CAD Techniques for Power Optimization in Virtex-5 FPGAs

Subodh Gupta[1], Jason Anderson[2], Linda Farragher[2], and Qiang Wang[1]

[1]Xilinx, Inc., San Jose, CA, USA
[2]Xilinx, Inc., Toronto, ON, Canada
{subodhg, janders, lindaf, qiangw}@xilinx.com

Abstract

We consider dynamic power dissipation in FPGAs and present CAD techniques for dynamic power reduction. The proposed techniques, comprising power-aware placement, routing, and a novel post-routing transformation, are applied to optimize the power consumed by industrial designs implemented in the Xilinx® Virtex™-5 FPGA. Board-level power measurements on a suite of industrial designs show that the techniques reduce power by 10%, on average.

I. Introduction

In more than 20 years since the introduction of the FPGA, research and development has produced dramatic improvements in FPGA speed and area efficiency, narrowing the gap between FPGAs and ASICs and making FPGAs the platform of choice for implementing digital circuits. Today, power consumption is a key concern for FPGA vendors and customers. Reducing the power of FPGAs is key to lowering packaging and cooling costs, improving device reliability, and opening the door to new markets such as mobile electronics [1].

Power dissipation in CMOS circuits comprises both static (leakage) power and dynamic power. Dynamic power is caused by transitions on the signals of a circuit and is governed by the equation [2]:

$$P_{avg} = \frac{1}{2} \sum_{i \in signals} C_i \cdot f_i \cdot V^2 \quad (1)$$

where C_i represents the capacitance of a signal, i; f_i is referred to as "switching activity" and represents the rate of transitions on signal i; and V is the supply voltage. Static power, on the other hand, is consumed when a circuit is in a quiescent, idle state. Static power results from leakage currents in off transistors, primarily sub-threshold and gate oxide leakage. An off MOS transistor is an imperfect insulator, allowing sub-threshold leakage current to flow between its drain and source terminals and tunneling current to flow through the gate oxide. Technology scaling, such as the recent move to $65nm$, means lower supply voltages and smaller transistor dimensions, leading to shorter wire lengths, less capacitance, and an overall reduction in dynamic power for implementing a given circuit. However, the higher integration leads to a net increase in dynamic power density (Watts/cm^2). Smaller process geometries also mean shorter transistor channel lengths and thinner gate oxides, producing an increase in static power as technology scales.

Computer-aided design (CAD) techniques for FPGA power optimization have appeared in the literature (e.g., [3]) and have been applied successfully to academic FPGA models; however, there is little published work on power-aware CAD for commercial FPGAs. In this paper, we describe CAD techniques that provide dynamic power reduction in the Xilinx Virtex-5 FPGA – a state-of-the-art $65nm$ FPGA. The techniques have been incorporated into the Xilinx ISE™ 9.2i software. The paper is organized as follows: Section II presents background and related work. Section III describes techniques for reducing power during the place and route steps of the flow. Section IV presents a novel technique for reducing power within an FPGA's logic blocks; it can be applied at the post-routing stage without any area or speed penalty. Power reduction results are given in Section V. Conclusions and suggestions for future work are offered in Section VI.

II. Power Dissipation in FPGAs

The programmability and flexibility of FPGAs make them less power-efficient than custom ASICs for implementing a given logic circuit. The FPGA configuration circuitry and configuration memory consume silicon area, producing longer wire lengths and higher interconnect capacitance. Programmable routing switches attach to the pre-fabricated metal wire segments in the FPGA interconnect and add to the capacitive load incurred by signals. To be sure, most dynamic power in an FPGA is consumed in the programmable routing fabric [4]. Likewise, static power is proportional to total transistor width. The interconnect comprises a considerable fraction of an FPGA's transistors and therefore dominates leakage [5, 6]. Consequently, the interconnection fabric should be a prime target for FPGA power optimization.

Certainly, power can be addressed through process technology changes, at the hardware architecture level, or at the circuit level. Virtex-5 FPGAs, for example, contain "diagonal" interconnect resources, allowing connections to be made with fewer routing conductors and reduced interconnect capacitance. At the transistor level, Virtex-4 and Virtex-5 FPGAs employ triple-oxide process technology for leakage mitigation. Three possible oxide thicknesses are available for each transistor, depending on its speed, power, and reliability requirements. The proliferation and increased use of hard IP blocks, such as DSPs and processors, can also lower power consumption versus implementing such functionality in the standard FPGA fabric. An overview of approaches for optimizing FPGA power can be found in [7] and [8].

It is also possible to reduce power without incurring any costly hardware or process changes. Power issues can be tackled through power-driven CAD algorithms and design flows.

III. Power Optimization in Placement and Routing

A. Placement

The core algorithm in the Xilinx placer uses analytical (mathematical) techniques. It begins with an initial overlapped placement of a design and then uses a force abstraction to move

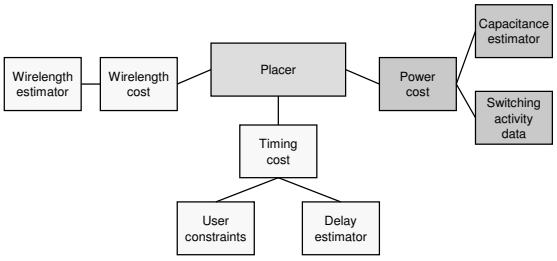


Fig. 1. Placer costing scheme incorporating power.

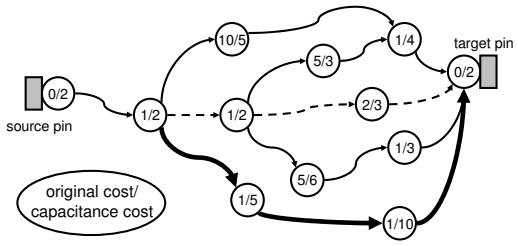


Fig. 2. Routing graph with capacitance costs on nodes.

logic blocks away from highly congested regions, ultimately producing a feasible, overlap-free placement. Once analytical placement is finished, swap-based local optimization is run on the placed design to further refine the placement. The traditional cost function used in our placer considers wirelength and timing as follows:

$$Cost = a \cdot W + b \cdot T \tag{2}$$

where W and T are the wirelength and timing costs, respectively, and a and b are scalar weights. The values of a and b can be set according to the relative priority of timing versus wirelength. The placer costing scheme is illustrated in Fig. 1.

As actual routes are not available during placement, the wirelength cost depends on wirelength estimation (e.g., using half-perimeter bounding box or minimum spanning tree). Likewise, the timing cost depends on estimates of connection delays and user-supplied constraints. To optimize power, we have extended the analytical placement and local optimizations by adding a power component to the cost function, as shown on the right-hand side of Fig. 1. The modified cost function is as follows:

$$Cost = a \cdot W + b \cdot T + c \cdot P_{avg} \tag{3}$$

where P_{avg} is the estimated dynamic power computed through (1) and c is a scalar weight. Computing P_{avg} requires a switching activity value for each signal and a user can extract signal switching activity data from simulation and supply it to the tool. Alternately, if a user does not provide any switching activity data, then the tool assumes a default switching activity for the primary inputs and sequential outputs and propagates activity to the rest of the signals, considering the logic functionality (e.g., using [9]). For best results, user-supplied switching activity data is required. During placement, the capacitance of a signal is not known and must therefore be estimated. Using the approach [10], we constructed an empirically derived capacitance estimation model based on signal parameters available during placement:

$$C_i = f(FO_i, XS_i, YS_i) \tag{4}$$

where f represents a generic mathematical function, C_i is the capacitance of signal i, FO_i is the number of fanouts of signal i, and XS_i is the X-span and YS_i is the Y-span of signal i in the placement, respectively. These parameters are architecture-independent and are available during placement.

To build the model, we extracted the capacitance, number of fanouts, X-span, and Y-span for each signal in a suite of designs collected from Xilinx customers. We then used least-squares regression analysis to fit the capacitance to a quadratic function of the model parameters. On average, (4) gives a 30% error for a wide range of designs.

B. Routing

Once the logic blocks are assigned to physical locations on the FPGA, we must route the connections between the blocks. The router uses a negotiated congestion routing algorithm, where during initial iterations, shorts between signals are permitted. In later iterations, the penalty for creating shorts is increased, until no routing conductor is used by more than one signal. Timing-critical connections are routed to minimize their delay, which involves computationally intensive RC delay calculations. Most connections, however, are not timing-critical.

The different types of routing resources in Virtex-5 have different capacitances associated with them. For example, a PENT resource (a conductor spanning 5 tiles) has a higher capacitance than a DOUBLE resource (spans 2 tiles). In the power-aware router, we choose to optimize the capacitance of non-critical connections. To achieve this, we changed the router's cost function for non-timing critical connections to consider capacitance, as opposed to the prior approach of basing cost on other factors such as estimated delay, length or scarcity. To illustrate the algorithm, consider the routing graph of Fig. 2.

Each node in the graph represents a routing conductor or logic block pin and each edge represents a programmable routing switch. The router must select a path between the source and target pins. The original cost and capacitance cost of each node is shown internal to each node in the figure. To minimize the original costs, the routing between the source and target pin would have taken the bold path. However, in the power-aware flow, the router will use the path shown dashed, which has lower overall capacitance.

In the negotiated congestion router, as shorts between signals are gradually resolved, the routing paths for signals generally become more circuitous as diversions around highly-congested regions are necessitated. We use switching activity data to guide the negotiation process in the power-aware router. Specifically, signals with high switching activity

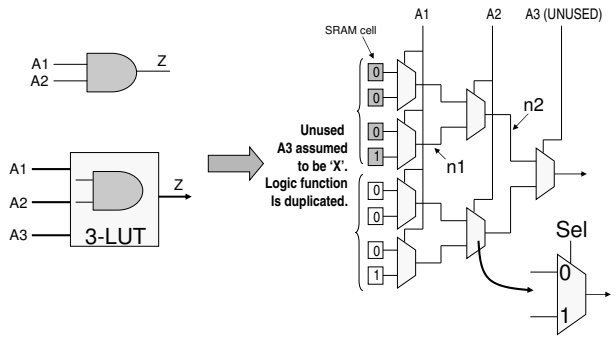


Fig. 3. Optimizing power within a LUT.

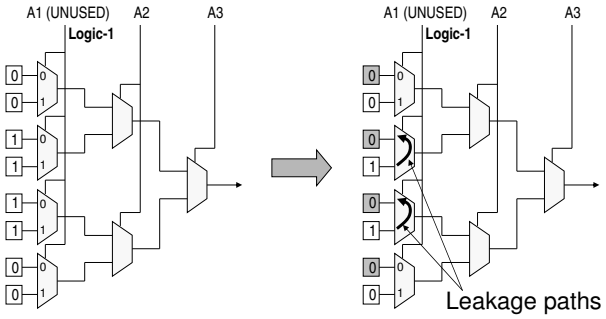


Fig. 4. Avoiding additional leakage paths.

are given preference for retaining low-capacitance routing resources. That is, we establish a preference for removing shorts by diverting low activity signals, where possible.

IV. Reducing Power Within Logic Blocks

The place and route optimizations described above lower power in the interconnection fabric. We also devised an approach to reduce power within logic blocks, specifically in look-up-tables (LUTs) when not all LUT inputs are used (Fig. 3). A K-input LUT is a small memory, with 2^K SRAM bits, capable of implementing any logic function with up to K inputs. LUTs form the basis of logic blocks in commercial FPGAs and are used to implement combinational logic. Fig. 3 shows how a hypothetical three-input LUT (having inputs A1, A2, and A3) implements a two-input logical-AND function. The truth table for the logical-AND is shown in the LUT SRAM contents on the left side of the multiplexer tree.

Note that input A3 is unused in Fig. 3. Normally, unused inputs are treated as "don't cares" and assumed to be either logic-0 or logic-1. Consequently, to account for this in the case of Fig. 3, the Xilinx software "replicates" the logic function in both the top and bottom half of the LUT SRAM memory contents. Unused LUT inputs occur frequently in customer designs, especially in Virtex-5 designs, where LUTs have six inputs. We analyzed over 90 Virtex-5 industrial customer designs and found that about 55% of LUTs had at least 1 unused input.

In the Virtex-5 hardware, unused LUT inputs are not allowed to float to an unspecified voltage. Rather, unused inputs are pulled high to logic-1, and this property is the root of our optimization. If A3 is pulled to logic-1, the top input to the deepest two-input multiplexer in the tree is never selected. However, because the logic function is replicated in both the top and bottom half of the LUT memory contents, toggling can occur on internal multiplexer nodes n1 and n2 based on changes in signal inputs A1 and A2. This toggling consumes dynamic power needlessly, as transitions on n1 and n2 are never propagated to the LUT output.

The optimization entails detecting unused LUT inputs at the post-routing stage and intelligently setting certain LUT memory contents to be logic-0 so as to eliminate unnecessary internal toggling, without disrupting the logic functionality. Returning to the example of Fig. 3, the top half of the LUT memory contents, shown shaded, would be set to logic-0. No toggling would occur on internal nodes n1 and n2, and therefore no dynamic power would be consumed by charging or discharging n1 and n2.

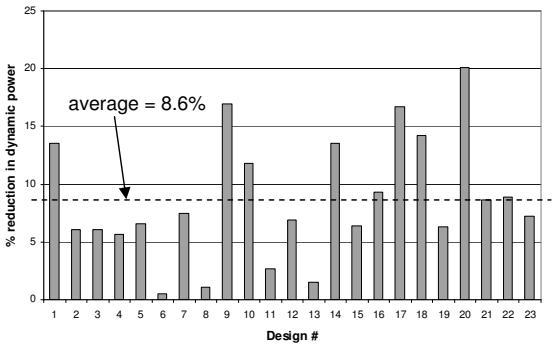
A crucial observation is that this optimization should not be applied in all cases, and specifically, it should not be applied in the case of *low order* unused LUT inputs[1]. Consider the example shown in the left of Fig. 4, where input A1 is unused and the truth table reflects a different logic function from that above. In this case, applying the optimization produces the result shown on the right of Fig. 4, where the shaded SRAM cells have their content set to logic-0. Notice that in this case, there is no benefit to dynamic power because whether the shaded SRAM cells are logic-0 or logic-1 does not affect toggling inside the LUT. More importantly, leakage power is actually made worse in this case, as applying the optimization creates new leakage paths from logic-1 to logic-0 in the first multiplexer stage of the LUT, as shown in the figure. In general, the optimization should only be applied for an unused input, P, if there exists a *used* input of lower order than P.

It is worth mentioning that this approach is a "free" power optimization technique. It can be executed at the post-routing stage and has no performance, area or hardware cost.

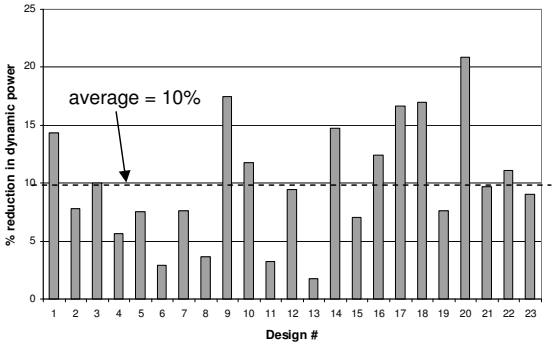
V. Results

We evaluated the power optimization techniques using 23 industrial designs collected from Xilinx customers targeted to Virtex-5. The designs range in size from about 1200 6-LUTs to 54000 6-LUTs. The designs were augmented with built-in automatic input vector generation by attaching a linear feedback shift register-based (LFSR-based) pseudo-random vector generator to the primary inputs. This permitted us to perform board-level measurements of power without requiring a large number of externally applied waveforms. The results presented here are based on measurements of current drawn from the V_{ccint} supply during circuit operation. Dynamic power was separated from static power by taking each design and clocking it at multiple frequencies, making a current measurement at each frequency, and then using the property that dynamic power scales linearly with frequency to break out dynamic from static current.

[1] A LUT input P is of lower order than a LUT input Q, if P is further away from the LUT's output than Q.



a) Results for power-aware place & route



b) Combined results including logic block power optimization

Fig. 5. Dynamic power optimization results.

Each industrial design was initially placed and routed in a baseline, non-power-aware run with tough-to-meet performance constraints. This initial implementation was simulated with ModelSim and switching activity data was extracted for all of its signals. The switching activity data was provided as input to a second, power-aware run of placement and routing. The results compare the dynamic power consumed by the baseline implementation, with that consumed by the power-aware implementation.

We begin by evaluating the benefits of power-aware placement and routing alone, without the LUT power reduction optimization scheme described in Section IV. The results are shown in Fig. 5(a). The horizontal axis represents each of the 23 designs; the vertical axis shows the percentage reduction in dynamic power due to power-aware layout synthesis. On average, across all 23 designs, dynamic power is reduced by up to 20% and 8.6%, on average. Note, however, that the power reductions do come with a performance cost: on average, the speed performance hit was between 3% and 4%, which we believe is acceptable in power-conscious designs.

Fig. 5(b) shows the combined results for all optimizations, including logic block power optimization. On average, across all 23 designs, dynamic power is reduced by up to 21% and 10%, on average. Comparing the results with those in Fig. 5(a), we observe that the logic block power reduction technique accounts for 1.4% of the overall 10%, without any additional speed performance degradation. Considering that these are initial results coming from software changes localized to the back-end of the CAD flow, we consider the power benefits quite encouraging.

VI. Conclusion

Many future directions exist for further reducing power in design tools. In front-end HDL synthesis, optimizations from the ASIC domain can be adapted for FPGAs, such as clock gating and operand isolation. Power-aware logic synthesis and activity-driven technology mapping to LUTs are well-studied in the literature and will yield considerable power reductions for Xilinx FPGAs. Improved capacitance estimation accuracy in placement will yield even higher power reductions.

Two areas that we feel are particularly fertile are glitch and leakage optimization. Glitches are spurious transitions that occur on signals caused by unequal path delays in circuits. Such transitions are unnecessary, yet they play a significant role in dynamic power. CAD techniques to mitigate glitches include equalizing path delays or inserting registers along the most "glitchy" paths [11]. Leakage in a digital CMOS circuit depends strongly on the circuit's applied input state. Therefore, one approach to leakage reduction in CAD is to automatically alter the circuit so that its signal values spend more time in low-leakage states.

The results given above, comprising 10% savings in dynamic power on average, show that significant strides have already been made in reducing power through CAD algorithms. We are optimistic that the future is bright for achieving additional power reductions in software. Power-conscious solutions comprising power-aware CAD algorithms and power-optimized devices such as Virtex-5 FPGAs form a compelling story from the power perspective. Continued advances in low-power software and hardware will open the door for Xilinx FPGAs entering new power-sensitive markets.

Acknowledgments

The authors thank Philip Costello and Manoj Chirania for their contribution to this work.

References

[1] ITRS. International technology roadmap for semiconductors. *http://public.itrs.net*, 2005.

[2] G. Yeap. *Practical Low Power Digital VLSI Design*. Kluwer Academic Publishers, Boston, MA, 1998.

[3] J. Lamoureux and S.J.E. Wilton. On the interaction between power-aware FPGA CAD algorithms. In *IEEE International Conference on Computer-Aided Design*, pages 701–708, San Jose, CA, 2003.

[4] L. Shang, A. Kaviani, and K. Bathala. Dynamic power consumption of the Virtex-2 FPGA family. In *ACM International Symposium on Field-Programmable Gate Arrays*, pages 157–164, Monterey, CA, 2002.

[5] T. Tuan and B. Lai. Leakage power analysis of a 90nm FPGA. In *IEEE Custom Integrated Circuits Conference*, pages 57–60, San Jose, CA, 2003.

[6] A. Rahman and V. Polavarapuv. Evaluation of low-leakage design techniques for field-programmable gate arrays. In *ACM International Symposium on Field Programmable Gate Arrays*, pages 23–30, Monterey, CA, 2004.

[7] J. H. Anderson. Power prediction and optimization techniques for FPGAs. In *Ph.D. Thesis*. Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ontario, Canada, 2005.

[8] T. Tuan and S. Trimberger. The power of FPGA architectures. *Xcell Journal*, Volume 60, First Quarter 2007.

[9] F. Najm. Transition density: A new measure of activity in digital circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12:310–323, February 1993.

[10] J.H. Anderson and F.N. Najm. Power estimation techniques for FPGAs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 10(10):1015–1027, October 2004.

[11] J. Lamoureux, G. Lemieux, and S. Wilton. Glitchless: An active glitch minimization technique for FPGAs. In *ACM International Symposium on Field-Programmable Gate Arrays*, pages 156–165, Monterey, CA, 2007.