Improving FPGA Performance with a S44 LUT Structure

Wenyi Feng, Jonathan Greene
Microsemi Corporation SOC Products Group, San Jose
{wenyi.feng, jonathan.greene}@microsemi.com

Alan Mishchenko
Department of EECS, University of California, Berkeley
alanmi@berkeley.edu

ABSTRACT

FPGA performance depends in part on the choice of basic logic cell. Previous work dating back to 1999-2005 found that the best look-up table (LUT) sizes for area-delay product are 4-6, with 4 better for area and 6 for performance. Since that time several things have changed. A new "LUT structure" mapping technique can target cells with a larger number of inputs (cut size) without assuming that the cell implements all possible functions of those inputs. We consider in particular a 7-input function composed of two tightly-coupled 4-input LUTs. Changes in process technology have increased the relative importance of wiring delay and configuration memory area. Finally, modern benchmark applications include carry chains, math and memory blocks. Due to these changes, we show that mapping to a 7-input LUT structure can approach the performance of 6-input LUTs while retaining the area and static power advantage of 4-input LUTs.

ACM Reference format:
Wenyi Feng, Jonathan Greene, and Alan Mishchenko. 2018. Improving FPGA Performance with a S44 LUT Structure. In *FPGA'18: 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Feb. 25–27, 2018, Monterey, CA, USA. ACM, New York, NY, USA, 6 pages.* DOI: https://doi.org/10.1145/3174243.3174272

1. INTRODUCTION

Modern FPGA architectures [1-4] use clusters of look-up tables (LUTs). Previous studies [6,7] sought combinations of LUT size (the number of inputs) and cluster size (the number of LUTs in the cluster) providing the best area-delay tradeoffs. LUT sizes of 4-6 were found to offer the best area-delay product with LUT4 slightly better for area and LUT6 for performance. Since static power tends to correlate with area, LUT4 is also better for static power.

LUT4s are used widely in commercial FPGAs, including Altera's Stratix [2], Lattice's ECP series [3], Microsemi's IGLOO2 and PolarFire families [1], and Xilinx's early Virtex series [4]. Starting about 2005, LUT6-based architectures were developed for improved performance, including by Altera since StratixII [9] and by Xilinx since Virtex5 [4]. Since the relevant netlists still contain a significant fraction of smaller LUTs which would under-utilize a simple LUT6, these architectures used different techniques to enhance area efficiency. Altera developed an adaptive logic module (ALM) [9], while Xilinx employed a dual-output LUT6 [4]. More sophisticated software is required to leverage these cells (e.g., [10]), and there is some performance cost due to the additional constraints on clustering and placement. Since in this paper we are concerned with performance, we sidestep these issues and focus on a simple LUT6. (But we comment on this matter further in the Discussion section below.)

Since the advent of LUT6 architectures, several things have changed. First, process technology has scaled considerably since the 180nm node considered in [7], with current design activity at 14 and 7nm. Wire delay has come to dominate logic delay. Has this caused the 15% performance benefit of LUT6 vs LUT4 reported in [7] to grow or shrink?

Another impact of advancing technology is that configuration bit cell area has not been keeping up with scaling. From 150nm to 16nm a shrink of 88x would be expected but SRAM FPGA configuration bit cell area has shrunk by only about 36x. Other things being equal, slower scaling of the bit cell will tend to make larger LUTs more costly since the number of bits in a LUT grows exponentially with the number of inputs. This is another motivation to check if the performance benefit of LUT6 is still significant.

Second, new logic synthesis and mapping algorithms have been developed. The main advantage of larger LUTs is the reduction in the number of levels of logic in the critical path. The authors of [7] suggested that if there is "a way to achieve the depth properties of a LUT7 without paying the heavy area price, then such a seven-input function may well be a good choice." A recent algorithm for mapping to "LUT structures" provides a way to do just that [12]. Consider the "S44" structure shown in Fig. 1(a). This is a 7-input structure composed of two tightly-coupled 4-input LUTs. While it cannot implement all 7-input functions, it can implement almost all 5-input functions, 98% of 6-input functions, and 75% of 7-input functions observed in the designs studied here (re-evaluated by the methods of [12]). The addition of a two-input mux and an additional output, as shown in Fig. 1(b), allow this structure to also implement two ordinary LUT4s.

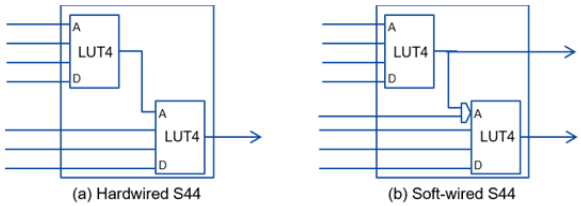

Figure 1. Hardwired and soft-wired S44s

Third, modern designs contain more than just the simple LUTs considered in [6,7]. They now include carry chains, which have a significant impact on the critical path [20], as well as embedded math and memory blocks.

These three changes motivate us to do a practical evaluation of S44 mapping, and to reexamine the performance benefits of LUT6 architectures in the context of 14nm technology, S44

FPGA'18, February 25–27, 2018, Monterey, CA, USA

ACM ISBN 978-1-4503-5614-5/18/02 ...$15.00
https://doi.org/10.1145/3174243.3174272

mapping, and industrial benchmark designs. Our contributions are as follows:

- We show how scaling has affected the relative delays of logic, intra-cluster wiring, and inter-cluster wiring, and explain why this would tend to reduce the benefit of LUT6.
- It was shown in [12] that the reduction in logic levels with the S44 structure incurred an increase in area for public benchmark designs (e.g., MCNC20 designs). We show that for more realistic industrial designs, S44 mapping provides both a delay and area benefit, and explain why.
- The prior study [12] considered only mapping. We show that the delay benefits of mapping to a soft-wired S44 are sustained through a complete clustering, placement and routing flow in a commercial architecture setting.
- We show that the post-routing performance benefit of LUT6 (or S44) over LUT4 is often much less for industrial designs that include carry chains and embedded blocks than for public designs that do not.
- We show that the combined effect of 14nm technology, S44 mapping and industrial benchmarks is to significantly narrow the performance benefit of LUT6 vs LUT4.

2. TECHNOLOGY SCALING AND ITS IMPACT ON FPGA ARCHITECTURES

2.1 Scaling of Various Delay Components

As is well-known, wire resistance is not scaling well, and as a result interconnect delay is increasing relative to logic delay [21]. Table 1 shows the ratio of various delays in similar architectures [17] optimized for 65nm and 14nm. Values are given for the average delay through a LUT4, a representative intra-cluster connection, and a representative longer connection of length 5 clusters.

Table 1: Delay Scaling from 65nm to 14nm

Delay	Ratio (65nm/14nm)
LUT4	4.1
intra-cluster routing	3.3
inter-cluster routing	2.4

It is apparent that the same architecture at a more advanced technology would exhibit critical paths with an increased contribution from inter-cluster routing and decreased contribution from logic.

How does this trend affect the relative speed of architectures using different LUT sizes? The simple explanation for the performance benefit of a larger LUT is that fewer levels of logic are required. The implicit assumption is that delay is proportional to the number of logic levels, as is commonly assumed in mapping algorithms [14]. However, this assumption may not be valid, especially in clustered architectures. The eliminated levels will more likely be intra-cluster connections (which are relatively fast) than inter-cluster connections (which are relatively slow). To gain intuition into how many intra- vs inter-cluster connections appear in critical paths of LUT4 vs LUT6 architectures, we propose the following thought experiment.

2.2 A Thought Experiment

Consider three architectures:

- ArchA: cluster of 8 LUT6s
- ArchB: cluster of 8 hard-wired S44 cells. Within each S44, one input from the first LUT4 is merged with one of the three free inputs from the second LUT4 to form a 6-input cell.
- ArchC: cluster of 6 soft-wired S44 cells

Each LUT6 in ArchA corresponds to an S44 in ArchB. Since ArchA and ArchB have the same number of logic cell inputs and outputs per cluster, they can use identical routing networks.

Given a LUT6 netlist placed in ArchA, we attempt to convert it to a functionally identical netlist placed in ArchB as follows. Consider each instance of a LUT6:

- If the LUT6 has no more than 4 used inputs, we can map it to one of the two corresponding LUT4s in ArchB trivially.
- If the LUT6 has 5 or 6 used inputs, we check whether the same function can be implemented in an S44. If so, we can use the S44 with no problem. As mentioned above, this will happen >98% of the time. We ignore the remaining cases for now since they will not change the big picture.

It is apparent that the resulting LUT4 netlist in ArchB has the same number of inter-cluster connections as in ArchA, up to 8 connections per cluster between LUT4s in the same S44, and the same number of other intra-cluster connections as in ArchA. The routing delays of the two implementations are similar, with the difference only in logic delays and the (very fast) direct connections internal to an S44.

One limitation of the conversion is pin swappability. As we map a 5- or 6-input function to a S44 structure, the S44 mapping might require some inputs to be assigned to the first LUT, some to the second LUT and some to both LUTs. So there is some potential reduction in routing flexibility. However, at worst we could solve this by adding additional muxes at the S44 inputs to guarantee the routing can stay the same during conversion, and lump the delay of these muxes into the cell delay.

Now consider converting the implementation from ArchB to ArchC. A typical LUT6 netlist has less than 50% LUTs using 5 or 6 inputs [15]. So a cluster from ArchA or ArchB can typically be reimplemented by a cluster from ArchC, packing two independent LUT4s into an S44 when necessary.

This conversion may occasionally fail (for example, when all S44 instances in an ArchB cluster use 5 or 6 inputs). However, it gives us some intuition why most additional logic levels in a LUT4 netlist can be routed using short connections. As technology scales, the longer routing delays will increasingly dominate over the short connection and logic delays, and the performance disadvantage of LUT4 will tend to shrink.

2.3 Prior Results Using VPR/VTR

While intuition is nice, it is desirable to confirm it by actual experiments using architectures tuned for two different process nodes, benchmarks, and a CAD flow such as VTR. Fortunately, such data is available. We compiled the detailed LUT4 vs. LUT6 performance results from the original 180nm study ([7], Figure 14; detail in [8], appendix E), and a recent 65nm study ([18], Figure 6.6(b); numerical values provided by private communication). Table 2 compares them. The 180nm study provides results for cluster size 1-10; while the 65nm study provides data for cluster size 4-15. Both use a similar

methodology and MCNC benchmarks. We compute the ratio of critical path delays for LUT4 vs LUT6 for each cluster size, and summarize three ways: "AvgAll" is the average of all available cluster sizes (1-10 for 180nm, 4-15 for 65nm); "AvgCommon" is the average of cluster size 4-10 (common to both); and "Best Delay" is for the best achieved LUT4 or LUT6 performance of any cluster size. All three averages show the difference shrinking from 180nm to 65nm, corroborating our hypothesis. Following this trend, further reductions of the difference are expected at smaller nodes.

Table 2: Critical Path Delays

	Ahmed, 2001 [8]			Zgheib, 2017 [18]		
Node	180nm			65nm		
Cluster Size	LUT4 (ns)	LUT6 (ns)	Ratio	LUT4 (ns)	LUT6 (ns)	Ratio
1	25.9	21.0	124%			
2	22.3	18.2	123%			
3	19.6	17.0	116%			
4	19.4	17.3	112%	4.9	4.3	114%
5	19.8	16.5	120%	4.9	4.1	121%
6	19.2	15.6	123%	4.8	4.2	115%
7	18.5	15.6	119%	4.6	4.1	112%
8	17.9	15.9	113%	4.5	4.1	109%
9	18.3	15.8	116%	4.5	4.1	110%
10	17.5	15.3	114%	4.9	4.4	110%
11				4.5	4.3	106%
12				4.7	4.4	107%
13				4.6	4.2	110%
14				4.6	4.3	107%
15				5.0	4.4	114%
AvgAll			117.8%			111.2%
AvgCommon			116.6%			113.0%
Best Delay	17.5	15.3	114.4%	4.5	4.1	111.2%

3. REVIEW OF LUT STRUCTURE MAPPING

3.1 LUT Structures

Approximating a large LUT with a combination of smaller LUTs and fast internal connections is a natural idea that has existed for some time. An XC4000 CLB has 2 LUT4s plus a LUT3 with two inputs driven directly by the two LUT4s [4]. A hard-wired S44 was studied in [8], but area-delay results were found to be consistently worse than for simple LUT4s. In its CAD flow, LUT structures were formed during the packing stage. The author stated that direct mapping into LUT structures held the promise of better results, but such a capability was not then available.

Various other LUT structures are proposed in [12] and [13].

3.2 Mapping into LUT Structures

ABC is a system for synthesis and verification [16]. It initially supported mapping into simple LUTs [14,15]. More recently, ABC has been extended to support direct mapping into LUT structures by these two modifications [12]:

- A checker determines whether a cut can be implemented using the structure. If the cut is no larger than the base LUT size, the check may be skipped.
- A "library file" is required to specify an area and delay cost for each number of used inputs up to the total number of inputs of the targeted structure. Mapping into a simple LUT has the same area and delay cost for any number of inputs up to the LUT size.

S44 mapping has two flavors depending on the library used, one for area optimization and one for delay optimization. Since our goal is performance we use the latter, which is reflected in Table 3. For 1-4 inputs, area and delay costs are set to 1. For 5-7 inputs, the area cost is set to 2 (since both constituent LUTs in the S44 are used), and delay cost is set to 1.2. The incremental delay cost of 0.2 approximates the delay of the additional LUT plus the direct connection between LUTs internal to the S44 relative to the delay of a LUT plus a normal routing connection. Further details can be found in [12].

Table 3: Mapping Costs for LUT4, S44, and LUT6

	LUT4		S44		LUT6	
Inputs	Area	Delay	Area	Delay	Area	Delay
1-4	1	1	1	1	1	1
5-6	N/A	N/A	2	1.2	1	1
7	N/A	N/A	2	1.2	N/A	N/A

3.3 Prior Results for LUT Structure Mapping

Mapping into an S44 structure reduces the logic depth by 28% at the expense of 5% area for a set of public benchmarks compared to simple LUT4 mapping [12].

Two factors affect the area of an S44 mapping. The ability to examine cuts of size up to 7 allows greater scope for optimization than cuts of size up to 4 for simple LUT4 mapping; this is good for area. On the other hand, achieving optimal delay in S44 mapping may require that some logic be duplicated. For example, suppose a node in the And-Inverter-Graph used by the mapper has two fanouts and both are critical. S44 mapping might have to cover the node twice in two S44 structures for optimal delay; this is bad for area.

S44 mapping requires about 3 times the runtime of simple LUT4 mapping, but is still quite practical even for industrial-sized designs.

4. EXPERIMENTAL METHODS

4.1 Architectures

The experimental architecture is roughly the same as that of [17] but with technology scaled to 14nm. A cluster has 12 LUT4s and 12 flip-flops. The inter-cluster routing consists of various length segments. The input interconnect block has three levels, providing excellent routing flexibility. A direct connection is available from each LUT's output (Y) to the fast input (A) of the next LUT in the cluster. Thus any adjacent pairs of LUTs (up to 6 pairs per cluster) can implement a soft-wired S44, and remaining LUTs can implement independent LUT4s. The architecture also supports carry chains and embedded blocks. The carry cell is a LUT4 with an additional carry input CI, carry output CO and sum output S (Figure 3 in [5]).

For comparison, an architecture with clusters of 8 LUT6s and 8 flip-flops is also created. The inter-cluster routing and input interconnect block remain unchanged. The quantity and fan-in of the output muxes are also unchanged, but to continue to use them fully the fanout of the LUTs and flip-flops is increased in a balanced way. Such an architecture is reasonable due to the

similar logic capacity of the two clusters (12xLUT4 vs 8xLUT6). The floor plan of the cluster and resulting area and delay models are updated to reflect the changes.

The cluster layouts assume non-volatile configuration memory. But since performance depends mainly on the routing architecture and logic cell rather than configuration bits, we would expect to see similar results for equivalent SRAM architectures as well.

Due to CAD limitations, we use the same 4-input carry cell even in the LUT6 architecture. (This has negligible impact on our results; see below.)

4.2 CAD Flow

The CAD flow used in our experiments takes as input a netlist produced by a commercial synthesis tool that infers carry, math and memory blocks. The flow consists of the following: re-synthesis and mapping (using ABC), packing, placement and routing. The latter three steps are done using a modified version of the Libero® SoC Design Suite [1]. Because ABC does a re-synthesis from an And-Inverter-Graph, any possible bias in the incoming netlist should be neutralized.

ABC is enhanced to handle "boxes" representing carry chains or embedded blocks. Carry chains are treated as white boxes, which are kept intact during optimization but whose function and delay are considered by ABC. (See [19] for a description of white boxes.) Delay costs of the carry cell are normalized relative to the average LUT delay as follows: 1.5 from LUT inputs to CO, 0.1 from CI to CO, and 0.2 from CI to S. Embedded blocks are registered at their inputs and outputs. Critical paths may start at a block output, or end at a block input, but do not go through any block.

For the LUT4 (baseline) case, mapping is done with command (dch; if)^4 using the LUT4 library [12]. The placer and router are aware of the Y-to-A direct connects inside the clusters and attempt to use them effectively.

For the S44 case, mapping is done with commands (dch; if -S 44)^4 using the S44 library. The mapped netlist represents a mixture of S44 and ordinary LUT4 instances. During packing, each individual LUT4 has weight 1 and each true S44 cell (consisting of 2 LUT4s) has weight 2. The placer ensures the two LUTs comprising an S44 cell are adjacent so the direct Y-to-A connection can be used during routing.

For the LUT6 case, mapping is done with command (dch; if)^4 using the LUT6 library. Packing, placement, and routing work with clusters of size 8 instead of 12.

To reduce the impact of random fluctuations in the CAD flow, we run placement five times per design with different random seeds and report average values.

4.3 Benchmark Designs

We use two suites of designs in our experiments. The "public" suite consists of the MCNC20 set excluding a few designs (clma, eliptic, and s298) with fewer than 120 LUTs. These designs lack carry and embedded blocks, but are useful for comparison with prior work. The "industrial" suite consists of proprietary designs including serial protocols, error correction, MACs, soft processors and complete customer applications. They include carry and embedded blocks. The suite includes designs using up to 54% of the LUT4s for muxes.

5. EXPERIMENTAL RESULTS

Results for the public suite are shown in Table 4, and for the industrial suite in

Table 5. "S44 area" is determined as per the S44 mapping area cost in Table 3, and may be compared to the number of cells in the LUT4 mapping. "S44 cells" counts any S44 as one, and may be compared to the number of cells in the LUT6 mapping. The number of "Carry Cells" is not affected by the type of mapping. "Logic levels" are determined as per the appropriate delay cost in Table 3, and for the carry cell as described above.

Table 4: Results for Public Suite

	Non-carry Cells				Carry	Logic Levels			Crit Path Delay	
Design	LUT4	S44 area	S44 cells	LUT6	Cells	LUT4	S44 / LUT4	LUT6 / LUT4	S44 / LUT4	LUT6 / LUT4
alu4	432	471	308	311	0	8.0	0.73	0.63	0.93	0.88
apex2	419	448	300	289	0	8.0	0.85	0.75	0.92	0.90
apex4	852	931	623	617	0	6.0	0.77	0.67	0.92	0.90
bigkey	1263	1305	974	788	0	4.0	0.85	0.75	0.97	0.94
des	1538	1577	1212	1172	0	6.0	0.77	0.83	0.91	0.96
diffeq	658	740	507	499	0	9.0	0.78	0.67	0.91	0.92
dsip	1121	1284	1046	891	0	4.0	0.80	0.75	1.01	1.02
ex1010	1005	1052	716	731	0	7.0	0.80	0.71	0.97	0.92
ex5p	182	194	167	145	0	3.0	0.80	1.00	0.96	0.92
frisc	2277	2463	1805	1806	0	15.0	0.72	0.73	0.91	0.97
misex3	320	332	223	235	0	5.0	0.92	0.80	0.94	0.93
pdc	318	355	248	247	0	5.0	0.72	0.80	0.92	0.94
s38417	2648	2804	1956	1866	0	8.0	0.73	0.75	0.92	0.97
s38584_1	3367	3249	2355	2302	0	8.0	0.73	0.75	0.94	0.95
seq	796	855	613	646	0	6.0	0.77	0.67	0.91	0.89
spla	328	323	246	247	0	5.0	0.88	0.80	0.95	0.93
tseng	653	659	550	549	0	10.0	0.84	0.80	0.89	0.90
Total	18177	19042	13849	13341	0					
Ratio	1.00	1.05		0.73	0.00					
Ratio	1.36		1.04	1.00	0.00					
Geomean							0.79	0.75	0.93	0.93

Table 5: Results for Industrial Suite

	Non-carry Cells				Carry Cells	Logic Levels			Crit Path Delay	
Design	LUT4	S44 area	S44 cells	LUT6		LUT4	S44 / LUT4	LUT6 / LUT4	S44 / LUT4	LUT6 / LUT4
D1	12915	12395	8493	8913	108	9.1	0.88	0.86	0.96	0.97
D2	19386	18522	12572	13014	117	10.2	0.78	0.80	0.95	0.94
D3	4581	4396	3342	3141	340	7.0	0.90	0.91	0.99	1.07
D4	10327	9843	7536	7396	460	7.3	0.92	0.86	1.02	1.02
D5	9461	9090	7147	6992	400	8.0	0.74	0.71	0.93	1.02
D6	4117	3886	2711	2728	390	13.1	1.00	1.00	0.96	1.01
D7	97770	93937	74782	73636	11233	13.0	0.80	0.74	0.97	1.02
D8	195785	187973	149695	147320	22484	14.0	0.76	0.76	0.94	1.02
D9	389862	373636	297763	293324	44967	14.0	0.81	0.79	0.97	0.94
D10	2824	2748	2157	2014	122	5.8	0.83	0.86	0.99	0.93
D11	4439	4287	3417	3180	143	5.9	0.83	0.83	0.99	0.94
D12	9820	9297	6843	6475	134	6.7	0.78	0.75	0.96	0.90
D13	2065	2026	1473	1470	61	13.0	0.72	0.77	0.91	0.96
D14	4723	4639	3441	3343	16	8.0	0.75	0.75	0.90	0.90
D15	4300	3570	2476	2468	0	5.0	0.92	0.80	0.99	0.93
D16	171937	165451	119162	107562	4822	10.0	0.89	0.84	1.08	1.07
D17	5555	5387	3567	3852	26	6.1	1.00	1.00	0.98	0.93
D18	11153	10490	7956	6569	52	5.1	1.00	1.00	1.03	1.00
D19	5177	5037	3848	3791	635	8.1	0.85	0.81	0.92	0.93
D20	6005	5683	4386	4149	574	11.4	0.93	0.91	0.98	1.01
Total	972202	932293	722767	701337	87084					
Ratio	1.00	0.96		0.72	0.09					
Ratio	1.39		1.03	1.00	0.12					
Geomean							0.85	0.83	0.97	0.97

"Crit Path Delay" is determined using post-route timing models (based on transistor-level circuits) for the applicable architecture.

For the public suite, comparing S44 vs LUT4, we see results similar to [12] with a reduction in logic levels (0.79) but some increase in area (1.05). Comparing LUT6 vs LUT4, we see a somewhat better reduction in logic levels (0.75).

Results for the industrial suite show two important differences. First, the area is lower for S44 than LUT4 (0.96) rather than higher. This appears to be due to less logic at critical or near critical paths that might trigger duplication. Indeed, the proportion of such logic (with a slack of 0 or 1 logic level) is found to be <10% for the industrial suite while >40% for the public designs. This makes mapping to S44 a win for area as well as logic levels. Second, the logic level reduction is smaller for both S44 and LUT6 mapping. This is due to the expected ([20]) significant contribution to critical paths from carry logic.

Recall that CAD limitations precluded the possibility of merging additional logic into the carry cell in our LUT6 mappings. To bound the impact of this potential issue, we separately mapped the industrial suite using another synthesis tool that did handle LUT6 carry cells, and checked for any occasions where a critical path contained at least one carry and had fewer logic levels than in our regular flow. This occurred only once and could have only minimal impact on the overall results.

For the public suite, LUT6 reduces critical path delay by a factor of 0.93 compared to LUT4, or 7%. This is smaller than the 11% reported for 65nm in [18], but is reasonable in light of the further scaling to 14nm here.

For the industrial suite, LUT6 reduces critical path delay by a factor of 0.97 compared to LUT4, or only 3%, less than the 7% seen for the public suite. Some of this is again due to the introduction of carry logic. Carry accounts for about 40% of the combinational logic delay in the critical paths of the industrial suite (in line with the findings of [20]). Another reason is that the industrial suite contains embedded blocks. When a critical path starts or ends at flip-flops, the flip-flops can be placed in the same cluster as the connected LUTs. This is not the case for embedded blocks, which have their own special routing clusters. This forces the relevant connections to be inter-cluster, incurring bigger delays that cannot be reduced by S44 or LUT6 mapping.

Comparing S44 and LUT6 mappings, we find that S44 ranges from 7% slower (D15) to 9% faster (D8) than LUT6 based on the design. To better understand why S44 can approximate the speed of LUT6, we show a breakdown of the critical path delays for the public suite in Table 6.

Table 6: Delay Breakdown for Public Suite

	LUT4	S44-way1	S44-way2	LUT6
Total Delay (ns)	144.8	134.7	134.7	134.9
Logic Delay (ns)	34.0	31.7	36.6	31.9
Intra-cluster delay (ns)	18.3	12.5	7.6	11.3
Inter-cluster delay (ns)	92.5	90.6	90.6	91.7
Total connections	638	616	461	493
Intra-cluster connections	301	305	150	182
Inter-cluster connections	337	311	311	311

"Total Delay" is the sum of the critical path delay over all runs of all designs. S44 is accounted for in two ways: "way1" is to treat the S44 netlist as a LUT4 netlist, considering the soft-wired net delay within the S44 as part of intra-cluster routing delay; "way2" is to treat S44 as single cell and the soft-wired net delay as part of the cell delay. The number of true S44 used is 155 (=305-150, the difference of intra-cluster connections between way1 and way2). Comparing S44-way1 with LUT4, we see the benefits of S44 mapping: reduced logic delay (due to fewer logic levels and more use of the fastest A-to-Y LUT delay), reduced intra-cluster routing delay (due to the extensive use of fast Y-to-A connections internal to the S44), and reduced inter-cluster routing delay (due to the reduction in inter-cluster connections, offset by higher average inter-cluster connection delays).

Alternatively, we can compare S44-way2 with LUT6. Total delay for LUT6 is similar. As suggested by our thought experiment, we see that the number and total delay of inter-cluster connections are very similar between S44 and LUT6. The only significant disadvantage for S44 is in logic delay, the relative importance of which is expected to decrease with further scaling.

One other explanation for the improved performance of S44 is its ability to implement a 4-input mux. The fast connection from Y to A within the S44 reduces the delay by about 10% for typical bus muxes compared with conventional LUT4 mappings.

6. DISCUSSION

As mentioned above, some commercial LUT6 architectures employ a dual-output LUT6 to improve area efficiency. Algorithms to pack two smaller LUTs into a dual-output LUT6 are discussed in [10]. These can achieve a 9.5% area saving at the expense of 1.6% performance loss, or 15.6% area saving at 12% performance loss in a more aggressive version. We have two observations on these results.

First, these more complex architectures can save area but are unlikely to improve performance compared to a simple LUT6. So the performance comparisons reported above should still be valid.

Second, does the dual-output technique eliminate the area cost of LUT6 vs LUT4? Using values from

Table 5, see that on average it takes (701337/8)/(932293/12) = 1.13 times as many clusters of 8xLUT6 versus 12xLUT4 to accommodate a given design. Furthermore, from trial layouts at 14nm we estimate the LUT6 cluster would occupy at least 10% more silicon area. The combined 23% area cost for LUT6 is clearly not outweighed by the 10-15% area savings from dual-output LUTs, which anyway would cost performance.

An obvious question is whether the LUT structure idea can be applied to a LUT6 architecture as well, using an S66 cell. We believe some improvement is possible but it will be much less than the improvement seen with S44 over LUT4. The simple reason is that there is a large logic level reduction from LUT4 mapping to LUT7 mapping, and S44 can capture most of the reduction. The reduction will be much less from LUT6 mapping to LUT11 mapping, making S66 much less interesting.

7. CONCLUSIONS

We conclude that:

- Contrary to earlier results with public benchmarks, we find that with industrial benchmarks S44 mapping saves area as well as logic levels. This is due to the fact that the industrial benchmarks have fewer near critical paths and require less logic duplication for optimal delay mapping.
- S44 mapping can effectively optimize use of fast direct connections between LUTs, and its benefits are sustained after placement and routing.
- The combined effect of technology scaling, S44 mapping, and use of industrial benchmarks allows 4-input LUTs to approach the performance of 6-input LUTs while retaining their area and static power advantage.

8. ACKNOWLEDGEMENTS

The authors would like to thank Sinan Kaptanoglu, Joel Landry, and Fei Li for their support and extensive discussions throughout this work.

9. REFERENCES

[1] Microsemi SoC products group (formerly Actel). http://www.microsemi.com/products/fpga-soc.

[2] Intel FPGA and SoC (formerly Altera). http://www.altera.com.

[3] Lattice Semiconductor Corp. http://www.latticesemi.com.

[4] Xilinx Corp. http://www.xilinx.com.

[5] PolarFire FPGA Fabric User Guide, downloadable from https://www.microsemi.com/products/fpga-soc/fpga/polarfire-fpga#documentation

[6] V. Betz, J. Rose and A. Marquardt. Architecture and CAD for Deep-submicron FPGAs. Kluwer Academic Publishers, February, 1999.

[7] E. Ahmed and J. Rose, The effect of LUT and cluster size on deep-submicron FPGA performance and density, *IEEE Trans. on VLSI*, vol. 12, pp. 288-298, 2004.

[8] E. Ahmed, The effect of logic granurality on deep-submicron FPGA performance and density, Master Thesis, Univ. of Toronto, 2001.

[9] D. Lewis, et al., The Stratix II logic and routing architecture, *FPGA* 2005, pp. 14-20.

[10] T. Ahmed, P. Kundarewich, J. Anderson, Packing techniques for Virtex-5 FPGAs, *ACM TRETS*, vol.2, No. 3, Article 18, 2009.

[11] J. Luu, et al., VTR 7.0: Next Generation Architecture and CAD System for FPGAs, *ACM TRETS*, vol. 7, No. 2, Article 6, 2014.

[12] S. Ray, et al., Mapping into LUT structures, *DATE* 2012, pp. 1579-1584.

[13] A. Mishchenko, LUT structure for delay: cluster or cascade, *IWLS* 2012, pp. 84-88.

[14] A. Mishchenko, et al., Combinational and sequential mapping with priority cuts, *ICCAD* 2007, pp. 354-361.

[15] S. Jang, et al., WireMap: FPGA technology mapping for improved routability and enhanced LUT merging, *ACM TRETS*, vol. 2, No. 2, Article 14, 2009.

[16] Berkeley Logic Synthesis and Verification Group, ABC: A System for Sequential Synthesis and Verification. http://www.eecs.berkeley.edu/~alanmi/abc/

[17] J. Greene, et al., A 65nm flash based FPGA fabric optimized for low cost and power, *FPGA* 2011, pp. 87-96.

[18] G. Zgheib, Leading the blind: automated transisttor-level modeling for FPGA architects, Ph.D Thesis, EPFL, 2017.

[19] S. Jang, et al., SmartOpt: An industrial strength framework for logic synthesis, *FPGA* 2009, pp. 237-240.

[20] K. Murray, et al., Timing-driven Titan: enabling large benchmarks and exploring the gap between academic and commercial CAD, *ACM TRETS*, vol. 8, No. 2, Article 10, 2015.

[21] G. Yeric, Moore's Law at 50: Are we planning for retirement?, IEEE Int'l Electron Devices Meeting, 2015.