# SHADOW AND-INVERTER CONES

*Hadi Parandeh-Afshar, Grace Zgheib, David Novo, Madhura Purnaprajna, Paolo Ienne*

School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
1015 Lausanne, Switzerland
{hadi.parandehafshar, grace.zgheib, david.novobruna, madhura.purnaprajna, paolo.ienne}@epfl.ch

## ABSTRACT

Despite their many advantages, FPGAs still come with significant overheads in area, delay, and power consumption due to an extreme programmability in both the routing and logic. From the performance perspective, large logic blocks, capable of covering big portions of circuits, lead to fewer hops in the routing network, and thus, to a shorter critical path. Recent work has shown that And-Inverter Cones (AICs) can considerably reduce the number of logic block levels compared to Look-Up Tables (LUTs), in a radically altered FPGAs architecture. In this paper, we use AICs as shadow logic for LUTs, which incurs minimal architectural changes with respect to current FPGAs, while exploiting the benefits of both AICs and LUTs. We also propose changes in the AIC architecture, for a more compact technology mapping. The new architecture reduces the average circuit delay by up to 35% with respect to standard FPGAs at the expense of a 3x increase in the number of the logic clusters. Other benchmarks show more moderate area overheads, e.g., 16% delay improvement for 20% area overhead.

## 1. INTRODUCTION

The trade-off between flexibility and performance is a constant challenge in the design of general-purpose architectures like FPGAs. In this paper, we propose shadow *And-Inverter Cones* (AICs) to improve the performance of present FPGAs without compromising their flexibility. Unlike the application of *shadow logic* to improve area efficiency [1], this paper entirely focuses on improving performance.

And-Inverter Cones were proposed as an alternative to *Look-Up Table* (LUT)-based FPGAs [2]. The main advantage of AICs is their scalability; the area of an AIC increases linearly with the number of inputs and not exponentially as in LUTs. However, replacing LUTs by AICs requires radical changes in the FPGA architecture, which jeopardizes its early adoption by FPGA manufacturers. As an alternative, we propose to include AICs as shadow logic to LUTs in existing FPGAs. A 6-AIC block, having 64 inputs, is placed "behind" some of the LUTs in a cluster. The area of the AIC is only around 7% of the LUT cluster area, and the delay penalty of adding the AIC block to the cluster is negligible. With an AIC as a shadow logic block to a group of LUTs, the mapping tool has the option of using the AIC instead of the group of LUTs whenever it brings a performance advantage. Furthermore, we suggest an enhancement to the original AIC architecture, which results in area reduction and better timing.

The rest of the paper is organized as follows: Section 2 introduces the new shadow logic cluster. Our modified technology mapping approach and area recovery are briefly described in Section 3. Section 4 presents the experimental methodology and evaluates the results for the new architecture. Finally, Section 5 draws conclusions.

## 2. THE NEW SHADOW LOGIC CLUSTER

The new shadow logic cluster is used to improve the performance of FPGAs by adding an AIC block as shadow to some of the LUTs of an existing logic cluster. The following subsections describe the enhanced AIC structure as well as the general architecture of the shadow cluster.

### 2.1. Enhanced And-Inverter Cones

The AIC [2] is a full binary tree of cells which can be configured as 2-input NAND or AND gates. The architecture of the AICs is inspired from the *And-Inverter Graphs (AIGs)* [3], where all nodes are 2-input AND gates with an optional inversion at the output.

However, in this original AIC architecture, inverters are not available at the inputs of the cells. Therefore, nodes whose fanouts include both inverted and non-inverted edges cannot be represented as-is. To accommodate these cases, some AIG transformations—such as duplicating the node that has the fanout or adding an inverter node at its output—are required, which increase the size of the AIG. This is only a constraint for AIC mapping and not for LUTs, as the inversion can be fixed internally by an appropriate LUT configuration.

978-1-4799-0004-6/13/$31.00 

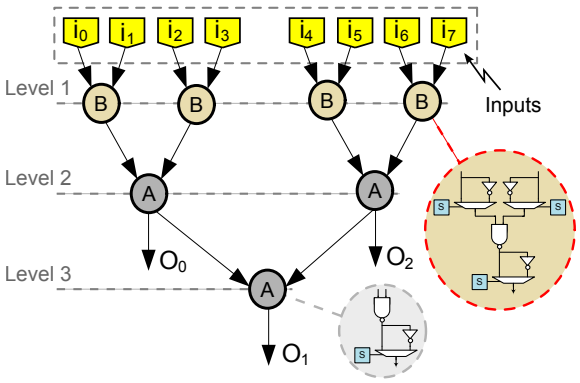


**Fig. 1**. ***Enhanced* 3-AIC architecture**. A 3-AIC has three levels of cells and eight inputs. We add a programmable inversion at the input of the first level (B) nodes, improving the flexibility of the cone.

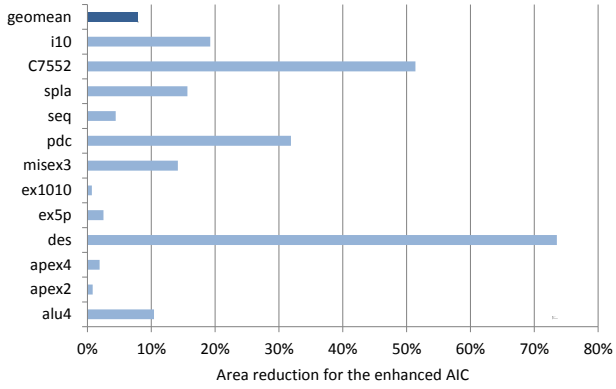


**Fig. 2**. **Effect of the *Enhanced* AIC architecture**. Area saving in terms of number of clusters that is achieved when mapping on the enhanced AIC as opposed to mapping on the original AIC.

In this work, we change the architecture of the AIC nodes at the first level of the cone to support programmable input inversions as well as programmable output inversions. This provides more flexibility and allows the AIC to map a larger subset of functions without the need for duplication or additional inverter nodes, which reduces the mapping cost in terms of both area and delay. Figure 1 shows the new AIC architecture. Nodes labelled as "B" have the new architecture, while the other nodes remain unchanged.

Figure 2 indicates the potential area reduction when the selected benchmarks are mapped on the enhanced AIC as opposed to the original AIC. Area reduction is more pronounced in benchmarks having a high rate of node duplication and input inversions when mapped on the original AIC.

### 2.2. Shadow Cluster Architecture

The new shadow cluster consists of the traditional FPGA cluster with one AIC as a shadow logic to a set of LUTs. For such an architecture, the mapper selects the AIC or the LUTs depending on the area and delay requirements. We chose the *Altera Stratix-III* [4] as the reference FPGA architecture where, each cluster has 10 logic blocks, called *Adaptive Logic Modules (ALM)*. Each ALM has eight inputs and two outputs. A 6-AIC block is used as a shadow to eight ALMs while the two remaining ALMs can be used exclusively for LUT-only implementation. This 6-AIC has 64 inputs and 31 outputs, which allows it to share those inputs with the shadowed LUTs while keeping the same cluster interface. However, since the shadowed ALMs have 16 outputs, a small sparse crossbar is used at the output of the AIC to select 16 outputs out of the 31. These 16 crossbar outputs are multiplexed with the 16 ALM outputs. Figure 3 illustrates the proposed shadow logic cluster; which is a hybrid AIC-LUT cluster. This means that the only area overhead introduced to the reference cluster is that of the AIC, its crossbar and multiplexers. However, the increase in area on account of these additional components is only about 7%. Given a technology mapped netlist which contains AIC and LUT blocks with different sizes, the cluster packer decides how to group the blocks and use the resources of the cluster shown in Figure 3.

## 3. TECHNOLOGY-MAPPING: HYBRID LUT-AIC SHADOW CLUSTER

In this work, the main design objective is to improve the delay of the mapped circuits using the mixture of LUTs and AICs that exist in the new shadow logic cluster. Hence, our technology mapping algorithm searches for hybrid solutions (LUT+AIC) that reduce the depth of the circuits. Area reduction is considered as a secondary optimization objective. Consequently, the mapping algorithm includes an area recovery phase, in which the non-critical paths are re-mapped to minimise area requirements.

The input to the technology mapping phase is the AIG-based representation of the circuit. As a standard approach, for a node $v$ in the AIG, numerous mapping candidates are extracted and the best one that meets the design objective is chosen. These candidates are conic subgraphs which are rooted at $v$. For LUT mapping, the cone candidate is IO bounded–$K$-feasible for a $K$-LUT–and depth $D$ bounded for a $D$-AIC.

After all mapping candidates are extracted, the algorithm tries to identify the best candidates of each node. It then covers the graph with the best candidates of a subset of nodes that are determined in the final stage of the mapping. This mapping approach [2] is a modified version of the classic LUT mapping algorithm [5], adapted for the LUT/AIC hy-

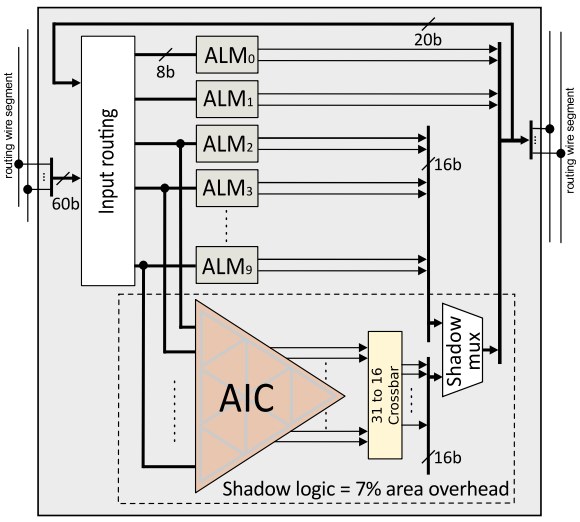


**Fig. 3**. **Shadow cluster architecture.** A 6-AIC is added to the reference cluster as a shadow logic to 8 ALMs, sharing the input crossbar. An additional crossbar is used to select 16 AIC outputs to be multiplexed with the ALM's outputs.

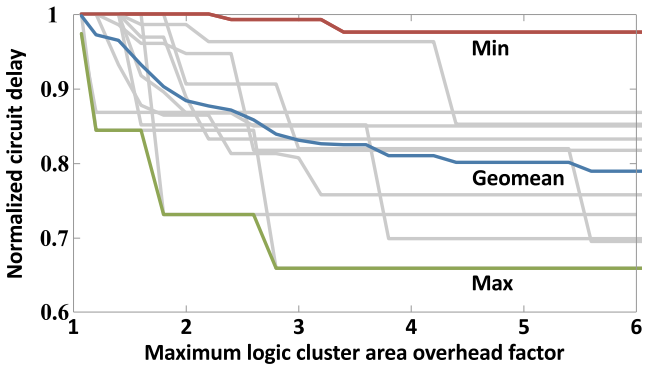


**Fig. 4**. **Delay to area trade-offs.** Best achievable—normalized—delay of multiple mappings of all benchmarks onto the shadow-AIC clusters for different area overheads with respect to the reference architecture, along with their minimum, maximum and geometric mean. The proposed architecture allows to trade off area for delay by sweeping the mapping parameter that sets the maximum logic depth kept during area recovery.

brid structure. We have adapted this mapping algorithm so that it trades-off delay in the non-critical paths to area, for the cases where the area overhead is significant. This modification is included in the area recovery phase, an add-on to the original mapping algorithm.

The AIC mapping algorithm performs a depth-oriented cone selection. If two cones are equivalent in depth, then the one with the lowest area is selected. Selecting the cone that minimizes the depth at each node guarantees the mapping solution to be depth optimal.

As such, mapping starts, as a first step, by generating all cone candidates—AIG subgraphs—of the nodes in the AIG. These subgraphs can be mapped to LUTs, AICs, or both. Then, as a second step, it finds the best candidate cone of each node in an AIG. This is done by traversing the AIG in topological order and selecting the candidates that minimize the depth of each node. The selected cone can be mapped either on a LUT or an AIC. If two candidates give the same depth to the node, the one with lower *area flow* is selected. Area flow [6] is an estimation of the area contribution of a mapping option. The final step is to map the entire AIG using the best mapping candidates identified for the nodes in the circuit. For this, the AIG is traversed backwards, from outputs to inputs, and is covered by the best cones of the nodes that become *visible* (and part of the solution) during the backward traversal. In this step, it is possible to reuse (for free) the AIC candidate of a previously visited node for the mapping of the current node. In such an AIC, the current node is mapped to a side output and is not the root of the AIC.

A single pass area recovery is integrated into the third step of the above mapping algorithm. The idea is to relax the mapping of the non-critical paths by using candidates that are not delay-optimal, but result in reduced area without affecting the (critical) depth of the circuit.

Statistics showed that on average, more than 20% area reduction is achieved when applying this area recovery step.

## 4. RESULTS

To implement a circuit on the new shadow cluster architecture, we first perform logic synthesis and optimization using ABC [7] to generate an AIG netlist in BLIF format. This BLIF file is forwarded to our technology mapping tool developed in C++, before being fed to VPR for packing, placement, and routing. In VPR, the timing-driven packer groups logic blocks in the technology mapped netlist into hybrid logic clusters with both AICs and LUTs. The output of the packer is a netlist consisting of packed complex blocks, functionally equivalent to the input netlist. This packed netlist is forwarded to the placement and routing algorithm with 10 different placement seeds. We report the average delay that was obtained from these different seeds. We used the MCNC benchmark suit [8] to evaluate the new FPGA architecture. Initially we started with the 20 biggest MCNC benchmarks, but due to the VPR-related packing problems that we encountered on sequential benchmarks, we only report the results for combinatorial circuits.

### 4.1. Evaluation

We compare the delay and area of the new mapping scheme for the presented shadow AIC architecture with the mapping

on the reference LUT-only architecture. Our mapping approach tries to minimize circuit delay by re-mapping logic onto the faster AICs; however, this entails an area overhead for two reasons: (1) the cluster area of our architecture is 7% bigger than the reference architecture and (2) the extensive use of AICs can increase the number of clusters required to map a particular circuit, as the architecture only provides a single AIC per cluster. The latter is alleviated by an area recovery phase, which tries to avoid AICs from noncritical paths. Still, the number of clusters required to achieve the maximum delay reduction can be significantly higher than in the reference mapping. Accordingly, we add a parameter $\beta$ to the mapping tool that can increase the maximal depth constraint used during the area recovery phase. Thereby, $\beta$ can be used to trade-off logical depth for AIC usage, which typically results in circuits that utilize fewer logic clusters (smaller) but with a larger critical path than the minimal one (slower). However, the technology mapping process is unaware of the cluster packing and routing, and as result, a decrease in logical depth does not always translate into a critical path reduction in the final implementation.

Each benchmark is mapped multiple times by varying values of $\beta$. Figure 4 shows the circuit delay of the Pareto optimal mappings with respect to a mapping on the reference architecture for different logic cluster area overhead factors. Note, that while sweeping $\beta$, some of the mappings are not Pareto optimal in the area-delay space, due to the delay contributions of local and global routing. The figure also shows the minimum, maximum and geometric mean of the normalized delay of all benchmarks depending on the maximum area overhead factor. The best delay improvement is 35% with respect to the reference delay at a non-negligible 3x logic cluster area increase. However, one benchmark is able to achieve a 16% delay improvement for a maximum factor area increase of 1.2x. In general, the results show significant increases in logic cluster area; however, the FPGA area also includes global routing area, which will reduce the total area overhead.

Nevertheless, we present a new architecture that offers diverse set of design solutions with varying delay and area trade-offs. Importantly, the LUT-only traditional mapping solution can always be adopted for a minimal area overhead of 7%. Accordingly, at the system level, one could achieve a sizable delay improvement by aggressively optimizing the delay critical components while keeping a moderate area increase on the rest of the design. Although the extreme solutions are probably unpalatable to most applications due to the significant area growth, they still show that the potentials for gain in timing are in some cases excellent (up to 35% delay reduction). The challenge will be to improve our mapping algorithm so that a minimum number of AICs are allocated to reduce the actual critical path and not just the logic depth.

## 5. CONCLUSIONS

This paper proposes a new logic cluster architecture for FPGAs. A 6-level And-Inverter Cone is placed as shadow logic "behind" eight of the ten logic blocks that exist in each logic cluster of the *Altera Stratix-III*. This requires a minimal architectural modification to the original logic cluster who's size increases by 7%. We also enhanced the original AIC architecture for a more efficient mapping and cone utilization. Our results show that the new cluster architecture is able to reduce the average circuit delay by 20% with respect to the standard FPGA cluster. In some cases, the delay reduction reaches 35%. Such a delay advantage comes at the price of using more clusters; for, on average, this delay is achieved at 3x the logic area cost. However, we also provide several design solutions that present a wide trade-off between delay and area, knowing that a purely LUT-based solution is always possible.

In the future, we will focus on developing a more intelligent technology mapping tool that can produce smaller circuits, while trying to maintain the current performance improvements. We will also consider the possibility of using AIC blocks as shadow logic for hard logic blocks in FPGAs.

## 6. REFERENCES


[1] P. A. Jamieson and J. Rose, "Enhancing the area efficiency of FPGAs with hard circuits using shadow clusters," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 12, pp. 1696–1709, Dec. 2010.

[2] H. Parandeh-Afshar, H. Benbihi, D. Novo, and P. Ienne, "Rethinking FPGAs: Elude the flexibility excess of LUTs with And-Inverter Cones," in *FPGA12*, Monterey, Calif., Feb. 2012.

[3] L. Hellerman, "A catalog of three-variable Or-Invert and And-Invert logical circuits," *IEEE Transactions on Electronic Computers*, vol. EC-12, no. 3, pp. 198–223, June 1963.

[4] *Stratix III Device Handbook, vols. 1 and 2*, Altera Corporation, `http://www.altera.com/literature/`.

[5] J. Cong and Y. Ding, "An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs," in *Proceedings of the International Conference on Computer Aided Design*, Santa Clara, Calif., Nov. 1992, pp. 49–53.

[6] V. Manohararajah and S. Brown, "Heuristics for area minimization in LUT-based FPGA technology mapping," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2331–40, Nov. 2006.

[7] A. Mishchenko, S. Chatterjee, and R. Brayton, "DAG-aware AIG rewriting: A fresh look at combinational logic synthesis," in *Proceedings of the 43rd Design Automation Conference*, San Francisco, Calif., July 2006, pp. 532–36.

[8] S. Yang, "Logic synthesis and optimization benchmarks user guide, version 3.0," Microelectronics Center of North Carolina, Research Triangle Park, N.C., Technical Report, Jan. 1991.