Measuring the Gap Between FPGAs and ASICs

Ian Kuon, *Student Member, IEEE*, and Jonathan Rose, *Senior Member, IEEE*

***Abstract*—This paper presents experimental measurements of the differences between a 90-nm CMOS field programmable gate array (FPGA) and 90-nm CMOS standard-cell application-specific integrated circuits (ASICs) in terms of logic density, circuit speed, and power consumption for core logic. We are motivated to make these measurements to enable system designers to make better informed choices between these two media and to give insight to FPGA makers on the deficiencies to attack and, thereby, improve FPGAs. We describe the methodology by which the measurements were obtained and show that, for circuits containing only look-up table-based logic and flip-flops, the ratio of silicon area required to implement them in FPGAs and ASICs is on average 35. Modern FPGAs also contain "hard" blocks such as multiplier/accumulators and block memories. We find that these blocks reduce this average area gap significantly to as little as 18 for our benchmarks, and we estimate that extensive use of these hard blocks could potentially lower the gap to below five. The ratio of critical-path delay, from FPGA to ASIC, is roughly three to four with less influence from block memory and hard multipliers. The dynamic power consumption ratio is approximately 14 times and, with hard blocks, this gap generally becomes smaller.**

***Index Terms*—Application-specific integrated circuits (ASIC), area comparison, delay comparison, field programmable gate array (FPGA), power comparison.**

I. Introduction

COMPARED to application-specific integrated circuits (ASICs) or full-custom design, field programmable gate arrays (FPGAs) offer many advantages including reduced nonrecurring engineering and shorter time to market. However, these advantages come at the cost of an increase in silicon area, a decrease in performance, and an increase in power consumption when designs are implemented on FPGAs. The existence of these inefficiencies in FPGA-based implementations is widely known and accepted, but there have been few attempts to quantify these differences.

These differences lead to an area, performance, and power-consumption gap between ASIC or full-custom designs and FPGAs. The gaps between ASIC and full-custom designs have been studied extensively [1]–[4], but little is known about the gap between FPGAs and ASICs. In this paper, we measure the area, performance, and power gap between FPGAs and ASICs in the core while mostly ignoring input-otput (I/O) issues.

Manuscript received March 15, 2006; revised June 29, 2006. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under a Discovery Grant. The work of I. Kuon was supported by NSERC. This paper was recommended by Associate Editor A. DeHon.

The authors are with the Edward S. Rogers, Sr., Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mail: ikuon@eecg.utoronto.ca; jayar@eecg.utoronto.ca).

Digital Object Identifier 10.1109/TCAD.2006.884574

We were motivated to measure this FPGA to ASIC gap for the following reasons.

1) In the early stages of system design, when system architects choose their implementation medium, they often choose between FPGAs and ASICs. Such decisions are based on the differences in cost (which is related to area), performance, and power consumption between these implementation media, but to date, there have been few attempts to quantify these differences. A system architect can use these measurements to assess whether implementation in an FPGA is feasible. These measurements can also be useful for those building ASICs that contain programmable logic by quantifying the impact of leaving part of a design to be implemented in the programmable fabric.
2) FPGA makers seeking to improve FPGAs can gain insight by quantitative measurements of these metrics, particularly when it comes to understanding the benefit of less programmable (but more efficient) hard heterogeneous blocks such as block memory [5]–[7], multipliers/accumulators [5]–[7], and multiplexers [7] that modern FPGA often employ.

In this paper, the area, performance, and power gap between a 90-nm CMOS SRAM-programmable FPGA and a 90-nm CMOS standard-cell technology will be measured. An SRAM-based FPGA is used, because such FPGAs dominate the market and limiting the scope of the comparison was necessary to make this work tractable. Similarly, CMOS standard-cell implementation is the standard approach for ASIC designs [1], [8]. The use of newer "structured ASIC" platforms [9] is not as widespread or mature as the market continues to rapidly evolve [10]. This comparison will exclusively focus on core logic while I/O issues will not be examined. We recognize that I/O area constraints or power demands can be crucial considerations; nevertheless, the core programmable logic of an FPGA remains fundamentally important.

A fair comparison between the two very different implementation platforms is challenging. To address concerns about the accuracy of this comparison, we provide careful descriptions of the comparison process. However, as always, the specific benchmarks used can significantly impact the results and, as will be shown in our results, the magnitude of the FPGA to ASIC gap can vary significantly from circuit to circuit and application to application. Given this variability, we perform the comparison using a large set of benchmark designs from a range of application domains. However, using a large set of designs means that it is not feasible to individually optimize each design. A team of designers focusing on any single design could likely optimize the area, performance, and power consumption of a design more thoroughly, but this is true of both the ASIC

and FPGA implementations. Nevertheless, this focus on multiple designs instead of single-point comparisons as was done historically increases the usefulness of these measurements.

This paper is organized as follows: Section II describes past comparisons between FPGAs and ASICs. The comparison methodology used for this paper is detailed in Section III. This methodology is experimentally based and, in Sections IV and V, the computer-aided design (CAD) flows used to implement the benchmarks are described. The approach used for measuring area, delay, and power is defined in Section VI. In Section VII, the FPGA to ASIC comparison results are presented and analyzed. Finally, Section VIII concludes this paper. An early version of this paper appeared in [11]. In this version, the quality of the area and delay comparison is significantly improved, the benefits of heterogeneous content are demonstrated more clearly, issues with the memory cores have been resolved, and the limitations of focusing on the core logic only are clarified.

II. Historical Measurements

Since FPGA were first developed, their area, speed, and power disadvantage relative to less programmable designs has been recognized. The limited number of past attempts to quantify this gap is reviewed here.

One of the earliest statements quantifying the gap between FPGAs and prefabricated media was by Brown *et al.* [12]. Their work reported the logic-density gap between FPGAs and mask programmable gate arrays (MPGAs) to be between eight to 12 times, and the circuit-performance gap to be approximately a factor of three. The basis for these numbers was a cursory comparison of the largest available gate counts in each technology and the anecdotal reports of the approximate operating frequencies in the two technologies at the time. While the latter may have been reasonable, the former potentially suffered from optimistic gate counting in FPGAs.

In this paper, we are seeking to measure the gap against standard-cell implementations, rather than the less common MPGAs. Standard-cell implementations are reported to be in the order of 33% to 62% smaller and 9% to 13% faster than MPGA implementations [13]. Combined with the FPGA to MPGA comparison, these estimates suggest an area gap between FPGAs and standard-cell ASICs of 12 to 38. However, the reliance of these estimates on only five circuits in [13] and the use of potentially suspect gate counts in [12] makes this estimate of the area gap unreliable. Combining the MPGA:ASIC and FPGA:MPGA delay-gap estimates, the overall delay gap of FPGAs to ASICs is approximately 3.3 to 3.5 times. Ignoring the reliance on anecdotal evidence [12], the past comparison is dated because it does not consider the impact of hard dedicated circuit structures such as multipliers and block memories that are now common [5], [7]. In this paper, we address this issue by explicitly considering the incremental impact of such blocks.

More recently, a detailed comparison of FPGA and ASIC implementations was performed by Zuchowski *et al.* [14]. They found that the delay of an FPGA lookup table (LUT) was approximately 12 to 14 times the delay of an ASIC gate. Their work found that this ratio has remained relatively constant across CMOS process generations from 0.25 μm to 90 nm. ASIC-gate density was found to be approximately 45 times greater than that possible in FPGAs when measured in terms of kilo-gates per square micrometer. Finally, the dynamic power consumption of an LUT was found to be over 500 times greater than the power of an ASIC gate. Both the density and the power consumption exhibited variability across process generations, but the cause of such variability was unclear. The main issue with this paper is that it also depends on the number of gates that can be implemented by an LUT. In this paper, we remove this issue by instead focusing on the area, speed, and power consumption of application circuits.

Wilton *et al.* [15] also examined the area and delay penalty of using programmable logic. The approach taken for the analysis was to replace part of a non-programmable design with programmable logic. They examined the area and delay of the programmable implementation relative to the non-programmable circuitry it replaced. This was only performed for a single module in the design consisting of the next state logic for a chip-testing interface. They estimated that when the same logic is implemented on an FPGA fabric and directly in standard cells, the FPGA implementation is 88 times larger. They measured the delay ratio of FPGAs to ASICs to be two times. This paper improves on this by comparing more circuits and using an actual commercial FPGA for the comparison.

Compton and Hauck [16] have also measured the area differences between FPGA and standard-cell designs. They implemented multiple circuits from eight different application domains, including areas such as radar and image processing, on the Xilinx Virtex-II FPGA, in standard cells on a 0.18-μm CMOS process from TSMC, and on a custom configurable platform. Since the Xilinx Virtex-II is designed in 0.15-μm CMOS technology, the area results are scaled up to allow direct comparison with 0.18-μm CMOS. Using this approach, they found that the FPGA implementation is only 7.2 times larger on average than a standard-cell implementation. The authors believe that one of the key factors in narrowing this gap is the availability of heterogeneous blocks such as memory and multipliers in modern FPGAs and, in this paper, we quantify these claims.

While this paper aims to measure the gap between FPGAs and ASICs, it is noteworthy that the area, speed, and power penalty of FPGAs is even larger when compared to the best possible custom implementation using full-custom design. It has been observed that full-custom designs tend to be three to eight times faster than comparable standard-cell ASIC designs [1]. In terms of area, a full-custom design methodology has been found to achieve 14.5 times greater density than a standard-cell ASIC methodology [2]. Finally, the power consumption of standard-cell designs has been observed as being between three to ten times greater than full-custom designs [3], [4].

III. New Comparison Methodology

As described in the previous section, past measurements of the gaps between FPGAs and ASICs have been based on simple estimates or single-point comparisons. In this paper, the gap is measured more definitively using an empirical method

that includes the results from many benchmark designs. Each benchmark design is implemented in an FPGA and in standard cells. The silicon area, maximum operating frequency, and power consumption of the two implementations are compared to quantify the area, delay, and power gaps between FPGAs and ASICs.

Both the ASIC and FPGA-based implementations are built using 90-nm CMOS technology. For the FPGA, the Altera Stratix II [5], [17] FPGA was selected based on the availability of specific device data [18]. This device is fabricated using TSMC's Nexsys 90-nm process [19]. The IC process we use for the standard cells is STMicroelectronic's CMOS090 Design Platform [20]. Standard-cell libraries provided by STMicroelectronics are used. Since the Altera Stratix II is implemented using a multi-threshold voltage process [21], we will assume a dual-V_t process for the ASIC to ensure a fair comparison. The TSMC and STMicroelectronics processes are not identical, but we believe they are sufficiently similar to allow implementations from the two processes to be compared. The results from both platforms will assume a nominal supply voltage of 1.2 V.

The Altera Stratix II is built around a base logic unit known as an adaptive logic module (ALM). An ALM consists of a flexible LUT with two possible outputs, two flip-flops, and a small amount of other logic circuitry [5], [17]. The block can be divided into two adaptive look-up tables (ALUTs). ALMs are grouped into clusters of eight known as logic array blocks (LABs). In addition to this, the Stratix II also contains multiplier-accumulator blocks known as DSP blocks and multiple types of memory blocks [5]. Collectively, we will refer to these resources as heterogeneous or hard content to distinguish them from the basic ALM-based logic. The ALM-based logic will simply be called logic or soft logic as appropriate.

A. Benchmark Selection

The selection of benchmarks can significantly impact the results of this empirical investigation and, therefore, before considering the implementation of these benchmarks, we describe how the benchmarks were initially selected. Benchmarks were obtained from a range of sources including publicly available designs from Opencores (http://www.opencores.org/) and designs developed for projects at the University of Toronto. All the benchmarks were written in either Verilog or VHDL.

In selecting the benchmarks, two critical factors were considered. The first was ensuring that the Verilog or VHDL RTL was synthesized similarly by the different tools used for the FPGA and the ASIC implementations, since we did not have access to a single synthesis tool that could adequately target both platforms. To check this, we compared the number of registers inferred by the two synthesis processes, which we describe in Sections IV and V-A. We rejected any design in which the register counts deviated by more than 5%. Some differences in the register count are expected, because different implementations are appropriate on the different platforms. For example, FPGA designs tend to use one-hot encodings for state machines because of the low incremental cost for flip-flops.

The other issue impacting benchmark selection was ensuring that the designs can make use of the block memories and dedicated multipliers on the Stratix II. This is important because one of the aims of this paper is to analyze the improvements possible when these hard dedicated blocks are used. However, not all designs will use such features, which made it important to ensure that the set of benchmarks includes both cases when these hard structures are used and not used.

TABLE I
BENCHMARK SUMMARY

Design	ALUTs	Total 9x9 Multipliers	Memory Bits
booth	68	0	0
rs_encoder	703	0	0
cordic18	2 105	0	0
cordic8	455	0	0
des_area	595	0	0
des_perf	2 604	0	0
fir_restruct	673	0	0
mac1	1 885	0	0
aes192	1 456	0	0
fir3	84	4	0
diffeq	192	24	0
diffeq2	288	24	0
molecular	8 965	128	0
rs_decoder1	706	13	0
rs_decoder2	946	9	0
atm	16 544	0	3 204
aes	809	0	32 768
aes_inv	943	0	34 176
ethernet	2 122	0	9 216
serialproc	680	0	2 880
fir24	1 235	50	96
pipe5proc	837	8	2 304
raytracer	16 346	171	54 758

Based on these two factors, the set of benchmarks in Table I was selected for use in this paper. To provide an indication of the size of the benchmarks, the table also lists the number of Altera Stratix II ALUTs, 9 × 9 multipliers, and memory bits used by each design. The 9 × 9 multipliers are the smallest possible division of the Stratix II's DSP block. These basic blocks can be combined to form larger multipliers (four can be used to make an 18 × 18 multiplier and eight are needed to make a 36 × 36 multiplier). While all the benchmarks are relatively modest in size, we believe that the circuits are sufficiently large to give us an accurate measure of the gap between FPGAs and ASICs.

IV. FPGA CAD Flow

The benchmark designs were all implemented on Altera Stratix II parts using the Altera Quartus II v5.0SP1 software for all stages of the CAD flow. Synthesis was performed using Quartus II Integrated Synthesis with all the settings left at their default values. The default settings perform balanced optimization, which focuses on speed for timing critical portions of the design and area optimization for noncritical sections. The defaults also allow the tool to infer DSP blocks (which contain multiplier-accumulator circuits), ROMs, and RAMs automatically from the RTL.

Placement and routing with Quartus II was performed using the `Standard Fit` effort level. This effort setting forces the tool to obtain the best possible timing results regardless of

timing constraints [22]. No timing constraints were placed on the design in the reported results. We obtained similar speed results when the clocks in the designs were constrained to an unattainable 1 GHz. The final delay measurements were obtained using the Quartus Timing Analyzer. As will be described in Section VI, area is measured according to the number of logic clusters used and, therefore, we set the packer to cluster elements into as few LABs as possible without significantly impacting speed. This is done using special variables provided by Altera that mimic the effect of implementing our design on a highly utilized FPGA. In addition to this, we used the LogicLock feature of Quartus II to restrict the placement of a design to a rectangular region of LABs, DSP blocks, and memories [22]. By limiting the size of the region for each benchmark, the implementation will more closely emulate the results expected for larger designs that heavily utilize a complete FPGA. We allow Quartus II to autosize the region, because we found that automatic sizing generally delivered results with greater or equal density than when we manually defined the region sizes to be nearly square with slightly more LABs than necessary.

The selection of a specific Stratix II device is performed by the placement and routing tool. The specific Stratix II part selected can have a significant impact on the cost of an FPGA-based design, and for industrial designs, the smallest (and cheapest) part would typically be selected. However, this issue is not as important for our comparison because, as will be described later, the comparison optimistically ignores the problem of device-size granularity.

Most FPGAs including the Stratix II are available in multiple speed grades, since the parts are tested after manufacturing and sold according to their speed. We will present results comparing ASICs to both the fastest and slowest FPGA speed grades. A comparison with the fastest speed-grade parts is useful in understanding the best case differences between FPGAs and ASICs because such parts are available off-the-shelf. However, ASICs are not typically speed binned and the parts are limited by the worst case temperature, voltage, and process. Therefore, we also present results relative to the slowest FPGA speed grade to provide a fairer measurement of the inherent speed differences between the FPGA and ASIC platforms.

Finally, the operating frequency of a design varies depending on the random seed given to the placement tool. To reduce the impact of this variability on our results, the entire FPGA CAD flow is repeated five times using five different placement seeds. All the results (area, speed, and power) are taken based on the placement and routing that resulted in the fastest operating frequency.

V. ASIC CAD Flow

While the FPGA CAD flow is straightforward, the CAD flow for creating the standard-cell ASIC implementations is significantly more complicated. The CAD flow is based on standard Synopsys and Cadence tools for synthesis, placement, routing, extraction, timing analysis, and power analysis. The steps involved along with the tools used are shown in Fig. 1. The CAD tools were provided by CMC Microsystems (http://www.cmc.ca).

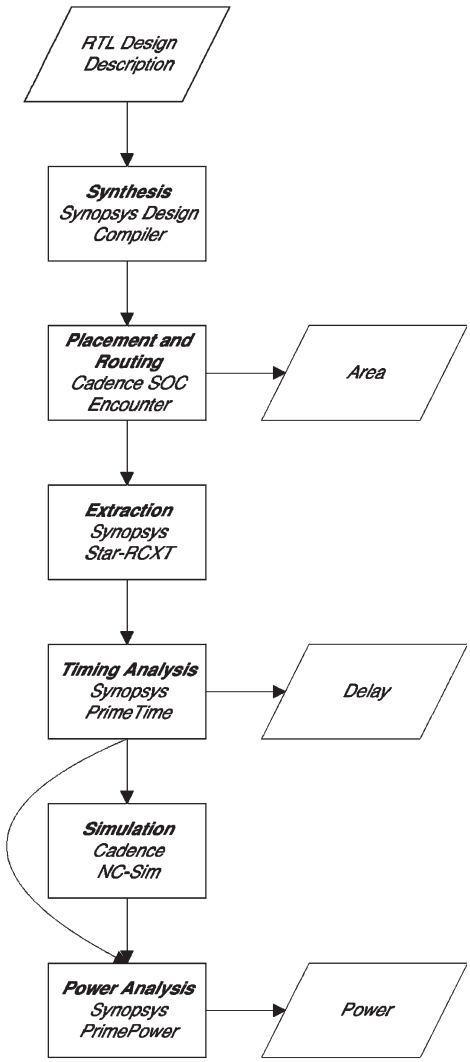

Fig. 1. ASIC CAD flow.

We used a range of sources when determining how to properly use these tools. These sources included vendor documentation, tutorials created by CMC Microsystems, and tool-demonstration sessions provided by the vendors. In the following sections, we will describe all the significant steps in this CAD flow.

A. ASIC Synthesis

Synthesis for the ASIC implementation was completed using Synopsys Design Compiler V-2004.06-SP1. All the benchmarks were synthesized using a common compile script. A top–down approach was used for this compilation. This approach compiles all the modules together starting from the highest level of hierarchy, while preserving the design hierarchy [23]. This top–down approach is reasonable in terms of CPU time and memory size, because all the benchmarks have relatively modest sizes.

The compile script starts by analyzing the hardware description language (HDL) source files for each benchmark. Elaboration and linking of the top-level module is then performed.

After linking, timing constraints are applied to the design. Initially, all the clocks in a design are constrained to 2 GHz. This constraint is unattainable, but, by over-constraining the design, we aim to create the fastest design possible. An area constraint of zero units is also placed on the design. This constraint is also unattainable, but this is a standard practice for enabling area optimization [23].

The version of the STMicroelectronics 90-nm design kit available to us contains four standard-cell libraries. Two of the libraries contain general-purpose standard cells. One version of the library uses low-leakage high-V_t transistors, while the other uses higher performing standard-V_t transistors. The other set of two libraries includes more complex gates and is available in high and standard-V_t versions. For compilation with Design Compiler, all four libraries were set as target libraries meaning that the tool is free to select cells from any of these libraries as it sees fit. The process from STMicroelectronics also has the option for low-V_t transistors; however, standard-cell libraries based on these transistors were not available to us at the time of this paper. Such cells would have offered even greater performance at the expense of static power consumption.

Once the specific target cells and clock and area constraints are specified, the design is compiled with Design Compiler. The compilation was performed using the `high-effort` setting. After the compile completed, an additional high-effort incremental compilation is performed. This incremental compilation maintains or improves the performance of the design by performing various gate-level optimizations [24].

Virtually, all modern ASIC designs require Design-for-Testability techniques to simplify post-manufacturing tests. At a minimum, scan chains are typically used to facilitate these tests [25]. This requires that all the sequential cells in the design are replaced by their scan-equivalent implementations. Accordingly, for all compilations with Design Compiler, the Test-Ready-Compile option is used, which automatically replaces sequential elements with scan-equivalent versions. Such measures were not needed for the FPGA-based implementation, because testing is performed by the manufacturer.

After the high-effort compilations are complete, the timing constraints are adjusted. The desired clock period is changed to the delay that was obtained under the unattainable constraints. With this new timing constraint, a final high-effort compilation is performed. `Sequential area recovery` optimizations are enabled for this compile, which allows Design Compiler to save area by remapping sequential elements that are not on the critical path. After this final compilation is complete, the scan-enabled flip-flops are connected to form the scan chains. The final netlist and the associated constraints are then saved for use during placement and routing.

For circuits that used memory, the appropriate memory cores were generated by STMicroelectronics using their custom memory compilers. CMC Microsystems and Circuits Multi-Projets (CMP) (http://cmp.imag.fr) coordinated the generation of these memory cores with STMicroelectronics. When selecting from the available memories, we chose compilers that delivered higher speed instead of higher density or lower power consumption. The memories were ordered to be as square as possible.

B. ASIC Placement and Routing

The synthesized netlist is next placed and routed based on the constraints with Cadence SOC Encounter GPS v4.1.5. The placement and routing CAD flow was adapted from that described in the Encounter design flow guide and tutorial [26]. The key steps in this flow are described as follows.

The modest sizes of the benchmarks allow us to implement each design as an individual block, and the run times and memory usage were reasonable despite the lack of design partitioning. For larger benchmarks, hierarchical-chip-floorplanning steps might be necessary.

Before placement, a floorplan must be created. For this floorplan, we selected a target row utilization of 85% and a target aspect ratio of 1.0. The 85% target utilization was selected to minimize any routing problems. Higher utilizations tend to make placement and routing more challenging [27]. Designs with large memory macro blocks proved to be more difficult to place and route; therefore, the target utilization was lowered to 75% for those designs.

After the floorplan is created, under these constraints, placement is performed. This placement is timing driven, and optimization is performed based on the worst case timing models. Scan-chain reordering is performed after placement to reduce the wirelength required for the scan chain. The placement is further optimized using Encounter's optDesign macro command, which performs optimizations such as buffer additions, gate resizing, and netlist restructuring. Once these optimizations are complete, the clock tree is inserted. Based on the new estimated clock delays from the actual clock tree, setup and hold time violations are then corrected. Finally, filler cells are added to the placement in preparation for routing.

Encounter's Nanoroute engine is used for routing. The router is configured to use all seven metal layers available in the STMicroelectronics process used for this paper. Once the routing completes, metal fill is added to satisfy metal-density requirements. Detailed extraction is then performed. This extraction is not of the same quality as the sign-off extraction but is sufficient for guiding the later timing-driven optimizations. The extracted parasitic information is used to drive post-routing optimizations that aim to improve the critical path of the design. These in-place optimizations include drive-strength adjustments. After these optimizations, routing is again performed and the design is checked for connectivity or design-rule violations. The design is then saved in various forms as required for the subsequent steps of the CAD flow.

C. Extraction and Timing Analysis

In our design environment, the parasitic extraction performed within SOC Encounter GPS is not sufficiently accurate for the final sign-off timing and power analysis. Therefore, after placement and routing is complete, the final sign-off quality extraction is performed using Synopsys Star-RCXT V-2004.06. This final extraction is saved for use during the timing and power analysis that is performed using Synopsys PrimeTime SI version X-2005.06 and Synopsys PrimePower version V-2004.06SP1, respectively.

VI. Comparison Metrics

After implementing each design as an ASIC and using an FPGA, the area, delay, and power of each implementation were compared. The specific measurement approach can significantly impact results; therefore, in this section, the measurement methodology for each of the metrics is described in detail.

A. Area

The area for the standard-cell implementation is defined in this paper to be the final core area of the placed and routed design. This includes the area for any memory macros that may be required for a design. The area of the inputs and outputs is intentionally excluded, because the focus in this paper is on the differences in the core logic.

Measuring the area of the FPGA implementation is less straightforward because the benchmark designs used in this paper generally do not fully utilize the logic on an FPGA. Including the entire area of an FPGA that is not fully utilized would obscure the comparison. Instead, for the area measurements, only the silicon area for any logic resources used by a design is included. The area of a design is computed as the number of LABs, M512, M4K, MRAM memories, and DSP blocks each multiplied by the silicon area of that specific block. Again, the area of I/Os is excluded to allow us to focus on the core programmable logic. The silicon areas for each block were provided by Altera [18]. These areas include the routing resources that surround each of the blocks. The entire area of a block (such as a memory or LAB) is included in the area measurement regardless of whether only a portion of the block is used. This block-level granularity is potentially pessimistic, and in Section VII-A, the impact of this choice is examined. To avoid disclosing any proprietary information, no absolute areas are reported and only the ratio of the FPGA area to ASIC area will be presented.

This approach of only considering the resources used may also be considered optimistic for a few reasons. It ignores the fact that FPGAs unlike ASICs are not available in arbitrary sizes and, instead, a designer must select one particular discrete size, even if it is larger than required for the design. This optimism is acceptable, because we are focusing on the cost of the programmable fabric itself. As well, we optimistically measure the area used for the hard heterogeneous blocks such as multipliers and memories. In commercial FPGAs, the ratio of logic to memories to multipliers is fixed. A designer must tolerate this ratio regardless of the needs of their particular design. For the area calculations in this paper, these fixed ratios are ignored and the area for a heterogeneous structure is only included as needed. This allows us to measure the best case impact of these hard blocks.

B. Delay

The critical path of each ASIC and FPGA design is obtained from static-timing analysis assuming worst case operating conditions. This determines the maximum clock frequency for each design. For the ethernet benchmark, which contains multiple clocks, the geometric average of all the clocks in each implementation is compared. For the FPGA, timing analysis was preformed using the timing analyzer integrated in Altera Quartus II. Timing analysis for the ASIC was performed using Synopsys PrimeTime SI, which accounts for signal-integrity effects such as crosstalk when computing the delay.

C. Power

Power is an important issue for both FPGA and ASIC designs, but it is challenging to fairly compare measurements between the platforms. This section describes in detail the method used to measure the power consumption of the designs. For these measurements, we separate the dynamic and static contributions to the power consumption both to simplify the analysis and because we are only able to report meaningful results for the dynamic power consumption comparison. In an attempt to ensure a fair and useful comparison, we adjusted the measurements of the static power and we describe our adjustments later in this section so as to explain the limited static power consumption results we are able to report.

It is important to note that in these measurements, we aim to compare the power-consumption gap as opposed to energy-consumption gap. To make this comparison fair, we compare the power with both the ASIC and the FPGA performing the same computation over the same time interval. An analysis of the energy-consumption gap would have to reflect the slower operating frequencies of the FPGA. The slower frequency means that more time or more parallelism would be required to perform the same amount of work as the ASIC design. To simplify the analysis in this paper, only the power-consumption gap will be considered.

Also, it is significant that we perform this comparison using FPGA and ASIC implementations designed to operate at the highest speed possible. This is done because our goal is to measure the power gap between typical ASIC and FPGA implementations as opposed to the largest possible power gap. Our results would likely be different if we performed the comparison using an ASIC designed to operate at the same frequency as the FPGA since power-saving techniques could be applied to the ASIC.

1) Dynamic and Static Power Measurement: The preferred measurement approach, particularly for dynamic power measurements, is to stimulate the post-placed and routed design with vectors representing typical usage of the design. This approach is used when appropriate testbenches are available and the results gathered using this method are labeled accordingly. However, in most cases, appropriate testbenches are not available and we are forced to rely on a less accurate approach of assuming constant toggle rates and static probabilities for all the nets in each design.

The dynamic power measurements are taken assuming worst case process, 85 °C and 1.2 V. Both the FPGA and ASIC implementations are simulated at the same operating frequency of 33 MHz. This frequency was selected since it was a valid operating frequency for all the designs on both platforms. Performing the comparison assuming the same frequency of operation for both the ASIC and FPGA ensures that both implementations perform the same amount of computation.

For the FPGA implementation, an exported version of the placed and routed design was simulated using Mentor Modelsim 6.0c when the simulation-based method was possible. That simulation was used to generate a value-change-dump (VCD) file containing the switching activities of all the circuit nodes. Based on this information, the Quartus II Power Analyzer measured the static and dynamic power consumption of the design. Glitch filtering was enabled for this computation, which ignores any transitions that do not fully propagate through the routing network. Altera recommends using this setting to ensure accurate power estimates [22]. Only core power (supplied by `VCCINT` was recorded, because we are only interested in the power consumption differences of the core programmable fabric. The power analyzer separates the dynamic and static contributions to the total power consumption.

The placed and routed netlist for the standard-cell implementation was simulated with back-annotated timing using Cadence NC-Sim 5.40. Again, a VCD file was generated to capture the state and transition information for the nets in the design. This file, along with parasitic information extracted by Star-RCXT, is used to perform power analysis with the Synopsys PrimePower tool, version V-2004.06SP1. PrimePower automatically handles glitches by scaling the dynamic power consumption when the interval between toggles is less than the rise and fall delays of the net. The tool also splits the power consumption up into static and dynamic components.

In most cases, proper testbenches were not available and, for those designs, power measurements were taken assuming all the nets in the design toggle at the same frequency and have the same static probability. This approach does not accurately reflect the true power consumption of a design, but we believe a comparison of the ASIC and FPGA measurements using this method is reasonable. It should be recognized that this approach may cause the power consumption of the clock networks to be less than typically observed. Nevertheless, this approach was selected instead of statistical-vectorless-estimation techniques that propagate toggle rates and static probabilities from source nodes to all nodes in design because the two power estimation tools produced significantly different activity estimates when using the statistical method.

2) Dynamic and Static Power-Comparison Methodology: Directly comparing the dynamic power consumption between the ASIC and the FPGA is reasonable, but the static power measurements on the FPGA require adjustments before a fair comparison is possible to account for the fact that the benchmarks do not fully utilize a specific FPGA device. Accordingly, the static power consumption reported by the Quartus Power Analyzer is scaled by the fraction of the core FPGA area used by the particular design. The fairness of this decision is arguable, since end users would be restricted to the fixed available sizes and would therefore incur the static power consumption of any unused portions of their design. However, the discrete nature of the device sizes obscures the underlying differences in the programmable logic that we aim to measure. Given the arbitrary nature of the FPGA sizes and the anticipation of power-shutdown capabilities in future FPGAs, we believe this adjustment to the static power is reasonable.

TABLE II
AREA RATIO (FPGA/ASIC)

Name	Logic Only	Logic & DSP	Logic & Memory	Logic, Memory & DSP
booth	33			
rs_encoder	32			
cordic18	19			
cordic8	25			
des_area	42			
des_perf	17			
fir_restruct	28			
mac1	43			
aes192	47			
fir3	45	17		
diffeq	41	12		
diffeq2	39	14		
molecular	47	36		
rs_decoder1	54	58		
rs_decoder2	41	37		
atm			70	
aes			24	
aes_inv			19	
ethernet			34	
serialproc			36	
fir24				9.5
pipe5proc				23
raytracer				26
Geomean	35	25	33	18

The static power adjustments are best illustrated by example. Assume a hypothetical FPGA in which one LAB and one MRAM block out of a possible ten LABs and two MRAM blocks are used. If the silicon area of the LAB and MRAM is 101 μm^2 and the area of all the LABs and MRAM is 210 μm^2, then we would scale the total static power consumption of the chip by $101/210 = 0.48$. This adjustment assumes that leakage power is approximately proportional to the total transistor width of a design, which is reasonable [28] and that the area of a design is a linear function of the total transistor width.

VII. RESULTS

All the benchmarks were implemented using the flow described in Sections IV and V. Area, delay, and power measurements were taken using the approach described in Section VI. In the following sections, the results for each of this metrics will be examined.

A. Area

The area gap between FPGAs and ASICs for the benchmark circuits is summarized in Table II. The gap is reported as the factor by which the area of the FPGA implementation is larger than the ASIC implementation. As described previously, this gap is sensitive to the benchmarks' use of heterogeneous blocks (memory and multipliers), and the results in the table are categorized in four ways. Those benchmarks that used only the basic logic fabric of clusters of LUTs and flip-flops are labeled "Logic Only." Those that used logic clusters and hard DSP blocks containing multiplier accumulators are labeled "Logic and DSP." Those that used clusters and memory blocks are labeled "Logic and Memory." Finally, those that used all three

are labeled "Logic, DSP, and Memory." We implemented the benchmarks that contained multiplication operations with and without the hard DSP blocks so results for these benchmarks appear in two columns, to enable a direct measurement of the benefit of these blocks.

First, consider those circuits that only use the basic logic LUT clusters. The area required to implement these circuits in FPGAs compared to standard-cell ASICs is on average a factor of 35 times larger, with the different designs covering a range from 17 to 54 times. This is significantly larger than the area gap suggested by [12], which used extant gate counts as its source. It is much closer to the numbers suggested by [14].

We can confirm the plausibility of this larger number based on our recent experience in designing and building complete FPGAs [29], [30]. As part of that paper, we created a design similar to the Xilinx Virtex-E, a relatively modern commercial architecture. If we consider such a design, only the LUTs and flip-flops perform the basic logic operations that would also be necessary in a standard-cell design. The FPGA, however, also requires additional circuitry to enable programmable connections between these LUTs and flip-flops. This excess circuitry is the fundamental reason for the area gap. Using our model of the Virtex-E, we calculated that the LUT and flip-flop only take up 3.4% of the total area for a Virtex-E cluster and its neighboring routing. The absolute area in the standard-cell design required to implement the functionality implemented by the LUT and flip-flop will be similar to area for the FPGA's LUT and flip-flop. This suggests the area gap should be at least $100\%/3.4\% = 29$. This is similar to our experimental measurement.

The hard heterogeneous blocks do significantly reduce this area gap. As shown in Table II, the benchmarks that make use of the hard multiplier accumulators and logic clusters are on average only 25 times larger than an ASIC. When hard memories are used, the average of 33 times larger is slightly lower than the average for regular logic, and when both multiplier accumulators and memories are used, we find the average is 18 times. Comparing the area gap between the benchmarks that make use of the hard multiplier-accumulator blocks and those same benchmarks when the hard blocks are not used best demonstrates the significant reduction in FPGA area when such hard blocks are available. In all but one case, the area gap is significantly reduced.[1] This reduced area gap was expected because these heterogeneous blocks are fundamentally similar to an ASIC implementation with the only difference being that the FPGA implementation requires a programmable interface to the outside blocks and routing.

It is noteworthy that significant variability in the area gap is observed in the benchmarks that make use of the heterogeneous blocks. One contributor to this variability is the varying amounts of heterogeneous content. The classification system used in Table II is binary in that a benchmark either makes use of a hard structure or it does not, but this fails to recognize the varying amounts of heterogeneity in the benchmarks. An alternative approach is to consider the fraction of a design's FPGA area that is used by heterogeneous blocks. The area gap is plotted versus this measure of heterogeneous content in Fig. 2. The figure demonstrates the expected trend that as designs make use of more heterogeneous blocks the area gap tends to decline.

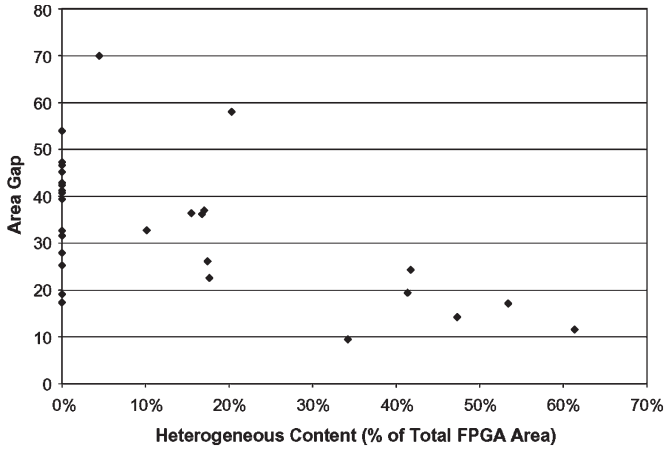

Fig. 2. Effect of hard blocks on area gap.

While these results demonstrate the importance of the introduction of these heterogeneous blocks in improving the competitiveness of FPGAs, it is important to recall that for these heterogeneous blocks, the analysis is optimistic for the FPGAs. As described earlier, we only consider the area of blocks that are used, and we ignore the effect of the fixed ratio of logic to heterogeneous blocks that a user is forced to tolerate and pay for. Therefore, the measurements will favor FPGAs for designs that do not fully utilize the available heterogeneous blocks. This is the case for many of the benchmarks used in this paper, particularly the benchmarks with memory.

However, this is also arguably unfair to the FPGAs since FPGA manufacturers likely tailor the ratios of regular logic to multiplier and memory blocks to the ratios observed in their customer's designs, and the area gap will be lower for any designs that fully utilize all the core resources. If we assume that the area gap for regular programmable logic is 35, as shown earlier, and that the area gap of heterogeneous blocks alone to an ASIC implementation is one for the large memory blocks and two for the other hard blocks, then for a fully utilized Stratix II device, the area gap would be approximately 4.7. Clearly, heterogeneous blocks can play a significant role in narrowing the area gap.

Other Considerations: The measurements of the core area gap are sensitive to many factors. One factor is the approach used to determine the area of a design on an FPGA. As described earlier, the approach used in this paper is to include the area for any resource used at the LAB, M512, M4K, M-RAM, or DSP block level. If any of these blocks is even partially used, the entire area of the block (including the surrounding routing) is included in the area measurement. This implicitly assumes the FPGA CAD tools attempt to minimize LAB usage, which is generally not the case for designs that are small relative to the device on which they are implemented. The configuration of the

[1]The area gap of the rs_decoder1 increases when the multiplier-accumulator blocks are used. This is atypical, and it appears to occur because the 5×5 bit multiplications in the benchmark are more efficiently implemented in regular logic instead of the Stratix II's 9×9 multiplier blocks.

TABLE III
AREA RATIO (FPGA/ASIC)—OPTIMISTIC FPGA AREA MEASUREMENT

Name	Logic Only	Logic & DSP	Logic & Memory	Logic, Memory & DSP
booth	32			
rs_encoder	31			
cordic18	19			
cordic8	25			
des_area	41			
des_perf	17			
fir_restruct	27			
mac1	43			
aes192	47			
fir3	28	17		
diffeq	32	11		
diffeq2	32	14		
molecular	40	36		
rs_decoder1	44	57		
rs_decoder2	36	37		
atm			69	
aes			24	
aes_inv			19	
ethernet			33	
serialproc			36	
fir24				8.5
pipe5proc				22
raytracer				26
Geomean	32	24	32	17

TABLE IV
CRITICAL-PATH-DELAY RATIO (FPGA/ASIC)—FASTEST SPEED GRADE

Name	Logic Only	Logic & DSP	Logic & Memory	Logic, Memory & DSP
booth	5.0			
rs_encoder	3.8			
cordic18	3.7			
cordic8	1.9			
des_area	2.0			
des_perf	3.1			
fir_restruct	4.0			
mac1	3.8			
aes192	4.4			
fir3	3.9	3.5		
diffeq	4.0	4.1		
diffeq2	3.9	4.0		
molecular	4.6	4.7		
rs_decoder1	2.5	2.9		
rs_decoder2	2.2	2.4		
atm			2.9	
aes			3.8	
aes_inv			4.3	
ethernet			4.3	
serialproc			2.8	
fir24				2.6
pipe5proc				2.9
raytracer				3.5
Geomean	3.4	3.5	3.5	3.0

Quartus II tools used in this paper mitigated this problem. In our past work, such settings were not used and the core-logic area gap was found to be 40 [11].

An alternative to measuring area by the number of LABs used is to instead consider the fraction of a LAB utilized based on the number of ALMs used in a LAB. The area gap results using this area metric are summarized in Table III. With this FPGA area-measurement technique, the area gap in all cases is reduced. The average area gap for circuits implemented in LUT-based logic is now only 32, and the averages for the cases when heterogeneous blocks are used have also become smaller. However, such measurements are an optimistic lower bound on the area gap because it assumes that all LABs can be fully utilized. As well, it ignores the impact such packing could have on the speed of a circuit.

These measurement alternatives for the FPGA do not apply to the ASIC-area measurements. However, the ASIC area may be impacted by issues related to the absolute size of the benchmarks used in this paper. The density of the ASIC may decrease for larger designs, because additional white space and larger buffers may be needed to maintain speed and signal integrity for the longer wires inherent to larger designs. The FPGA is already designed to handle those larger designs; therefore, it would not face the same area overhead for such designs. As well, with larger designs, hierarchical floorplanning techniques, in which the design is split into smaller blocks that are individually placed and routed, may become necessary for the ASIC. Such techniques often add area overhead because the initial area budgets for each block are typically conservative to avoid having to make adjustments to the global floorplan later in the design cycle. As well, it may be desirable to avoid global routing over placed and routed blocks to simplify design-rule checking. White space must then be added between the blocks for the global routing. That further decreases the density of the ASIC design while the FPGA would not suffer from the same effects. All these factors could potentially narrow the area gap between FPGAs and ASICs for larger designs.

As described earlier, the focus in this comparison is on the area gap between FPGAs and ASICs for the core area only. This area gap is important because it can have a significant impact on the cost diference between FPGAs and ASICs, but other factors can also be important. In particular, many designs on FPGAs require a large number of inputs/outputs and, as a result, such designs may be I/O or pad limited. This means that the die area of a device is set by the requirements for the I/O pads not by the core-logic area. In those cases, the additional core area required for the FPGAs is immaterial. Package costs are also a factor that can reduce the significance of the core area gap. For small devices, the cost of the package can be a significant fraction of the total cost for a packaged FPGA. The costs for silicon are then less important and, therefore, the large area gap between FPGAs and ASICs may not lead to a large cost difference between the two.

B. Delay

The speed gap for the benchmarks used in this paper is given in Table IV. The table reports the ratio of the FPGA's critical-path delay relative to the ASIC for each of the benchmark circuits. As was done for the area comparison, the results are categorized according to the types of heterogeneous blocks that were used on the FPGA.

Table IV shows that, for circuits with logic only, the average FPGA circuit is 3.4 times slower than the ASIC implementation. This generally confirms the earlier estimates from [12],

which were based on anecdotal evidence of *circa*-1991 maximum operating speeds of the two approaches. However, these results deviate substantially from those reported in [14], which is based on an apples-to-oranges LUT-to-gate comparison.

For circuits that make use of the hard DSP multiplier accumulator blocks, the average circuit was 3.5 times slower in the FPGA than in an ASIC; in general, the use of the hard block actually slowed down the design as can be seen by comparing the second and third column of Table IV. This result is surprising, since one would expect the faster hard multipliers to result in faster overall circuits. We examined each of the circuits that did not benefit from the hard multipliers to determine the reason this occurred. For the molecular benchmark, the delays with and without the DSP blocks were similar because there are more multipliers in the benchmark than there are DSP blocks. As a result, even when DSP blocks are used, the critical path on the FPGA is through a multiplier implemented using regular logic blocks. For the rs_decoder1 and rs_decoder2 benchmarks, only small 5×5 bit and 8×8 bit multiplications are performed and the DSP blocks which are based on 9×9 bit multipliers do not significantly speed up such small multiplications. In such cases where the speed improvement is minor, the extra routing that can be necessary to accommodate the fixed positions of the hard multiplier blocks can eliminate the speed advantage of the hard multipliers. Finally, the diffeq and diffeq2 benchmarks perform slower when the DSP blocks are used because the 32×32 bit multiplications performed in the benchmarks are not able to fully take advantage of the hard multipliers which were designed for 36×36 bit multiplication. As well, those two benchmarks contain two unpipelined stages of multiplication and it appears that implementation in the regular logic clusters is efficient in such a case. We believe that with a larger set of benchmark circuits, we would have encountered more benchmarks that could benefit from the use of the hard multipliers, particularly, if any designs were more tailored to the DSP block's functionality. However, as these results demonstrated, the major benefit of these hard DSP blocks is not the performance improvement, if any, but rather the significant improvement in area efficiency.

For the circuits that make use of the block memory, the FPGA-based designs are on average 3.5 times slower, and for the few circuits using both memory and multipliers, the FPGA is on average 3.0 times slower. Clearly, the benefit of these memory blocks is similar to the DSP blocks in that they only narrow the speed gap slightly, if at all, and their primary benefit is improved area efficiency.

To further demonstrate the limited benefit of heterogeneous blocks in narrowing the speed gap, Fig. 3 plots the speed gap against the amount of heterogeneous content in a design. As described previously, the amount of heterogeneous content is measured as the fraction of the area used in the FPGA design for the hard memory and DSP blocks. Unlike the results shown for the area gap, as the amount of hard content is increased, the delay gap does not narrow appreciably.

1) Speed Grades: As described earlier, the FPGA delay measurements assume the fastest speed-grade part is used. Comparing to the fastest speed grade is useful for understanding the best case disparity between FPGAs and ASICs, but it is not entirely fair. ASICs are generally designed for the worst case process, and it may be fairer to compare the ASIC performance to that of the slowest FPGA speed grade. Table V presents this comparison. For logic-only circuits, the ASIC performance is now 4.6 times greater than the FPGA. When the circuits make use of the DSP blocks, the gap is 4.6 times, and when memory blocks are used, the performance difference is 4.8 times. For the circuits that use both the memory and the multipliers, the average is 4.1 times. As expected, the slower speed-grade parts cause a larger performance gap between ASICs and FPGAs.

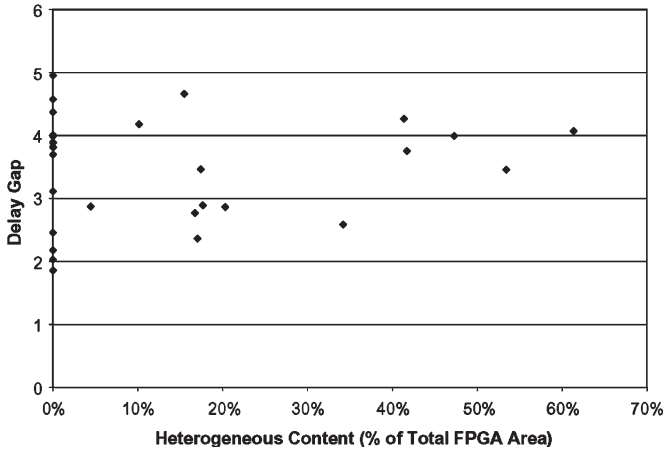

Fig. 3. Effect of hard blocks on delay gap.

TABLE V
CRITICAL-PATH-DELAY RATIO (FPGA/ASIC)—SLOWEST SPEED GRADE

Name	Logic Only	Logic & DSP	Logic & Memory	Logic, Memory & DSP
booth	6.7			
rs_encoder	5.3			
cordic18	5.1			
cordic8	2.5			
des_area	2.8			
des_perf	4.1			
fir_restruct	5.2			
mac1	5.2			
aes192	6.0			
fir3	5.3	4.6		
diffeq	5.5	5.4		
diffeq2	5.3	5.4		
molecular	6.2	6.3		
rs_decoder1	3.4	3.7		
rs_decoder2	3.0	3.0		
atm			4.0	
aes			5.1	
aes_inv			5.7	
ethernet			5.6	
serialproc			3.8	
fir24				3.8
pipe5proc				3.9
raytracer				4.7
Geomean	4.6	4.6	4.8	4.1

C. Dynamic Power Consumption

In Table VI, we list the ratio of FPGA dynamic power consumption to ASIC power consumption for the benchmark circuits. Again, we categorize the results based on which hard

TABLE VI
DYNAMIC POWER CONSUMPTION RATIO (FPGA/ASIC)

Name	Method	Logic Only	Logic & DSP	Logic & Memory	Logic, Memory & DSP
booth	Sim	26			
rs_encoder	Sim	52			
cordic18	Const	6.3			
cordic8	Const	5.7			
des_area	Const	27			
des_perf	Const	9.3			
fir_restruct	Const	9.6			
mac1	Const	19			
aes192	Sim	12			
fir3	Const	12	7.5		
diffeq	Const	15	12		
diffeq2	Const	16	12		
molecular	Const	15	16		
rs_decoder1	Const	13	16		
rs_decoder2	Const	11	11		
atm	Const			15	
aes	Sim			13	
aes_inv	Sim			12	
ethernet	Const			16	
serialproc	Const			16	
fir24	Const				5.3
pipe5proc	Const				8.2
raytracer	Const				8.3
Geomean		14	12	14	7.1

FPGA blocks were used. As described earlier, two approaches are used for power-consumption measurements and the table indicates which method was used. "Sim" means that the simulation-based method (with full simulation vectors) was used and "Const" indicates that a constant toggle rate and static probability was applied to all nets in the design. Static power results are not presented for reasons that will be described later.

The results indicate that on average FPGAs consume 14 times more dynamic power than ASICs when the circuits contain only logic. If we consider the subset of designs for which the simulation-based power measurements were used, we observe that the results are on par with the results from the constant-toggle-rate method. We are more confident in the results when this technique is used. When we compared the simulation-based results to the constant-toggle-rate measurement for each benchmark, we did not observe any bias toward over or under prediction.

When we consider designs that include hard blocks such as DSP blocks and memory blocks, we observe that the gap is 12, 14, and 7.1 times for the cases when multipliers, memories, and both memories and multipliers are used, respectively. The area savings that these hard blocks enabled suggested that some power savings should occur, because a smaller area difference implies less interconnect and fewer excess transistors, which in turn means that the capacitive load on the signals in the design will be less. With a lower load, dynamic power consumption is reduced and we observe this in general. In particular, we note that the circuits that use DSP blocks consume equal or less power when the area efficient DSP blocks are used as compared to when those same circuits are implemented without the DSP blocks. The exceptions are rs_decoder1, which suffered from an inefficient use of the DSP blocks and molecular.

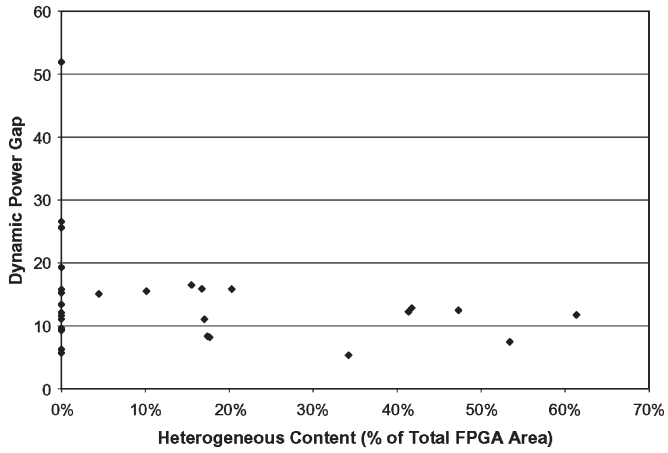

Fig. 4. Effect of hard blocks on power gap.

In Fig. 4, the power gap is plotted against the amount of heterogeneous content in a design (with heterogeneous content again measured in terms of area). The chart demonstrates that as designs use more heterogeneous resources, there is a slight reduction in the FPGA to ASIC dynamic power gap. Such a relationship was expected because of the previously shown reduction in the area gap with increased hard content.

Other Considerations: The clock network in the FPGA is designed to handle much larger circuits than were used for this comparison. As a result, for these modestly sized benchmarks, the dynamic power consumption of this large network may be disproportionately large. With larger designs, the incremental power consumption of the clock network may be relatively small and the dynamic power gap could potentially narrow, as it becomes necessary in the ASIC to construct equally large clock networks.

It is also important to recognize that core dynamic power consumption is only one contributor to a device's total dynamic power consumption. The other source of dynamic power is the input/output cells. Past studies have estimated that I/O power consumption is approximately 7%–14% of the total dynamic power consumption [31], [32], but this can be very design dependent. While the dynamic power consumption gap for the I/Os was not measured in this paper, we anticipate that it would not be as large as the core-logic dynamic power gap, because like the multipliers and memories, I/O cells are hard blocks with only limited programmability. Therefore, including the effect of I/O power consumption is likely to narrow the overall dynamic power gap.

VIII. STATIC POWER CONSUMPTION

In addition to the dynamic power, we measured the static power consumption of the designs for both the FPGA and the ASIC implementations; however, as will be described, we were unable to draw any useful conclusions. We performed these measurements for both typical silicon at 25 °C and worst case silicon at 85 °C. For leakage-power measurements, the worst case silicon is the fast-process corner. To account for the fact that the provided worst case standard-cell libraries were characterized for a higher temperature, the standard-cell results

were scaled by a factor determined from HSPICE simulations of a small sample of cells. We did not need to scale the results for typical silicon. The results we observed for these two cases deviated significantly. For logic-only designs, on average, the FPGA-based implementations consumed 87 times more static power than the equivalent ASIC when measured for typical conditions and typical silicon, but this difference was only 5.4 times under worst case conditions for worst case silicon.

The usefulness of either of these results is unclear. Designers are generally most concerned about worst case conditions, which makes the typical case measurements uninformative and potentially subject to error since more time is spent ensuring the accuracy of the worst case models. The worst case results measured in this paper suffer from error introduced by our temperature scaling. As well, static power, which is predominantly due to subthreshold leakage for current technologies [33], is very process dependent, and this makes it difficult to ensure a fair comparison given the available information. In particular, we do not know the confidence level of either worst case leakage estimate. These estimates are influenced by a variety of factors including the maturity of a process and, therefore, a comparison of leakage estimates from two different foundries, as we attempt to do here, may reflect the underlying differences between the foundries and not the differences between FPGAs and ASICs that we seek to measure. Another issue that makes comparison difficult is that, if static power is a concern for either FPGAs or ASICs, manufacturers may opt to test the power consumption and eliminate any parts that exceed a fixed limit. Both business and technical factors could impact those fixed limits. Given all these factors, to perform a comparison in which we could be confident, we would need to perform HSPICE simulations using identical process models. We did not have these same concerns about dynamic power, because process and temperature variations have significantly less impact on dynamic power.

Despite our inability to reliably measure the absolute static power consumption gap, we did find that, as expected, the static power gap and the area gap are somewhat correlated (the correlation coefficient of the area gap to the static power gap was 0.80 and 0.81 for the typical and worst case measurements, respectively). This was expected, because transistor width is generally proportional to the static power consumption [28] and the area gap partially reflects the difference in total transistor width between an FPGA and an ASIC. This relationship is important, because it demonstrates that hard blocks such as multipliers and block memories which reduced the area gap, have reduced the static power consumption gap as well.

IX. Conclusion

In this paper, we have presented empirical measurements quantifying the gap between FPGAs and ASICs for core logic. We found that for circuits implemented purely using the LUT-based logic elements, an FPGA is approximately 35 times larger and between 3.4 to 4.6 times slower on average than a standard-cell implementation. Both of these factors are significant, and together, they indicate that to achieve the same performance in an FPGA as an ASIC, there is an area gap of at least 119 and this assumes that ideal parallelization is possible. For designs that are I/O limited, this may not be a significant issue, but if the overall market for FPGAs is to grow, it will be crucial for this large area gap to be narrowed. It was also observed that an FPGA consumes 14 times more dynamic power than an equivalent ASIC on average. We confirmed that the use of hard multipliers and dedicated memories enable a substantial reduction in area and power consumption, but these blocks have a relatively minor impact on the delay differences between ASICs and FPGAs.

Acknowledgment

The authors would like to thank everyone who provided feedback on this paper at FPGA 2006 and during their visits to Actel, Altera, and Xilinx. This feedback has allowed them to significantly improve this paper. In particular, they would like to thank V. Betz and S. Trimberger. The authors would also like to thank to J. Pristupa for the extensive support he provided for both the technology kits and the numerous CAD tools required for this paper. This comparison would not have been possible without the area measurements of the Stratix II provided by R. Cliff from Altera and the technology files and memory cores provided by CMC Microsystems and CMP. P. Chow, P. Jamieson, A. Rodionov, and P. Yiannacouras provided some of the benchmarks we used in this paper.

References

[1] D. Chinnery and K. Keutzer, *Closing the Gap Between ASIC & Custom Tools and Techniques for High-Performance ASIC Design*. Norwell, MA: Kluwer, 2002.

[2] W. J. Dally and A. Chang, "The role of custom design in ASIC chips," in *Proc. 37th Des. Autom. Conf.*, 2000, pp. 643–647.

[3] A. Chang and W. J. Dally, "Explaining the gap between ASIC and custom power: A custom perspective," in *Proc. 42nd Des. Autom. Conf.*, 2005, pp. 281–284.

[4] D. G. Chinnery and K. Keutzer, "Closing the power gap between ASIC and custom: An ASIC perspective," in *Proc. 42nd Des. Autom. Conf.*, 2005, pp. 275–280.

[5] Altera Corporation. (2005, May). Stratix II Device Handbook, 3rd ed. [Online]. Available: http://www.altera.com/literature/hb/stx2/stratix2_handbook.pdf

[6] Lattice Semiconductor Corporation. (2005, May). Lattice ECP/EC Family Data Sheet, version 01.6.

[7] *Virtex-4 Family Overview*, Jun. 2005, Xilinx. [Online]. Available: http://www.xilinx.com/bvdocs/publications/ds112.pdf

[8] M. J. S. Smith, *Application-Specific Integrated Circuits*. Reading, MA: Addison-Wesley, 1997.

[9] NEC Electronics. (2005). ISSP (Structured ASIC). [Online]. Available: http://www.necel. com/issp/english/

[10] LSI Logic. (2005). Rapid Chip Platform ASIC. [Online]. Available: http://www.lsilogic.com/products/rapidchip_platform_asic/index.html

[11] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," in *Proc. ACM/SIGDA 14th Int. Symp. FPGA*, 2006, pp. 21–30.

[12] S. D. Brown, R. Francis, J. Rose, and Z. Vranesic, *Field-Programmable Gate Arrays*. Norwell, MA: Kluwer, 1992.

[13] H. S. Jones, Jr., P. R. Nagle, and H. T. Nguyen, "A comparison of standard cell and gate array implementions in a common CAD system," in *Proc. IEEE Custom Integr. Circuits Conf.*, 1986, pp. 228–232.

[14] P. S. Zuchowski, C. B. Reynolds, R. J. Grupp, S. G. Davis, B. Cremen, and B. Troxel, "A hybrid ASIC and FPGA architecture," in *Proc. ICCAD*, Nov. 2002, pp. 187–194.

[15] S. J. Wilton, N. Kafafi, J. C. H. Wu, K. A. Bozman, V. Aken'Ova, and R. Saleh, "Design considerations for soft embedded programmable logic cores," *IEEE J. Solid-State Circuits*, vol. 40, no. 2, pp. 485–497, Feb. 2005.

[16] K. Compton and S. Hauck, "Automatic design of area-efficient configurable ASIC cores," *IEEE Trans. Comput*. submitted for publication. [Online]. Available http://www.ee.washington.edu/people/faculty/hauck/publications/cASIC_Journal.pdf
[17] D. Lewis, E. Ahmed, G. Baeckler, V. Betz, M. Bourgeault, D. Cashman, D. Galloway, M. Hutton, C. Lane, A. Lee, P. Leventis, S. Marquardt, C. McClintock, K. Padalia, B. Pedersen, G. Powell, B. Ratchev, S. Reddy, J. Schleicher, K. Stevens, R. Yuan, R. Cliff, and J. Rose, "The Stratix II logic and routing architecture," in *Proc. ACM/SIGDA 13th Int. Symp. FPGA*, 2005, pp. 14–20.
[18] R. Cliff, Altera Corporation, San Jose, CA, private communication, Apr. 2005.
[19] Altera Corporation. (2004, Jun.). Partnership With TSMC Yields First Silicon Success on Altera's 90 nm, Low-k Products. [Online]. Available: http://www.altera.com/corporate/news_room/releases/releases_archive/2004/products/nr-tsmc_partnership.html
[20] STMicroelectronics. (2005). 90 nm CMOS090 Design Platform. [Online]. Available: http://www.st.com/stonline/prodpres/dedicate/soc/asic/90plat.htm
[21] Altera Corporation. (2005, Jan.). Altera Demonstrates 90 nm Leadership by Shipping World's Highest-Density, Highest-Performance FPGA. [Online]. Available: http://www.altera.com/corporate/news_room/releases/releases_archive/2005/products/nr-ep2s180_shipping.html
[22] Altera Corporation. (2005, May). Quartus II Development Software Handbook (5th ed.). [Online]. Available: http://www.altera.com/literature/hb/qts/quartusii_handbook.pdf
[23] *Design Compiler Reference Manual: Constraints and Timing*, Synopsys, Mountain View, CA, Jun. 2004. Version v-2004.06 ed.
[24] *Design Compiler User Guide*, Synopsys, Mountain View, CA, Jun. 2004. Version v-2004.06 ed.
[25] N. H. E. Weste and D. Harris, *CMOS VLSI Design A Circuits and Systems Perspective*. Reading, MA: Addison-Wesley, 2005.
[26] *Encounter Design Flow Guide and Tutorial*, Cadence Design Syst., San Jose, CA, Feb. 2004. Product Version 3.3.1.
[27] X. Yang, B.-K. Choi, and M. Sarrafzadeh, "Routability-driven white space allocation for fixed-die standard-cell placement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 4, pp. 410–419, Apr. 2003.
[28] W. Jiang, V. Tiwari, E. de la Iglesia, and A. Sinha, "Topological analysis for leakage prediction of digital circuits," in *Proc. ASP-DAC/VLSI Des.*, 2002, p. 39.
[29] I. Kuon, A. Egier, and J. Rose, "Design, layout and verification of an FPGA using automated tools," in *Proc. ACM/SIGDA 13th Int. Symp. FPGA*, 2005, pp. 215–226.
[30] K. Padalia, R. Fung, M. Bourgeault, A. Egier, and J. Rose, "Automatic transistor and physical design of FPGA tiles from an architectural specification," in *Proc. ACM/SIGDA 11th Int. Symp. FPGA*, 2003, pp. 164–172.
[31] Altera Corporation. (2005, Aug.). Stratix II vs. Virtex-4 Power Comparison & Estimation Accuracy, White Paper. [Online]. Available: http://www.altera.com/literature/wp/wp_s2v4_pwr_acc.pdf
[32] L. Shang, A. S. Kaviani, and K. Bathala, "Dynamic power consumption in Virtex-II FPGA family," in *Proc. ACM/SIGDA 10th Int. Symp. FPGA*, 2002, pp. 157–164.
[33] V. De and S. Borkar, "Technology and design challenges for low power and high performance," in *Proc. ISLPED*, 1999, pp. 163–168.

Ian Kuon (S'99) received the B.Sc. degree in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 2002 and the M.A.Sc. degree in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 2004. Since then, he has been working toward a Ph.D. degree at the University of Toronto.

He has held a variety of co-op and intern positions including eight months in 2000 with PMC Sierra in product engineering, eight months in 2001 with Research in Motion in their digital ASIC design group, four months in 2004 with the Altera Toronto Technology Centre, and four months in 2006 with Altera in their IC Design department.

Mr. Kuon is the recipient of the Natural Sciences and Engineering Research Council of Canada (NSERC) PGS A postgraduate scholarship and currently holds an NSERC CGS D scholarship.

Jonathan Rose (S'86–M'86–SM'06) received the Ph.D. degree in electrical engineering from the University of Toronto, Toronto, ON, Canada, in 1986.

He was a Postdoctoral Scholar and then a Research Associate in the Computer Systems Laboratory at Stanford University, from 1986 to 1989. In 1989, he joined the faculty of the University of Toronto. He spent the 1995–1996 year as a Senior Research Scientist at Xilinx, in San Jose, CA, working on the Virtex field-programmable gate-array (FPGA) architecture. He is the co-founder of the ACM FPGA Symposium and remains part of that Symposium on its steering committee. In October 1998, he co-founded Right Track CAD Corporation, which delivered architecture for FPGAs and packing, placement, and routing software for FPGAs to FPGA device vendors. He was President and CEO of Right Track until May 1, 2000. Right Track was purchased by Altera and became part of the Altera Toronto Technology Centre. His group at Altera Toronto shared responsibility for the development of the architecture for the Altera Stratix, Stratix II, Stratix GX, and Cyclone FPGAs. His group was also responsible for placement, routing, delay-annotation software, and benchmarking for these devices. From May 1, 2003 to April 30, 2004, he held the part-time position of Senior Research Scientist at Altera Toronto. He has worked for Bell-Northern Research and a number of FPGA companies on a consulting basis. He is currently a Professor and Chair of the Edward S. Rogers, Sr., Department of Electrical and Computer Engineering, University of Toronto.