Area-Optimized Technology Mapping for Hybrid FPGAs

Srini Krishnamoorthy, Sriram Swaminathan, and Russell Tessier

Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, MA. 01003
sikrishn@ecs.umass.edu

Abstract. As integration levels in FPGA devices have increased over the past decade, the structure of programmable logic resources has become more diversified. Recently, Altera Corporation has introduced a new family of LUT-based FPGAs that have been augmented with user-configurable programmable logic array blocks (PLAs). In this paper a novel FPGA technology mapping approach is described that automatically partitions user designs into netlist subgraphs appropriately-sized for implementation on both types of available user resources. The subgraphs are subsequently mapped to assigned target resources. It is shown that fast *estimation* of *post-minimization* product term counts plays an especially important role in the mapping of designs to PLAs.

1 Introduction

Recent innovations in FPGA architecture have led to the development of new FPGA families that combine diverse sets of logic resources on the same silicon substrate. To support wide fan-in, low logic-density subcircuits, such as finite state machines, several contemporary *hybrid* FPGA architectures [7] contain SRAM-configurable programmable logic arrays (PLAs), product term based structures optimized for area-efficient design implementation [6]. When coupled with fine-grained look-up tables, PLAs provide an integrated programmable resource that can be used in many digital system designs to support control logic for LUT-based datapaths.

In this paper, the general technology mapping problem for hybrid FPGAs is described and an automated mapping algorithm for hybrid FPGAs which optimizes design area is presented. While previous work in hybrid technology mapping has focused on single-output logic cones [7] [8], our heuristic approach uses the concept of Maximum Fanout Free Subgraphs (MFFSs) [5] to quickly identify circuit regions that are well-suited to PLA implementation. To evaluate an input design, a flow-based search is performed on the input netlist to locate design subcircuits that contain high fan-in but require limited logical area. After subgraphs are ranked and clustered, PLA subgraphs and the remaining portion of the original design are mapped to PLAs and LUTs respectively. While our approach is specifically optimized to target Altera APEX20KE FPGAs, it

is general enough to be easily adapted to other hybrid programmable architectures as well. Subgraph evaluation for PLA implementation is significantly complicated by the limited number of product terms available in each PLA. For successful PLA implementation, subgraphs must meet both I/O limitations *and* product term limitations. While it is possible to quickly evaluate subgraph Pterm counts prior to PLA optimization, these counts may differ substantially from final counts found after PLA optimization using logic minimization approaches. To aid in subgraph evaluation we have developed a product term *estimator* that can accurately predict the post-minimization product term count of subgraphs without performing time-consuming full logic minimization.

Quartus, Altera's software tool for mapping circuits to APEX20KE devices can map a given circuit to either LUTs or PLAs, but it cannot automatically partition circuits to target both LUTs and PLAs. We have used our subgraph based partitioning tool, *hybridmap*, to partition a given circuit into both LUTs and PLAs. The output of this tool is fed to Quartus to complete mapping to APEX20KE devices containing both LUTs and PLAs.

2 Background

Technology mapping for LUT-based devices has been explored extensively since the introduction of commercial FPGAs fifteen years ago. The existing approaches can roughly be categorized as tree-based, flow-based [3], and cut-based [4] mapping. The most extensively-used approach, flow-based mapping, has been applied to both LUT-only [3] and LUT/ROM-based FPGA architectures and is adapted in this paper for LUT/PLA-based devices. While memory blocks not used to implement memory functions can be leveraged to implement combinational functions with extended logical depth [5] [10], limited memory input counts currently restrict the breadth of logic functions that can be implemented. As a result, wide fan-in subcircuits such as finite state machines must be migrated into accompanying device LUTs in many design implementations, motivating a design migration to newer hybrid LUT/PLA devices.

Recently, Kaviani [7] [8] has investigated both the architectural parameters of hybrid FPGA architecture and supporting technology mapping approaches. In the architectural study it was shown that *low-fanout* PLAs make area-efficient replacements for small numbers of lookup tables and are well-suited to implement logic nodes with wide fan-in and sparse logical density. The described technology mapping approach for these hybrid LUT/PLA architectures applies partial collapsing and partitioning to isolate wide fan-in logic nodes with single outputs. Input sharing is then used to determine which nodes should be combined into the PLA. In contrast to the Kaviani architecture, the Altera APEX20KE contains PLAs with relatively large numbers of product terms (up to 32) and outputs (up to 16) in relation to input count (32). The structure of the PLA is largely influenced by a desire to allow the circuitry forming the PLA to operate either as a PLA *or* as a small memory containing 2048 SRAM bits [6]. In Section 5, it will be shown that as the number of outputs increase in a PLA, product

term sharing becomes an important issue in efficient PLA mapping. As a result *subgraph-based* rather than single-output node-based approaches that consider both input *and* product term counts are appropriate for mapping to wide-fanout PLAs.

3 Problem Definition

For single-output PLAs well-known two-level minimization techniques can be employed to map designs to minimized sum-of-products representations [2]. More extensive combinational node search approaches are needed to determine which multi-fanout subgraphs are appropriate for implementation in PLAs. In our *hybridmap* system a search is optimized to locate wide-fanin local subgraphs of logic embedded within a design netlist that require a minimal product term count (e.g. ≤ 32). By extracting this circuitry from the LUT-implementation space, more design logic can be squeezed into an FPGA device or a smaller device can be used for the same design. Following allocation of portions of the user design to specific FPGA resources, individual technology mapping tools are used to optimize each design portion to a PLA or collection of LUTs independently.

The same terminology that has previously been used in [10] and [5] to describe graph operations involved in technology mapping will be applied in this paper. Input to the technology mapping system is a combinational circuit, represented as a directed acyclic graph $G(V, E)$ containing combinational nodes V and interconnection edges E. For each node v in G, a *cone*, rooted at v, consists of v and at least one of its predecessors. If all edges driven by nodes in the cone (except for the root) fan out to other nodes in the cone, the cone is said to be *fan-out free*. The fan-out free cone rooted at v containing the largest possible number of nodes is the *maximum fan-out free cone* or $MFFC(v)$. The concept of a fan-out free cone can be further extended to subgraphs if multiple nodes are considered to form a *root set*, S. A *fan-out free subgraph*, $FFS(S)$, is a subgraph containing S and a number of predecessors in which all edges driven by nodes in the subgraph (except S) fan out to other nodes in the subgraph. The maximum fan-out free subgraph, $MFFS(S)$, is the fan-out free subgraph rooted at S that contains the largest number of nodes. A subgraph that contains no more that d inputs can be characterized as *d-feasible* and a subgraph that contains no more than m product terms after logic minimization can be characterized as *m-packable*. For a specific device, a subgraph is considered *PLA-feasible* if the number of subgraph inputs and outputs are each less than the number of PLA inputs (i_m) and outputs (o_m) and the number of product terms needed to implement the subgraph is fewer than the number (p_m) found in a PLA.

4 Methodology

Hybridmap uses a collection of new heuristics and existing technology mapping tools to perform hybrid technology mapping. The high-level flow of the steps taken by our tool is shown in Figure 1. Initially, the circuit under consideration

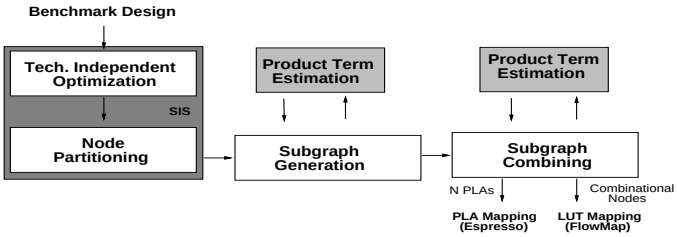

Fig. 1. Software Flow

is represented as a directed acyclic graph containing combinational nodes. Technology independent optimization is performed by *SIS* [9] using the optimization scripts *script.rugged* or *script.algebraic* to reduce the logic complexity of input designs. As a second preprocessing step, the *SIS* kernel extraction tool, *xl_split* [9] is used to recursively split wide-fanin nodes into multiple nodes, each possessing input counts which meet PLA i_m constraints.

Following preprocessing, a multi-step graph-based analysis of circuit nodes is performed to locate suitable subgraphs for PLA implementation. As a first step, a graph traversal starting from each graph node, v, is performed. For each traversal, a breadth-first search is used to locate fan-out nodes that are direct graph descendents of v. This traversal leads to identification of the subgraph root set associated with v. Following the breadth-first search, a maximum fan-out-free subgraph based on the root set is determined by tracing backward from the root set along fan-in edges until all inclusive nodes and edges are identified. After MFFS evaluation for all graph nodes v, all isolated subgraphs are ranked based on input and product term count. An important tool in product term evaluation is Pterm *estimation* which estimates the post-minimization product term count that could be expected by applying logic minimization tools such as *Espresso* [2]. To promote full PLA utilization, individual subgraphs are clustered based on input and product term sharing to determine the final collection of subgraphs to be implemented in PLAs. The three steps that form the core of our system flow, subgraph generation, product term estimation, and subgraph combining, are described in greater detail in the next two sections. Additional algorithm details can be found in [11] also.

4.1 Subgraph Generation

Identification of feasible PLA subgraphs starts as a localized circuit DAG search designed to locate collections of shared nodes that drive PLA outputs and have limited fanout. These nodes form a subgraph root set and serve as a basis for the identification of fan-in signals and nodes that may be absorbed into PLAs. To promote product term reuse, nodes in the root set ideally share numerous fan-in signals and fan-in nodes. The heuristic used to identify root set nodes in our system is similar to the root set determination algorithm outlined in [5]. For each node v in G, a transitive fan-out set is determined by traversing the fan-out

of v in a breadth-first fashion until a prespecified tree depth, n is reached. As shown in Figure 2a for $n = 2$, this node set includes all combinational nodes that are driven by v and all direct successors of these nodes. At intermediate points in the search the covered tree that contains the largest number of nodes while still driving no more than o_m outputs is saved. Following search completion the leaves of the saved tree are designated as the root set. In Figure 2a the darkly-shaded leaf nodes can be identified as the root set of this traversal. Given the feed-forward nature of this step, root set determination across all V nodes in graph G can be completed in $O(V)$ time.

Once a root set has been determined, an inverse traversal of G starting from S is performed to determine the MFFS associated with the root set. At each iterative step of the post-order DAG traversal a test is made to determine if candidate node fan-out is limited to other current subgraph nodes. When no other nodes that meet this criteria can be located or the fan-in to the subgraph exceeds i_m, inverse graph traversal is terminated. As shown in Figure 2b, all predecessors of the root set contained by an MFFS can be targeted to a PLA since intermediate signals driven by the subgraph are not needed outside the subgraph. For worst case traversal, the algorithm will require $O(E)$ time to evaluate MFFSs for each DAG node in G, where E is the number of edges in G.

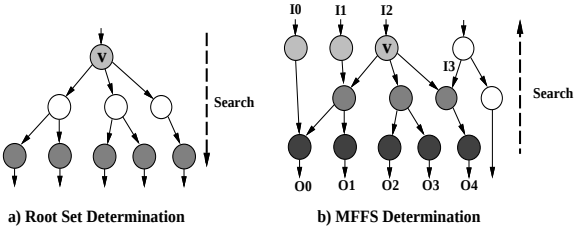

Fig. 2. Root Set and MFFS Determination

4.2 Product Term Estimation

As described in Section 1, PLA structures in hybrid FPGA devices are well-suited to the implementation of wide fan-in, low logic-density circuit constructs such as finite state machines. In considering a set of subgraphs with equivalent input and output characteristics, it is therefore desirable to select subgraphs with limited product term counts over those with larger counts. To illustrate the range of product term counts that can be found in subgraphs with approximately (within 1 or 2) $i_s = 32$ inputs and $o_s = 16$ outputs, product term counts for 168 disjoint subgraphs extracted from the benchmarks listed in Table 3 were derived and plotted in Figure 3. In the figure it can be seen that 34% of subgraphs contain less than 32 product terms and that many of these m-packable

subgraphs contain Pterm counts quite close to the 32 Pterm boundary. Through experimentation it has been determined that after two-level minimization, product term counts can vary significantly from original values. As a result to select subgraphs that minimize post-minimization product term count, a heuristic tool based on Espresso has been developed that can estimate to within 10% accuracy the post-minimization product term count of subgraphs on average in less than 10% of the time taken by Espresso. Our estimation tool performs the same basic

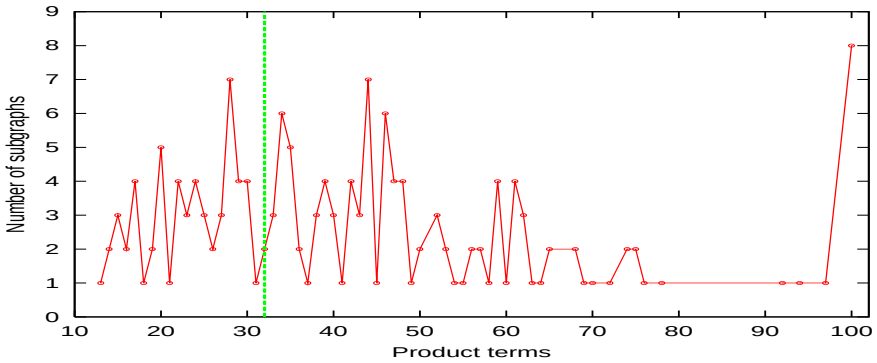

Fig. 3. Pterm Count for Subgraphs with $i_s = 32$, $o_s = 16$

PLA optimizations as Espresso on target subgraphs including Pterm sharing, Pterm covering, and graph expansion. Unlike Espresso, our tool significantly limits the reduction search space by restricting Pterm expansion and output complementation to only those combinations most likely to lead to an overall Pterm reduction. These minimizations are illustrated through use of examples. The input required by Espresso is an encoded truth table, as shown in Figure 4. Each row indicates a product term (or *cube*) with true (1), complemented (0) or don't care (-) conditions and each output column represents a single output. For circuits containing large numbers of outputs (e.g. 32) *don't care* expansion by Espresso (Fig 4) can lead to a large search space. In performing estimation, we choose to expand only a single cube at a time. The minimizations listed above (cube sharing, Pterm covering, graph expansion) are then applied incrementally to reduce Pterm count. A second approach uses lookup tables to programmably invert PLA outputs. During the minimization process, for a n output subgraph, only n complementations are considered compared to the 2^n complementation patterns tried by Espresso. Each output is complemented starting from the one driven by the most Pterms. As shown in the example, by choosing to complement only the first output, the logic expressed by the first three Pterms is now covered by the fourth.

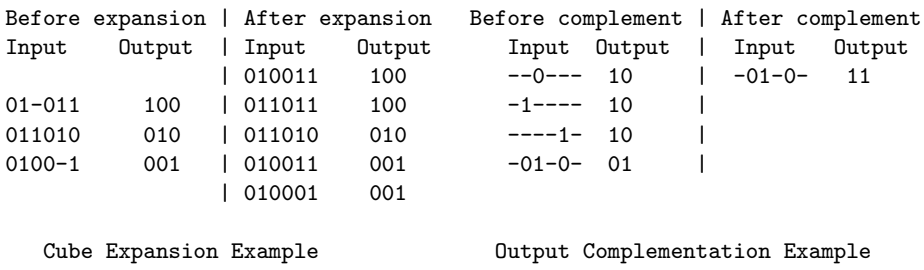

Fig. 4. Minimization Examples

4.3 Subgraph Combining

Following graph reduction, all resulting subgraphs under consideration can feasibly be implemented in a target PLA. Multiple, smaller subgraphs may be combined together through bin packing to form combined implementations that still meet i_m, o_m, and p_m PLA requirements. Merged subgraph combinations are evaluated using the following cost equation which encourages the growth of wide fan-in subgraphs while penalizing overly Pterm-dense combinations:

$$Cost_{jk} = feas(j,k) \times \frac{c \times i_{jk}}{d \times p_{jk}} \quad (1)$$

where c and d are scaling constants and i_{jk} and p_{jk} are merged subgraph input and Pterm counts respectively. A merged subgraph is judged to be feasible if input, output and post-minimization Pterm counts are less that available PLA counts. While input and output limits are straightforward to evaluate, possible product term merging across subgraphs may require additional invocation of the product term estimator to eliminate combined graphs that overflow product term counts from consideration. Given r target PLAs, following combining the r feasible subgraphs that cover the most inputs while minimizing the number of internal product terms are selected. Selected subgraphs are extracted from the original subgraphs and optimized by a full pass of Espresso. The remainder of the circuitry is mapped to four-input look-up tables using FlowMap [3].

5 Results

Hybridmap has been implemented and applied to 11 MCNC logic synthesis benchmarks. All experiments were performed on a 250MHz Sun UltraSparc with 320 MB of memory. To measure the importance of subgraph-based technology mapping for hybrid FPGAs, the number of output signals driven by each product term across 150 mapped subgraphs was determined for subgraphs with over a range of subgraph outputs counts (o_s). In Figure 5 shared output values greater than one indicate that product terms are shared across a number of outputs and that subgraph analysis approaches are necessary to consider subcircuits

that drive multiple outputs simultaneously. For low PLA output counts, little product term sharing occurs but as the number of PLA outputs increase, sharing becomes more prevalent. Such product term sharing would indicate a need for subgraph rather than cone-based optimization. In Section 4.2, an al-

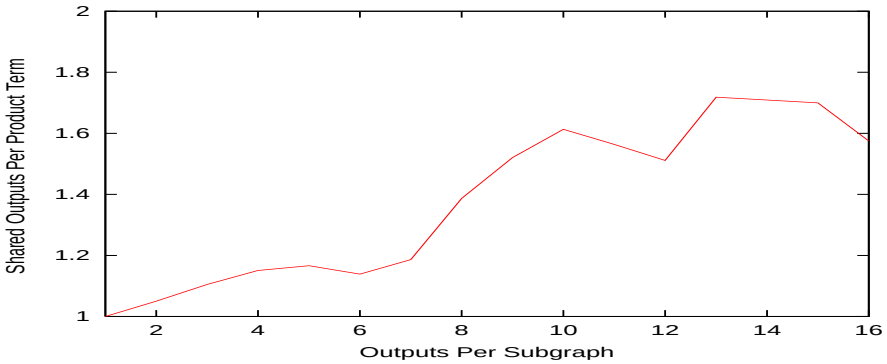

Fig. 5. Average Outputs Driven by Each Product Term

gorithm for fast product term estimation was outlined that can estimate the post-minimization product term count of a subgraph prior to the application of Espresso. A total of 15 subgraphs with input and output values close to 32 and 16 (the I/O counts of PLAs in APEX20KE) were evaluated for each MCNC benchmark and averaged results appear in Table 1. On average, the estimator determined a product term count that was within 3.7% of the total determined by Espresso in a small fraction of Espresso run time. The maximum error found by the estimator for any single design was determined to be 7 product terms. The run times of the estimator are small enough to allow for the Pterm evaluation of a small set of subgraphs during the subgraph generation and combining steps.

To test the quality of *hybridmap* in reducing LUT utilization, four large benchmarks were mapped to the APEX20KE family. Quartus was used to map the partitioned circuit created by hybridmap onto the APEX20KE devices. From Table 2 it can be seen that PLAs can effectively be used to reduce the overall LUT count in a mapped design allowing area for additional design circuitry to be implemented in leftover LUTs. By automatically mapping subgraphs to PLAs about 10% additional LUTs may be mapped to a target device versus a mapping that uses LUTs only. As a final experiment, *hybridmap* was compared to recent results published in [8] for hybrid devices containing relatively low-fanout PLAs with $i_m = 16$ inputs, $o_m = 3$ outputs, and $p_m = 10$ Pterms. As shown in Table 3 even though the subgraph-based tools were designed to target PLAs with wide fanouts, the results from *hybridmap* were competitive with previously-reported work. The inputs used for these experiments were previously used in [8] and were provided from Kalenteri.

6 Conclusions

Recent advances in FPGA architecture and technology have made PLAs attractive devices with hybrid logic *hybridmaps* a reality. In this paper we have outlined heuristic techniques to automatically identify portions of a design netlist that are appropriate for implementation in PLA-based logic resources found in Altera

Circuit Name	Orig. Pterms	Espresso		Estimator			
		Pterms	Time (s)	Pterms	Time (s)	Ave. Diff (Pterms)	Max. Error (Pterms)
apex4	32	25	772	25	2	0	0
cordic	29	26	21	29	1	3	3
cps	21	20	347	20	2	0	0
frg2	45	39	726	39	1	1	3
misex3	27	26	1384	26	12	0	0
sbc	45	35	268	40	9	5	7
scf	27	21	52	21	13	0	1
spla	43	25	106	25	11	0	0
total	269	217	3676	225	51	-	7

Table 1. Comparison of Estimator with Espresso

		LUTs-only	Hybrid	
Circuit	APEX Device	Quartus 4-LUTs	PLAs	Left Over 4-LUTs
spla	20KE100	1769	10	1648
pdc	20KE100	2303	10	2168
frisc	20KE100	2294	10	2025
des	20KE100	1150	10	1038

Table 2. Hybrid Mapping for the APEX20KE Architecture

Circuit	# of 4-LUTs	Node-based		Hybridmap	
		PLAs	4-LUTs	PLAs	4-LUTs
s1423	154	19	72	19	95
frg2	324	30	123	30	107
x3	282	25	98	25	109
dalu	357	27	106	27	104
sbc	266	26	105	26	124
cps	520	53	209	53	221
s1488	219	21	81	21	85
scf	300	33	126	33	116
apex2	905	90	366	90	355
alu4	666	69	286	69	245
Total	3993		1572		1561

Table 3. Comparison with Node-based Hybrid Mapping

APEX20KE devices. By integrating our search method with a fast product term estimator it is possible to quickly identify wide-fanin, low logic density subgraphs that are well-suited to PLA implementation.

7 Acknowledgements

We are thankful to Alireza Kaviani for providing us with the circuits that were previously used in [8].

References

1. *APEX 20K Data Sheet.* Altera Corporation, 1999.
2. R. Brayton, A. Sangiovanni-Vincentelli, G. Hachtel, and C. McMullin. *Logic Minimization Algorithms for Digital Circuits.* Kluwer Academic Publishers, Boston, MA, 1984.
3. J. Cong and Y. Ding. FlowMap: an Optimized Technology Mapping Algorithm for Delay Optimization in Lookup-table Based FPGA Designs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13:1–12, Jan. 1994.
4. J. Cong, C. Wu, and Y. Ding. Cut Ranking and Pruning: Enabling a General and Efficient FPGA Mapping Solution. In *ACM 7th International Symposium on Field-Programmable Gate Arrays*, Monterey, Ca., Feb. 1999.
5. J. Cong and S. Xu. Technology Mapping for FPGAs with Embedded Memory Blocks. In *ACM 6th International Symposium on Field-Programmable Gate Arrays*, Monterey, Ca., Feb. 1998.
6. F. Heile and A. Leaver. Hybrid Product Term and LUT Based Architectures Using Embedded Memory Blocks. In *ACM 7th International Symposium on Field-Programmable Gate Arrays*, Monterey, Ca., Feb. 1999.
7. A. Kaviani and S. Brown. The Hybrid Field-Programmable Architecture. *IEEE Design and Test of Computers*, pages 74–83, Apr. 1999.
8. A. Kaviani and S. Brown. Technology Mapping Issues for an FPGA with Lookup Tables and PLA-like Blocks. In *ACM/SIGDA 8th International Symposium on Field-Programmable Gate Arrays*, Monterey, Ca., Feb. 2000.
9. E. Sentovich. *SIS: A system for sequential circuit analysis.* Tech. Rep. UCB/ERL M92/41, Electronics Research Laboratory, University of California, Berkeley, may 1992.
10. S. Wilton. SMAP: Heterogeneous Technology Mapping for Area Reduction in FPGAs with Embedded Memories. In *ACM 6th International Symposium on Field-Programmable Gate Arrays*, Monterey, Ca., Feb. 1998.
11. Srini Krishnamoorthy, Sriram Swaminathan and Russell Tessier. *Area-Optimized Technology Mapping for Hybrid FPGAs.* UMass Amherst ECE Dept Tech. Report TR-CSE-00-4, 2000