DESIGN AND IMPLEMENTATION OF HYBRID LUT/MULTIPLEXER FPGA LOGIC ARCHITECTURES USING VERILOG HDL

M. Sravani, Dept. of ECE, SITAMS, Chittoor, Email: medikurthi.sravani@gmail.com
Dr. S. Vijay Kumar, Associate Professor, Dept. of ECE, SITAMS, Chittoor, Email: vijaysuresh1975@gmail.com

Abstract:-- Hybrid configurable logic block architectures for field-programmable gate arrays that contain a mixture of lookup tables and hardened multiplexers are evaluated toward the goal of higher logic density and area reduction. Multiple hybrid configurable logic block architectures, both non-fracturable and fracturable with varying MUX:LUT logic element ratios are evaluated across two benchmark suites (VTR and CHStone) using a custom tool flow consisting of LegUp-HLS, Odin-II front-end synthesis, ABC logic synthesis and technology mapping, and VPR for packing, placement, routing, and architecture exploration. Experimentally, we show that for non-fracturable architectures, without any mapper optimizations, we naturally save area post place and route; both accounting for complex logic block and routing area while maintaining mapping depth. With architecture-aware technology mapper optimizations in ABC, additional area is saved, post-place-and-route. For both non-fracturable and fracturable architectures, we see minimal impact on timing performance for the architectures with best area-efficiency.

Index Terms: Hybrid complex logic block, Multiplexer (MUX).

I. INTRODUCTION

A field-programmable gate array (FPGA) is a block of programmable logic that can actualize multi-level logic functions. FPGAs are most regularly utilized as partitioned item chips that can be customized to actualize vast functions. Be that as it may, little blocks of FPGA logic can be valuable segments on-chip to enable the users of the chip to customize some portion of the chip's logical function. A FPGA block should execute both combinational logic functions and interconnect to have the function to build multi-level logic functions. There are a few unique advances for programming FPGAs, yet most logic forms are probably not going to execute antifuses or comparative hard programming technologies. All through the historical backdrop of field-programmable gate arrays (FPGAs), lookup tables (LUTs) have been the essential logic element (LE) used to acknowledge combinational logic. A K-input LUT is non-specific and exceptionally adaptable ready to actualize any K-input Boolean function. The utilization of LUTs improves technology mapping as the issue is decreased to a chart covering issue. In any case, an exponential territory cost is paid as bigger LUTs are considered. The estimation of K in the vicinity of 4 and 6 is commonly found in industry and the scholarly community, and this range has been exhibited to offer a decent area/execution trade off. As of late, various different works have investigated elective FPGA LE structures for execution change to close the huge hole amongst FPGAs and application-specific integrated circuits (ASICs).

LOOKUP TABLES

The essential strategy used to fabricate a combinational logic block (CLB) additionally called a logic element in a SRAM-based FPGA is the lookup table (LUT). As appeared in Figure, the lookup table is a SRAM that is utilized to implement a truth table. Each address in the SRAM represents a combination of inputs to the logic element. The value stored at that
address represents the value of the function for that input combination. An n-input function requires an SRAM with locations.

Since an essential SRAM isn't clocked, the lookup table logic element works much as some other logic gate as its inputs changes, its output changes after some delay.

![Fig.1: Lookup Table.](image1)

**PROGRAMMING A LOOKUP TABLE**

Unlike a typical logic gate, the function represented by the logic element can be changed by changing the values of the bits stored in the SRAM. Subsequently, the n-input logic element can speak to functions (however a portion of these functions are changes of each other).

![Fig.2: Programming A Lookup Table](image2)

A regular logic element has four data inputs. The delay through the lookup table is free of the bits stored in the SRAM, so the delay through the logic element is the same for all functions. This means, for instance, a lookup table-based logic element will display a similar delay for a 4-input XOR and a 4-input NAND. Obviously, the static logic gate is faster than the logic element. Logic elements for the most part contain registers, flip-flops, latches and in addition combinational logic.

A flip-flop or latch is small compared with the combinational logic element (in sharp difference to the circumstance in custom VLSI), so it bodes well to add it to the combinational logic element. Utilizing a different cell for the memory element would basically take up routing resources. The memory element is associated with the output; regardless of whether it stores a given value is controlled by its clock and enable inputs.

In this paper, we propose fusing (a few) solidified multiplexers (MUXs) in the FPGA logic blocks as a methods for expanding silicon region efficiency and logic density. However, their utilization in commercial chips has wound down, maybe halfway because of the simplicity with which logic functions can be mapped into LUTs. Nevertheless, it is generally comprehended that the LUTs are inefficient at executing MUXs, and that MUXs are every now and again utilized as a part of logic circuits. To underscore the inefficiency of LUTs executing MUXs, look at that as a six input LUT (6-LUT) is basically a 64-to-1 MUX (to choose 1 of 64 truth-table columns) and 64-SRAM design cells, yet it can just realize a 4-to1 MUX (4 data+2 select=6 inputs). In this paper, we display a six-input LE in view of a 4-to-1 MUX, MUX4, that can understand a subset of six input Boolean logic functions, and new hybrid complex logic block (CLB) that contains a mixture of MUX4s and 6-LUTs. The proposed MUX4s are small with a 6-LUT (15% of 6-LUT area), and can effectively map {2,3}-input functions and some {4,5,6}-input functions. Moreover, we investigate fracturability of Les the function to part the LEs into small elements in both
LUTs and MUX4s to expand logic density. The proportion of LEs that ought to be LUTs versus MUX4s is additionally investigated towards improving logic density for both nonfracturable and fracturable FPGA designs. To encourage the architecture investigation, we built up a CAD stream for mapping into the proposed hybrid CLBs, made utilizing ABC and VPR, and depict technology mapping methods that empower the choice of logic works that can be implanted into the MUX4 elements. The principle commitments in this paper are as per the following.

1) Two hybrid CLB structures (nonfracturable and fracturable) that contain a mixture of MUX4 LEs and the conventional LUTs outputting up to 8% area savings.

2) Mapping methods called Natural Mux and MuxMap focused toward the hybrid CLB design that upgrade for area, while saving the first mapping depth.

3) A full post-place-and-route architecture assessment with VTR7, and CHStone benchmarks encouraged by LegUp-HLS, the Verilog-to-Routing venture indicating sway on both area and delay.

Compared with the preliminary publication, we have performed transistor level modeling of the MUX4 LE, additionally contemplated the fracturable designs, and brought together the open source tool flow from C through LegUp-HLS to the VTR stream. Meager crossbars (versus full crossbars in the past work) have additionally been incorporated into our CLBs, expanding modeling accuracy. The new transistor-level modeling of the MUX4 additionally furnishes more exact outcomes as contrasted and the past work. Results have additionally been extended with the incorporation of timing comes about and also bigger structural proportion clears.

II. Literature Review

Ongoing works have demonstrated that the heterogeneous designs and synthesis can significantly affect enhancing logic density and delay, narrowing the ASIC–FPGA gap. Works by Anderson and Wang with "gated" LUTs, at that point with deviated LUT LEs, demonstrate that the LUT elements exhibit in business FPGAs give superfluous adaptability. Toward enhanced delay and area, the macro cell-based FPGA structures have been proposed. These examinations depict noteworthy changes to the customary FPGA structures, while the progressions proposed here expand on models utilized as a part of industry and the scholarly world. Correspondingly, and-inverter cones have been proposed as substitutions for the LUTs, roused by and-inverter graphs (AIGs). Purnaprajna and lenne investigated the likelihood of repurposing the current MUXs contained inside the Xilinx Logic Slices. Like this work, they utilize the ABC need cut mapper and in addition VPR for packing, place, and route.

Nonetheless, their work is principally delayed based demonstrating a normal accelerate of 16% utilizing just ten of 19 VTR7 benchmarks. In this article, we think about the technology mapping issue for a novel field programmable gate array (FPGA) design that is based on k-input single-output programmable logic array (PLA-) like cells, or, k/m-macro cells. Every cell in this architecture can execute a single output function of up to k inputs and up to m product terms. We build up an exceptionally effective technology mapping algorithm, km stream, for this new kind of architecture. The exploratory outcomes demonstrate that our algorithm can accomplish depth optimality on all the experiments in an arrangement of 16 Microelectronics Center of North Carolina (MCNC) benchmarks.
Besides it is demonstrated that on this arrangement of benchmarks, with just a generally modest number of item terms \( (m \leq k+3) \), the k/m-macro cell based FPGAs can accomplish the same or comparative mapping depth contrasted with the primary k input single-output lookup table-(k-LUT-) based FPGAs. We additionally research the total area and delay of k/m-large scale cell-based FPGAs and contrast them and those of the generally utilized 4-LUT-based FPGAs. The exploratory outcomes demonstrate that k/m-full scale cell-based FPGAs can beat 4-LUT-based FPGAs as far as both delay and area after placement and routing by VPR on this arrangement of benchmarks. This paper presents test estimations of the contrasts between a 90-nm CMOS field programmable gate array (FPGA) and 90-nm CMOS standard-cell application specific integrated circuits (ASICs) as far as logic density, circuit speed, and power utilization for core logic. We are propelled to make these estimations to empower system designers to settle on better educated decisions between these two media and to offer knowledge to FPGA creators on the insufficiencies to assault and, in this way, enhance FPGAs. We depict the procedure by which the estimations were obtained and demonstrate that, for circuits containing just look-up table-based logic and flip-flops, the proportion of silicon area required to actualize them in FPGAs and ASICs is by and large 35.

Current FPGAs additionally contain "hard" blocks, for example, multiplier/accumulators and block memories. We find that these blocks reduce this normal area gap basically to as meager as 18 for our benchmarks, and we assess that broad utilization of these hard blocks could possibly bring down the gap to beneath five. The proportion of critical path delay, from FPGA to ASIC, is approximately three to four with less impact from block memory and hard multipliers. The dynamic power utilization proportion is roughly 14 times and, with hard obstructs, this gap for the most part winds up smaller. In this paper the new structural proposition are routinely created in both scholarly world and industry. For FPGA’s to continue to grow, it is important that these new architectural ideas are fairly and accurately evaluated, so that those worthy ideas can be included in future chips. Typically, this evaluation is done using experimentation. However, the use of experimentation is dangerous, since it requires making assumptions regarding the tools and architecture of the device in question. If these assumptions are not accurate, the conclusions from the experiments may not be meaningful. In this paper, we investigate the sensitivity of FPGA architectural conclusions to experimental variations. To make our study concrete, we evaluate the sensitivity of four previously published and well-known FPGA architectural results: lookup-table size, switch block topology, cluster size, and memory size. It is shown that these experiments are significantly affected by the assumptions, tools, and techniques used in the experiments.

III. PROPOSED ARCHITECTURES.
A. MUX4: 4-to-1 Multiplexer Logic Element.

The MUX4 LE appeared in Fig. 3 comprises of a 4-to-1 MUX with discretionary reversal on its input that permit the realization of any \{2, 3\}-input functions, some \{4,5\}-input functions, and one 6-input function with a 4-to-1 MUX itself with discretionary reversal on the data inputs. A 4-to-1 MUX matches the input pin count of a 6-LUT, taking into account reasonable correlations as for the network and intra cluster routing. Any two input Boolean function can be effectively executed in the MUX4: the two function inputs of info can
be fixing to the select lines and truth table values (logic 0 or logic 1) can be routed to the data inputs as needs be. For three-input functions consider that Shannon decomposition around one variable produces cofactors with at most two factors. A second decomposition of the cofactors around one of their two remaining variables produces cofactors with at most one variable. Such single-variable cofactors can be sustained to the data inputs (the optional inversion might be required), with the decomposition variables feeding the selected inputs. In like manner, elements of in excess of four data inputs can be actualized in the MUX4 as long as Shannon decomposition as for any two input sources produces cofactors with at most one input.

Fig. 3. MUX4 LE depicting optional data input inversions

B. Logic Elements, Fracturability, and MUX4-Based Variants

Two families of architectures were created:
1) Without fracturable LEs
2) With fracturable LEs.

In this paper, the fracturable LEs refer to an architectural element on which at minimum one logic function can be alternatively mapped. Nonfracturable LEs refer to a design element on which just a single logic work is mapped. In the nonfracturable designs, the MUX4 element appeared in Fig. 3 is utilized together with nonfracturable 6-LUTs. This element has an indistinguishable number of contributions from a 6-LUT loaning for reasonable correlation concerning the info network. For the fracturable architecture, we think about an eight-input LE, firmly coordinated with the versatile logic module in ongoing Altera Stratix FPGA families.

For the MUX4 variation, Dual MUX4, we utilize two MUX4s inside a single eight-input LE. In the arrangement, appeared in Fig. 4, the two MUX4s are wired to have devoted select inputs and shared data inputs. This arrangement enables this structure to map two independent (no shared inputs) three-input functions, while bigger functions might be mapped depending on the shared inputs between the two functions. A design in which a 4-to-1 MUX (MUX4) is fractures into two smaller 2-to-1 MUXs was considered.

Fig. 4. Double MUX4 LE that uses devoted select inputs and shared data Inputs

C. Hybrid Complex Logic Block

A wide range of designs were viewed as the first being a nonfracturable architecture. In the nonfracturable design, the CLB has 40 inputs and ten basic LEs (BLEs), with each BLE having six inputs and one output. Fig. 5 demonstrates this nonfracturable CLB design with BLEs that
contain an optional register. We change the proportion of MUX4s to LUTs inside the ten elements CLB from 1:9 to 5:5 MUX4s:6-LUTs. The MUX4 element is proposed to work in conjunction with 6-LUTs, making a hybrid CLB with a mixture of 6-LUTs and MUX4s (or MUX4 variations).

Fig.5: Hybrid CLB with a 50% depopulated Intra-CLB crossbar depicting BLE internals for Non-fracturable (one optional register and one Output) architecture.

Fig. 6 demonstrates the organization of our CLB and internal BLEs. For fracturable structures, the CLB has 80 inputs and ten BLEs, with each BLE having eight inputs and two outputs imitating an Altera Stratix Adaptive-LUT. A similar compass of MUX4 to LUT proportions was additionally performed. Fig. 4 demonstrates the fracturable architecture with eight inputs to each BLE that contains two optional registers. We assess fracturability of LEs versus nonfracturable LEs with regards to MUX4 elements since fracturable LUTs are common in commercial designs. For instance, Altera Adaptive 6-LUTs in Stratix IV and Xilinx Virtex 5 6-LUTs can be fractured into two smaller LUTs with a few constraints on inputs.

D. Area Modeling

1) MUX4 Logic Element:

Initial evaluations of the MUX4 element demonstrated that the MUX4 is ~10% the area of a 6-LUT in general. A 4-to-1 MUX can be realized with three 2-to-1 MUXs. Thus, the MUX4 element contains seven 2-to-1 MUXs, four SRAM cells and four inverters altogether (see Fig. 1). The optional inversion utilizes the four SRAM cells, while whatever is left of the LE arrangement is performed through routing. Likewise, the depth of the MUX tree is halved and compared with the 6-LUT, which has six 2-to-1 MUXs on its longest path. Minimalistically, expecting consistent pass transistor sizing and that the area of a 2-to-1 MUX and six transistor SRAM cell are generally identical, the MUX4 element has (1/16)th the SRAM region and (1/8)th the MUX area of a 6-LUT. These evaluations were amended utilizing transistor level modeling of the circuit blocks.

Transistor level optimization of the constituent circuit blocks of a FPGA requires a comprehension of the ideal area delay tradeoffs for every individual circuit block. This requires extracting a delegate critical path, which is a way whose structure of blocks and topology will be like the critical path of a particular design. Separating the representative critical path enables us to judge to what degree every individual block is timing critical, which accordingly builds up a area delay tradeoff objectives for each block. This is in accordance with the transistor-level optimization tool developed previously. We utilize the consequences of earlier work to set up the optimal area delay tradeoff for 6-LUTs in traditional island-style.
FPGA architecture with typical structural parameters.

The subsequent 6-LUT delay fills in as a perspective for optimization for the circuits considered in this paper: in light of a legitimate concern for expanding region diminishment while enabling execution to be kept up (overlooking the distinctions in cell tallies between mapping to a customary LUT and the LEs proposed in this paper), we endeavor to coordinate the delay of a 6-LUT while limiting the area of every one of the variations of the MUX4 circuits. Transistor level modeling and optimizations depended on a prescient 22-nm superior process [21], while the area display exhibited in earlier work [20] was utilized to appraise the region of different circuit structures. With this approach, we decided an area delay optimal 6-LUT has a region of 930 minimum width transistors, and a most pessimistic scenario delay of 261 ps. For the MUX4 cell and Dual MUX4 cell, a base area and minimum delay cell was made.

The minimum area MUX4 cell has an area of 95 minimum width transistors and a delay of 204 ps; all transistors were minimum width for this situation, and as the minimum area answer for this circuit could meet (and enhance) the worst case delay target of a 6-LUT. Essentially, the Dual MUX4 cell has an area of 249 minimum width transistors while meeting the most pessimistic scenario delay prerequisite. Be that as it may, we utilized the delay design plan for both the MUX4 and Dual MUX4 elements for whatever remains of the examination as there is definitely not a huge increment in region over the minimum area design.

2) FPGA Area Model:

Although determining the area of a MUX4 element relative to a 6-LUT is important, we need to also examine global FPGA area considering the number of CLB tiles, area overheads within the CLB and routing area per CLB. Throughout this paper, global FPGA area was estimated assuming that, per tile, 50% of the area is inter cluster and intra cluster routing, 30% of the area is used for LUTs, and 20% for registers and other miscellaneous logic, following Anderson and Wang and a private communication. It is important to note that this 50%–30%–20% model is an estimate based on a traditional full FPGA design where-by the routing and internal CLB crossbars are optimized toward 6-LUTs. Production of an optimized FPGA utilizing our new MUX4 elements would surely change said model. However, optimizing the entire routing architecture toward our MUX4 variants, measuring the routing architecture, and closing the loop by creating a more accurate model is out of the scope of this work. Using this model, we can make some observations about the hybrid CLB architecture. The 30% that normally would account for ten 6-LUT LEs within the tile is now split between the smaller MUX4 elements and 6-LUTs.

IV. Technology mapping using ABC

ABC was used for technology mapping, with modifications that allow for MUX4 embeddable function identification and MUX2 embeddable function identification in the case of fracturable MUX4s and custom mapping. The internal data structure used within the ABC is an AIG, where the logic circuit is represented using 2-input AND gates with inverters. Priority Cuts mapping in ABC (invoked with the if command) was modified to perform our custom technology mapping. This mapper traverses the AIG from primary inputs to primary outputs finding intermediate mappings for internal nodes and finally the primary outputs, using a dynamic programming approach. The priority cuts mapper performs multiple passes on the AIG to find the best cut per node. For depth-oriented mapping, the mapper first prioritizes
mapping depth then optimizes for area discarding cuts whose selection would increase the overall depth of the mapped network. Based on this standard mapper, two mapper variants were produced and evaluated. The first variant, Natural Mux, evaluates and identifies internal functions that are MUX4-embeddable, agnostic of the target architecture; i.e., this flow uses the default priority cuts mapping and performs a post processing step to identify MUX4-embeddable functions. From this mapping, we can evaluate what area savings are possible without any mapper changes. The second variant MuxMap, area-weights the MUX4-embeddable cuts relative to 6-LUT cuts, thereby establishing a preference for selection/creation of MUX4-embeddable solutions.

V. Modeling using VPR

VPR was used to perform architectural evaluation. The standard ten 6-LUT CLB architecture in 40-nm included with the VPR distribution was used for baseline modeling. The hybrid CLBs shown in Figs. 3 and 4 were modeled using the XML-based VPR architectural language. The snippet from the architecture file for the physical block hardened MUX4 element, this code specifies a MUX4 as a six-input one-output black box to the VPR. In addition, since all MUX4s can also be mapped to the 6-LUTs, an additional mode was added to the 6-LUT physical block. The mode concept allows the VPR packer to pack LUTs into LUTs (as usual), but also enables MUX4s to be packed into the LUTs. The architectures with CLBs having MUX4: LUT ratios from 1:9 to 5:5 were created from the baseline 40-nm architectures with delays obtained through circuit simulations of the MUX4 variants. Importantly, we made minor modifications to the VPR packing algorithm itself, so that the MUX4 net list elements are preferred to be packed into the MUX4 Les in the architecture (while limiting packing MUX4 net list elements into LUTs). The modifications involved changing the attraction function during the CLB packing. One change was to ensure that the logic functions that were MUX4 embeddable were preferentially packed into a physical MUX4 element and not into an LUT. Another was to apply a negative weight on MUX4-embeddable functions when the current CLB’s physical MUX4 elements are all occupied also preventing MUX4-embeddable functions from being placed into the LUTs. Without this, the MUX4 net list elements might needlessly consume LUTs, which should be reserved, where possible, for those net list elements that demand their flexibility. This becomes doubly important for fracturable architectures, since their packing problem is more complex. Without this modification, a significant CLB usage increase was observed across all benchmark sets.

VI. RESULTS

Fig. 7: Simulation Results for Dual mux4

Fig. 8: Simulation Results for Hybrid CLB6
VII. CONCLUSION

We have proposed another mixture of CLB design containing MUX4 hard MUX elements and indicated strategies for productively mapping to these structures. Weighting of MUX4-embeddable functions with our MuxMap method joined with a select mapping technique gave help to circuits low normal MUX4-embeddable proportions. We additionally gave examination of the benchmark suites post mapping, talking about the circulation of functions inside every benchmark suite. The region decreases for nonfracturable structures, is 8% and MUX4:LUT proportion is 4:6 and on account of fracturable architecture the region diminishments are 2%. The CHStone benchmarks being high level incorporated with LegUp-HLS likewise demonstrated barely better execution and this could be because of the way LegUp performs HLS on the CHStone benchmarks themselves. In general, the expansion of MUX4s to FPGA models insignificantly affect FMax and show potential for enhancing logic thickness in nonfracturable designs and unassuming potential for enhancing logic thickness in fracturable architecture.

REFERENCES


BIOGRAPHIES

Medikurthi Sravani is pursuing M.Tech in VLSI SYSTEM DESIGN from SRINIVASA INSTITUTE OF TECHNOLOGY and MANAGEMENT STUDIES, CHITTOOR, Andhra Pradesh.