NAND-NOR: A Compact, Fast, and Delay Balanced FPGA Logic Element

Zhihong Huang† Xing Wei† Grace Zgheib‡ Wei Li† Yu Lin†
Zhenghong Jiang† Kaihui Tu† Paolo Ienne‡ Haigang Yang†

†Chinese Academy of Sciences
Institute of Electronics, Beijing, China
{huangzhihong, liw, linyu, kelv, yanghg}@mail.ie.ac.cn

‡Ecole Polytechnique Fédérale de Lausanne (EPFL)
School of Computer and Communication Sciences, 1015 Lausanne, Switzerland
{grace.zgheib, paolo.ienne}@epfl.ch

ABSTRACT

The And-Inverter Cone has been introduced as an alternative logic element to the look-up table in FPGAs, since it improves their performance and resource utilization. However, further analysis of the AIC design showed that it suffers from the delay discrepancy problem. Furthermore, the existing AIC cluster design is not properly optimized and has some unnecessary logic that impedes its performance. Thus, we propose in this work a more efficient logic element called NAND-NOR and a delay-balanced dual-phased multiplexers for the input crossbar. Our simulations show that the NAND-NOR brings substantial reduction in delay discrepancy with a 14% to 46% delay improvement when compared to AICs. And, along with the other modifications, it reduces the total cluster area by about 27%, when compared to the reference AIC cluster. Testing the new architecture on a large set of benchmarks shows an improvement of the delay-area product by about 44% and 21% for the MCNC and VTR benchmarks, respectively, when compared to LUT-based cluster. This improvement reaches 31% and 19%, respectively, when compared to the AIC-based architecture.

1. INTRODUCTION

Since their first introduction in 1984, *Field-Programmable Gate Arrays (FPGAs)* have increasingly become the main computing power in various fields due to their flexibility and programmability. Having a short time-to-market, reconfigurable capabilities and fast programmability makes the FPGA a reliable computing device in modern digital systems [4, 11].

Most current commercial FPGAs are *SRAM-based (Static Random Access Memory)* FPGAs and have an island-style architecture. In both commercial FPGAs as well as academic research, the FPGA architecture relies mainly on the *Look-up Tables (LUTs)* as the main logic elements in the clusters [2]. A K-input LUT can implement any combinational logic function with K inputs. However, the flexibility of the LUTs comes at the expense of circuit area and delay. Undeniably, the improvement of both the area efficiency and the performance of reconfigurable logic architectures has always been one of the main targets of academic and industrial research [1, 6].

Inspired by modern synthesis tools [3], a new logic element was recently proposed as an alternative logic element for FPGAs [12]. Having a conic structure and composed of multi-level configurable AND and inverter gates, this element is called *And-Inverter Cone (AIC)*. When compared with LUTs, AICs have many advantages: (1) the AIC has a high number of inputs and outputs, which allows it to implement larger, multi-output functions; (2) the AIC structure is similar to the regular expression of Boolean algebra, which allows it to satisfy the logic synthesis requirements and abide by its optimizations for an improved performance; (3) the AIC's area and delay increase linearly and logarithmically with the number of inputs, as opposed to the exponential and linear increase of the LUTs, respectively; (4) intermediate results can be directly reused through the intermediate outputs (known as side outputs) of the AIC, reducing the logic duplication and improving the overall circuit area.

Taking all these advantages into consideration, the AIC is presented as a promising alternative to LUTs in FPGAs. Zgheib et al. [15] presented several AIC cluster architectures, compared them with a state-of-the-art LUT-based FPGA architecture, and concluded that both the AIC and LUT-based architectures have their respective advantages and disadvantages, depending on the application. A new depth-constrained technology-mapping tool was also proposed [7] to optimize the circuits for AICs and improve their results, especially in terms of used area. However, a potential issue in the AIC design has been ignored in the existing research, so far: the input-to-output delay of the AIC can vary a lot, depending on the cone configuration. This variation can have a major impact and is of high importance, since, given

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

FPGA '17, February 22-24, 2017, Monterey, CA, USA
© 2017 ACM. ISBN 978-1-4503-4354-1/17/02…$15.00
DOI: http://dx.doi.org/10.1145/3020078.3021750

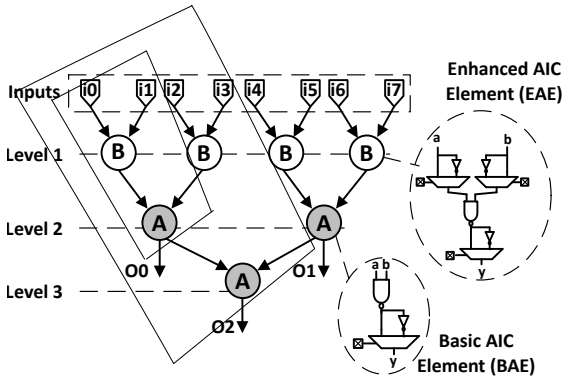

Figure 1: A 3-level AIC (AIC3) with its two types of nodes: Enhanced AIC Element (EAE) and Basic AIC Element (BAE).

its multi-level conic structure, any delay variation that exists in a single AIC node propagates from one level to the other and gets accumulated in the process before reaching the final output. The delay glitch caused by this discrepancy in the cone affects the stability of the overall function and may induce additional dynamic power consumption.

In this paper, we aim at solving the AIC's delay discrepancy problem and propose a more efficient logic element with balanced delays, called NAND-NOR. We also present an improved cluster architecture that reduces the area overhead added with the introduction of AICs into the logic clusters.

The paper starts by presenting the original AIC structure and the delay discrepancy problem, in Section 2. Then, Section 3 proposes the NAND-NOR logic element and analyzes it characteristics. We then introduce a DDM input crossbar and optimized NAND-NOR-based cluster architecture in Section 4. We compare the new architecture, with all its new features, against an LUT-based architecture [15], similar to Stratix-IV, and AIC-based architecture [15] in Section 5. Finally, we summarize our results and conclude in Section 6.

2. LIMITATIONS IN THE AIC DESIGN

The And-Inverter Cone is a multi-level binary tree of cells where each cell is composed of an AND gate with programmable inversions [15], as shown in Figure 1. The AIC has two types of nodes: the *Enhanced AIC Element (EAE)* and the *Basic AIC Element (BAE)*, where the EAE is basically a BAE with programmable input inversions [13].

Figure 2 shows the transistor implementation of the BAE which can be configured as either an AND or NAND gate by selecting either the inverted or non-inverted output, respectively. Despite their many advantages, AICs still have limitations, some of which are already know [15]. However, in this section, we highlight the delay discrepancy problem as well as the inefficiency in the AIC-based cluster design.

2.1 Delay discrepancy problem

Analysis of the AIC shows that the propagation delay of a single node varies with the configuration of the AIC. If the output inversion is needed, then the signal has an additional transistor to go through, increasing the input-to-output delay. This difference in delay can accumulate with every node, as the signal traverses the different AIC levels. To better evaluate the delay difference, we implement the

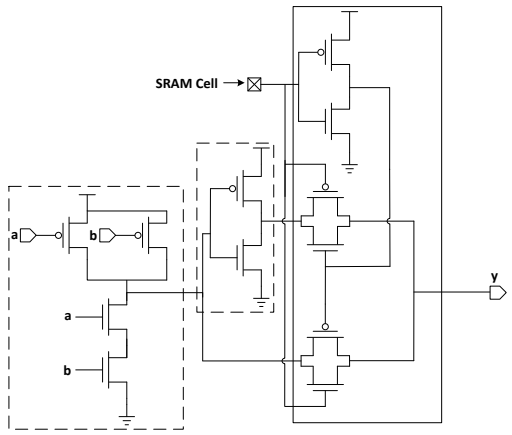

Figure 2: The transistor-level implementation of a BAE.

Table 1: Delay of the AIC, with up to 6 levels, for both the best and worst-case scenarios (ps).

AIC level	Delay of IO path with no inversions	Delay of IO path with all inversions	Average delay	ΔT	σ
AIC6	389.6	577.6	483.6	188	94
AIC5	341.6	504.8	423.2	163.2	81.6
AIC4	274.4	410.4	342.4	136	68
AIC3	215.2	323.2	269.2	108	54
AIC2	148.32	228.8	188.56	80.48	40.24
AIC1	90.64	143.92	117.28	53.28	26.64

AIC cluster with a 40nm standard CMOS technology using the Cadence Virtuoso platform in custom design flow. The delay is measured using Spectre simulator, in typical technology corner. To be able to compare our results with the latest work on AICs [15], we kept the same design environment and simulation parameters. Table 1 shows the simulated delay of up to a 6-level AIC, in the two extreme cases: (1) when no programmable inversion is used in any of the levels and (2) when all the programmable inversions are used in every level, between the input and output. It also shows the difference in delay (ΔT) between the worst and best-case scenarios, as well as the standard deviation.

The results show that, as the number of levels increases, the delay difference between the two extreme cases increases. This is due to the accumulation effect mentioned earlier, which makes the delay discrepancy a critical problem for AICs. For the 6-level AIC, the delay difference can reach 188ps, which means that the delay of the AIC increases by about 50% when all the inversions as added, as opposed to the case when none of the inversions is used. This delay difference is further aggravated in the case of cascaded multi-level AICs.

In combinational circuits, having different arrival times of input signals that must transit simultaneously can result in signal competition and cause signal spikes known as glitches. Such glitches can affect the stability of the circuit or even cause errors. In general, any discrepancy caused by the routing paths can be handled by router's timing analysis. However, since the existing CAD tools do not take into consideration the delay discrepancy due to the logic cell configuration, this problem must be handled separately,

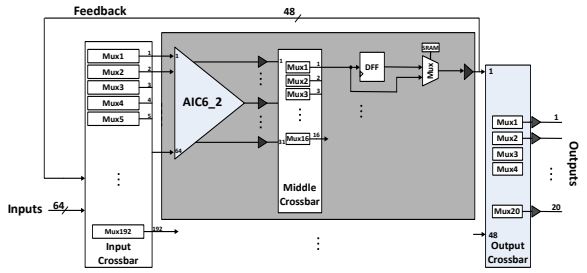

Figure 3: The AIC cluster architecture adopted in the latest AIC paper [15].

otherwise, it will affect the reliability of the AIC-based FPGAs.

2.2 Inefficiency in existing AIC cluster design

The reference AIC cluster architecture [15] is shown in Figure 3. Although the multiple AIC outputs are highly advantageous in reducing the circuit area, managing all these outputs at the cluster level requires some dedicated hardware. For instance, the middle crossbar is a set of 2-level multiplexers, dedicated solely to reducing the number of AIC outputs; it has a substantial contribution to the delay and area of the cluster.

An analysis of this AIC cluster design shows that the crossbars contribute to about 79% of the total cluster area, and about 43% to 70% of the delay, depending on the selected path. This shows that the crossbars have a major effect on the performance of the AIC-based cluster. Thus, any reduction in the size or number of the crossbar multiplexers used can result in a substantial area reduction and efficiency improvement.

3. NAND-NOR CONE

To overcome the limitations of the AICs discussed in Section 2, especially the delay discrepancy problem, we propose a new cone-based logic element called NAND-NOR.

3.1 NAND-NOR Cone Structure

The NAND-NOR is a new logic element with a conic structure, similar to the AIC, but with nodes that can operate either as a NAND or NOR function, as shown in Figure 4. The NAND-NOR has as well an enhanced type of nodes, used at the first level of the cone, to which programmable input inversions are added. According to De Morgan's theorem [8, 5], any logic function expressed as a combination of AND and NAND operations, which is the representation used by AICs, can be transformed into a function of NAND and NOR operations.

Having programmable inversions at the inputs of the Enhanced NAND-NOR Element adds to its flexibility and allows it to implement eight basic functions: AB, $\bar{A}\bar{B}$, $\bar{A}B$, $A\bar{B}$, $A+B$, $\bar{A}+\bar{B}$, $\bar{A}+B$ and $A+\bar{B}$. Thus, despite the difference in design and configuration, the NAND-NOR and AIC can be logically equivalent and the NAND-NOR can replace the AIC without major CAD tool modifications.

3.2 NAND-NOR Design and Optimization

The transistor-level implementation of the NAND-NOR is shown in Figure 5. As mentioned earlier, and shown in Figures 5b and 5c, the NAND-NOR can be configured either as a NAND gate or a NOR gate, respectively. Compared

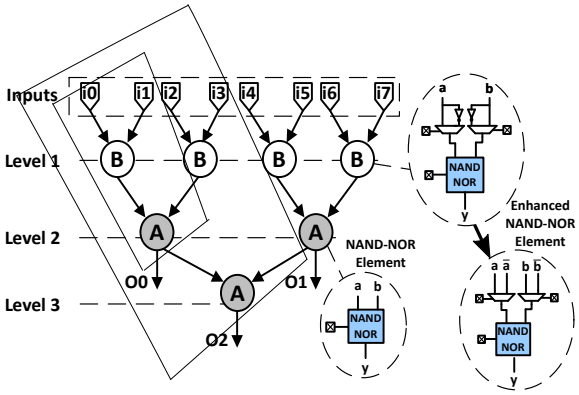

Figure 4: A 3-level NAND-NOR with its different nodes.

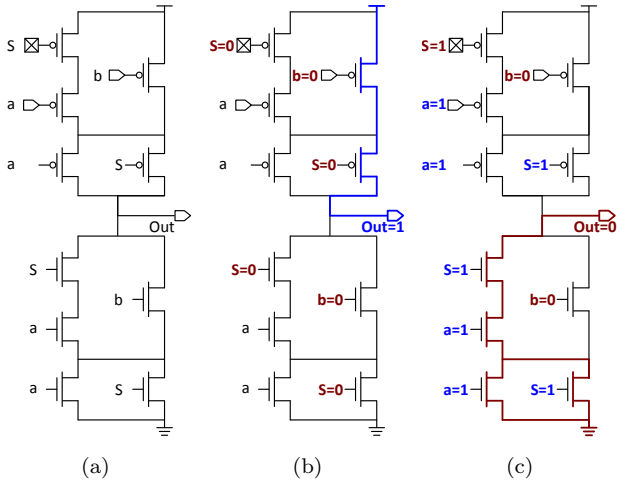

Figure 5: The transistor-level implementation of the NAND-NOR with two configurations: (b) as a NAND gate with input b equals to zero, and (c) as a NOR gate with input a equals to one and b equals to zero.

with the BAE, the NAND-NOR implementation has a simpler structure. The critical path traversed by the signal is reduced from 4 transistors to 2, while the total number of transistors is also reduced from 12 to 10.

We design the NAND-NOR cone circuit using the same technology node and simulate it using the same steps explained in Section 2. Our simulations show that some delay discrepancy exists also between the different NAND-NOR configurations. However this discrepancy can be minimized or even eliminated, theoretically, by optimizing the different transistor sizes. Table 2 shows the post-layout delay results for every level of a 6-level NAND-NOR while temporarily ignoring the input inversions. In this table, the *nor_a* column shows the delay when the path goes through the a input of the NOR gate of every NAND-NOR level. The same logic applies for the remaining columns. ΔT is the maximum delay difference among the different paths and configurations, with respect to the average delay.

When compared with the AIC [15], the NAND-NOR offers between 14% and 46% improvement in average delay, with 43% to 59% reduction in delay discrepancy, and 64% to 71% improvement in delay's standard deviation, for the different levels. When considering a 6-level cone, the NAND-NOR offers also about 23% area reduction.

Table 2: Delay of the different configurations of a 6-level NAND-NOR (ps).

NAND-NOR Level	nor_a	nor_b	nand_a	nand_b	Average Delay	Average Delay Improvement	ΔT	ΔT Improvement	σ	σ Improvement
6	433.6	423.2	372.8	466.4	424	14.06%	51.2	45.53%	33.59	64.27%
5	375.2	372	321.6	404	368.2	14.94%	46.6	42.89%	29.65	63.66%
4	300.8	298.4	259.2	324.8	295.8	15.75%	36.6	46.18%	23.52	65.42%
3	225.6	224.8	196.8	246.4	223.4	20.50%	26.6	50.74%	17.63	67.35%
2	151.36	151.84	135.04	168	151.56	24.41%	16.52	58.95%	11.65	71.04%
1	67.92	84	76.64	92.24	80.2	46.23%	12.28	53.90%	8.98	66.27%

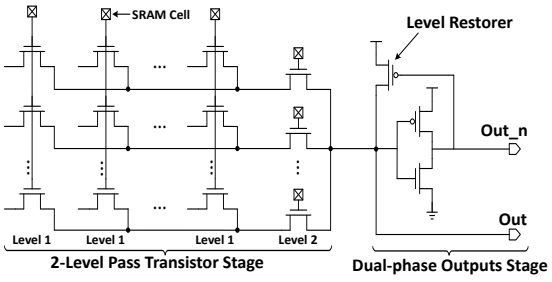

Figure 6: Delay-balanced Dual-phased Multiplexer (DDM).

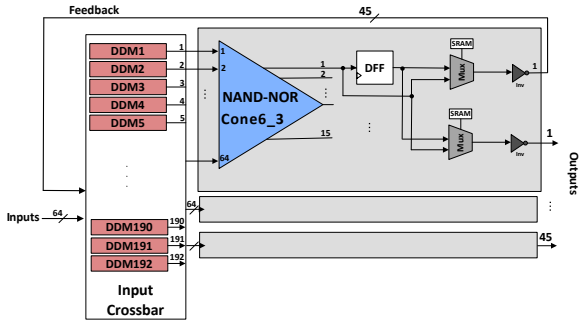

Figure 7: Optimized NAND-NOR cone cluster.

4. DDM CROSSBAR AND CLUSTER OPTIMIZATION

In this section, we try to introduce additional design improvements by balancing the delays of the input signals of the cone through an enhanced input crossbar design. We also try to optimize the overall cluster and improve its inefficiencies through some architectural modifications.

4.1 DDM input crossbar

While testing the NAND-NOR, we noticed that the programmable input inversions for the first level nodes introduce additional delay variations into the design. These cone inputs are delivered using an input crossbar, composed of 192 multiplexers, as shown in Figure 3. Each multiplexer connects only to a single input of the cone, which will then be either inverted or not, depending on the configuration. By that, the load of such multiplexers remains within an acceptable range. Taking all these characteristics into consideration, we suggest using a Delay-balanced Dual-phased Multiplexer (DDM) to deliver the cone inputs.

Figure 6 details the transistor level design of a DDM multiplexer. The 2-level pass-transistor stage is similar to the one used in the Stratix series products [9]. Having the dual-phase outputs stage, we optimize the transistor sizing of the circuit and get both the output and its negation with relatively similar delays.

Having the input crossbar designed as a DDM, the programmable input inversions of the first level nodes of the NAND-NOR cone can be removed, as shown in Figure 4. This results in about 28% area reduction for a 6-level cone, when comparing with the area of the AIC [15]. On top of that, the delay of the crossbar itself is also improved by about 56% due to (i) the removal of the second inverter of the output buffer and (ii) additional transistor optimizations.

4.2 Optimized Logic Cluster

We also try to improve the AIC cluster architecture of Figure 3 by removing the output crossbar and connecting the outputs directly to the connection boxes as the cluster outputs. By doing so, we can reduce the area and delay overhead by limiting the routing flexibility of the outputs.

We also notice that the outputs of the AIC's second level require about half of the cluster resources used to manage the AIC outputs, while being used only to map relatively simple functions. With the help of the new depth-constrained technology-mapping tool [7], we were able to restrict the NAND-NOR's output level to at least the third level, as shown in Figure 7. In this case, the 6-level cone, labeled *cone6_3*, has its outputs generated from level 3 to level 6. This reduces the total number of outputs (of the three cones) from 93 to 45, allowing us to remove the middle crossbar of the cluster.

In addition to that, the existing cluster design uses a single 2-to-1 multiplexer to select between the combinational and sequential versions of each output, for both the feedback and direct output signals. However, such a design can be rather inefficient since, whenever an application requires the same signal in both its combinational and sequential versions, the function generating the output is usually duplicated. To avoid such situations, we add a second multiplexer, as shown in Figure 7, to split between the signal used for feedback and the one used for direct output, allowing for better flexibility.

All the modifications to the cluster, resulting in an optimized NAND-NOR-based architecture, are depicted in Figure 7. The new cluster includes a half populated DDM input crossbar and three NAND-NOR Cone6_3, with outputs starting at the third level.

Figure 8 shows the delay and area comparison between the existing AIC-based cluster architecture [15], a NAND-NOR-based cluster but with a regular input crossbar (labeled *NAND-NOR only*), and the cluster design of Figure 7 with both the NAND-NOR and a DDM input crossbar. The results show that our optimized logic cluster is the most optimal in terms of both delay and area, among the three designs. When compared with the existing AIC cluster [15], our optimized NAND-NOR cluster results in a 27% area re-

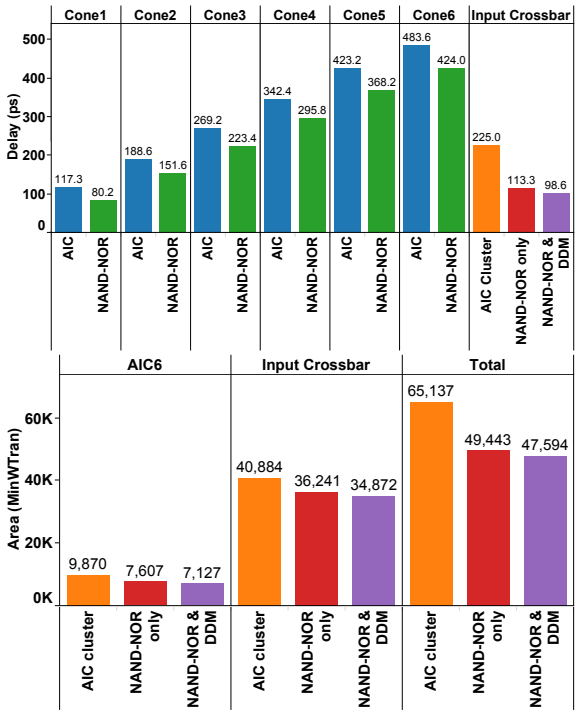

Figure 8: Area and delay comparison of cluster architectures. Note: AIC cluster refers to the reference AIC architecture [15] shown in Figure 3.

duction. Although the delay improvements vary depending on the selected cone configuration, we show that, on average, the NAND-NOR and DDM crossbar are the most delay efficient.

5. EXPERIMENTS AND RESULTS

Now that the cluster is designed and optimized, we test it on a set of benchmarks and compare it to the state-of-the-art architectures.

5.1 Experimental Setup

To test the efficiency of our NAND-NOR cluster, we run the experiments using the VTR flow [10]. However, we replace the default technology mapper, ABC, with the depth-constrained AIC mapper [7]. In fact, since the architectures of the NAND-NOR and AIC are similar and their functionalities can be logically equivalent, we were able to directly use the same netlist generated by the AIC technology mapper [7] and just change the configuration of the cones to support the NAND-NOR. In both cases, the mapper uses the side outputs of the cones.

The overall CAD flow used in the experimental setup is detailed in Figure 9. The area and delay parameters derived in the simulations of Section 4.2 are added to the architecture file, so that they can be used in the packing and routing steps. The other architectural parameters are set to be identical to the reference architecture [15] for a better comparison; more specifically, we use unidirectional wires, a segment length of 4, Wilton switch boxes, and a variable channel width. We also define F_s, Fc_{in} and Fc_{out} to be 3, 0.15, and 0.1 respectively. We select the big 20 MCNC

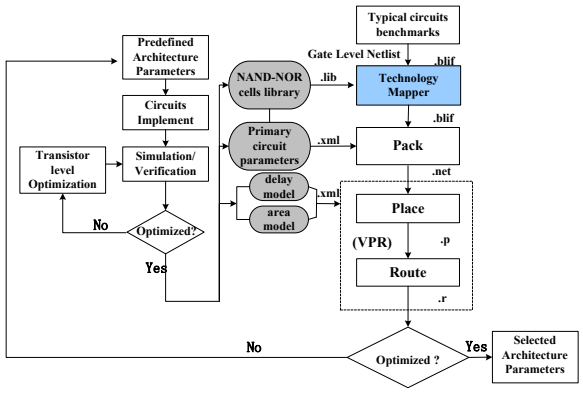

Figure 9: Experimental CAD flow.

benchmarks and the VTR benchmarks [14] that finish within a reasonable time, to test the architecture.

5.2 Results

The selected benchmarks were tested on both the reference AIC architecture and the NAND-NOR architecture. Table 3 shows the delay and area improvement for the MCNC and VTR benchmarks when tested on the NAND-NOR-based FPGA with DDM input crossbar, compared with the LUT-based Stratix IV-like architecture [15] and the reference AIC-based architecture [15]. It also lists the number of clusters needed to implement the circuits with each architecture.

Although, generally, the NAND-NOR architecture does not reduce the number of used clusters, the cluster itself is smaller, as shown in Section 4, which results in an average total area reduction. When compared to the LUT-based FPGA, the delay-area product of the NAND-NOR-based architecture shows 44% and 21% improvement, for the MCNC and VTR benchmarks, respectively. And when compared to the AIC-based architecture, the delay-area product is improved by 31% and 19%, for the MCNC and VTR benchmarks, respectively.

The NAND-NOR architecture can bring worse results in one of these two cases: (i) the delay is compromised for the sake of area improvement (or vice versa) which is mainly decided by the packing and routing algorithms like, for example, for the *steriovision3* benchmark; and (ii) both the delay and area are worse than the LUT-based architecture, which happens in the case of *steriovision1*, a known case for which the technology mapper performs poorly [7].

6. CONCLUSIONS

Several AIC-based FPGA architectures have been previously explored; however, none of these explorations considered the delay discrepancy problem which is particularly important in conic structures like the AIC, due to the multi-level accumulation effect.

An improved and delay balanced logic cone was introduced along with an optimized FPGA cluster architecture, with a DDM input crossbar. With these modifications, we were able to significantly reduce the delay discrepancy, improve the system stability, and provide significant area and delay reduction.

When compared to the AIC structure, the optimized NAND-NOR cone provides about 14% to 46% delay improvement, depending on the number of levels, while the

Table 3: Area, delay and number of clusters for the NAND-NOR architecture vs. the Stratix IV and the AIC architectures.

MCNC Benchmark	Delay Reduction (%) w.r.t. LUT	Delay Reduction (%) w.r.t. AIC	Area Reduction (%) w.r.t. LUT	Area Reduction (%) w.r.t. AIC	Number of Clusters LUT	Number of Clusters AIC	Number of Clusters NAND-NOR
s298	23.45	7.57	50.85	28.27	52	43	49
pdc	30.19	19.94	30.17	27.82	193	151	153
diffeq	29	5.45	18.42	4.72	58	40	48
alu4	32.79	5.62	24.65	-0.14	62	44	51
misex3	31.94	17.7	34.71	13.47	56	48	47
apex2	22.31	14.74	53.49	32.9	80	76	74
seq	32.81	19.02	42.37	21.69	76	62	61
s38417	-23.39	9.45	39.09	36	222	278	315
bigkey	10.4	4.57	44.77	23.37	80	66	75
s385841	12.93	-2.64	26.55	16.94	198	152	200
apex4	18.51	16.92	37.45	14.52	57	56	55
tseng	3.45	-8.8	8.16	23.94	50	41	55
ex1010	3.06	20.17	44.14	23.25	212	236	235
elliptic	26.45	5.93	27.96	19.12	140	93	112
dsip	18.69	10.97	48.65	22.69	83	66	76
clma	-1.42	26.71	30.18	24.2	285	381	345
spla	-8.41	2.1	27.6	20.26	141	155	175
des	-13.67	26.09	24.33	-12.16	69	119	111
frisc	27.15	4.91	24.28	23.89	155	106	130
ex5p	-32.9	38.03	33.83	30.98	47	102	73
Geomean	14.07	12.92	34.56	20.5	98	92	98

VTR Benchmark	Delay Reduction (%) w.r.t. LUT	Delay Reduction (%) w.r.t. AIC	Area Reduction (%) w.r.t. LUT	Area Reduction (%) w.r.t. AIC	Number of Clusters LUT	Number of Clusters AIC	Number of Clusters NAND-NOR
stereovision0	-40.41	6.45	17.98	17.84	1041	1417	1554
ch_intrinsics	28.84	4.36	40.98	13.84	16	13	13
diffeq1	0.01	5.53	10.45	4.88	37	55	67
mkSMAdapter4B	22.48	11.56	24.57	13.65	114	91	111
boundtop	10.89	14.18	32.3	23.39	172	170	182
or1200	0.9	4.06	-16.15	18.56	205	198	243
stereovision1	-48.47	3.67	-19.82	3.5	986	1495	1693
mkDelayWorker32B	44.84	-0.94	43.49	15.04	55	39	39
raygentop	-0.36	9.83	21.17	20.3	158	194	215
stereovision3	-6.18	-24.91	39.8	15.06	15	12	18
diffeq2	-2.36	7.1	1.36	9.52	25	55	64
mkPktMerge	41.39	2.66	2.58	5.41	11	11	11
sha	-37.7	10.61	21.11	26.09	157	232	261
blob_merge	-29.91	2.85	4.28	18.72	387	481	586
Geomean	3.13	4.44	18.21	14.96	96	108	123

NAND-NOR cluster offers about 27% area reduction. The new architecture was tested on the MCNC and VTR benchmarks and showed a delay-area product improvement of 44% and 21%, respectively, when compared with the state-of-the-art LUT-based architecture, and 31% and 19% when compared with the reference AIC-based architecture.

7. REFERENCES

[1] E. Ahmed and J. Rose. The effect of LUT and cluster size on deep-submicron FPGA performance and density. *IEEE Transactions on Very Large Scale Integration Systems*, 12(3):288–298, 2004.

[2] V. Betz, J. Rose, and A. Marquardt. *Architecture and CAD for deep-submicron FPGAs*. Kluwer Academic, Boston, Mass., 1999.

[3] R. K. Brayton and A. Mishchenko. ABC: An academic industrial-strength verification tool. In *Proceedings of the International Conference on Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 24–40. Springer, July 2010.

[4] D. Chinnery and K. Keutzer. *Closing the Gap Between ASIC & Custom: Tools and Techniques for High-Performance ASIC Design*. Springer US, New York, NY., 2002.

[5] P. Hurley. *A Concise Introduction to Logic*. Wadsworth, 2011.

[6] M. Hutton, J. Schleicher, D. Lewis, B. Pedersen, R. Yuan, S. Kaptanoglu, G. Baeckler, B. Ratchev, K. Padalia, and M. Bourgeault. Improving FPGA performance and area using an adaptive logic module. In *Proceedings Field-Programmable Logic and Applications*, pages 135–144. Springer, 2004.

[7] Z. Jiang, G. Zgheib, C. Yu Lin, D. Novo, L. Yang, Z. Huang, H. Yang, and P. Ienne. A technology mapper for depth-constrained FPGA logic cells. In *Proceedings of the 25th International Conference on Field-Programmable Logic and Applications*, pages 1–8, London, Sept. 2015.

[8] K. Levitz and H. Levitz. *Logic and Boolean algebra*. Barron's Educational Series, 1979.

[9] D. Lewis et al. The Stratix II logic and routing architecture. In *Proceedings of the 13th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 14–20, Monterey, Calif., Feb. 2005.

[10] J. Luu, J. Goeders, M. Wainberg, A. Somerville, T. Yu, K. Nasartschuk, M. Nasr, S. Wang, T. Liu, N. Ahmed, K. B. Kent, J. Anderson, J. Rose, and V. Betz. VTR 7.0: Next generation architecture and CAD system for FPGAs. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 7(2):6:1–6:30, June 2014.

[11] F. Mayer-Lindenberg. Design and application of a scalable embedded systems? Architecture with an FPGA based operating infrastucture. In *9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools.*, pages 189–196, 2006.

[12] H. Parandeh-Afshar, H. Benbihi, D. Novo, and P. Ienne. Rethinking FPGAs: Elude the flexibility excess of LUTs with And-Inverter Cones. In *Proceedings of the 20th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 119–28, Monterey, Calif., Feb. 2012.

[13] H. Parandeh-Afshar, G. Zgheib, D. Novo, M. Purnaprajna, and P. Ienne. Shadow And-Inverter Cones. In *Proceedings of the 23rd International Conference on Field-Programmable Logic and Applications*, pages 1–4, Porto, Portugal, Sept. 2013.

[14] J. Rose, J. Luu, C. W. Yu, O. Densmore, J. Goeders, A. Somerville, K. B. Kent, P. Jamieson, and J. Anderson. The VTR project: architecture and CAD for FPGAs from Verilog to routing. In *Proceedings of the 20th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 77–86, 2012.

[15] G. Zgheib, L. Yang, Z. Huang, D. Novo Bruna, H. Parandeh-Afshar, H. Yang, and P. Ienne. Revisiting And-Inverter Cones. In *Proceedings of the 22nd ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 45–54, Monterey, Calif., Feb. 2014.