Enhancing the Area-Efficiency of FPGAs with Hard Circuits Using Shadow Clusters

Peter Jamieson, Jonathan Rose

Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto
Toronto, Ontario, Canada M5S 3G4
jamieson@eecg.toronto.edu, jayar@eecg.toronto.edu

***Abstract*— There is a dramatic logic density gap between FPGAs and ASICs, and this gap is the main reason FPGAs are not cost-effective in high volume applications. Modern FPGAs narrow this gap by including "hard" circuits such as memories and multipliers, which are very efficient *when* they are used. However, if these hard circuits are not used, they go wasted (including the very expensive programmable routing that surrounds the logic) and have a negative impact on logic density.**

In this paper we propose a new architectural concept, called *shadow clusters*, that seeks to mitigate this loss. A shadow cluster is a standard FPGA logic "cluster" that is placed "behind" every hard circuit and can programmably, through simple, small multiplexers, replace the hard circuit in the event it isn't needed.

We measure the area-efficiency of FPGAs with and without shadow clusters and show that a modern commercial architecture (with a fixed ratio of multipliers to soft logic) would gain 4.7% in area-efficiency by employing shadow clusters. Indeed, every architecture we studied under "reasonable" conditions never showed a loss of area-efficiency. Furthermore, we show that most area-efficient architecture that employs the shadow cluster concept is 12.5% better than the most area-efficient architecture without shadow clusters.

I. Introduction

Field-Programmable Gate Arrays (FPGAs) have penetrated the digital IC market to the point where they are a compelling choice for low and medium-volume chips. They have failed to take over the market for large capacity, high-volume chips for one simple reason: their high cost due to their large silicon area. Recent research [1] has shown that simple LUT/FF FPGA and programmable routing fabric requires about 35 times the area of a cell-based ASIC to implement the same circuit. That same work showed that modern FPGAs [2], [3] can narrow this gap by employing "hard" circuits such as memories and multipliers in heterogeneous FPGAs as illustrated in Fig. 1. The figure shows logic "tiles" that are either basic soft logic or columns of either multipliers or memories.

These hard circuits reduce the area gap in a significant way, when actually used. If the circuit isn't used, however, then it is wasted making the area problem worse! Indeed, the central question of FPGA architecture is when to include more specific structures on the FPGA and how to make them as flexible as possible so that most application circuits can employ them.

In this paper, we propose a new architectural concept, called *shadow clusters*, that is a way to create that flexibility, and is illustrated in Fig. 2. A shadow cluster is a standard FPGA logic "cluster" (typically consisting of a group of LUTs and flip-flops) that exists within the same logical tile as a hard circuit such as a multiplier. When the hard circuit cannot be used by an application circuit, simple fabric-programmable multiplexers "swap" in the shadow cluster which can be used just like the regular soft logic fabric. While this may seem like a terrible waste of area itself (because the shadow cluster goes unused when the hard circuit is used), the additional area is very small: well over 70% of the area of the soft logic of an FPGA is dedicated to the routing that would be shared between these two structures.

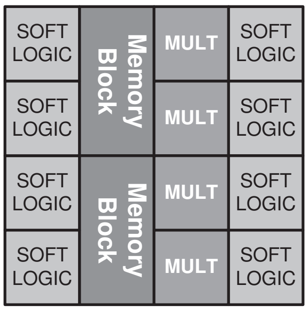

Fig. 1. Representation of Modern Heterogeneous FPGA

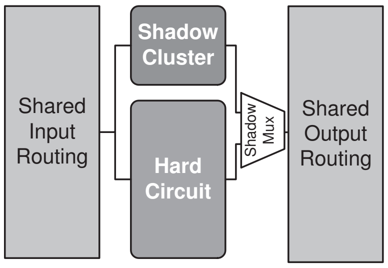

Fig. 2. Illustration of Shadow Cluster Concept

We demonstrate this concept by modeling the area of FPGAs with and without shadow clusters and implementing a suite of circuits, represented by their demand for soft and hard logic. The efficacy of this concept depends on the correct measurement of FPGA area, so we carefully size the transistors in the FPGA and the hard circuits.

Two other key concepts are needed to implement the idea correctly: architecting the logic block with shadow cluster and hard circuit so that their *demand* for programmable routing is about the same, and making the correct choice

TABLE I
FPGA ARCHITECTURAL PARAMETERS USED IN PAPER

Parameter	W	N	K	F_{cin}	F_{cout}	F_s	Mult Size
Value	180	10	4	0.18	0.1	3	18x18

of how to map application circuits into hard or soft logic. To concretize our measurements we focus on multiplier-based hard circuits, which are now common in modern FPGAs, but the concepts should apply to many kinds of circuits, including block memory.

Previous solutions to improve the usage of hard circuits include adding extra functionality to hard circuits (examples include Altera's DSP block and Xilinx's XtremeDSP [3], [2]) and novel targetting of hard circuits in CAD flow stages [4], [5], [6]. Another approach is taken in the Virtex-II where programmable routing bits are shared between hard circuits to save on programmable routing area [7].

The remainder of this paper is organized as follow: In Section II, we describe relevant basic terminology of FPGA architecture and introduce terminology that allows us to speak about heterogeneous architectures. Section III describes more detailed architectural issues relating to shadow clusters. In section IV, we describe the experimental methodology that is used to assess the area-efficiency of shadow clusters, and Section V presents the results of these experiments and analysis.

II. FPGA ARCHITECTURAL BASICS AND TERMS

In this section, we define relevant architectural terminology for FPGAs, for both classical soft logic and new terms for describing heterogeneous architectures.

The basic soft logic *tile* of an FPGA consists of a logic block surrounded by programmable routing. An array of tiles is connected together to form the soft fabric of an FPGA (which is illustrated in portions labeled "soft logic" in Figure 1). The soft logic alone is capable of implementing all logic functions. A typical soft logic tile consists of a cluster of several Basic Logic Elements (BLEs) which in turn are often some form of Lookup Table (LUT) together with a flip-flop [8].

The architecture of the soft logic fabric has many parameters, including the number of BLEs (N) per soft logic cluster tile, the input size of the LUT (K), the number of routing tracks per channel (W), the input connectivity to the soft logic cluster tile (F_{cin}), the output connectivity (F_{cout}), the switch block flexibility (F_s) [9] among several other parameters.

For the soft logic fabric we use to illustrate the concepts in this paper, we will select a single set of parameters chosen to be close to the typical parameters of a modern FPGA, including the use of direct-drive (also known as unidirectional) routing [10]. The parameters we use in this paper are given in Table I.

A. Hard Circuits in FPGAs

While the soft logic fabric described above can implement any logic function, there are some functions that will be particularly inefficient. For this reason, FPGA vendors have added more specific hard circuits to efficiently execute these functions. Smaller specific circuits are added directly into the soft logic cluster tile (such as carry chains, adders, and multiplexers [3], [2], [11]). Larger functions (such as memories and multipliers [3], [2], [11] or even floating point units [12]) are incorporated as a differentiated tile, which separately abuts the soft logic cluster tiles, as illustrated in Figure 1.

That figure illustrates that differentiated tiles are typically added to the FPGA as a unique column of tiles. In the Stratix II [3] and Virtex4 FPGAs [2] the multipliers tiles are also somewhat flexible and can implement different sizes of multipliers as well as functionality such as multiply accumulation. Throughout this paper we will use a radix 4 booth encoded multiplier architecture [13].

TABLE II
AVG. MULT. SUPPLY RATIOS FOR INDUSTRIAL FPGAS

FPGA	Supply Ratio
Stratix I	1:66
Stratix II	1:45
VirtexII	1:23
Virtex4 SX	1:15
Virtex4 FX	1:60
Virtex4 LX	1:104

B. Supply and Demand Ratio

A key architectural parameter of a heterogeneous FPGA is the ratio of the number of hard circuit tiles to soft logic cluster tiles, which we call the supply ratio, R_s. For example, an FPGA architecture with a supply ratio equal to 1:10 will have one multiplier for every ten soft logic clusters.

Table II gives the supply ratios for several commercial FPGAs. These ratios are calculated by geometrically averaging the supply ratios for each FPGA in the family. For all the FPGA families other than the Virtex4 LX family the supply ratio remains relatively constant over the family. In each case, the supply ratio is normalized to a cluster size of N=10 (we assume StratixII cluster has N=16) and a size 18x18 multiplier.

We observe that the supply ratio is increasing (more multipliers) with each new FPGA generation. Additionally, Virtex4 consists of three families in which the multiplier supply ratio is different for each family. This diversification is another approach to the problem of wasted hard circuits and tries to more closely target a range of applications and their multiplier demands.

We can also describe a given digital design in terms of its demand for hard circuits. We define the demand ratio, R_d, as the number of hard tiles to the number of soft logic cluster tiles that a digital design requires when implemented on an FPGA, if all circuits capable of being implemented in the hard tile actually are.

When discussing either ratio (R_s or R_d) we will say that a certain architecture or circuit has a greater supply or demand. This means that there are more multipliers available on an architecture or desired by a circuit. For example, a supply ratio of 1:8 is a greater supply of hard circuits compared to a supply ratio of 1:10.

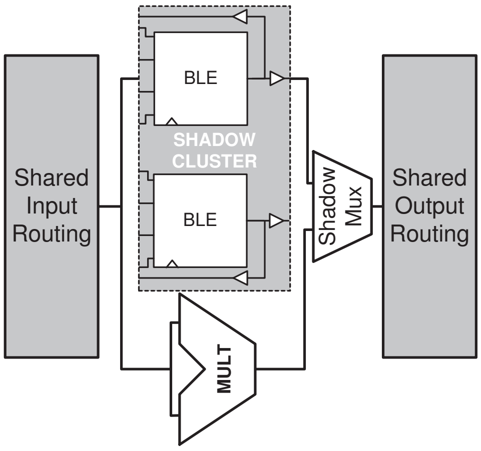

Fig. 3. Multiplier tile combined with a shadow cluster.

An FPGA which has a supply of hard circuits greater than the demand of an application circuit will waste the hard circuit and its surrounding programmable routing, while one with fewer than needed will require the design's hard circuit to be built inefficiently in the soft fabric. A given FPGA will be required to serve applications drawn from a distribution of demand ratios, and so a single value for supply ratio will never be perfect. Our new notion of shadow clusters seeks to mitigate the negative affect of this variation in demand.

III. Shadow Cluster Architecture

Our goal is to improve the area-efficiency of a heterogeneous FPGA with hard circuits by reducing the penalty in area of unused hard circuits. The proposed technique leverages the fact that the largest area cost in FPGAs is in the programmable routing. The idea is to add soft logic, which we call a *shadow cluster*, in combination *with* the hard circuit, so that either the hard circuit or the soft logic will be used and ensuring that the expensive programmable routing is used. In this section, we discuss the architectural design of both the hard circuit and the shadow cluster.

Fig. 3 illustrates the shadow cluster concept with a tile containing a shadow cluster and a multiplier. In this Figure, the programmable input routing drives both the BLEs of the shadow cluster and the multiplier, and a multiplexer selects which output to employ, under the usual programmable control (Clearly only one will be active at a time). We believe that the input and output routing for both hard circuits and soft logic cluster tiles are almost identical since both are designed to flexibly route global routing to specific pins.

There are two architectural issues to deal with here: first, the size of the multiplier tile compared to the regular soft logic cluster tile, and the size of the shadow cluster to place within the multiplier tile. For the remainder of this paper we will focus specifically on multiplier hard circuits, and select a specific size of multiplier to work with to illustrate our concept.

To address the issue of size of the multiplier and soft logic, consider that a key aspect of an FPGA's architecture is the number of routing tracks per channel – set as 180 in Table I. The number of tracks must be sufficient to supply the routing needs of circuits mapped to this FPGA.

A homogeneous soft logic fabric FPGA with a given soft logic cluster tile size (N) and LUT size (K) in each tile will require a specific number of tracks per channel to route most benchmark circuits. The parameters N and K, in conjunction with some of the routing architecture parameters, present a certain number of pins to be routed to the programmable routing, which we call the *pin demand*. This pin demand includes the input pins entering the tile and output pins emanating from the tile. The number of tracks needed in an architecture is a function of pin demand and the other routing architecture parameters given in Table I. This number is usually determined experimentally by the FPGA architect.

To accommodate a hard multiplier tile with higher pin demand than the soft logic cluster tile an architecture needs more tracks per channel, or the FPGA will be difficult to route for circuits that employ all those pins. Similarly, if the multiplier pin demand is less than that of the soft logic cluster tile, then the routing channel will be underutilized. Thus it is important to choose the multiplier tile so that it presents roughly the same pin demand per tile as the soft logic fabric.

For example, an FPGA architecture with ten 4-input LUTs (like that in Table I) typically has an input pin demand of 22 [9]. To match this soft logic cluster tile's input pin demand to that of an n by n multiplier suggests setting n to 11, since the multiplier has 2n inputs. However, since the number of outputs for the soft logic cluster tile is 10 (usually the same as N) and the multiplier would have 22 outputs, this would result in a serious mismatch. It would be more appropriate to make choices that more closely match the input and output pin demands of the multiplier and the soft logic cluster tile. A 9 by 9 multiplier (which has 18 inputs and 18 outputs for a total of 36 pins) more closely matches to the 22 input, 10 output soft logic cluster tile (which has a total of 32 pins).

In this paper, we use an 18x18 multiplier. Since this has 36 input pins and 36 output pins, and our soft logic cluster tile has 22 inputs pins and 10 output pins, the multiplier is "stretched" over two tiles to better match the pin demand.

This same argument, in reverse form, tells us that the shadow cluster should be the same size (number of BLEs) as the soft logic cluster tile. So, each of the two tiles that together contain the 18x18 multiplier will each have a 10 BLE shadow cluster.

As an aside, we believe that the extra multiplexing and area required to implement a shadow cluster will incur a minor speed penalty. As can be seen in Figure 3, there is no extra active path on the input side (and so only a minor capacitive load increase may occur), and only a 2:1 multiplexer is added on the output side, which is just one of many such multiplexers. Any power concerns for shared inputs could be dealt with simple power saving techniques like gating [14].

IV. Measurement Methodology

To determine the effectiveness of the shadow cluster concept we employ an empirical approach: we measure the area

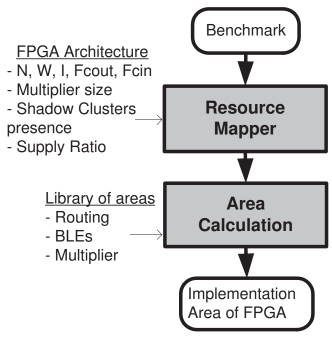

Fig. 4. Experimental measurement flow

consumed by a set of benchmarks after mapping into a heterogeneous FPGA with hard multipliers, with and without shadow clusters. The following sections describe the mapping of benchmarks into the heterogeneous FPGA and then the determination of the FPGA's area. The subsequent section describes the set of benchmarks we employ. The benchmarks are modeled as the number of soft logic clusters and hard multipliers required.

A. Circuit Mapping Flow

Figure 4 gives the experimental flow we use to measure the area for implementing benchmarks on an FPGA architecture. The first step is to map a benchmark design to the different FPGA tiles available on the FPGA, and then, calculate the area of the FPGA based on the tiles used. We discuss the calculation of the area in the next section.

A benchmark is modeled as requiring a number of soft logic cluster tiles and multiplier tiles. A mapping step is required because the FPGA's multiplier supply ratio (defined in Section II) will rarely be the same as the benchmark's multiplier demand ratio. Depending on the supply ratio, the appropriate size of FPGA is determined.

Here it is important to note that we will follow the usual practice in FPGA architecture research [15], and allow the *size* of the FPGA to be matched to the size of each benchmark, while enforcing the key FPGA architectural parameters to be maintained. This is common practice because fixed-size FPGAs otherwise create a result-obscuring quantization effect. The key parameter to maintain in a heterogeneous FPGA architecture when the device size is changing is the supply ratio of hard circuits to the soft logic. Whenever the FPGA size must be increased it is done by adding hard blocks and soft logic clusters in this ratio.

Note that the multiplier hard circuit we employ in this work is a simple 18x18 multiplier. It is not decomposable into smaller multipliers such as the DSP block in Stratix I and II [3], and so no special mapping is needed.

The mapping of the benchmark to the FPGA is determined by setting the number of multipliers equal to the number present in the benchmark, and then determining the number of soft logic clusters based on the supply ratio. An alternative is to fix the number of soft logic clusters to fit the benchmark and use that to determine the number of multipliers, but that may result in a slower circuit because there are insufficient hard multipliers for those in the benchmark.

TABLE III
PERCENTAGE AREA WITHIN A TILE AND RELATIVE AREA

Tile Type	BLEs	Mult.	Routing	Relative Size per Tile
Cluster (N=10)	13%	-	87%	1.0
Multiplier 18x18 (1 of 2 tiles)	-	55%	45%	1.9
Multiplier + Shadow 18x18 (1 of 2 tiles)	6%	52%	42%	2.0

In the case where the architecture has shadow clusters, our mapping algorithm takes this into account where each multiplier can be converted to use as a cluster if that benefits the final area.

Once the number of tiles of each type is known, and the area of each tile is known the total area for each benchmark can be calculated. The next section describes how the area of each tile type is determined.

B. Transistor and Cell Area Estimation of Tiles

We need to determine the relative area between regular multiplier tiles, multiplier tiles with shadow clusters, and regular soft logic cluster tiles. The sizes were determined in a 90nm CMOS process that was available to us [16]. Two methods of size determination are used: for all the FPGA-specific components (programmable routing in the multiplier tile and soft logic cluster tiles as well as the LUT-based logic) we carefully sized a transistor-level circuit using an automated transistor sizing approach. Space limitations prevent the detailed description of this automation.

The area of the multiplier portion of the multiplier tile is determined by using a cell-based approach with commercial standard cells in the 90nm process [16]. The multiplier was described in Verilog and synthesized using Synopsys Design Compiler V-2004.06-SP1.

The area of the multiplier tile includes the cell-based multiplier and the programmable routing area. The programmable routing is built to match the input and output pin demand of the multiplier. For example, a 9x9 multiplier has 18 input pins, so the input programmable routing area is determined for 18 inputs. The area of a multiplier tile that contains a shadow cluster will contain, in addition, the area for the logic and extra multiplexers needed for the shadowing.

Table III shows the make-up of different tiles (the soft logic tile, the pure hard multiplier tile, and the shadowed multiplier tile) on a percentage basis. The final column shows the size of each tile relative to the soft logic cluster size N=10. Note that the 18x18 multiplier is stretched over two tiles, and 1.9x represents how much one half of the multiplier is compared to the soft logic cluster tile.

Notice that, for the soft logic tile, the programmable routing takes over 80% of the area, as we discussed in the introduction! More importantly, it is important to note that the shadow

TABLE IV

BENCHMARKS DETAILS

Benchmark	BLEs	Mults	Size Range Mults
fft	2374	32	8x8
iirA	289	5	8x8 to 10x10
iirB	297	5	8x16 to 16x16
firA	84	4	8x8
firB	1598	25	16x16
firC	998	17	8x8
diffeqA	221	5	32x32
diffeqB	512	5	32x32
stereoVisionA	17765	152	8x8
stereoVisionB	34379	528	4x7 to 16x9
rayTraceA	2118	18	7x8 to 16x16
rayTraceB	21557	31	16x16 to 29x33
oc45_cpu	2191	1	16x16
reedsolDecoderA	1151	13	4x4
reedsolDecoderB	1799	9	8x8
moleculeDynamics	10542	19	38x38 to 43x50
cordicA	591	0	-
cordicB	2830	0	-
multAccumulateA	2864	0	-
multAccumulateB	9828	0	-
crc33_d264	102	0	-
desArea	1481	0	-
desPerf	4592	0	-
stereoVisionC	7281	0	-
stereoVisionD	170	0	-
rayTraceC	766	0	-
rayTraceD	1807	0	-

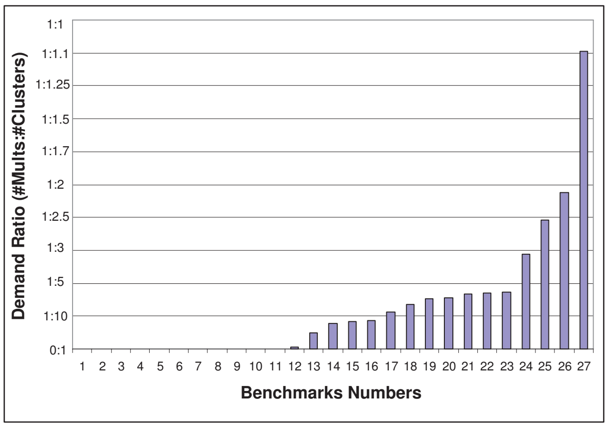

Fig. 5. Demand Ratios From Our Benchmarks.

TABLE V

DETAILS OF BENCHMARK SUITES

Name	Num. Bmarks	Avg. Demand	Variance	BLE Range	Mult. Range
B8	27	1:8	0.069	10542 to 34379	0 to 528
SB45	250	1:45	0.012	10000 to 25000	0 to 145
SB15_V0	250	1:15	0	10000 to 25000	0 to 350
SB15_V1	250	1:15	0.0009	10000 to 25000	0 to 350
SB15_V2	250	1:15	0.0030	10000 to 25000	0 to 350
SB15_V3	250	1:15	0.0078	10000 to 25000	0 to 350
SB15_V4	10	1:15	0.0076	10000 to 25000	0 to 350

cluster increases the area of the hard multiplier tile by only 1.1% (2.0/1.9) relative to the soft logic cluster tile.

C. Benchmarks

We have a set of *real* benchmarks applications written in Verilog gathered from various sources including: The Opencores organization [17], SCU-RTL [18], Texas-97 [19], and the Benchmarks for Placement 2001 [20]. We have also converted applications developed locally from VHDL to Verilog. These designs include Raytrace [21], Stereo Vision [22], and Molecular Dynamic system [23]. Table IV shows the number of BLEs, and the number and size range of multipliers in each benchmark.

As discussed above, the benchmarks after logic synthesis need only be represented as the number and size of multipliers and the number of BLEs (which can be used to calculate the number of soft logic cluster tiles) in each design. To obtain the number of BLEs, we pass our real benchmarks through Altera's Quartus tool [24]. We determine the number of multipliers in each circuit by manually counting and identifying multipliers.

Figure 5 show the demand ratios of the real benchmarks ordered from least to greatest. The demand is calculated based on a soft logic cluster size of N=10 and multiplier size of 18 by 18.

This data shows that 12 of 27 of the real benchmarks have no multipliers (R_d = 0), which we believe is realistic because only a subset of applications require multiplication. The remainder of the benchmarks have demand ratios between roughly 1:20 to 1:1 The arithmetic average of demand ratio for all the circuits is approximately R_d = 1:8, which is significantly larger than any of the supply ratios for existing industrial FPGAs as shown in Table II.

We also synthetically generate benchmark suites with different demand ratios than the original benchmarks for some of our experimental results. These benchmark suites are drawn from a distribution either based on the demand ratios shown in Figure 5, which is our most realistic model to draw from, or a distribution that we have created (to experiment with demand ratio variance). We do this to achieve better statistical results, and it also allows us to experiment with the effect of different distributions in demand ratio, to postulate the effect of different application domains.

Table V shows all the benchmark suites we will use in our experiments. Within the table we report the average demand ratio of the benchmark suite, the variance among the benchmark's demand ratios, the range of BLEs, and the range of multipliers.

The benchmark suite, B8, consists of our original benchmarks with the small benchmarks artificially inflated in size (by multiplying the number of multipliers and BLEs by a constant) to eliminate quantization noise in the experiments. The benchmark suites SB45 and SB15_V2 are synthetically created using a distribution based on our original benchmarks. The remaining benchmark suites all have an average demand ratio of 1:15 (the largest supply ratio present in the industrial FPGAs), but these benchmarks were drawn from different distribution curves to change the variance, where greater variance (a larger decimal value for variance) means that the benchmark's demand ratios are more widely distributed from the average. SB15_V0 has no variance while SB15_V1, SB15_V3, and SB15_V4 were drawn from inverse distribution functions that include benchmarks with zero demand for multipliers.

V. Results

In this section, we measure the effectiveness of adding shadow clusters to heterogeneous FPGAs with multiplier tiles, under various scenarios of FPGA architecture (as expressed by the supply ratio) and demand for multipliers (as expressed by the demand ratio statistics of a particular benchmark suite).

We begin with modern FPGAs which have supply ratios ranging from 1:15 at the high end and 1:104 at the low end as shown in Table I. We will measure the effectiveness of adding shadow clusters to these FPGAs under two demand scenarios: the first assumes that the average demand ratio is exactly the same as the supply ratio, assuming this is how industrial FPGA architects target their applications.

A. Avg. Demand Ratio Equal to Commercial Supply Ratio

In the first scenario, the average demand ratio of the benchmarks is set to be the same as the supply ratio of the FPGA. Consider the supply ratio of the Virtex 4 SX, which is 1:15. Table VI shows the results for a suite of 10 benchmarks (suite SB15_V4 from Table V) that have an average demand ratio of 1:15.

The table first gives the benchmark name, its number of soft logic cluster tiles in the benchmark, the number of 18x18 multipliers it requires, and the calculated demand ratio. The next column gives the area of the FPGA without shadow clusters required to implement the benchmark, assuming, as discussed above, that the FPGA can grow to accommodate the size of the benchmark. The next column gives the area required for an FPGA with shadow clusters and the final column gives the ratio between the "with" and "without" shadow clusters area.

Table VI illustrates when shadow clusters give a benefit - if the demand ratio is less than the supply ratio of the architecture, they allow the multiplier tiles' routing to be used and result in an area-efficiency gain: the first 6 circuits in Table VI gain 2% to 12% area-efficiency.

When the supply and demand ratios are equal or when the demand ratio is greater than the supply then the shadow clusters cause a loss of efficiency because the shadow logic goes wasted. In each case where we lose area-efficiency, the ratio is the same (0.986). The reason the ratio is the same is that each benchmark is mapped to shadow and non-shadow clustered FPGAs with enough hard multipliers for each multiplier in the benchmark. This means that both the shadow and non-shadow cluster architectures will have the same number of multipliers and soft logic clusters due to the supply ratio. In the shadow cluster architecture, each multiplier wastes space for each shadow cluster in a multiplier, and this waste is proportional to the supply ratio.

Overall, for benchmark suite (SB15_V4), the shadow cluster FPGA is 5.4% more area-efficient than the non-shadow clustered FPGA. It is worthwhile to note that the shadow cluster concepts wins, if slightly, under the scenario in which the average demand ratio *matches* the supply ratio, and therefore shows promise. The win arises because there is variance about the demand, which is unarguably true for the various designs targeting FPGAs. The amount of win is a function of the supply ratio (which determines the amount of potential waste with unused multipliers), the average demand ratio (as discussed above) and the variance of the demand. In the next section, we explore the effect of varying supply ratio and demand ratio, keeping the variance constant.

TABLE VII
Area-efficiency of different benchmarks on architectures with different supply ratios

	Area-efficiency Ratio		
R_s	R_d=1:8 (B8)	R_d=1:15 (SB15_V2)	R_d=1:45 (SB45)
1:15	1.043	1.047	1.083
1:23	1.027	1.025	1.046
1:45	1.013	1.012	1.017
1:60	1.011	1.009	1.011
1:66	1.007	1.007	1.010
1:104	1.004	1.004	1.006

B. Effect of Several Average Demand Ratios

Table VII illustrates the results of a series of experiments, each one like that given in Table VI, but with different supply ratio architectures and average demand ratios. Each number in the table is the geometric average of the area ratios (without shadow clusters/with shadow clusters) across all of the circuits in a given benchmark suite. Table V shows the characteristics of different suites, with different demand ratios. For example, Table VII entry with R_s = 1:15 tested under a average demand ratio of 1:45 has an area-efficiency ratio of 1.083, indicating that a shadow cluster architecture is about 8.3% more area-efficient than a non-shadowed architecture.

It is remarkable to note that every ratio in Table VII is greater than one, indicating that shadow clusters *never* reduce area-efficiency under all these scenarios! This suggests that industrial architects should seriously consider employing this notion.

These experiments compare FPGAs with fixed supply ratios with and without shadow clusters. Of more interest is to determine the best shadow clustered architecture (across all supply ratios) against the best non-shadowed cluster architecture (across all supply ratios), which follows below.

C. Best Shadowed and Non-Shadowed Architectures

In this section, we measure the area-efficiency of FPGAs with supply ratios that vary for both shadowed and non-shadowed architectures. We will compare a suite of architectures with different supply ratios against a non-shadowed architecture with a fixed supply ratio of 1:15. We will refer to this 1:15 non-shadowed architecture as the base_non_15 architecture. We chose this architecture as the basis for comparison

TABLE VI

RESULTS FOR INDIVIDUAL BENCHMARKS ON SHADOWED AND NON-SHADOWED FPGAS WITH SUPPLY RATIO EQUAL TO 1:15

Benchmark Name	# Soft Logic Clusters	# Multipliers (18x18)	Demand Ratio	Area no Shadow Clusters (10e5 u^2)	Area with Shadow Clusters (10e5 u^2)	Area-Efficiency Ratio
SB15_V4_1	1849	0	0	170	152	1.118
SB15_V4_2	1420	0	0	131	117	1.119
SB15_V4_3	1638	0	0	151	135	1.086
SB15_V4_4	1042	0	0	96	87	1.110
SB15_V4_5	1904	0	0	174	156	1.115
SB15_V4_6	1925	89	1:21.6	175	171	1.023
SB15_V4_7	1523	141	1:10.8	204	207	0.986
SB15_V4_8	1304	121	1:10.8	175	177	0.986
SB15_V4_9	1528	284	1:5.4	411	417	0.986
SB15_V4_10	1502	349	1:4.3	506	513	0.986
Average			1:15			1.054

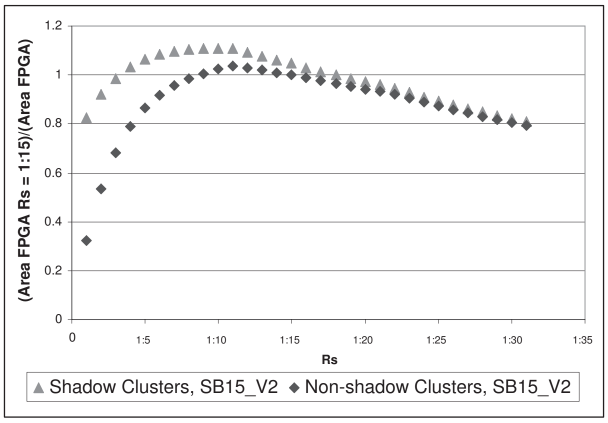

Fig. 6. Area-efficiency For Varying Supply Ratios

to illustrate the concept of best architecture, and in the next section we will look at a range of benchmark suites using this measurement methodology.

Figure 6 shows data points in two curves; the data points marked by triangles compares shadowed architectures against base_non_15. Each triangle point represents an experiment, and shows the average area required to implement the benchmark suite, SB15_V2, on the base_non_15 architecture divided by the average implementation area on a shadow architecture with a specific supply ratio (x-axis). A value greater than 1 means that the architecture specified by the x-axis has better area-efficiency than base_non_15.

Similarly, the data points marked by diamonds compares non-shadowed architectures against base_non_15. Each diamond point represents an experiment comparing the average area required to implement the same benchmark suite on base_non_15 divided by a non-shadowed architecture with a specific supply ratio.

The triangle data of Figure 6 shows that the shadow cluster architectures with supply ratios ranging from 1:4 through 1:16 are have better area-efficiency than base_non_15.

Of these architectures, the one with a supply ratio equal to 1:9 is, on average, the most area-efficient shadow cluster architecture. The diamond data points, representing non-shadow

TABLE VIII

SMALLEST IMPLEMENTATION ARCHITECTURE FOR BENCHMARK SUITES

	Non-shadow Cluster Supply Ratio	Shadow Cluster Supply Ratio	Base vs Shadow
B8	1:11	1:9	12.5%
SB15_V2	1:13	1:11	7.2%
SB45	1:28	1:19	4.6%

clustered architectures with varying supply ratios smallest implementing architecture has a supply ratio equal to 1:11.

It is interesting to note how the shadowing concept significantly changes the best supply ratio, in shadowed architectures vs. non-shadowed, and how shadowing appears to support lower ratios.

D. Effect of Demand Ratios

In this section, we study how the distribution of demand ratios in our benchmark suites affects the area-efficiency of FPGAs with shadow clusters. We measure the area-efficiency of benchmark suites with different average demand ratios mapped to shadow and non-shadow FPGAs with varying supply ratios.

Table VIII shows a summary of our results in which we record the supply ratio of shadow and non-shadow architectures which result in, on average, the smallest area implementation for each benchmark suite. Column 1 shows the benchmark suites; Column 2 and 3 show the supply ratio at which the non-shadow cluster architecture and shadow architecture achieve the smallest implementation area, and Column 4 shows what percentage smaller the smallest shadow cluster architecture is compared to the smallest non-shadow cluster architecture.

For our original benchmarks (B8), the smallest shadow cluster architecture is 12.5% smaller than the smallest non-shadow architecture. The benchmark suite SB45 implemented on the smallest shadow cluster architecture is 4.6% smaller than the smallest non-shadow cluster architecture. For each benchmark suite, the smallest shadow cluster architecture is more area-efficient than the smallest non-shadow cluster archi-

TABLE IX
SMALLEST IMPLEMENTATION ARCHITECTURE FOR DIFFERENT DEMAND VARIANCES

Variance	Non-shadow Cluster Supply Ratio	Shadow Cluster Supply Ratio	Base vs Shadow
SB15_V0	1:15	1:15	-1.4%
SB15_V1	1:10	1:10	4.5%
SB15_V2	1:13	1:11	7.2%
SB15_V3	1:11	1:9	11.4%

tecture, and in general, shadow cluster architectures improve implementation area for each benchmark suite.

E. Demand Ratio Variance within Benchmark Suites

Next, we continue our study of the effects of demand ratio distribution within our benchmark suites. Here, we measure the effectiveness of shadow clusters for benchmark suites with the same average demand ratio but different variances.

Table IX shows a summary of our results, similar to the previous table, in which we report supply ratios of the architectures that on average result in the smallest implementation of benchmark suites with average demand ratio equal to 1:15 and different variances. In the second row of the table, benchmark suite SB15_V0 has no variance, and as we continue down the rows in the table, variance increases, meaning the demand ratios of each benchmark within the benchmark suite are more widely distributed from the average. In general, as variance within the benchmark suite increases (as we go down the table) the area improvement for shadow cluster architectures compared to non-shadow cluster architecture increases.

Greater variance means there are more circuits with demand ratios farther from the mean. Therefore, there are more circuits with demand ratios lower than supply ratio, including benchmarks with zero demand for multipliers that will benefit from the presence of shadow clusters and outweigh the losses for high demand benchmarks that waste shadow clusters.

For SB15_V0 with a variance of 0 (meaning that all benchmarks have a demand ratio equal to 1:15) we can see that the shadow cluster architecture results in a larger implementation area. This 1.4% increase in area represents the area cost for adding shadow clusters to an architecture with a 1:15 supply ratio.

VI. CONCLUSIONS

In this paper, we have presented an architectural concept, called *shadow clusters*, that improves the area-efficiency of FPGAs by mitigating the area loss due to unused programmable routing surrounding hard circuit tiles. The key reason this concept benefits architectures is that the area cost for programmable routing is not wasted for unused hard circuit tiles when shadow clusters are present, and the range of designs targeting FPGAs include many that do not use all the available hard circuits.

We measured the effectiveness of shadow clusters showing that, under realistic scenarios, they always improve area-efficiency of existing industrial FPGAs. Additionally, our results show that shadow clusters improve area-efficiency to various degrees as a function of the supply ratio and its relation to the average demand ratio and variance of demand ratios within a benchmark suite.

REFERENCES

[1] I. Kuon and J. Rose, "Measuring the gap between fpgas and asics," in *FPGA'06*, Feb 2006, pp. 21–30.
[2] *Virtex-4 Family Overview*, Xilinx, March 2005.
[3] *Stratix II Device Handbook*, Altera, 2004.
[4] G. Morris, G. Constantinides, and P. Cheung, "Using dsp blocks for rom replacement: A novel synthesis flow," in *FLP'05*, Aug 2005, pp. 77–82.
[5] J. Cong and S. Xu, "Performance-driven technology mapping for heterogeneous fpgas," *IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems*, vol. 19, no. 11, pp. 1268–1280, Nov. 2000.
[6] S. J. Wilton, "Implementing logic in fpga memory arrays: Heterogeneous memory architectures," in *IEEE Custom Integrated Circuits Conference*, May 2002, pp. 142–149.
[7] *Virtex-II Pro Platform FPGAs: Functional Description*, Xilinx, Oct 2003.
[8] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in fpgas," in *Proceedings of FPGA'02*, 2002, pp. 59–66.
[9] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.
[10] G. Lemieux and D. Lewis, "Directional and single-driver wires in fpga interconnect," in *IEEE International Conference on Field-Programmable Technology*, Dec 2004, pp. 41–48.
[11] *Eclipse Family Data Sheet*, QuickLogic, 2003.
[12] M. J. Beauchamp, S. Hauck, K. Underwood, and K. Hemmert, "Embedded floating point units in fpgas," in *FPGA'06*, Feb 2006, pp. 12–20.
[13] A. D. Booth, "A signed binary multiplication technique," *Quarterly Journal of Mechanics and Applied Mathematics*, pp. 236–240.
[14] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 8, pp. 847–854, 1995.
[15] J. S. Rose, R. J. Francis, P. Chow, and D. Lewis, "The Effect of Logic Block Complexity on Area of Programmable Gate Arrays," in *IEEE Custom Integrated Conference*, May. 1989, pp. 5.3.1 – 5.3.5.
[16] STMicroelectronics, "90nm CMOS090 Design Platform," 2005, http://www.st.com/stonline/prodpres/dedicate/soc/asic/90plat.htm.
[17] "www.opencores.org."
[18] "www.engr.scu.edu/mourad/benchmark/RTL-Bench.html."
[19] "www-cad.eecs.berkeley.edu/Respep/Research/vis/texas-97/."
[20] "www.cs.nthu.edu.tw/∼ylin/."
[21] J. Fender and J. Rose, "A high-speed ray tracing engine built on a field-programmable system," in *IEEE International Conf. On Field-Programmable Technology*, 2003, pp. 188–195.
[22] A. Dharabiha, J. Rose, and W. MacLean, "Video-rate stereo depth measurement on programmable hardware," in *IEEE Computer Society Conference on Computer Vision & Pattern Recognition*, 2003, pp. 203–210.
[23] N. Azizi, I. Kuon, A. Egier, A. Darabiha, and P. Chow, "Reconfigurable molecular dynamics simulator," in *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, April 2004, pp. 197–206.
[24] Altera, *Quartus II Handbook, Volumes 1, 2, and 3*, 2004.