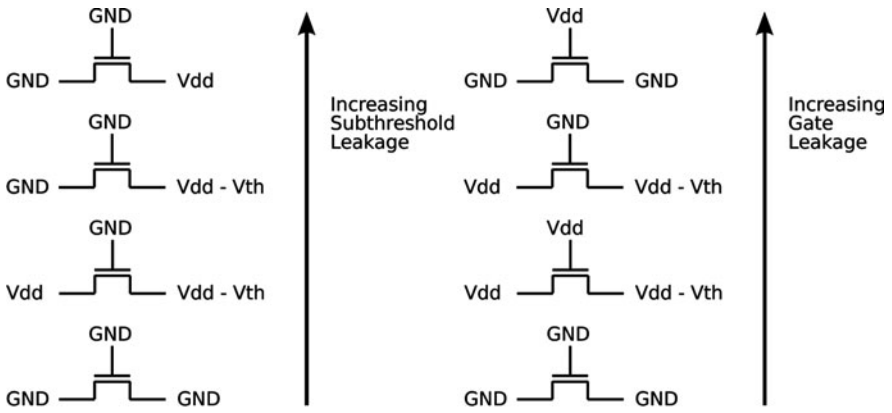


Fig. 10.7 Leakage paths in NMOS pass transistors [22]. (**a**) Subthreshold leakage; (**b**) Gate Leakage

heavily on the voltages on each of the pass transistor nodes. Gate leakage and subthreshold leakage are the two primary sources of leakage power. Figure 10.7 shows different voltage configurations of NMOS pass transistors and how they rank in terms of both subthreshold and gate leakage. We see that gate leakage is maximized when V_{GS} is large and V_{DS} is small, while subthreshold leakage is maximized with large V_{DS}. For most technologies subthreshold leakage dominates gate leakage, so large V_{DS} voltages should be avoided.

If we consider pairs of connected pass transistors in the LUT multiplexer (e.g., the transistors connected SRAM-00 and SRAM-01 in Fig. 10.2a), we see that leakage can be minimized by configuring the LUT so that inputs to pairs of pass transistors are either both “0” or “1” (which minimizes V_{DS} for both transistors). This information can be exploited by using an algorithm to rearrange the LUT inputs such that the LUT is configured for minimal leakage [6, 22]. The algorithm examines all 2^k input permutations and determines which permutation pairs together the most “0” and “1” configuration bits. Through this technique an average of 50.3% leakage energy can be saved.

10.3.4 Placement

The next step after obtaining a clustered CLB netlist is to place the netlist on a physical FPGA architecture. Placement algorithms have traditionally tried to minimize delay by placing critical CLBs close together, reducing the amount of capacitive interconnect a net must pass through. Low-power placement algorithms attempt to place CLBs with high activity connections between them close together. Similar to clustering, the goal is to minimize the amount of activated general-purpose interconnect – shorter nets will charge and discharge less interconnect capacitance, which is desirable for highly active nets. Several works have implemented simulated annealing [28] placers with cost functions modified to reflect switching activity [21, 33, 53]. Each of these placers calculate the cost of a swap by adding a power term to the existing wirelength and delay terms. The power term includes the switching activity of the nets involved in the swap along with their capacitance. However, because net capacitance is not yet known it must be estimated; [53] and [21] include more accurate methods of capacitance estimation that lead to improved results.

Low-power placers must be careful to balance the intents of minimizing wirelength of critical nets and minimizing wirelength of highly active nets. Attempting to place circuits using low-power annealers may increase circuit delay for designs with many critical, low-activity nets. On average, placers without capacitance estimation reduce total power by 3.0% with 4% delay cost; placers with capacitance estimation can achieve reductions of 8.6–13% with delay cost of 1–3% [21, 53].

10.3.5 Routing

The final FPGA CAD step connects placed CLBs together via the programmable routing fabric; FPGA routers typically use the PathFinder algorithm [45]. Nets are routed one by one through the routing fabric using a least-cost path algorithm and are iteratively rerouted until convergence conditions are met. Nets are allowed to share resources so long as those resources are on the least-cost path. The cost of a routing resource (e.g., switch, wire segment) is calculated using the delay of that resource, the criticality of the net currently being routed, and the number of nets currently sharing the resource. Over time the cost of sharing increases rapidly, helping to negotiate congestion and separate the nets.

PathFinder can be modified for low-power routing by adjusting the cost function to account for net activity and capacitance [21, 33], similar to the placement case. The cost of a resource is modified to include the activity of the net being routed and the capacitance of the candidate routing resource. In modern FPGA architectures, routing resources are not identical and have differing capacitances; hence, lower capacitance paths can be selected. Ideally, high activity nets should be mapped to low capacitance routes. However, as in placement, power-aware routers must be careful to balance delay and power minimization; low activity routes may end up being critical. In fact, power-aware routers have been shown to reduce total power by 2.6% but with a delay increase of 3.8%.

10.3.6 Glitch Reduction

A glitch is a switching event in a combinational circuit that does not contribute to the computation of that circuit's final value. Glitches occur when inputs to a gate arrive at different times; Fig. 10.8 shows an example of a glitch. Glitch energy has been shown to be around 20% [32, 43] of total dynamic energy.

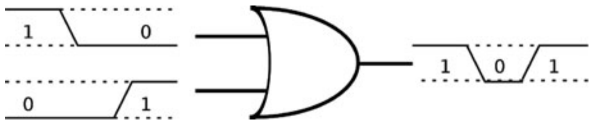


Fig. 10.8 Example of a glitch

Several optimization techniques have been proposed to reduce energy lost to glitches in FPGAs. Wilton et al. proposed pipelining circuits to reduces glitches [54]. A highly pipelined design has fewer glitches because it limits the number of logic levels between registers. Pipelining can reduce energy per operation substantially (40 − 90%) for circuits with significant glitching at a cost of latency. Glitches can also be minimized during CAD steps in either technology mapping [12] or routing [16]. Technology mappers already target reduced switching activity, so only minor modifications need to be made to model glitches. Glitch-aware routers attempt to route early or late arriving signals through alternative paths in the routing fabric such that all signals arrive simultaneously. Glitch-aware CAD has been shown to reduce dynamic power by 11–18.7%.

10.3.7 Full CAD Flows

Lamoureux et al. studied the impact of combining each of these power-aware CAD steps into one uniform low-power CAD flow [33]. They discovered that the power reductions from each of the individual CAD steps were not cumulative. Table 10.2 reviews their results. Technology mapping and clustering yield the most significant power reductions individually of 7.5 and 12.6%, respectively. However, when combined they reduce power by only 17.6%. This is largely due to the fact that

Table 10.2 Energy savings from using low-power CAD tools [33]

Mapping	Clustering	Placement	Routing	Energy savings (%)
Base	Base	Base	Base	0.0
Base	Base	Base	Power	2.6
Base	Base	Power	Base	2.9
Base	Base	Power	Power	5.7
Base	Power	Base	Base	12.6
Base	Power	Base	Power	14.8
Base	Power	Power	Base	15.9
Base	Power	Power	Power	17.9
Power	Base	Base	Base	7.5
Power	Base	Base	Power	9.9
Power	Base	Power	Base	10.1
Power	Base	Power	Power	12.6
Power	Power	Base	Base	17.6
Power	Power	Base	Power	19.5
Power	Power	Power	Base	20.6
Power	Power	Power	Power	22.6

power-aware technology mapping already hides many high activity nodes within LUTs, so power-aware clustering has fewer high activity nodes to optimize by clustering together. Place and route individually save 2.6 and 2.9% energy; when combined with mapping and clustering they add an additional 5.0% rather than 5.6%. For a complete power-aware CAD flow a combined power reduction of 22.6% can be achieved.

10.4 Architecture

While CAD transformations can provide power savings by optimizing the way in which circuits are mapped to FPGAs, changing the physical FPGA architecture can provide significant additional benefits. Architectural changes affect the structure of the FPGA on a global scale by reorganizing the design of logic and interconnect.

10.4.1 Logic Block

FPGA logic is organized into CLBs that contain N k-input LUTs (Fig. 10.1 and 10.6). Values chosen for N and k can impact the overall dynamic power dissipation of an FPGA architecture. Several researchers have explored this design space to determine energy favorable values of k and N [36, 43, 46].

Changing k can have the following tradeoffs in terms of power: first, if k is increased, CLBs require more local routing to connect all LUT input pins (Fig. 10.6), increasing the dynamic energy of the interconnect local to the CLB. However, larger LUTs can implement more complex functions, reducing the total number of LUTs and CLBs. Fewer CLBs mean less demand for global routing resources, saving dynamic interconnect energy. The tradeoff in selecting N is similar: large values of N increase CLB capacity and functionality, which increases local interconnect energy but reduces global interconnect energy. k and N impact leakage energy solely by changing the total area of a design—larger area means more devices leaking. It has been shown that $k = 4$ minimizes area [3] and therefore leakage energy [43].

Lin et al. examined total FPGA energy as a function of k and N [43] (Fig. 10.9). They found that a LUT input size of $k = 4$ minimizes total energy, and that selecting either $k = 3$ or $k = 5$ can increase energy dissipation by as much as 50%. They also determined that smaller cluster sizes reduce energy but with diminishing returns; for $k = 4$ cluster sizes from $N = 6 - 10$ typically use the least energy.

10.4.2 Interconnect Topology

Because interconnect power dominates total FPGA power (Fig. 10.3), it is important to select an appropriate interconnect topology and configuration. FPGA global interconnect is typically parameterized by the length of a wire segment, the

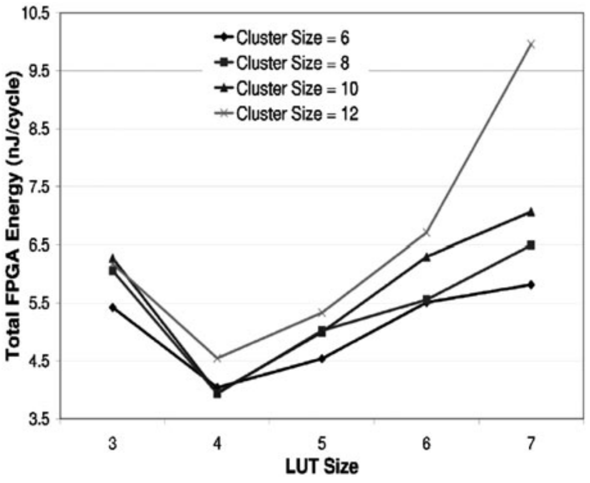


Fig. 10.9 FPGA energy as a function of LUT input size (k) and cluster size (N) [43]

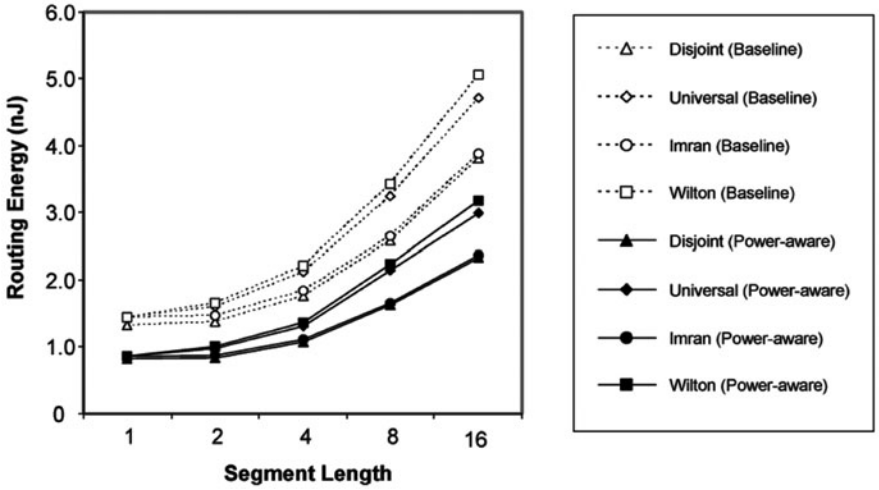


Fig. 10.10 FPGA energy as a function of segment length and switchbox topology [46]

connectivity of switchboxes, and the directionality of the segment (bi-directional or directional).

Poon et al. [46] studied the impact of segment length and switchbox configurations on FPGA energy (Fig. 10.10). A longer wire segment connects to more switches, increasing its capacitance; however, longer segments should lead to fewer total switches. They found the shortest segments $L_{seg} = 1$ and disjoint style switchboxes are the most energy efficient.

In the past, FPGAs were manufactured with bi-directional switches. These switches suffer from the inefficiency that, once configured, an FPGA only uses a

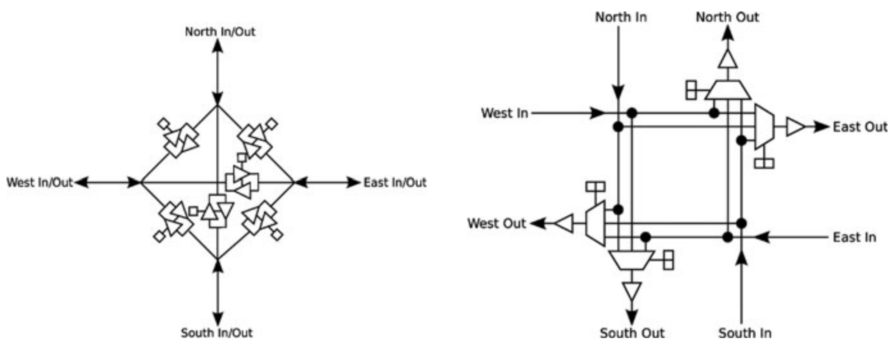


Fig. 10.11 FPGA switches [34]. (**a**) Bi-directional (**b**) Directional

switch in one direction; hence, only 50% of drivers are ever utilized. Directional switches drive segments in a single direction, ensuring that all drivers could be utilized. Figure 10.11 compares the two switch types. Figure 10.2b depicted a single direction of a bidirectional switch; two of these switches are connected in a loop to create a bidirectional switch as shown in Fig. 10.11a. The multiplexer in the directional switch can be implemented with either pass transistor or static logic. While directional drivers would seem to require more wiring because wires can only be utilized in one direction, in practice they use the same number of wires as the bi-directional case, saving area and improving delay [34]. Jamieson et al. [26] studied the effect of directional versus bi-directional switches on interconnect energy, and found that directional wires also save significant energy.

10.4.3 Dynamic Voltage Scaling

A technique commonly used in processors to reduce dynamic energy is dynamic voltage scaling (DVS) (Chapter 2, section 2.1). DVS reduces V_{dd}, saving both dynamic and static energy (Equations (10.1) and (10.2)) but at the expense of increasing delay. DVS is useful in scenarios where a design needs to operate at a target frequency. In these cases, V_{dd} can be lowered to the point where the target is still achieved, minimizing the energy wasted on margins. The exact value of V_{dd} can be different between chips due to variation and can change over time due to environmental variation; hence, an on-chip control circuit with delay feedback is typically used to adjust V_{dd}.

DVS in FPGAs was examined by [13]. They use a design-specific measurement circuit that tracks the delay of a design's critical path to provide feedback to the voltage controller. Figure 10.12 shows the control and measurement circuit; for measurement, a chain of 128 inverters is constructed with identically clocked flip-flops on each inverter output. In a single clock cycle the input to the inverter chain will propagate a specific distance dependent on the delays of the inverters due to environmental conditions. The distance through the chain is set to be identical to the critical path delay in a nominal environment, and it is assumed that the measurement circuit

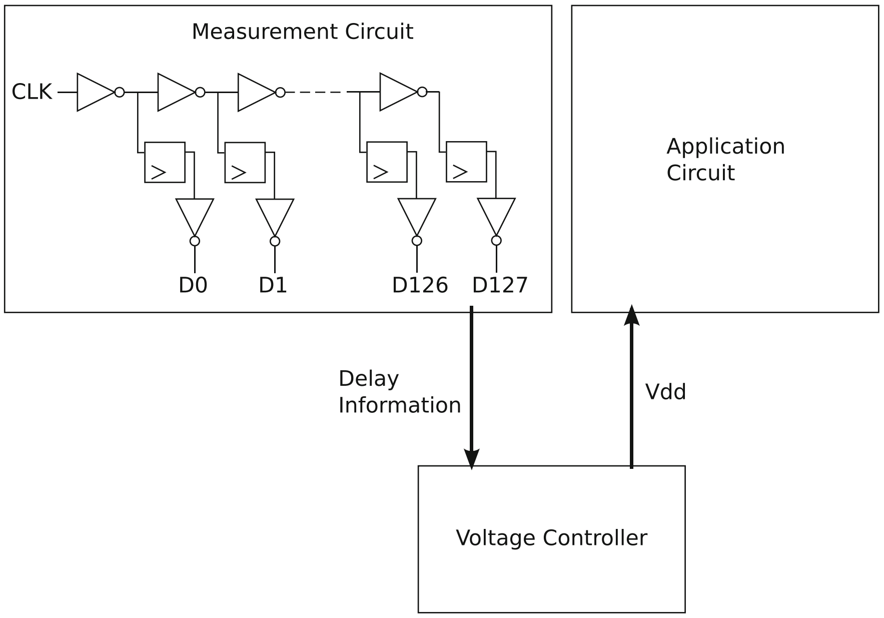


Fig. 10.12 Dynamic voltage scaling for FPGAs [13]

tracks the critical path delay under environmental variation. Through this technique energy savings of 4–54% can be observed.

The correctness of this design depends critically on how well matched the reference circuit chain is to the circuit critical path. More robust techniques will use the path itself as the reference for tuning voltage. The most robust designs augment the registers in the architecture to help monitor timing [7, 9].

10.4.4 Power Gating

Leakage power is a significant source of FPGA power dissipation, and is expected to increase as technology scales [51]. As previously discussed, FPGAs use a significant amount of area to provide programmability. In addition, to be widely usable FPGAs over-provision resources so that customers can map large and complex designs. As a consequence significant portions of an FPGA are often unused. Instead of leaving these unused portions to sit idle and leak, it is possible to use power gating (Chapter 2, section 2.4.5) to disconnect the power supply from these regions and eliminate leakage of unused devices.

Figure 10.13 shows a basic power gating circuit. To support power gating a high V_{th} sleep transistor must be inserted between the power supply and the block to be gated. The high threshold ensures that leakage through the sleep transistor will be negligible. The gate of the transistor is tied to a control bit that can either be set at

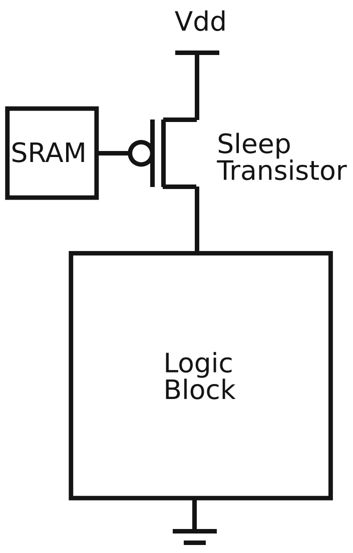


Fig. 10.13 Power gating circuitry

configuration time (i.e., static power gating) or during runtime (i.e., dynamic power gating).

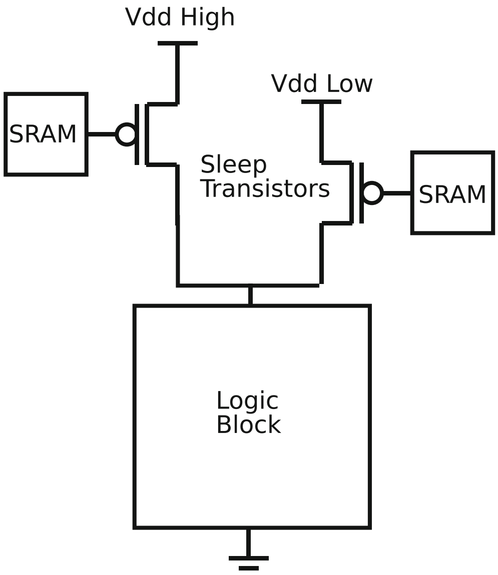
Sleep transistors increase area and also add delay to the gated block. Sizing up the transistors negates the speed penalty, but it also increases the area penalty and reduces the effectiveness of leakage control (wider transistors leak more). Another tradeoff in applying power gating is selecting the appropriate block size to gate. Gating off smaller blocks yields more coverage in disabling unused devices and increases power savings, but at the cost of using many sleep transistors. Gating off larger blocks amortizes the cost of the sleep transistors but makes it difficult to disable the largest number of unused devices.

Many researchers have explored different points in the design space for power gating granularity in FPGAs. Calhoun et al. [10] perform power gating at the gate level; Gayasen et al. [19] use larger power gating blocks, choosing to gate off regions of four CLBs at a time. To enhance leakage reduction they develop a region-constrained placement algorithm to increase the number of regions that can be power gated. They show 20% leakage power savings and that coarse-grained power gating with improved placement achieves the same results as fine-grained power gating. Rahman et al. [47] suggest that a combination of fine and coarse-grained power gating provides the best results.

10.4.5 Dual V_{dd}

Instead of simply using sleep transistors to disable or enable blocks of logic, they can be used to select from different supply voltages (V_{dd}) to power the block. Figure 10.14 shows the basic circuit for providing dual supply voltages. Reducing V_{dd} saves dynamic and static energy, but at the cost of performance. However, not all

Fig. 10.14 Dual V_{dd} design



paths are critical, and hence only elements on critical paths need to be placed in high V_{dd} regions. Paths with timing slack can travel through low V_{dd} blocks. Identifying those elements and assigning an appropriate V_{dd} are therefore important for dual V_{dd} designs. Level conversion is necessary when moving from a low V_{dd} region to a high V_{dd} region, and level converters add delay, area, and energy overhead.

Dual V_{dd} design has been studied extensively in the FPGA literature [5, 18, 25, 37, 38, 44]. Early work placed level converters at the inputs to each wire segment [37, 38], but later work [44] demonstrated that this placement costs significant area and leakage. More recent work has developed techniques for placing level converters only at the inputs to CLBs and developed V_{dd} assignment algorithms to guarantee that no high V_{dd} switch follows a low V_{dd} switch [25]. Furthermore, transformations can be performed on the netlist to distribute timing slack to all nets such that total power is minimized. Dual V_{dd} FPGA architectures typically achieve $\approx 50\%$ power savings.

10.4.6 Dual V_{th} and Body Biasing

Similar to dual V_{dd} architectures for reducing total power, dual V_{th} and body-biased FPGAs have been proposed to reduce leakage power. Dual V_{th} FPGAs define low and high V_{th} regions at fabrication time. High V_{th} regions reduce leakage at the cost of increased delay. Body-biased FPGAs use embedded circuitry to change the body to source voltage for regions at configuration time. The effect of changing the body to source voltage V_{bs} on V_{th} is

$$V_{\mathrm{th}} = V_{\mathrm{th0}} + \gamma\left(\sqrt{|\Phi_{\mathrm{s}} - V_{\mathrm{bs}}|} - \sqrt{|\Phi_{\mathrm{s}}|}\right) \qquad (10.3)$$

Where V_{th0} is the ideal V_{th} at $V_{bs} = 0$, γ is the body bias coefficient, and Φ_s is the surface potential. Hence, by applying negative values of V_{bs} to a region its V_{th} can be increased.

Figure 10.15 shows the circuitry required for body biasing. To determine the correct bias voltage, a representative critical path from the circuit to be biased is replicated and its delay is compared to a reference clock using a phase detector. The output of the phase detector is fed into a digital counter, which is converted into an analog signal using a R–2R resistor network and op-amp. This signal is then used to select the body bias voltage for the target circuit block.

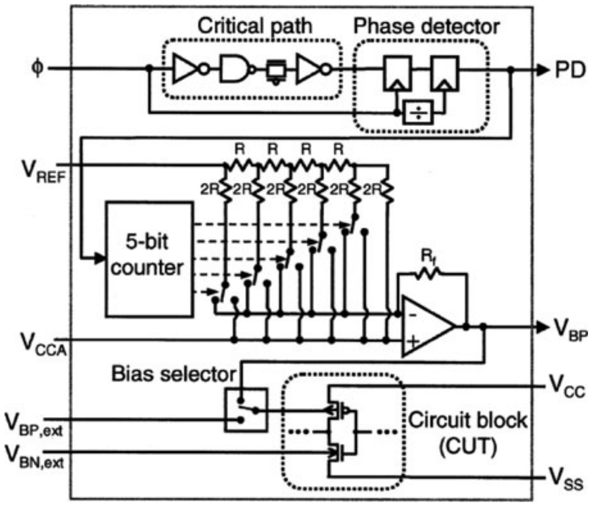


Fig. 10.15 Body bias circuit [50]

Much work has also been done in dual V_{th} design for FPGAs [24, 27, 39, 52] with body biasing being integrated into commercial FPGAs [35]. Similar tradeoffs exist when mapping circuits to either dual V_{th}/body biased architectures or dual V_{dd} architectures. Critical paths must be placed on low V_{th} blocks to ensure minimal delay reduction, and timing slack must be available to save leakage energy on non-critical paths. Block granularity is important [24] and selection of approriate body bias voltages is important [27].

While the previous discussion of V_{th} and V_{dd} selection applies at the level of the mapped design, it is also profitable to use different V_{th} devices for circuits that play different roles in the FPGA architecture. Notably, high V_{th} devices can significantly reduce the configuration SRAM bit leakage reduced without impacting area or delay [52]. Configuration bits are a prime candidate for high V_{th} transistors because they constitute a significant fraction of FPGA area and are always on and leaking. Fortunately, configuration bits are set only at configuration time and do not contribute any delay or switching energy to mapped circuits. Increasing configuration SRAM V_{th} can reduce total leakage energy by up to 43%. Today's commercial FPGAs are fabricated with three different threshold voltages [29].

10.5 Circuits

Below the level of architectural techniques are circuit optimizations. These techniques typically attempt to redesign LUT and interconnect switch circuits to reduce both static and dynamic power.

10.5.1 Low Swing Interconnect

Reducing interconnect capacitance is one way to reduce the dynamic energy dissipated by FPGA switches. However, capacitance is only a linear term in dynamic energy (Equation (10.1)); V_{dd} is quadratic. Hence, greater energy savings could be achieved by reducing the voltage switched in the interconnect.

A low swing interconnect segment consists of a driver and receiver operating at nominal voltages, with the wire between them operating at reduced voltage. The driver converts a full swing input into a low swing interconnect signal and the receiver converts it back. With this technique the amount of dynamic energy dissipated in interconnect segments can be reduced significantly. A common drawback of low swing interconnect is the slow speed of the receiver circuit. Figure 10.16 shows a low swing segment that uses cascode and differential circuitry to improve receiver speed [20]. By employing this technique throughout an FPGA, interconnect energy can be reduced by a factor of 2.

10.5.2 Glitch Reduction

Section 10.3.6 introduced techniques to transform the design to reduce glitches. With circuit-level support, we can introduce additional opportunities to reduce glitches.

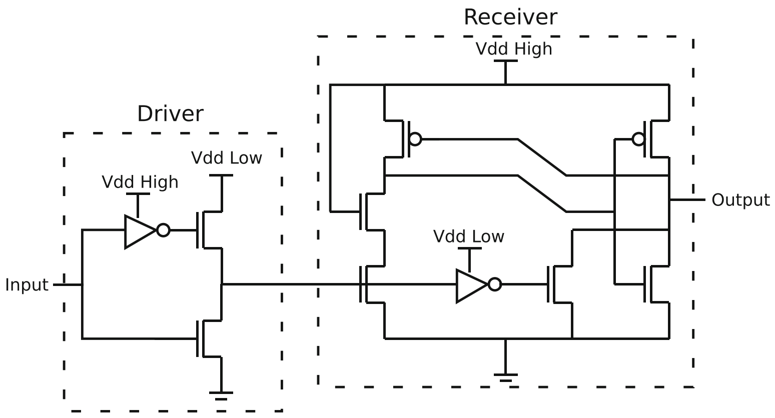


Fig. 10.16 Low swing interconnect circuit [20]

GlitchLess is a circuit level technique to eliminate glitches [32]. GlitchLess inserts programmable delay elements at the inputs of each logic element, and programs them, post-fabrication, such that arriving signals are aligned. Figure 10.8 showed an example of a glitch; Fig. 10.17 shows how to eliminate this glitch by adding a delay element. On average this eliminates 87% of glitches, reducing overall FPGA power by 17%.

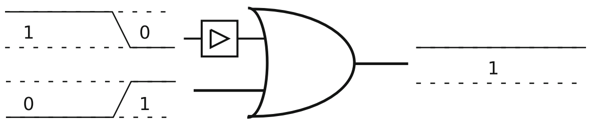


Fig. 10.17 Removing glitches through programmable delay elements

10.6 Devices and Technologies

The lowest level at which FPGA designers can attempt to reduce power consumption is at the device and technology level. Fundamental advances in several emerging technologies have made is possible to build FPGAs out of more than just conventional CMOS transistors. We briefly examine a few technologies where studies suggest the 3D ICs may reduce FPGA power consumption.

10.6.1 Three-Dimensional Integration

A three-dimensional integrated circuit (3D IC) is a chip with two or more active layers of devices. 3D ICs have two distinct physical advantages over two-dimensional integrated circuits (2D ICs). First, they can achieve significantly higher logic density from adding devices in a third dimension. Second, average wirelength decreases because logic elements effectively move closer together. A reduction in average wirelength means a reduction in interconnect capacitance, which is the dominant source of FPGA power.

There are three ways to stack 3D ICs: wafer-to-wafer, die-to-die, and monolithic. Wafer-to-wafer and die-to-die methods manufacture wafers and dies separately and stack them together, connecting layers using through-silicon vias (TSVs). The primary challenge with these two techniques is that TSVs are generally very large and only limited connections can be made between layers. Monolithic stacking directly fabricates active layers of silicon together, which results in much higher inter-layer connection density. The primary challenge with monolithic stacking is that the temperature needed to create an active layer can damage layers and wiring beneath it.

Ababei et al. explored the benefits of a die-to-die 3D FPGA architecture [1]. They developed an optimized 3D place-and-route tools to aid in wirelength reduction.

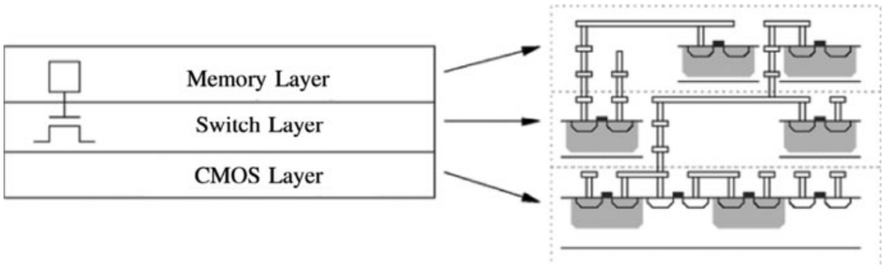


Fig. 10.18 Monolithic 3D FPGA [41]

Assuming a ten layer device they demonstrated that a die stacked 3D FPGA benefits from a 25% average wirelength reduction.

Lin et al. studied a monolithically stacked FPGA [41, 42] where logic, routing, and configuration all occupy independent layers (Fig. 10.18). This is possible, in part, because configuration memories make up a significant fraction of the area in FPGAs. The configuration memories do not require high-speed logic transistors, so can use alternate device technologies that do not require high temperature processing. Removing the memory area from the logic reduces one of the large components of area overhead that drive longer wires and hence larger capacitance in FPGA designs compared to ASICs. They demonstrate that a 3D monolithic FPGA can reduce area, delay, and dynamic power by 3.3×, 2.51×, and 2.93×, respectively.

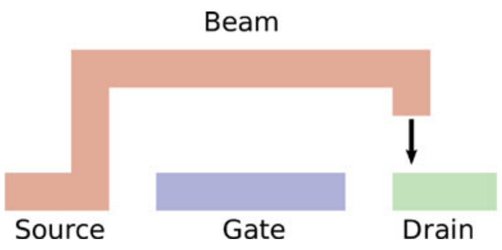
The main challenge in 3D ICs is manufacturing; however, an important secondary challenge is cooling. ICs are typically cooled on the package surface, and with conventional cooling techniques (e.g., heatsinks and fans) power density is limited to approximately 100 W/cm^2. 3D ICs have significantly higher power densities than 2D chips because of many more active transistors per unit surface area. While 3D ICs reduce wire capacitance such that overall power efficiency is improved, it may prove difficult to provide sufficient cooling proportional to the increase in power density.

10.6.2 Nanoelectromechanical Relays

Nanoelectromechanical (NEM) relays are electrostatically actuated mechanical switches that are excellent candidates to replace FPGA interconnect switches. NEM relays hold their on/off state (i.e., they are hysteretic), enabling them to replace both the NMOS pass transistor and SRAM configuration bit in a conventional FPGA switch with a single device. They have much lower on resistance than NMOS pass transistors, reducing delay. In terms of power, NEM relays exhibit zero leakage current. Hence, significant reductions in leakage power are possible.

NEM relays can be made out of conventional CMOS processing materials (e.g., silicon, metal) or carbon nanotubes, which are more difficult to integrate into a CMOS manufacturing flow. Figure 10.19 shows a three-terminal relay. The relay consists of source, gate, and drain electrodes and a mechanical deflecting beam.

Fig. 10.19 NEMS switch



When a voltage V_{gs} is applied to the switch, electrostatic force attracts the beam downward toward the gate. An opposing, intrinsic elastic force in the beam resists the deflection. When V_{gs} is greater than the pull-in voltage V_{pi}, the beam collapses and makes contact with the drain, enabling conduction between the source and drain. To disconnect the beam, Vgs is lowered below the pull-out voltage V_{po}. Because $V_{po} < V_{pi}$, the switch exhibits hysteresis with an operating range between V_{po} and V_{pi}.

Zhou et al. examined replacing switches in LUTs with carbon nanotube NEM relays, demonstrating a 91% reduction in LUT power [55]. Chen et al. explored conventional metal NEM relays as a replacement for interconnect switches [11]. They found that NEM relays can reduce area, delay, and leakage power by 43.6, 28, and 37%, respectively.

The main challenges in using NEM relays are integrating with a conventional CMOS manufacturing process and ensuring reliable mechanical contacts over many switching events. The deflecting beam can suffer from stiction (failure to release after deflection), tip bouncing, and other mechanical failures.

10.7 Summary

Table 10.3 summarizes the techniques covered in this chapter, the level at which they operate, the terms in the power equations they attempt to reduce (from Equations (10.1) and (10.2)), the type of power reduction, and the demonstrated benefit. It is important to note that these benefits will not all be additive.

While FPGAs dissipate considerably more power than ASICs when fabricated in the same process and solving identical problems, a substantial amount of research in low-power FPGA design in the last decade has begun to narrow this gap, making it easier for differences in technology and problem solved to result in lower energy for the FPGA than the ASIC. This chapter summarized the progress in reducing FPGA power across all levels of design, from CAD and architecture down to circuits and devices, and all components of the FPGA architecture.

Demand for low-power devices will only increase in the future as mobile devices become even more ubiquitous. As technology scales, power limitations will constrain the performance and growth of CPUs, which are already at a power disadvantage to FPGAs. ASICs are declining in popularity due to enormous design and fabrication costs, which will only increase. FPGAs have the unique opportunity to

Table 10.3 Roundup of low-power FPGA techniques

Technique	Level	Reduces	Benefit type	Benefit(%)
Technology mapping	CAD	C, α	P_{total}	7.6–17
Clustering	CAD	C, α	P_{total}	13
LUT input transformation	CAD	I_{leak}	P_{static}	50
Placement	CAD	C, α	P_{total}	3.0–13
Routing	CAD	C, α	P_{total}	3
Glitch routing/placement	CAD	α	P_{dynamic}	11–19
Logic block architecture	Architecture	C	P_{total}	48
Interconnect architecture	Architecture	C	P_{total}	12
Dynamic voltage scaling	Architecture	V_{dd}	P_{total}	4–54
Power gating	Architecture	I_{leak}	P_{static}	20
Dual V_{dd}	Architecture	V_{dd}	P_{total}	50
Dual V_{th}/body biasing	Architecture	I_{leak}	P_{static}	43
Low swing interconnect	Circuits	V_{dd}	P_{total}	50
Glitchless	Circuits	α	P_{total}	17
3D integration	Devices	C	P_{dynamic}	66
NEM relays	Devices	I_{leak}	P_{static}	37

become the low-power devices of future by continuing the close the power gap with innovations in power reduction. Chapter 12 shows how efficient handling of variation and wear can further narrow or reverse the energy gap and increasingly favor more FPGA-like architectures.

References

1. Ababei C, Mogal H, Bazargan K (2006) Three-dimensional place and route for FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 25(6):1132–1140
2. Abnous A, Zhang H, Wan M, Varghese G, Prabhu V, Rabaey J (eds) (2002) The application of programmable DSPs in mobile communications, Chichester, UK. doi: 10.1002/0470845902 chap. 17: The pleiades architecture. Wiley, pp 327–359
3. Ahmed E, Rose J (2000) The effect of LUT and cluster size on deep-submicron FPGA performance and density. In: Proceedings of the international symposium on field-programmable gate arrays. ACM, New York, NY, pp 3–12
4. Anderson J, Najm F (2002) Power-aware technology mapping for LUT-based FPGAs. In: Proceedings of the international conference on field-programmable technology, pp 211–218
5. Anderson J, Najm F (2004) Low-power programmable routing circuitry for FPGAs. In: Proceedings of the international conference on computer-aided design
6. Anderson J, Najm F, Tuan T (2004) Active leakage power optimization for FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays
7. Austin T, Blaauw D, Mudge T, Flautner K (2004) Making typical silicon matter with Razor. IEEE Comput 37(3):57–65
8. Betz V (1999) VPR and T-VPack: versatile packing, placement and routing for FPGAs. <http://www.eecg.toronto.edu/~vaughn/vpr/vpr.html>. Version 4.30
9. Bowman KA, Tschanz JW, Kim NS, Lee JC, Wilkerson CB, Lu SLL, Karnik T, De VK (2009) Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. IEEE J Solid State Circuits 44(1):49–63

10. Calhoun B, Honore F, Chandrakasan A (2003) Design methodology for fine-grained leakage control in MTCMOS. In: Proceedings of the international symposium on low power electronics and design
11. Chen C, Wong HSP, Mitra S, Parsa R, Patil N, Chong S, Akarvardar K, Provine J, Lewis D, Watt J, Howe RT (2010) Efficient FPGAs using nanoelectromechanical relays. In: Proceedings of the international symposium on field-programmable gate arrays. ACM Press, New York, NY, p 273
12. Cheng L, Chen D, Wong MD (2007) GlitchMap: An FPGA technology mapper for low power considering glitches. In: Proceedings of the ACM/IEEE design automation conference, pp 318–323
13. Chow C, Tsui L, Leong P, Luk W, Wilton S (2005) Dynamic voltage scaling for commercial FPGAs. Proceedings of the international conference on field-programmable technology, pp 173–180
14. DeHon A (1998) Comparing computing machines. In: Configurable computing: technology and applications, proceedings of SPIE, vol. 3526, SPIE
15. DeHon A (1999) Balancing Interconnect and computation in a reconfigurable computing array (or, why you don't really want 100% LUT utilization). In: Proceedings of the international symposium on field-programmable gate arrays, pp 69–78
16. Dinh Q, Chen D, Wong M (2010) A routing approach to reduce glitches in low power FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 29(2):235–240
17. Farrahi AH, Sarrafzadeh M (1994) FPGA technology mapping for power minimization. In: Proceedings of the international conference on field-programmable logic and applications, Springer, London, pp 66–77
18. Gayasen A, Lee K, Vijaykrishnan N, Kandemir M, Irwin M, Tuan T (2004) A Dual-Vdd low power FPGA architecture. In: Proceedings of the international conference on field-programmable logic and applications, Springer, pp 145–157
19. Gayasen A, Tsai Y, Vijaykrishnan N, Kandemir M, Irwin MJ, Tuan T (2004) Reducing leakage energy in FPGAs using region-constrained placement. In: Proceedings of the international symposium on field-programmable gate arrays, pp 51–58
20. George V, Zhang H, Rabaey J (1999) The design of a low energy FPGA. In: Proceedings of the international symposium on low power electronics and design, pp 188–193
21. Gupta S, Anderson J, Farragher L, Wang Q (2007) CAD techniques for power optimization in Virtex-5 FPGAs. In: Proceedings of the IEEE custom integrated circuits conference, pp 85–88
22. Hassan H, Anis M, Elmasry M (2008) Input vector reordering for leakage power reduction in FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 27(9):1555–1564
23. Hauck S, DeHon A (eds) (2008) Reconfigurable computing: the theory and practice of FPGABased computation. systems-on-silicon. Elsevier Burlington, MA
24. Hioki M, Kawanami T, Tsutsumi T, Nakagawa T, Sekigawa T, Koike H (2006) Evaluation of granularity on threshold voltage control in flex power FPGA. In: Proceedings of the international conference on field-programmable technology, pp 17–24
25. Hu Y, Lin Y, He L, Tuan T (2008) Physical synthesis for FPGA interconnect power reduction by dual-Vdd budgeting and retiming. ACM Trans Des Autom Electron Syst 13(2):1–29
26. Jamieson P, Luk W, Wilton SJ, Constantinides GA (2009) An energy and power consumption analysis of FPGA routing architectures. In: Proceedings of the international conference on field- programmable technology, pp 324–327
27. Kawanami T, Hioki M, Matsumoto Y, Tsutsumi T, Nakagawa T, Sekigawa T, Koike H (2006) Optimal set of body bias voltages for an FPGA with field-programmable Vth components. Proceedings of the international conference on field-programmable technology, pp 329–332
28. Kirkpatrik S, Gellatt Jr, CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
29. Klein M (2005) The Virtex-4 power play. Xcell J (52):16–19
30. Kuon I, Rose J (2007) Measuring the gap between FPGAs and ASICs. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):203–215

31. Kusse E, Rabaey J (1998) Low-energy embedded FPGA structures. In: Proceedings of the international symposium on low power electronics and design, pp 155–160
32. Lamoureux J, Lemieux GGF, Wilton SJE (2008) GlitchLess: dynamic power minimization in FPGAs through edge alignment and glitch filtering. IEEE Trans VLSI Syst 16(11):1521–1534
33. Lamoureux J, Wilton S (2003) On the interaction between power-aware FPGA CAD algorithms. In: Proceedings of the international conference on computer-aided design. IEEE Computer Society, Washington, DC
34. Lemieux G, Lee E, Tom M, Yu A (2004) Directional and single-driver wires in fpga interconnect. In: Proceedings of the international conference on field-programmable technology, pp 41–48
35. Lewis D, Ahmed E, Cashman D, Vanderhoek T, Lane C, Lee A, Pan P (2009) Architectural enhancements in Stratix-III and Stratix-IV. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, pp 33–42
36. Li F, Chen D, He L, Cong J (2003) Architecture evaluation for power-efficient FPGAs. In: Proceedings of the International symposium on field-programmable gate arrays, ACM Press, New York, NY, p 175
37. Li F, Lin Y, He L (2004) FPGA power reduction using configurable dual-Vdd. In: Proceedings of the ACM/IEEE design automation conference, ACM, pp 735–740
38. Li F, Lin Y, He L (2004) Vdd programmability to reduce FPGA interconnect power. In: Proceedings of the International Conference on Computer-Aided Design, IEEE, pp 760–765
39. Li F, Lin Y, He L, Cong J (2004) Low-power FPGA using pre-defined dual-Vdd/dual-Vt fabrics. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, p 4250
40. Li H, Katkoori S, Mak WK (2004) Power minimization algorithms for LUT-based FPGA technology mapping. ACM Trans Des Autom Electron Syst 9(1):33–51
41. Lin M, El Gamal A (2009) A low-power field-programmable gate array routing fabric. IEEE Trans VLSI Syst 17(10):1481–1494
42. Lin M, El Gamal A, Lu YC, Wong S (2007) Performance benefits of monolithically stacked 3- D FPGA. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):216–229
43. Lin Y, Cong J (2005) Power modeling and characteristics of field programmable gate arrays. IEEE Trans Comput Aided Des Integr Circ Syst 24(11):1712–1724
44. Lin Y, He L (2006) Dual-Vdd interconnect with chip-level time slack allocation for FPGA power reduction. IEEE Trans Comput Aided Des Integr Circ Syst 25(10):2023–2034
45. McMurchie L, Ebeling C (1995) PathFinder: a negotiation-based performance-driven router for FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, pp 111–117
46. Poon K, Wilton S, Yan A (2005) A detailed power model for field-programmable gate arrays. ACM Trans Des Autom Electron Syst 10:279–302
47. Rahman A, Das S, Tuan T, Trimberger S (2006) Determination of power gating granularity for FPGA fabric. In: Proceedings of the IEEE custom integrated circuits conference, pp 9–12
48. Shang L, Kaviani A, Bathala K (2002) Dynamic power consumption in Virtex-II FPGA family. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, p 164
49. Singh A, Marek-Sadowska M (2002) Efficient circuit clustering for area and power reduction in FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, New York, NY, pp 59–66
50. Tschanz J, Kao J, Narendra S, Nair R, Antoniadis D, Chandrakasan A, De V (2002) Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. IEEE J Solid State Circuits 37(11):1396–1402
51. Tuan T, Lai B (2003) Leakage power analysis of a 90 nm FPGA. In: Proceedings of the IEEE custom integrated circuits conference, IEEE, p 57
52. Tuan T, Rahman A, Das S, Trimberger S, Kao S (2007) A 90-nm low-power FPGA for battery-powered applications. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):296–300

53. Vorwerk K, Raman M, Dunoyer J, Kundu A, Kennings A (2008) A technique for minimizing power during FPGA placement. In: Proceedings of the international conference on field-programmable logic and applications, pp 233–238
54. Wilton S, Ang S, Luk W (2004) The impact of pipelining on energy per operation in fieldprogrammable gate arrays. In: Proceedings of the international conference on field-programmable logic and applications, Springer, pp 719–728
55. Zhou Y, Thekkel S, Bhunia S (2007) Low power FPGA design using hybrid CMOS-NEMS approach. In: Proceedings of the international symposium on low power electronics and design, ACM, p 19