International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

Design of Hybrid LUT/MUX Based Configurable Logic Architectures for FPGAs

M. Dilip Kumar & G. Ramesh
[1]M.Tech, (Ph.D) PG Scholar, Dept of ECE, G. Pulla Reddy Engineering College(Autonomous), Kurnool, Andhra Pradesh, India
[2]Assistant Professor, Dept of ECE, G. Pulla Reddy Engineering College(Autonomous), Kurnool, Andhra Pradesh, India
dil.dilip555@gmail.com & rameshece007@gmail.com

ABSTRACT:

New types of logic block architectures for FPGAs are designed and evaluated using Verilog to Routing (VTR) tool. Two types of architectures namely Nonfracturable and fracturable are designed. A new logic element (MUX4) is used along with conventional LUT in the proposed logic block architectures. Fracturable logic elements have achieved improved logic density. MUX4 logic element along with LUT has reduced the area consumption as compared to conventional logic block architectures. The architectures are evaluated by implementing benchmark circuits on to them by VTR tool. A new CAD flow has been generated to map the circuits on to the proposed logic structures. The usage of 50% depopulated interconnect structure inside the logic cluster has provided the area savings up to 15%. The proposed non fracturable architectures have provided the area savings up to 5% and fracturable architectures have achieved 2% area savings.

Keywords – Depopulated interconnect structure, fracturable architectures, logic density, MUX4 logic element, VTR tool

I. INTRODUCTION

A logic block is responsible in implementing the functionality of an FPGA for which it is programmed. Lookup tables are the primary logic elements used in majority of FPGAs. Xilinx uses LUT based logic blocks. A K-input LUT is capable of implementing any K- input Boolean function. But their area increases exponentially with their size. Alternative logic elements are necessary in order to solve the problem. ACTEL uses MUX based logic blocks. They are area efficient than LUTs but the process of technology mapping a circuit on to them is complex. They are not as flexible as LUTs in implementing wide range of functions. Thus both type logic elements have their merits and demerits. By using both type of logic elements in a single cluster one can both of their advantages.

VTR tool has been used to design and evaluate the proposed hybrid logic block architectures and a new CAD flow has been designed to map logic functions on to proposed logic elements. The performance evaluation has been done using benchmark circuits.

II.EXISTING SYSTEM

i. LUT based logic blocks

A LUT is a logic gate that contains storage cells to implement logic functions. A k-LUT is direct implementation of a function truth table. Each storage cell holds a single logic value and the storage value is produced as the output of the storage cell. A k- input LUT can implement any k-input Boolean function. It contains 2^k storage cells and 2^k-1 2-to-1 multiplexers. It can implement 2^2^K different possible number of logic functions.

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

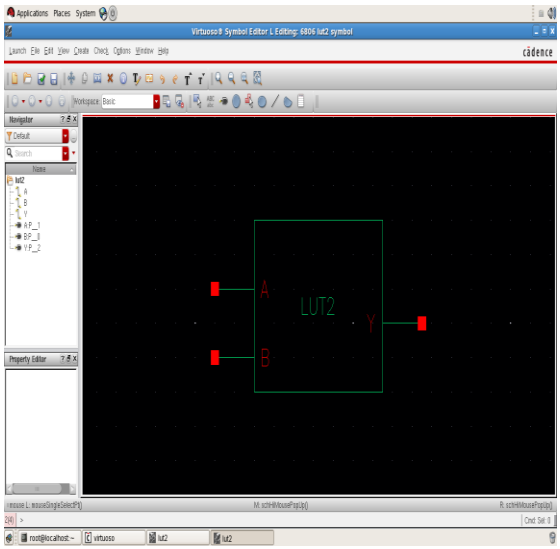

Fig.1 A two input LUT

The size of a LUT is determined by its number of its inputs. As the LUT size increases, the number of components increases and the area increases exponentially.

LUT size	SRAM cell count	MUX count	Transistor count
1	2	1	18
2	4	3	42
3	8	7	90

Table.1 Exponential area increase with LUT size

LUTs are inefficient in multiplexers which are one of the major digital circuits. A 4x1 multiplexer is implemented by a six input LUT which contains 64 SRAM cells and 63 2x1 multiplexers. Thus it requires large number of components to implement a multiplexer which results in higher area cost. Due to above disadvantages, the logic designers had started thinking about alternate logic elements to LUTs.

ii. MUX based logic blocks

ACTEL FPGAs uses multiplexer based logic blocks. Consider ACT1 logic module with eight inputs and one output. It contains three 2-to-1 multiplexers and a two input OR gate. It can implement limited number of logic functions.

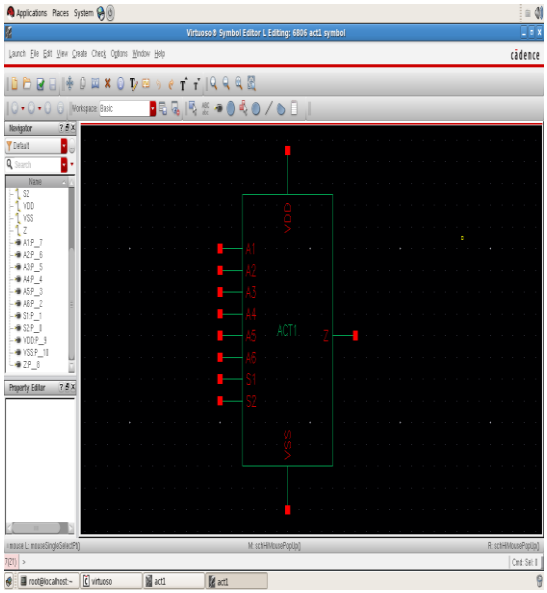

Fig.2 ACT1 logic module

It can implement any two and three input functions. It contains less number of components hence it consumes less area. The process of technology mapping a function on to it involves Shannon decomposition. As a result the compilation time increases. On the other hand it cannot implement wide range of functions as a LUT.

Thus by using LUT based and mux based logic elements within a single cluster both of their advantages can be obtained. The LUTs provide the flexibility and the muxes provides area savings. The mux based logic elements are dedicated as hard logic elements. The heterogeneous technology mapping has been employed in this work.

III.PROPOSED SYSTEM

The proposed logic block architectures contain the following logic elements

1. Non fracturable architectures

- LUT6
- MUX4 logic element

i. Six input LUT

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

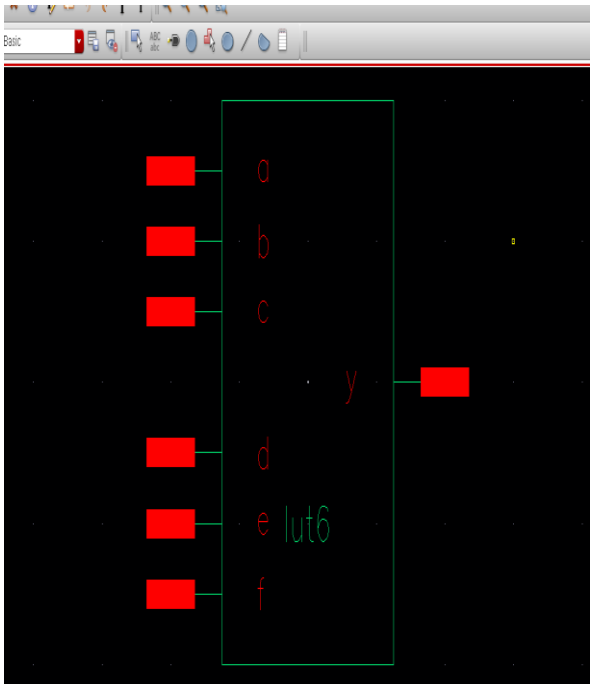

Fig.3 six input LUT

It can implement any six input logic function.

ii. MUX4 logic element

it is a six input logic element in compatible with six input LUT. It contains optional inversion on its data inputs. It can implement the following functions

- Any two and three input functions
- Some four and five input functions
- One six input function

Through Shannon decomposition a function can be implemented by it. for a two input function, the two inputs are to be given to the select inputs and the truth table values are given to the data inputs. This can also be achieved by performing Shannon decomposition about one of the two variables and it is fed to the select input and the cofactors are given to the data inputs. for three input functions, Shannon decomposition about one variable produces the cofactors with two variables.

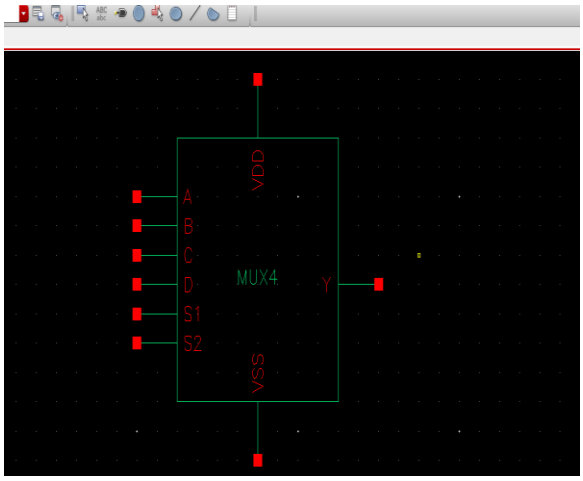

Fig.4 MUX4 logic element

Table.2 Area comparison of MUX4 logic element with LUT6

It contains four SRAM cells, four inverters and seven 2x1 multiplexers. It involves less number of components than a LUT6 hence it consumes less area.

Thus the MUX4 consumes approximately 10% of LUT6 area. By using such logic element along with LUT6 high area efficiency can be achieved.

2. Fracturable architectures

- Fracturable LUT6
- Dual MUX4 logic element

i. Fracturable LUT6

It is an eight input and two output logic element by splitting LUT6 into two five input LUTs with two shared inputs. Fracturable logic elements can map one or more logic functions optionally. The ability of splitting LUT6 into two LUT5 elements improves the logic density.

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

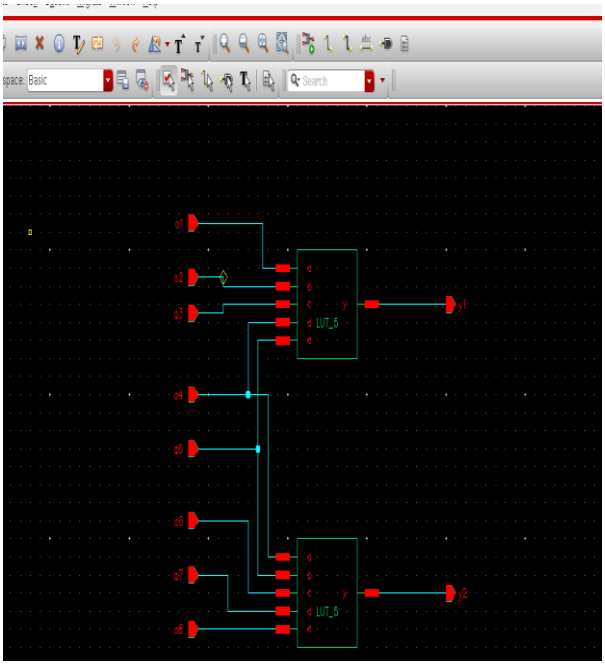

Fig.5 Fracturable LUT6

It can implement two five input functions with two inputs sharing. It can implement independent four input functions with no inputs sharing. While implementing two five input functions only one function can be mapped at a time. By using such logic structures more logic can be implemented with less number of components. By varying the number of shared inputs the logic density can be improved.

Component	SRAM cells	2x1 MUX	Inverters	Transistor count
MUX4	4	7	4	74
LUT6	64	63	-	762

ii. Dual MUX4 logic element

It is the variant of MUX4 logic element. It contains eight inputs and two outputs. It matches with the pin count of fracturable LUT6. It contains two MUX4 logic elements with dedicated select inputs and shared data inputs. It can implement two independent three input functions. As it is not a reprogrammable structure, all the MUX4 inputs are shared providing the fracturability factor as zero.

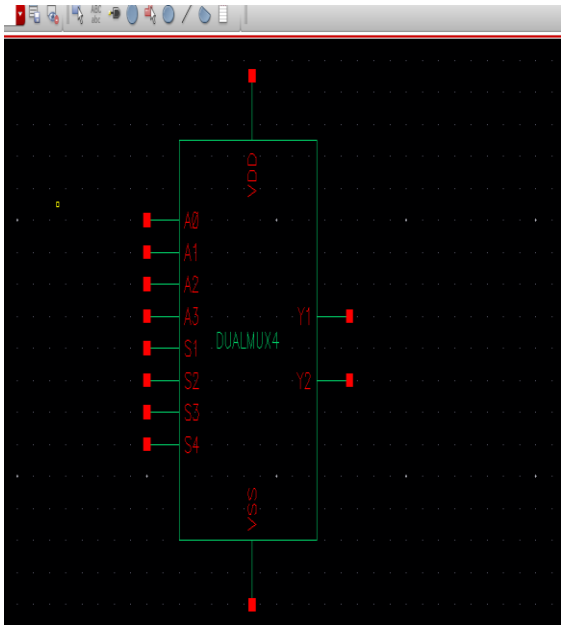

Fig.6 Dual MUX4 logic element

It can implement larger functions depending on the shared inputs.

Hybrid logic block architectures

i. Non fracturable architecture

This logic block has 40 inputs and 10 outputs. It contains ten Basic Logic Elements (BLE) with six inputs and one output each.

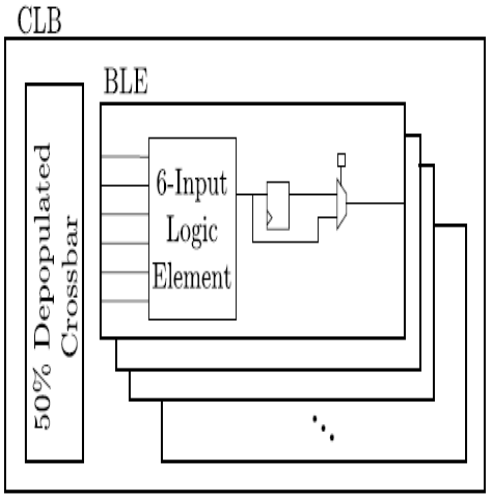

Fig.7 Non fracturable architecture

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

ii. Fracturable architecture

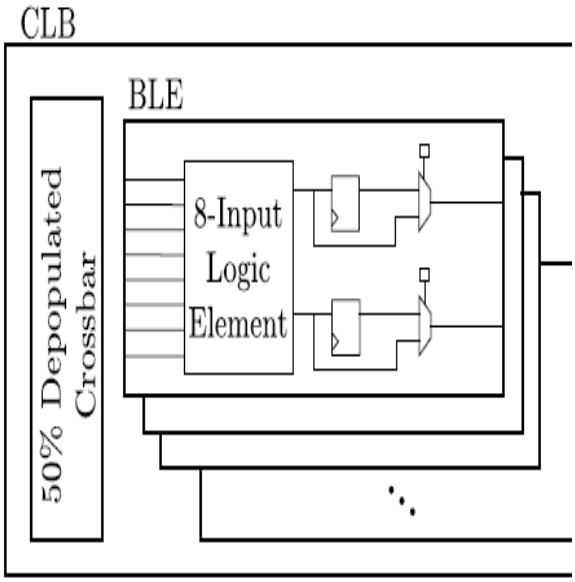

Fig.8 Fracturable architecture

This logic block has 80 inputs and 20 outputs. It contains ten BLEs with eight inputs and two outputs each. The ratio of MUX4 and LUT6 based logic elements is varied from 1:9 to 5:5 within the ten element logic blocks. The BLEs in both the architectures are interconnected in a 50% depopulated crossbar structure.

IV.VERILOG TO ROUTING (VTR) CAD TOOL

It is an open source CAD tool for FPGA research and development. The proposed logic block architectures are developed and evaluated using this tool. It takes an FPGA architecture description file and a Verilog description of a benchmark circuit as input and then implements the circuit on to the proposed FPGA architecture and generates delay and area results. It contains the following set of tools.

- Odin-II (synthesis)
- ABC (Logic optimization & Technology mapping)
- VPR (Packing, Placement and Routing)

The FPGA architectures are described in XML based VPR architectural description language. The architecture description script requires all the FPGA architecture parameters. The proposed logic block architectures are compared with LUT only architectures (both non fracturable and fracturable architectures).

V.EXPERIMENTAL EVALUATION

The proposed architectures are evaluated by implementing the VTR benchmark circuits on to them. The five of the benchmark circuits used here belong to different application area like image processing, mathematics, cryptography and computer vision.

i. Improved Logic Density

The fracturability of logic elements provides the improved logic density than the conventional logic elements. It allows the logic elements to implement more than one function within a logic element. Depending on the number of inputs shared independent functions can be implemented within a single logic element. Consider a benchmark circuit being implemented on a non fracturable architecture and a fracturable architecture. Fracturable architectures take less number of CLBs to implement a given circuit than non fracturable architectures as shown in the below table. Thus the number of logic blocks required to implement the same logic is less in fracturable architectures providing the improved logic density. The below considered fracturable architecture has logic element with all inputs shared between two LUT5 logic elements. The logic density can be further increased by reducing the number of shared inputs which facilitates to form a larger logic element and implementing larger independent functions. By reducing the number of shared inputs to two (FI = 2) a eight input logic element from a fracturable LUT6. Fracturable architecture with such logic element takes less number of CLBs than baseline fracturable architecture.

Architecture	Number of CLBs
Non fracturable	407
Fracturable	307
Improved fracturable	292

Table.3 Improved logic density due to fracturable logic elements

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 10
September 2017

Thus by improving the fracturability factor the number of CLBs required to implement a circuit will be reduced providing the improved logic density.

ii. Full crossbar vs. 50% depopulated crossbar

The local interconnect network in a cluster is fully connected like a full crossbar. It provides high degree of connectivity making the routing easy. But it consumes much area because higher connectivity requires more number of interconnecting switches and multiplexers. This area overhead is reduced by replacing the full crossbar with 50% depopulated crossbar in the proposed architectures. Consider the area comparison between full cross bar based architecture and depopulated crossbar based architecture.

Architecture	Routing area (MWTA)
Full cross bar based	116091
50% depopulated cross bar based	939341

Table.5 Area savings of depopulated crossbar

Thus the depopulated crossbar like interconnect within the cluster provided the area savings up to 15%.

iii. Area Comparison between Baseline and Proposed Architectures

A. Non fracturable architectures

The architectural evaluation has been done using five of the VTR benchmark circuits.

Benchmark circuits	Baseline Area	Arch 1:9 (% area)	Arch 2:8 (% area)	Arch 3:7 (% area)	Arch 4:6 (% area)	Arch 5:5 (% area)
Diffeq1	5.9124e	99.4	98.5	97.6	103.2	125.4
Ch_intrinsics	5.1323e	97.0	94.4	96.0	108.0	119.3
Sha	2.7212e	98.7	97.2	97.2	106.2	125.3
Blob_merge	8.1869e	97.9	102.0	107.3	120.6	132.6
Stereovision3	2.3866e	98.6	96.0	87.1	92.8	94.0

Table.6 Area comparison of baseline non fracturable architecture with proposed non fracturable architecture

B. Fracturable architectures

Thus the proposed arch 1:9 architecture provides the maximum area savings up to 5% in non fracturable and up to 2% in fracturable architectures.

Benchmark Circuits	Baseline Area	Arch 1:9 (% area)	Arch 2:8 (% area)	Arch 3:7 (% area)	Arch 4:6 (% area)	Arch 5:5 (% area)
Diffeq1	3.4947e	100	98	110	120	131
Ch_intrinsics	4.7517e	98.0	99	106	121	149
Sha	2.7398e	101	105	105	132	139
Blob_merge	8.3306e	105	112	120	130	150
Stereovision3	0.94913e	99	99	105	99	106

Table.6 Area comparison of baseline fracturable architecture with proposed fracturable architecture

VI.CONCLUSION

Hybrid logic block architectures that contain MUX4 and LUT6 are designed and evaluated using VTR benchmarks. The area efficiency is achieved by the usage of MUX4 logic element along with LUT6 in different ratios. Fracturable architectures have improved the logic density. The proposed non fracturable architectures provided the area savings up to ~5% and the fracturable architectures provide the area savings up to ~2%.

REFERENCES

[1] E. Ahmed and J. Rose, "The effect of LUT and Cluster size on deep submicron FPGA performance and density," IEEE Trans Very Large Scale Integration. (VLSI), vol. 12, no. 3, pp. 288–298, Mar. 2004.

[2] M. Purnaprajna and P. Ienne, "A case for heterogeneous technology mapping: Soft versus hard multiplexers," in Proc. IEEE 21st Annual. Int. Symposium. FCCM, Apr. 2013, pp. 53–56.

[3] S. A. Chin and J. H. Anderson, "A case for hardened multiplexers in FPGAs," in Proc. FPT, Dec. 2013, pp. 42–49.

[4] G. Lemieux and D. Lewis, " Using sparse crossbars within LUT," in Proc. 9th Int. Symposium. ACM/SIGDA FPGA, 2001, pp. 59–68.

[5] J. Rose et al., "The VTR project: Architecture and CAD for FPGAs from Verilog to routing," in Proc. ACM/SIGDA FPGA, 2012, pp. 77–86.

[6] J. Luu and J. Anderson, "Architecture description and packing for logic blocks with hierarchy, modes and complex interconnect" Department of Electrical and computing engineering, University of Toronto, Canada.

[7] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in Proc. 7th Int. Workshop *FPL*, 1997, pp. 213–222.

[8] Peter Jamieson, Lesley Shannon, "Odin-II – An open source CAD Tool for Verilog HDL synthesis Tool for CAD research" Department of Electrical and Computing Engineering, Miami University.

[9] Ana Petkovaska, "Getting started with ABC: A System for Sequential synthesis and verification".

[10] J. Anderson and Q. Wang, "Improving logic density through synthesis inspired architecture," in Proc. IEEE FPL, Aug./Sep. 2009, pp. 105–111.

[11] J. Anderson and Q. Wang, "Area-efficient FPGA logic elements: Architecture and synthesis," in Proc. ASP DAC, 2011, pp. 369–375.

[12] A. Mishchenko, S. Chatterjee, and R. Brayton, "DAG-aware AIG rewriting a fresh look at combinational logic synthesis," in Proc. 43rd Annual. *DAC*, 2006, pp. 532–535.

[13] C. Chiasson and V. Betz, "COFFE: Fully-automated transistor sizing for FPGAs," in Proc. Int. Conf. FPT, Dec. 2013, pp. 34–41.

[14] Predictive Technology Model. [online]. Available: http://ptm.asu.edu/, accessed 2015.

[15] Altera, private communication, Mar. 2014.

[16] A. Mishchenko, S. Cho, S. Chatterjee, and R. Brayton, "Combinational and sequential mapping with priority cuts," in Proc. IEEE/ACM Int. Conf. ICCAD, Nov. 2007, pp. 354–361.

[17] Virtex-6 FPGA User Guide. [online]. Available: http://www.xilinx.com

[18] VTR 7.0 user guide. January 2016.

[19] Robert J. Francis, "Technology mapping for LUT based FPGAs" Department of Electrical and computing engineering, University of Toronto, Canada.

[20] David Dickin, Lesley Shannon, "Exploring FPGA technology mapping for fracturable LUT minimization" School of engineering science, Simon Fraser University, Canada.