

An Overview of Low-Power Techniques for Field-Programmable Gate Arrays

Julien Lamoureux and Wayne Luk
{jlamoure, wl}@doc.ic.ac.uk
Imperial College London

Abstract

This paper provides an overview of low-power techniques for field-programmable gate arrays (FPGAs). It covers system-level design techniques and device-level design techniques that have targeted current commercial devices. It also describes current research on circuit-level and architecture-level design techniques. Recent studies on power modelling and on low-power computer-aided design (CAD) are also reported. Finally, it proposes future work that would enable the use of FPGA technology in applications where power and energy consumption is critical, such as mobile devices.

1. Introduction

Field-programmable gate arrays are ideal for adaptive systems, since they are reconfigurable and can be programmed to implement any digital logic. Applications of such FPGA-based adaptive systems include face image recognition [22], on-line failure recovery [49], and analysis of firefly synchronisation [62].

The main drawback of FPGAs is that they are less efficient than application-specific integrated circuits (ASICs) due to the added circuitry needed to make them reconfigurable. In a recent study [29], FPGAs are estimated to be 3-4 times slower, 5-35 times larger, and 7-14 times less energy efficient than ASICs depending on the application and the flexibility of the FPGA.

Traditionally, FPGA research focused on reducing the speed and area overhead [9]. In recent years, however, much of the focus has shifted to improving the energy efficiency. This shift is due to process scaling and increased demand for low-power applications. Although process scaling reduces the energy needed to perform a given computation (since wires and transistors are smaller), it increases power dissipation per unit area and therefore the overall power for a given die size [24]. At the same time, demand for low-power applications is increasing due to the proliferation of hand-held devices and increasing energy costs. For hand-held and other battery operated devices, reducing power increases battery life. For non-mobile

devices, reducing power consumption lowers operating, packaging, and cooling system costs.

There are many ways to make FPGAs more energy efficient. The various techniques can be divided into five categories: process, circuit, architecture, system, and computer-aided design (CAD). Process techniques refer to the use of new low-power process technologies offered by the semiconductor manufacturers. Circuit techniques refer to the transistor-level implementation of the logic and routing resources. Architecture techniques refer to functionality of the logic, I/O, and memory resources and the connectivity between these resources. System techniques refer to high-level low-power techniques such as dynamic voltage control, turning resources off when they are not being used, and run-time reconfiguration. Finally, CAD refers to enhancements made to the mapping tools which are used to configure the FPGA.

This paper is organized as follows. Section 2 describes the basic structure of an FPGA and summarizes the sources of power dissipation. Section 3 and Section 4 cover respectively system-level design techniques and device-level design techniques, both of which target mainly current commercial devices. Section 5 describes circuit-level and architecture-level design techniques for experimental devices. Section 6 and Section 7 present recent research respectively on power modeling and on low-power computer-aided design (CAD). Finally, Section 8 summarizes the paper and proposes future work.

2. FPGA Architecture and Power Dissipation

This section reviews the basic structure of an FPGA, focusing on what makes FPGAs power-hungry. FPGAs are made up of a large number of configurable logic blocks, which implement the logic part of digital circuits, and a configurable routing fabric, which implements the connections between the logic blocks. Modern FPGAs also have embedded fixed logic components, such as memories and arithmetic logic units. These embedded components are typically aligned with the logic tiles, and are often arranged in rows or columns. Figure 1 provides an abstract view of an FPGA with programmable logic and embedded fixed-function components.

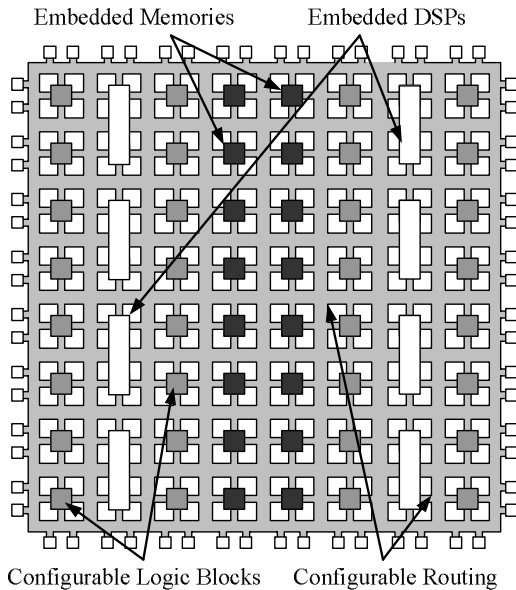


Figure 1: A generic FPGA with embedded components.

FPGAs dissipate more static power than ASICs for a number of reasons. FPGAs use a large amount of configuration memory to control every programmable routing switch and logic function in the FPGA. Each configuration bit dissipates static power. Another reason is that the programmable logic blocks are implemented using lookup-tables, which have significantly more transistors than the corresponding logic gates in an ASIC. Similarly, FPGA routing resources use significantly more transistors than in ASICs because of the large number of multiplexers needed to make the routing flexible.

FPGAs also dissipate more dynamic power than ASICs. In both an ASIC and FPGA, connections between gates are associated with some amount of parasitic capacitance due to the metal wire used to implement the connection as well as the driver and driven transistors. However, as described above, a connection in an FPGA also contains a large number of programmable switches. These switches significantly increase the parasitic capacitance on the wire segments and charging and discharging this parasitic capacitance consumes dynamic power.

Figure 2 shows a breakdown of core power consumption in a commercial 90-nm FPGA [61]. The figure shows that the routing resources dissipate the greatest amount of power, followed by logic and clock network resources. This study also reports that dynamic power accounts for 62% of the total power, while static power accounts for 38%. There is also recent work which considers FPGAs with embedded memories; such memories are found to account for 14% of core dynamic power [56].

3. Low-Power System-Level Design

This section describes various low-power design techniques that have been applied to current FPGA

technology by application developers. We classify these techniques into three categories: basic techniques, techniques involving run-time reconfigurability, and techniques for soft processors.

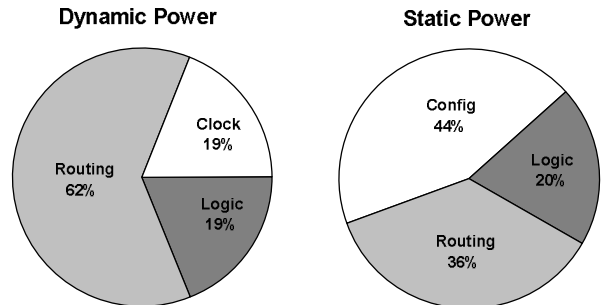


Figure 2: Breakdown of core power consumption in Xilinx Spartan-3 devices [61].

First, we provide five examples of basic techniques introduced below.

(a) It is usually preferable to use coarse-grained embedded blocks rather than the fine-grained configurable logic blocks in an FPGA, since the former are more power-efficient than the latter for the same function [29]. However one needs to ensure that in doing so, power consumption for routing would not increase significantly.

(b) Pipelining is a simple and effective way of reducing glitching, and hence minimising power consumption. It is found that, at a given clock speed, pipelining can reduce the amount of energy per operation by between 40% and 90% for applications such as integer multiplication, CORDIC, triple DES, and FIR filters [65].

(c) Word-length optimisation can be applied to obtain the best trade-off in speed, area, power consumption, flexibility, and accuracy. One approach is to study the sensitivity of outputs in a design to small errors due to rounding or truncation of internal variables for fixed-point hardware implementation. Improvements in power consumption of up to 98% (mean 87%) have been achieved for adaptive filters and polynomial evaluations [17].

(d) Clock gating can be used to reduce dynamic power consumption by disabling the clock for the inactive regions to prevent signal transitions. It can be combined with word-length optimisation (see (c) above) so that the circuitry in an operator is gated when not in use [47].

(e) Dynamic voltage scaling can be used to adapt the supply voltage to the FPGA as the temperature changes, to minimise power consumption. It has been shown that power reduction between 4% and 54% can be achieved for various arithmetic circuits [16].

Second, we provide two examples of low-power techniques involving run-time reconfigurability; the use of such reconfigurability has already been reported [49][62].

(a) Word-length optimisation can be combined with run-time reconfiguration so that the smallest design is adopted at a given time, as long as the energy reduction in execution is greater than the energy overhead for reconfiguration [48].

(b) Run-time reconfiguration can be applied to change a design in order to adapt to run-time conditions. For instance when a communication channel becomes more noisy, a more powerful but less energy efficient turbo coder can be used to maintain a fixed bit error rate, and vice versa [40]. Such run-time adaptation can further benefit from devices that support partial reconfiguration [46].

Third, we provide two examples of low-power techniques for FPGA-based soft processors; the use of such processors in adaptive systems has been reported [58].

(a) Based on an iterative improvement method, instruction set extensions to the MicroBlaze soft processor have been proposed [10]. Up to 40% reduction in energy and 12% reduction in peak power have been reported.

(b) Combined application of power-aware scheduling and instruction recoding techniques can be used to optimise a soft processor at multiple levels of abstraction. Dynamic power reduction of up to 74% has been obtained [18].

4. Device-Level Design: Commercial Devices

The latest FPGA devices from vendors such as Altera and Xilinx incorporate various low-power device-level technologies. This section reviews some of these technologies.

At the device level, Altera and Xilinx both utilize triple gate oxide technology, which provides a choice of three different gate thicknesses, to trade-off between performance and static power [3][67]. In earlier technologies, only two thicknesses were available. Transistors with thicker oxide were used for the large, higher voltage tolerant transistors in the I/O blocks and the thinner ones were used everywhere else. The new medium thickness oxide transistors provide slightly less performance than thin oxide transistors, but leak significantly less. In the latest FPGAs, these are used in the configuration memory and the switches that are controlled by this memory. Since the configuration memory remains static during the operation of the device, the oxide thickness does not affect the performance of the corresponding switches. To reduce dynamic power, FPGA vendors use a low-k dielectric between metal layers, which reduces the parasitic capacitance. This, in addition to smaller device geometries, reduces the average node capacitance and, correspondingly, dynamic power. Dynamic power of the core of the FPGAs can be reduced further by lowering the supply voltage because dynamic power has a quadratic relationship (CV^2f) with the supply voltage. Xilinx reduces the core supply voltage from 1.2V (in Virtex 4 FPGAs) to 1.0V (in Virtex 5 FPGAs), which cuts core power significantly. Similarly, the core supply voltage of Altera Stratix III FPGAs can be selected (by the user) to be either 1.1V, for high performance, or 0.9V, for lower power consumption.

Altera and Xilinx have also made a number of architecture-level changes to their latest devices to reduce static and dynamic power. Both vendors have recently increased the size of the LUTs (lookup tables) within the logic blocks [2][67]. By increasing the size of the basic logic elements, from 4-input LUTs to 6 and 7-input LUTs, both static and dynamic power are reduced since more logic is implemented within each LUT and less routing is needed between the LUTs. This reduces power since LUTs are implemented using smaller transistors (compared to transistors in the routing resources), which leak less and dissipate less dynamic power. Both vendors have also modified their routing architectures to increase the number of neighbouring logic blocks that can be reached in only one or two hops (each routing segment used counts as one hop). Using more 1-hop routes reduces the average capacitance of the routes, which improves both power and performance. Other architecture-level features that reduce overall power are the embedded memories, adders, and multipliers. Although each of these functions can be implemented using the programmable logic fabric, its implementation as a fixed-function embedded block is more power-efficient since circuitry to make it flexible is not needed, and it can be turned off when not used.

Finally, a number of low-power techniques have also been incorporated into the commercial FPGA CAD tools. Detailed power models have been integrated within the Altera Quartus II [2] and Xilinx ISE CAD tools [66]. Both vendors provide a spreadsheet utility to make early power predictions before the design is complete and a detailed power model that can be used when the design is complete. Early power estimates are based on estimated resource usage, I/O types, clock requirements, clock frequencies, and environmental conditions. The detailed power models provide estimates after the application has been placed, routed, and simulated. The estimations from the detailed power models are more accurate than those from the early power models, since detailed capacitance, leakage, and switching activity information is known for each node in the application circuit. In the case where simulation results are not available, only basic probability-based (vectorless) activity estimation is available and the accuracy of the power estimates is significantly reduced. This is especially true for sequential circuits. Power-aware CAD techniques have also been incorporated into the commercial CAD flows. In Quartus II, power is minimized during technology mapping, placement, and routing by minimizing the capacitance of high-activity signals using techniques similar to those described in the previous section. Power is also minimized by optimizing the mapping to the embedded memories, as described in [56], and, similarly, by optimizing the mapping to the embedded DSP blocks. In ISE, power is minimized during placement and routing by minimizing the capacitance of high-activity signals. Dynamic power dissipation is further minimized by strategically setting the configuration bits within partially used (some inputs are not used) LUTs to minimize switching activity. Both CAD tools also ensure that all

unused logic blocks, embedded blocks, routing resources, and clock network resources are turned off to save power.

Combining the above techniques, Altera reports that Stratix III FPGAs are over 50% more power efficient than Stratix II FPGAs [2]. Similarly, Xilinx reports that Virtex-5 FPGAs consume over 35% less dynamic power than Virtex-4 FPGAs, with even greater savings when embedded components are used [67]. Xilinx also points out that low-leakage techniques are already incorporated in their Virtex-4 FPGAs, resulting in 70% lower static power consumption when compared with competing FPGAs.

A low-power alternative to SRAM-based FPGAs is flash-based FPGA technology. Flash-based FPGAs, such as Actel's IGLOO devices, are inherently more efficient because flash-based memory dissipates significantly less leakage power compared to SRAM memory. For instance, Actel reports that their low-power FPGAs dissipate 4 times less leakage power than their nearest competitors [1].

5. Circuit- and Architecture-Level Design

The architecture and the circuit-level implementation of the FPGA is key in reducing power, since it directly affects the efficiency of mapping applications to FPGA resources, and the amount of circuitry to implement these resources.

A number of studies have investigated low-power FPGA architecture design. Energy-efficient FPGA routing architectures and low-swing signalling techniques to reduce power are described in [21][43]. In [55], a new FPGA routing architecture that utilizes a mixture of hardwired and traditional programmable switches is proposed, which reduces static and dynamic power by reducing the number of configurable routing elements. In [7], a novel FPGA routing switch with high-speed, low-power, or sleep modes is presented. The switch reduces dynamic power for non timing critical logic and standby power for logic when it is not being used. In [41], power-gating is applied to the switches in the routing resources to reduce static power; duplicate routing resources, that use either high or low V_{dd}, are used to reduce dynamic power. In [30], energy-efficient modules for embedded components in FPGAs are introduced to reduce power by optimizing the number of connections between the module and the routing resources, and by using reduced supply voltage circuit techniques. In [27], several power reduction techniques, such as register file elimination and efficient instruction fetch, are proposed for a coarse-grain reconfigurable cell-based architecture; up to 3.6 times lower energy than an ARM7 device, and up to 6 times lower energy than a C55X DSP, is reported.

Although significant improvements have already been made, many opportunities to further reduce power in FPGAs remain. The rest of this section describes two recent improvements: minimization of FPGA glitch power, and efficient FPGA clock network design.

The first improvement concerns FPGA glitch reduction. Glitching occurs when values at the inputs of a LUT toggle at different times due to uneven propagation delays of those signals. If the arrival times are far enough apart, spurious

transitions can be produced at the LUT output, as shown in Figure 3(a). A recent study suggests that glitching accounts for 31% of dynamic power dissipation in FPGAs [33].

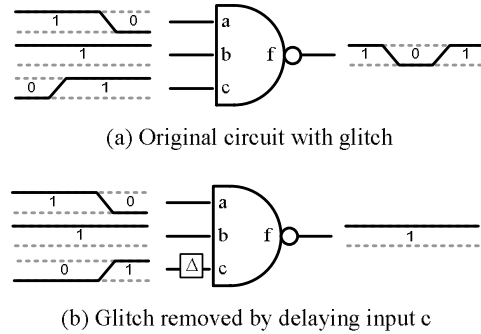


Figure 3: Example of delay insertion to eliminate glitching.

The study proposes a method for minimizing glitching which involves adding configurable delay elements to the inputs to each logic element in the FPGA (Figure 4). After place and route, detailed timing information is used to configure these delay elements so as to align the arrival times at the inputs of each logic element. This eliminates glitches as long as the arrival times can be aligned closely enough, as shown in Figure 3(b).

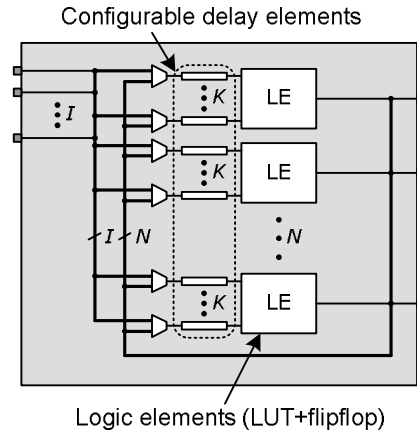


Figure 4: FPGA logic block with configurable delay elements.

The amount of glitching that can be eliminated depends on several factors. Specifically, the resolution, maximum delay, location, and amount of the programmable delay elements all have an affect on glitch elimination and overhead. It was found that, on average, the proposed technique eliminates 87% of the glitching, which reduces overall FPGA power by 17%, while the added circuitry increases the overall FPGA area by 6% and critical-path delay by less than 1%.

A 17% reduction in power is significant. Moreover, the method can be applied to all commercial FPGAs, and requires only minor changes to the CAD flow or the rest of the architecture. The gains are roughly independent of

those that can be obtained using process enhancement techniques. However, there may be some overlap in these gains with those that can be obtained using a power-aware CAD flow, since by reducing the activity of high-activity signals, there may be less “low-hanging fruit” available for the power-aware CAD flow.

The second recent improvement concerns low-power clock network design. New FPGAs are sophisticated enough to implement large system-level applications. These applications often have many clock domains. As an example, consider a communications application connected to several I/O ports. Each port might have its own clock, meaning the circuitry connected to each port must be controlled by a separate clock. FPGA vendors support such applications through the use of programmable clock networks that are flexible enough to support a wide range of applications, yet have low skew.

These clock networks have a significant impact on power since they connect to each flip-flop on the FPGA and toggle every clock cycle. In [61], the clock network in a current FPGA is shown to account for 19% of dynamic power (Figure 2). Moreover, depending on how flexible it is, the clock network can impose constraints that affect how applications can be placed within the FPGA. As an example, current FPGAs are divided into regions which can support a limited number of different clock domains. For applications with many clock domains, these constraints could force domains to be placed farther apart than they would otherwise be if the clock network is more flexible.

Recent work [34][35] examines the trade-off between the flexibility of FPGA clock networks and overall power consumption. This research has three parts. First, a parameterized framework for describing a wide range of FPGA clock networks. Second, a comparison of clock-aware placement techniques to determine their effectiveness: since clock networks impose hard constraints on the placement of logic blocks within the FPGA, a good clock-aware placement algorithm must obey these constraints and also optimize for speed, routability, and power consumption. Several techniques for combining these objectives are evaluated, in terms of their ability to find a placement that is fast, energy efficient, and legal. Third, experiments to determine what makes an efficient FPGA clock network. It is found that FPGA clock networks with more flexibility near the sources (pads or internal sources) and less flexibility near the sinks (flip-flops) are more efficient in terms of overall power consumption. Also dividing FPGA into clock regions that can be driven by global or local clock sources significantly reduces the area and power dissipation of the clock network.

Both the parameterized clock network framework and the clock-aware placement techniques have been incorporated into the popular VPR CAD tool [9] and is publicly available. The significance of this work is thus two-fold: (1) techniques that help FPGA vendors to provide more efficient clock networks, and (2) a new approach for

architectural exploration that helps to guide future researchers.

6. FPGA Power Modelling

Accurate power modelling is important in low-power FPGA design for a number of reasons. First, application designers need detailed power estimates to ensure their application meets various power budgets. Second, power-aware FPGA CAD tools require detailed power estimates in order to minimize power. Third, FPGA designers who development new FPGA architectures and CAD tools need power estimates to evaluate new low-power techniques.

A number of FPGA power models have recently been presented in the literature. In [50], a detailed power model that estimates static and dynamic power of the logic, routing, and clock network for a range of FPGAs with different architecture parameters is described. For static power, the model uses a first-order analytical technique which calculates leakage based on transistor size and various technology-specific parameters. For dynamic power, the model uses transistor-level capacitance information from the VPR place and route tool [9] and switching activity information obtained using vectorless activity estimation techniques. In [36], a similar FPGA power model estimates static and dynamic FPGA power by calculating the power for each clock cycle using simulated switching activity information, instead of vectorless techniques. This power model has been enhanced to support FPGAs with a programmable supply voltage [37] and programmable threshold voltages [38]. In [13][25][52], high-level FPGA power models that use macro-models to estimate power are described. These models characterize the power consumption of various FPGA components, such as adders, multipliers, and programmable logic, for low-power high-level synthesis or design space exploration.

One of the main challenges in power modelling is activity estimation, which involves determining how often each node in the FPGA toggles. This activity information is needed in order to calculate how much dynamic power an application dissipates when it operates. Estimating activities is challenging because there are large a number of nodes within each circuit and complex interactions between the nodes which are difficult to model.

One approach is *vectorless* activity estimation which involves estimating the switching activity of each node based on the switching activities of the inputs and the logic function of that node. The advantage of vectorless activity estimation is that it is typically fast and does not require input vectors. The disadvantage is that it is less accurate than simulation because it typically does not consider the complex interactions between nodes.

Vectorless techniques that model these complex interactions have been proposed [44][45][52][59][60]. In general, however, there is trade-off between speed and accuracy. Some vectorless techniques are even slower than brute force simulation. A recent study [32] compares some vectorless techniques to determine which are fast enough to

be used in FPGA CAD flow. Specifically, the aim of the study is to identify the most accurate techniques that do not noticeably slow down the FPGA design flow. These techniques and two novel techniques are then integrated into a new publicly available activity estimation tool called ACE-2.0.

7. Low-Power FPGA CAD

FPGA CAD tools, which map an application to the FPGA programmable fabric, can also have a significant impact on power consumption. This mapping generally occurs in five stages: high-level synthesis, technology mapping, clustering, placement, and routing. Each stage can be optimized to improve the final implementation.

Power-aware high-level synthesis algorithms for FPGAs are presented in [12][15]. In [12], power is reduced by minimizing the total power of the operations and the size of the multiplexers that connect them. The algorithm described in [15] targets FPGAs with programmable power supplies and minimizes power by assigning low-V_{dd} to as many operations as possible given resource and timing constraints. Low-power technology mapping algorithms are presented in [5][8][14][19][31][39][63][64]. In general, these algorithms minimize power by absorbing as many high-activity nodes as possible when the gates are packed into LUTs and/or by minimizing node-duplication, which tends to increase the amount of interconnect between the LUTs. Low-power clustering techniques have been described in [11][23][31][54]. These algorithms minimize power by absorbing as many small (low fan-out) and high-activity nets as possible when the LUTs are packed into clusters (logic blocks). Absorbing small nets tends to reduce number of inter-cluster nets (which dissipate the most power) and absorbing high-activity nets further reduces power. Low-power place and route techniques were presented in [28][31][53][57], which minimize power by reducing the distance between logic blocks connected by high-activity wires (during placement) and by routing high-activity wires as directly as possible (during routing). In [8], leakage power is minimized by choosing low-leakage LUT configurations. Finally, in [56], power-aware algorithms for mapping logical memories to the physical FPGA embedded memories were described. The algorithms minimize dynamic power consumed by embedded memories by evaluating a range of possible mappings and selecting the most power-efficient choice.

8. Conclusions and Future Work

Significant improvements have been made to improve power and energy efficiency of FPGAs. This paper describes many of these improvements, which range from low-level process and circuit design techniques through to high-level CAD techniques. While further improvements will likely be made at all levels, there seems to be significant potential for power savings at the system level.

At the system level, power reduction can be obtained by optimizing management and scheduling of system

resources. As an example, programmable logic devices (PLD) such as FPGAs can be used to reduce power dissipation in mobile applications by effective exploitation of “deep-sleep” mode in mobile processors. When the processor is not needed, a programmable logic device can be used to monitor external resources, such as battery gauges, sensors, and interrupts, and determine when the host processor (or other devices) should be brought out of deep-sleep mode. Although the host processor itself can be used to monitor the external resources, it could be 10-100 times less efficient since it uses more power (when it is not in deep-sleep) when compared to a small system monitor implemented in a PLD [4]. As another example, FPGAs can be used as coprocessors to perform compute intensive tasks more efficiently than in software. Because it is flexible, the hardware implementation of the coprocessor can be optimized for the given task and even for specific input parameters such as media format.

Research targeting these system-level power tradeoffs is required. In the previous example, there is a trade-off between the power savings that can be achieved by not using the host processor and the cost of initializing the PLD and the processor when tasks are passed from one to the other. Several factors can affect the power savings, including the scheduling of tasks to maximize the duration that the processor can sleep and minimizing the cost of configuring the programmable logic device. Moreover, as programmable logic devices become more sophisticated, many of these system-level issues can be effectively addressed by FPGA technology. FPGAs with embedded processors and soft-processors are already available [19]. This introduces similar system-level tradeoffs and the potential for significant power savings.

Research targeting low-power system-level benchmarks is also required. To our knowledge, current academic benchmarks do not support this type of research. Specifically, benchmarks are needed that perform complicated tasks and provide realistic input stimuli reflecting how the applications are used. These should include description of when and how often different computations are required, as well as realistic input data that must be processed by the application. Benchmarks involving adaptive systems are of particular interest, since it appears promising to study how functional adaptation can be extended to cover power and energy reduction.

Acknowledgement. The support of Canadian Natural Science and Engineering Research Council, UK Engineering and Physical Research Science Council, European Commission FP6 project hArtes, Agility Design Solutions, Celoxica and Xilinx is gratefully acknowledged.

References

- [1] Actel, “IGLOO Handbook,” 2008.
- [2] Altera, “Quartus II Handbook,” Vol. 2, Chapter 9, 2007.
- [3] Altera, “Quartus II Handbook,” Vol. 3, Chapter 10, 2007.
- [4] Altera Corp., “Cut power 100X using CPLD coprocessors in Portable Applications,” Webcast, Dec. 2007.

- [5] M.J. Alexander, "Power optimization for FPGA look-up tables," Proc. ACM Int. Symp. on Physical Design, pp. 156-162, 1997.
- [6] J.H. Anderson and F.N. Najm, "Power-aware technology mapping for LUT-based FPGAs," Proc. IEEE Int. Conference on Field-Prog. Technology, pp. 211-218, 2002.
- [7] J.H. Anderson and F.N. Najm, "A novel low-power FPGA routing switch," Proc. IEEE Custom Integrated Circuits Conf., pp. 719-722, 2004.
- [8] J.H. Anderson, F.N. Najm, and T. Tuan, "Active leakage power optimization for FPGAs," IEEE Trans. on Computer-Aided Design, vol. 25, no. 3, pp. 423-437, March 2006.
- [9] V. Betz., J. Rose, and A. Marquardt, "Architecture and CAD for deep-submicron FPGAs," Kluwer Academic Publishers, 1999.
- [10] P. Biswas et al, "Performance and energy benefits of instruction set extensions in an FPGA soft core," Proc. Int. Conf. on VLSI Design, pp. 651-656, 2006.
- [11] D. Chen and J. Cong, "Delay optimal low-power circuit clustering for FPGAs with dual supply voltages," Proc. Int. Symp. on Low Power Electronics and Design, pp. 70-73, 2004.
- [12] D. Chen, J. Cong, and Y. Fan, "Low-power high-level synthesis for FPGA architecture," Low Power Electronics and Design, pp. 134-139, 2003.
- [13] D. Chen, J. Cong, Y. Fan, and Z. Zhang, "High-level power estimation and low-power design space exploration for FPGAs," Proc. Asia South Pacific Design Automation Conf., pp. 529-534, 2007.
- [14] D. Chen, J. Cong, F. Li, and L. He, "Low-power technology mapping for FPGA architectures with dual supply voltages," Proc. ACM Int. Symp. on Field-Prog. Gate Arrays, pp. 109-117, 2004.
- [15] D. Chen, J. Cong, and J. Xu, "Optimal module and voltage assignment for low-power," Proc. Asia South Pacific Design Automation Conf., pp. 850-855, 2005.
- [16] C.T. Chow et al, "Dynamic voltage scaling for commercial FPGAs," Proc. IEEE Int. Conf. on Field Prog. Technology, 2005.
- [17] G. Constantinides, "Word-length optimization for differentiable nonlinear systems," ACM Trans. on Design Automation of Electronic Sys., vol. 11, no. 1, pp. 26-43, 2006.
- [18] R. Dimond, O. Mencer and W. Luk, "Combining instruction coding and scheduling to optimize energy in system-on-FPGA," Proc. IEEE Symp. on Field-Prog. Custom Computing Machines, IEEE Computer Society Press, 2006.
- [19] A.H. Farrahi and M. Sarrafzadeh, "FPGA technology mapping for power minimization," Proc. Int. Workshop on Field-Prog. Logic and Applications, pp. 167-174, 1994.
- [20] B.H. Fetcher, "FPGA embedded processors: revealing true system performance," Proc. Embedded Sys. Conf., ETP-357, 2005.
- [21] V. George, H. Zhang, and J. Rabaey, "The design of a low energy FPGA," Proc. Int. Symp. on Low Power Electronics and Design, pp. 188-193, 1999.
- [22] K. Glette, J. Torresen and M. Yasunaga, "Online evaluation for a high-speed image recognition system implemented on a Virtex-II Pro FPGA," Proc. NASA/ESA Conf. on Adaptive Hardware and Sys., pp. 463-470, IEEE, 2007.
- [23] H. Hassan, M. Anis, A. El Daher, and M. Elmasry, "Activity packing in FPGAs for leakage power reduction," Proc. Design Automation and Test in Europe, pp. 212-217, 2005.
- [24] International Technology Roadmap for Semiconductors, 2005.
- [25] T. Jiang, X. Tang, and P. Banerjee, "Macro-models for high level area and power estimation on FPGAs," Proc. ACM Great Lakes Symp. on VLSI, pp. 162-165, 2004.
- [26] B. Kapoor, "Improving the accuracy of circuit activity measurement," Proc. ACM Design Automation Conf., pp. 734-739, 1994.
- [27] S. Khawam et al, "The reconfigurable instruction cell array," IEEE Trans. on VLSI Sys., vol. 16, no. 1, pp. 75-85, 2008.
- [28] B. Kumthekar, and F. Somenzi, "Power and delay reduction via simultaneous logic and placement optimization in FPGAs," Proc. Design Automation and Test in Europe, pp. 202-207, 2000.
- [29] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," IEEE Trans. on Computer-Aided Design, vol. 26, no. 2, pp. 203-215, Feb. 2007.
- [30] E. Kusse and J. Rabaey, "Low-energy embedded FPGA structures," Proc. Int. Symp. Low Power Electronics and Design, pp. 155-160, 1999.
- [31] J. Lamoureux and S.J.E. Wilton, "On the Interaction between Power-Aware CAD Algorithms for FPGAs," Proc. IEEE Int. Conf. on Computer Aided Design, pp. 701-708, 2003.
- [32] J. Lamoureux and S.J.E. Wilton, "Activity estimation for Field-Programmable Gate Arrays," Proc. Intl. Conf. on Field-Prog. Logic and Applications, pp. 87-94, 2006.
- [33] J. Lamoureux, G.G. Lemieux, and S.J.E. Wilton, "GlitchLess: dynamic power minimization in FPGAs through edge alignment and glitch filtering," to appear in IEEE Trans. on Very Large Scale Integration Systems, 2008.
- [34] J. Lamoureux and S.J.E. Wilton, "FPGA clock network architecture: flexibility vs. area and power," Proc. ACM Int. Symp. on Field-Prog. Gate Arrays, pp. 101-108, 2006.
- [35] J. Lamoureux and S.J.E. Wilton, "Clock-aware placement for FPGAs," Proc. Int. Conf. on Field-Prog. Logic and Applications, pp. 124-131, 2007.
- [36] F. Li, D. Chen, L. He, and J. Cong, "Architecture evaluation for power-efficient FPGAs," Proc. ACM Int. Symp. on Field-Programmable Gate Arrays, pp. 175-184, 2003.
- [37] F. Li, Y. Lin, and L. He, "FPGA power reduction using configurable dual-Vdd," Proc. Design Automation Conf., pp. 735-740, 2004.
- [38] F. Li, Y. Lin, L. He, and J. Cong, "Low-power FPGA using pre-defined dual-Vdd/dual-Vt fabrics," Proc. ACM Int. Symp. on Field-Prog. Gate Arrays, pp. 42-50, 2004.
- [39] H. Li, S. Katkoori, and W.-K. Mak, "Power minimization algorithms for LUT-based FPGA technology mapping," ACM Trans. on Design Automation of Electronic Sys., vol. 9, no. 1, pp. 33-51, 2004.
- [40] J. Liang, R. Tessier and D. Goeckel, "A dynamically reconfigurable power efficient turbo coder," Proc. IEEE

- Symp. on Field-Prog. Custom Computing Machines, IEEE Computer Society Press, pp. 91-100, 2004.
- [41] Y. Lin, F. Li, and L. He, "Routing track duplication with fine-grained power-gating for FPGA interconnect power reduction," Proc. Asia South Pacific Design Automation Conf., pp. 645-650, 2005.
- [42] R. Marculescu, D. Marculescu, and M. Pedram, "Switching activity analysis considering spatiotemporal correlations," Proc. IEEE Int. Conf. Computer-Aided Design, pp. 294-299, 1994.
- [43] M. Meijer, R. Krishnan, and M. Bennebroek, "Energy efficient FPGA interconnect design," Proc. Conf. on Design and Test in Europe, pp. 1-6, 2006.
- [44] J. Monteiro and S. Devadas, "A methodology for efficient estimation of switching activity in sequential logic circuits," Proc. ACM/IEEE Design Automation Conf., pp. 12-17, 1994.
- [45] F. Najm, "Low-pass filter for computing the transition density in digital circuits," IEEE Trans. on Computer-Aided Design, vol. 13, no. 9, pp. 1123-1131, 1994.
- [46] J. Noguera and I.O. Kennedy, "Power reduction in network equipment through adaptive partial reconfiguration," Proc. Int. Conf. on Field Prog. Logic and Applications, IEEE, pp. 240-245, 2007.
- [47] W.G. Osborne, W. Luk, J.G.F. Coutinho and O. Mencer, "Power and branch aware word-length optimisation," Proc. IEEE Symp. on Field-Prog. Custom Computing Machines, IEEE Computer Society Press, 2008.
- [48] W.G. Osborne, W. Luk, J.G.F. Coutinho and O. Mencer, "Reconfigurable design with clock gating," Proc. Int. Symp. on Systems, Architectures, Modelling and Simulation, 2008.
- [49] K. Paulsson, M. Hubner and J. Becker, "Strategies to on-line failure recovery in self-adaptive systems based on dynamic and partial reconfiguration," Proc. NASA/ESA Conf. on Adaptive Hardware and Sys., IEEE, pp. 288-291, 2006.
- [50] K.K.W. Poon, S.J.E. Wilton, and A. Yan, "A detailed power model for field-programmable gate arrays," ACM Trans. on Design Automation of Electronic Systems, vol. 10, no. 2, pp. 279-302, April 2005.
- [51] A. Raghunathan, S. Dey and N. K. Jia, "Register transfer level power optimization with emphasis on glitch analysis and reduction," IEEE Trans. on Computer-Aided Design, vol. 18, no. 8, pp. 1114-1131, 1999.
- [52] A. Reimer, A. Schulz, and W. Nebel, "Modeling macromodules for high-level dynamic power estimation of FPGA-based digital designs," Proc. Int. Symp. on Low Power Electronics and Design, pp. 151-154, 2006.
- [53] K. Roy, "Power-dissipation driven FPGA place and route under timing constraints," IEEE Trans. on Circuits and Sys., vol. 46, no. 5, pp. 634-637, 1999.
- [54] A. Singh, G. Parthasarathy, and M. Marek-Sadowski, "Efficient circuit clustering for area and power reduction in FPGAs," ACM Trans. on Design Automation of Electronic Systems, vol. 7, no. 4, pp. 643-663, 2002.
- [55] S. Sivaswamy, G. Wang, C. Ababei, K. Bazargan, R. Kastner, and E. Bozargzadeh, "HARP: hard-wired routing pattern FPGAs," Proc. Int. Symp. on Field-Prog. Gate Arrays, pp. 21-29, 2005.
- [56] R. Tessier, V. Betz, D. Neto, A. Egier, and T. Gopalsamy, "Power-Efficient RAM mapping algorithms for FPGA embedded memory blocks," IEEE Trans. of Computer- Aided Design, vol. 26, no. 2, pp. 278-289, Feb 2007.
- [57] N. Togawa et al, "A simultaneous placement and global routing algorithm for FPGAs with power optimization," Proc. Asia Pacific Conf. on Circuits and Sys., pp. 125-128, 1998.
- [58] J. Torresen and J. Jakobsen, "An FPGA-implemented processor architecture with adaptive resolution," Proc. NASA/ESA Conf. on Adaptive Hardware and Sys., IEEE, pp. 386-389, 2006.
- [59] C.Y. Tsui et al, "Exact and approximate methods for calculating signal and transition probabilities in FSMs," Proc. ACM/IEEE Design Automation Conf., pp. 18-23, 1994.
- [60] C.Y. Tsui, M. Pedram, and A.M. Despaigne, "Efficient estimation of dynamic power consumption under a real delay model," Proc. IEEE Int. Conf. Computer-Aided Design, pp. 224-228, 1993.
- [61] T. Tuan, A. Rahman, S. Das, S. Trimberger, and S. Kao, "A 90-nm low-power FPGA for battery-powered applications," IEEE Trans. on Computer-Aided Design, vol. 26, no. 2, pp. 296-300, Feb. 2007.
- [62] A. Upegui and E. Sanchez, "Evolving hardware with self-configurable connectivity in Xilinx FPGAs," Proc. NASA/ESA Conf. on Adaptive Hardware and Sys., IEEE, pp. 153-162, 2006.
- [63] C-C. Wang and C-P Kwan, "Low power technology mapping by hiding high-transition paths in invisible edges for LUT-based FPGAs," Proc. IEEE Int. Symp. on Circuits and Sys., pp. 1536-1539, 1997.
- [64] Z-H. Wang, E-C. Liu, J. Lai, and T-C. Wang, "Power minimization in LUT-based FPGA technology mapping," Proc. ACM Asia South Pacific Design Automation Conf., pp. 635-640, 2001.
- [65] S.J.E. Wilton, S-S. Ang, and W. Luk. "The impact of pipelining on energy per operation in field programmable gate arrays". In Proc. Field Prog. Logic and Applications, LNCS 3203, pp. 719-728, 2004.
- [66] Xilinx, "Optimizing FPGA power with ISE design tools," Xcell Journal, Issue 60, pp. 16-19, 2007.
- [67] Xilinx, "Power Consumption in 65nm FPGAs," 2007.