International Journal of LNCT, Vol 2(5)
ISSN (Online): 2456-9895

Review paper on hybrid LUT/MUX combinational architecture

Shaili Jain[1], Shashilata Rawat[2] and Monika Kapoor[3]
M.Tech Scholar,Department of Electronics and Communication Engineering,LNCT Bhopal(M.P.),India[1]
Assistant Professor,Department of Electronics and Communication Engineering,LNCT Bhopal(M.P),India[2]
Professor and Head, Department of Electronics and Communication Engineering,LNCT Bhopal(M.P),India[3]

Abstract
Hybrid combinational architectures are used for FPGA's which consists of different combinations of LUT's and hardened multiplexers are evaluated for achieving the goal of greater logic density and area minimization. Various Technology mapping optimizations which target the proposed architectures are also implemented within the circuit. All calculating for CLB's and routing area while retaining mapping extent. For divisible architectures, the proposed architecture of this paper, analyzes the logic size, area and power consumption using Xilinx 14.2.

Keywords
Hybrid system, Combinational logic block (CLB), LUT, Multiplexer.

1.Introduction
A field-programmable gate array (FPGA) is a combination of various programmable logic devices which is capable of implementing multi-level logic functions. FPGAs are usually used as discrete service chips that can be programmed to execute large functions. Although, simpler blocks of FPGA logic can be useful components on-chip to allow the user of the chip to tailor part of the chip's logical function. FPGA block must implement both combinational logic functions and interconnect to be able to construct multi-level logic functions. There are various distinct technologies for programming FPGAs, but most logic processes are doubtful to implement anti fuses or similar hard programming technologies. During the history of FPGA's, LUT's have been the main logic element (LE) used to accomplish combinational logic. A K-input LUT is generic and very flexible able to implement any K-input Boolean function. The use of LUTs eases technology mapping as the problem is reduced to a graph covering problem. Though, a very high area price is paid when large area LUTs are considered. The value of K between 4 and 6 is usually observed in industry and academia, and this range has been verified to offer a good area vs performance compromise. In recent times, a number of other works have explored different FPGA LE architectures for performance improvement to close the large gap between FPGAs and application-specific integrated circuits (ASICs).

Look up Tables:
The basic method used to build a combinational logic block (CLB) also called a logic element in an SRAM-based FPGA is the lookup table (LUT). As shown in Figure 1, the lookup table is an SRAM that is used to implement a truth table. Every address stored 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.

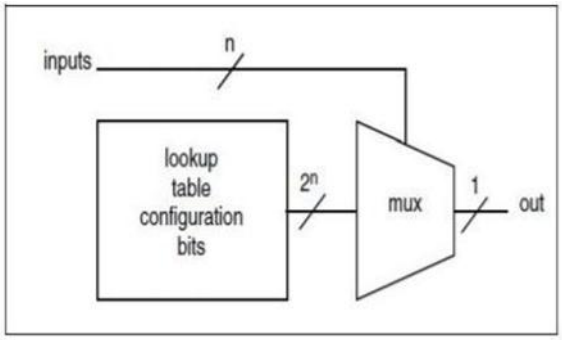

Figure 1 Lookup tables

Programming a lookup table
Disparate a typical logic gate, the function represented by the logic element can be modified by modifying the values of the bits stored in the SRAM. As a result, the n-input typical logic element has four inputs. The delay due to the LUT is autonomous of the number of bits stored in the SRAM, so the delay through the logic element is the constant for all functions. This indicates that, for example, a LUT-based logic element will display the same delay for a 4-input XOR and a 4-input NAND. In contrast, a 4-input XOR developed by using static CMOS logic is significantly slower than a 4-input NAND. Certainly, the static logic gate is usually quicker than the logic element. Logic elements normally contain registers flip-flops and latches as well as combinational logic. A flip-flop or latch is trivial as compared to the combinational logic element as compared to custom

VLSI, so it signifies to add it to the combinational logic element. Using a discrete cell for the memory element would basically engage routing resources. The memory element is connected to the output, whether it stores a given value is controlled by its clock and enable inputs. In this paper, we propose incorporating hardened multiplexers (MUXs) in the FPGA logic blocks as a means of increasing silicon area efficiency and logic density. The MUX-based logic blocks for the FPGAs have seen success in early commercial architectures, such as the Actel ACT-1/2/3 architectures, and efficient mapping to these structures has been studied in the early 1990s. However, their use in commercial chips has waned, perhaps partly due to the ease with which logic functions can be mapped into LUTs, simplifying the entire computer aided design (CAD) flow. Nevertheless, it is widely understood that the LUTs are inefficient at implementing MUXs, and that MUXs are frequently used in logic circuits. To underscore the inefficiency of LUTs implementing MUXs, consider that a six input LUT (6-LUT) is essentially a 64-to-1 MUX (to select 1 of 64 truth-table rows) and 64-SRAM configuration cells, yet it can only realize a 4-to-1 MUX (4 data+2 select=6 inputs). In this paper, we present a six-input LE based on a 4-to-1 MUX, MUX4, that can realize a subset of six-input Boolean logic functions, and a new hybrid complex logic block (CLB) that contains a mixture of MUX4s and 6-LUTs. The proposed MUX4s are small compared with a 6-LUT (15% of 6-LUT area), and can efficiently map all {2,3}-input functions and some {4,5,6}-input functions. In addition, we explore factorability of Les the ability tosplit the LEs into multiple smaller elements in both LUTs and MUX4s to raise logic density. The ratio of LEs that should be LUTs versus MUX4s is also looked towards optimizing logic density for both non-fragile and fragile FPGA architectures. To ease the architecture examination, we developed a CAD flow for mapping into the proposed hybrid CLBs, created by using ABC and VPR, and describe technology mapping method that encourages the selection of logic functions that can be embedded into the MUX4 elements. The main contributions in this paper are as follows:

- Two hybrid CLB architectures (non-fracturable and fracturable) that contain a mixture of MUX4 LEs and the traditional LUTs yielding up to 8% area savings.
- Mapping techniques called Natural Mux and Mux Map targeted toward the hybrid CLB architecture that optimize for area, while preserving the original mapping depth.
- A full post-place-and-route architecture evaluation with VTR7, and CH Stone benchmarks facilitated by Leg Up- HLS, the Verilog-to-Routing project showing impact on both area and delay.

Compared with the preliminary publication, we have performed transistor level modeling of the MUX4 LE, further studied the fracturable architectures, and unified the open source tool-flow from C through LegUp-HLS to the VTR flow. Sparse crossbars (versus full crossbars in the previous work) have also been included in our CLBs, increasing modeling accuracy. The new transistor-level modeling of the MUX4 also provides more accurate results as compared with the previous work. Results have also been expanded with the inclusion of timing results as well as larger architectural ratio sweeps.

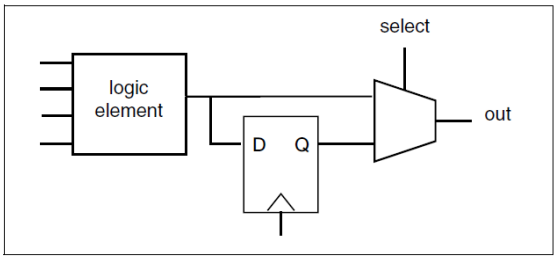

Figure 2 Programming a lookup table

2.Literature review

Stephen Alexander Chin et al. Hybrid configurable logicblock 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 Leg Up-HLS, Odin-II front-end synthesis, ABC logic synthesis and technology mapping, and VPR for packing, placement, routing, and architecture exploration. Technology mapping optimizations that target the proposed architectures are also implemented within ABC. Experimentally, we show that for non-fracturable architectures, without any mapper optimizations, we naturally save up to ~8% area post place and route; both accounting for complex logic block and routing area while maintaining mapping depth. With architecture-aware method mapper optimizations in ABC, additional area is saved, post-place-and-route. For fracturable architectures, experiments show that only marginal gains are seen after place-and-route up to ~2%. For both non-

fracturable and fracturable architectures, minimal impact on timing performance for the architectures with best area-efficiency.

Rose et al. Recent works have shown that the heterogeneous architectures and synthesis methods can have a significant impact on improving logic density and delay, narrowing the ASIC–FPGA gap. Works by Anderson and Wang with- gate LUTs, then with asymmetric LUT LEs, show that the LUT elements present in commercial FPGAs provide unnecessary flexibility. Towards improved delay and area, the macrocell-based FPGA architectures have been proposed. These studies describe significant changes to the traditional FPGA architectures, whereas the changes proposed here build on architectures used in industry and academia. Similarly, and-inverter cones have been proposed as replacements for the LUTs, inspired by and-inverter graphs (AIGs).

Y. Hara et al. Purnaprajna and Ienne explored thepossibility of repurposing the existing MUXs contained within the Xilinx Logic Slices. Similar to this work, they use the ABC priority cut mapper as well as VPR for packing, place, and route. However, their work is primarily delaying based showing an average speed up of 16% using only ten of 19 VTR7 benchmarks. In this article, we study the technology mapping problem for a novel field programmable gate array (FPGA) architecture that is based onk-input single-output programmable logic array (PLA) like cells, or, k/m-macrocells. Each cell in this architecture can implement a single output function of up to k inputs and up to m product terms. We develop a very efficient technology mapping algorithm, km flow, for this new type of architecture.

A. Canis et al. The experimental results show that ouralgorithm can achieve depth-optimality on almost all the test cases in a set of 16 Microelectronics Center of North Carolina (MCNC) benchmarks. Furthermore it is shown that on this set of benchmarks, with only a relatively small number of product terms ($m \leq k+3$), the k/m-macro cellbased FPGAs can achieve the same or similar mapping depth compared with the traditional k-input single-output lookup table- (k-LUT) based FPGAs. We also investigate the total area and delay of k/m-macro cell-based FPGAs and compare them with those of the commonly used 4-LUT-based FPGAs. The experimental results show that k/m-macro cell-based FPGAs can outperform 4-LUT-based FPGAs in terms of both delay and area after placement and routing by VPR on this set of benchmarks.

Ahmed et al. This paper presents experimentalmeasurements of the differences between a 90- nm CMOS field programmable gate array (FPGA) and 90-nm CMOS standard-cell application specific integrated circuits (ASICs) in terms of logic density, circuit speed, and power consumption for core logic. We are motivated to make these measurements to enable system designers to make better informed choices between these two media and to give insight to FPGA makers on the deficiencies to attack and, thereby, improve FPGAs. We describe the methodology by which the measurements were obtained and show that, for circuits containing only look-up table-based logic and flip-flops, the ratio of silicon area required to implement them in FPGAs and ASICs is on average 35. Modern FPGAs also contain - hard blocks such as multiplier/accumulators and block memories. We find that these blocks reduce this average area gap significantly to as little as 18 for our benchmarks, and we estimate that extensive use of these hard blocks could potentially lower the gap to below five. The ratio of critical-path delay, from FPGA to ASIC, is roughly three to four with less influence from block memory and hard multipliers.

J. Rose et al. The dynamic power consumption ratio isapproximately 14 times and, with hard blocks, this gap generally becomes smaller. In this paper the new architectural proposals are routinely generated in both academia 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.

3.Problem formulation

A K-input LUT is generic and very flexible, capable of implementing any K-input Boolean functions. The use of LUTs eases the technology mapping as the problem is minimized to a graph covering problem. Still, a very high price per unit area is paid when complex LUTs are examined. The value of K between 4 to 6 is normally observed in industrial and academic fields, and this range has been confirmed to offer a good area vs performance compromise [4],[5]. Not very long ago, a number of different works have been explored for alternate FPGA LE architectures for performance improvement [6]–[10] to close the large void between FPGAs and ASIC's [11]. In this paper, we are proposing for incorporating hardened multiplexers (MUXs) in the FPGA logic blocks as a means of increasing silicon area efficiency and logic density.

The MUX-based logic blocks for the FPGAs have observed achievement in early commercial architectures, such as the Actel ACT-1/2/3 architectures, and proficient mapping to these structures has been studied [12] in the early 1990s. However, their use in commercial chips has diminished, may be partly due to the ease with which logic functions can be mapped into LUTs, simplifying the entire CAD flow. However, it is widely observed and accepted that the LUTs are ineffective at implementing frequently implemented component MUXs.

4.Proposed methodology

A. Mux4: 4-To-1 multiplexer logic element

The MUX4 LE shown in Figure 3 consists of a 4- to-1 MUX with optional inversion on its inputs that allow the realization of any {2,3}-input function, some {4,5}-input functions, and one 6-input function a 4-to-1 MUX itself with optional inversion on the data inputs. A 4-to-1MUX matches the input pin count of a 6-LUT, allowing for fair comparisons with respect to the connectivity and intra cluster routing. Any two-input Boolean function can be easily implemented in the MUX4: the two function inputs can be tied to the select lines and the truth table values (logic-0or logic-1) can be routed to the data inputs accordingly. For three-input functions; consider that Shannon decomposition about one variable produces cofactors with at most two variables. A second decomposition of the cofactors about one of their two remaining variables produces cofactors with at most one variable. Such single-variable cofactors can be fed to the data inputs (the optional inversion may be needed), with the decomposition variables feeding the select inputs. Likewise, functions of more than four inputs can be implemented in the MUX4 as long as Shannon decomposition with respect to any two inputs produces cofactors with at most one input

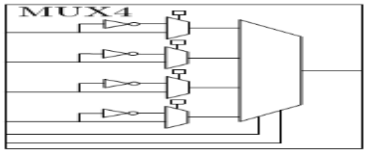

Figure 3 MUX4 LE depicting optional data input inversions

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 one or more logic functions can beoptionally mapped. Non-fracturable LEs refer to an architectural element on which only one logic function is mapped. In the non-fracturable architectures, the MUX4 element shown in the above Figure 3 is used collectively with non-fracturable 6-LUTs. This element shares the same number of inputs as a 6-LUT offering for reasonable comparison with respect to the input connectivity. For the fracturable architecture, we consider an eight-input LE, closely tally with the adaptive logic module in recent Altera Stratix FPGA families. For the MUX4 variant, Dual MUX4, we use two MUX4s within a single eight-input LE. In the configuration, shown in Figure 4, the two MUX4s are wired to have dedicated select inputs and shared data inputs. This configuration allows this structure to map two independent (no shared inputs) three-input functions, while larger functions may be mapped dependent on the shared inputs between both functions. An architecture in which a 4-to-1 MUX (MUX4) is fractured into two smaller 2-to-1 MUXs was considered.

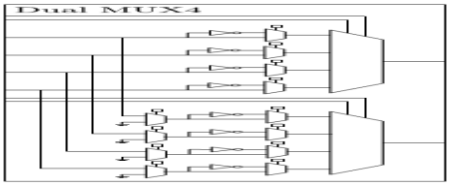

Figure 4 Dual MUX4 LE that utilizes dedicated select inputs and shared data Inputs

C. Hybrid complex logic block

A variety of different architectures were considered the first being a non-fracturable architecture. In the

non-fracturable architecture, the CLB has 40 inputs and ten basic LEs (BLEs), with each BLE having six inputs and one output. Figure 5 shows this non-fracturable CLB architecture with BLEs that contain an optional register. We vary the ratio of MUX4s to LUTs within 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, creating a hybrid CLB with a mixture of 6-LUTs and MUX4s (or MUX4 variants).

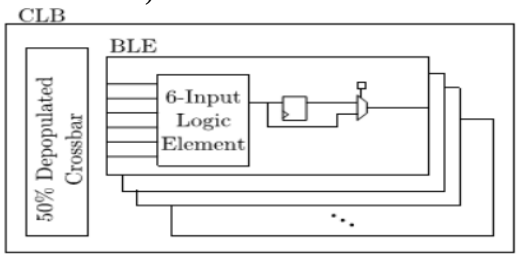

Figure 5 Hybrid CLB with a 50% depopulated intra-CLB crossbar depicting BLE internals for nonfracturable (one optional register and one output) architecture

Figure 6 Shows the organization of our CLB and internal BLEs. For fracturable architectures, the CLB has 80 inputs and ten BLEs, with each BLE having eight inputs and two outputs emulating an Altera Stratix Adaptive-LUT. The same sweep of MUX4 to LUT ratios was also performed. Fig. 4 shows the fracturable architecture with eight inputs to each BLE that contains two optional registers. We evaluate fracturability of LEs versus non-fracturable LEs in the context of MUX4 elements since fracturable LUTs are common in commercial architectures. For example, Altera Adaptive 6-LUTs in Stratix IV and Xilinx Virtex 5 6-LUTs can be fractured into two smaller LUTs with some limitations on inputs.

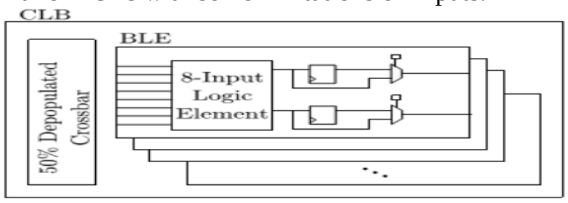

Figure 6 Hybrid CLB with a 50% depopulated intra-CLB crossbar depicting BLE internals for a fracturable (two optional registers and two outputs) architecture

D. Area Modeling

1) MUX4 Logic Element: Initial estimates of the MUX4 element showed that the MUX4 is~10% the area of a 6-LUT overall. A 4-to-1 MUX can be realized with three 2-to-1 MUXs. Hence, the MUX4 element contains seven 2- to-1 MUXs, four SRAM cells, and four inverters in total (see Fig. 1). The optional inversion uses the four SRAM cells, whereas the rest of the LE configuration is performed through routing. In addition, the depth of the MUX tree is halved compared with the 6-LUT, which has six 2-to-1 MUXs on its longest paths. Conservatively, assuming constant pass transistor sizing and that the area of a 2-to-1 MUX and six transistor SRAM cell are roughly equivalent, the MUX4 element has (1/16)th the SRAM area and(1/8)th the MUX area of a 6-LUT. These estimates were revised using transistor level modeling of the circuit blocks. Transistor-level optimization of the constituent circuit blocks of an FPGA requires an understanding of the optimal area-delay tradeoffs for each individual circuit block. This requires extracting a representative critical path, which is a path whose composition of blocks and topology will be similar to the critical path of a specific design. Extracting the representative critical path allows us to judge to what extent each individual block is timing critical, which thus establishes an area-delay tradeoff goals for each block. This is in line with the transistor-level optimization tool developed previously. We use the results of prior work to establish the optimal area-delay tradeoff for 6-LUTs in conventional island-style FPGA architecture with typical architectural parameters. The resulting 6-LUT delay serves as a point of reference for optimization for the circuits considered in this paper: in the interest of maximizing area reduction while allowing performance to be maintained (ignoring the differences in cell counts between mapping to a conventional LUT and the LEs proposed in this paper), we attempt to match the delay of a 6-LUT while minimizing the area of each of the variants of the MUX4 circuits. Transistor level modeling and optimizations were based on a9 predictive 22-nm high performance process, while the area model presented in prior work was used to estimate the area of various circuit structures. With this methodology, we determined an area-delay optimal 6-LUT has an area of 930 minimum-width transistors, and a worst-case delay of 261 ps. For the MUX4 cell and Dual MUX4 cell, a minimum area and minimum delay cell was created. The minimum area MUX4 cell has an area of 95 minimum width transistors and a delay of 204 ps; all transistors were minimum-width in this case, and as the minimum area solution for this circuit was able to meet (and improve upon) the worst-case delay target of a 6-LUT. Similarly, the Dual MUX4 cell has an area of 249 minimum-width transistors while meeting the worst-case delay

requirement. However, we chose to use the minimum delay design for both the MUX4 and Dual MUX4 elements for the rest of the study as there is not a significant increase in area 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 variantsmeasuring 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.

5.Conclusion

In this paper, we proposed a new hybrid CLB architecture containing MUX4 hard MUX elements and shown techniques for efficiently mapping to these architectures. We also provided analysis of the benchmark suites post mapping, discussing the distribution of functions within each benchmark suite. The area reductions for non-fracturable architectures, is 8% and MUX4:LUT ratio is 4:6 and in the case of fracturable architecture the area reductions are 2%.The CH Stone benchmarks being high level synthesized with LegUp-HLS also showed marginally better performance and this could be due to the way LegUp performs HLS on the CHStone benchmarks themselves. Overall, the addition of MUX4s to FPGA architectures minimally impact FMax and show potential for improving logic-density in non-fracturable architectures and modest potential for improving logic density in fracturable architecture.

References

[1] Chin SA, Luu J, Huda S, Anderson JH. Hybrid lut/multiplexer FPGA logic architectures. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2016; 24(4):1280-92.

[2] Rose J, Luu J, Yu CW, Densmore O, Goeders J, Somerville A, et al. The VTR project: architecture and CAD for FPGAs from verilog to routing. In Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays 2012 (pp. 77-86). ACM.

[3] Hara Y, Tomiyama H, Honda S, Takada H. Proposal and quantitative analysis of the CHStone benchmark program suite for practical C-based high-level synthesis. Journal of Information Processing. 2009; 17:242-54.

[4] Canis A, Choi J, Aldham M, Zhang V, Kammoona A, Anderson JH, et al. LegUp: high-level synthesis for FPGA-based processor/accelerator systems. InProceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays 2011 (pp. 33-6). ACM.

[5] Ahmed E, Rose J. The effect of LUT and cluster size on deep-submicron FPGA performance and density. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2004; 12(3):288-98.

[6] Rose J, Francis RJ, Lewis D, Chow P. Architecture of field-programmable gate arrays: The effect of logic block functionality on area efficiency. IEEE Journal of Solid-State Circuits. 1990; 25(5):1217-25.

[7] Parandeh-Afshar H, Benbihi H, Novo D, Ienne P. Rethinking FPGAs: elude the flexibility excess of LUTs with and-inverter cones. In proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays 2012 (pp. 119-28). ACM.

[8] Anderson JH, Wang Q. Improving logic density through synthesis-inspired architecture. In field programmable logic and applications, 2009. FPL 2009. International conference on 2009 (pp. 105-11). IEEE.

[9] Anderson JH, Wang Q. Area-efficient FPGA logic elements: Architecture and synthesis. In proceedings of the 16th asia and south pacific design automation conference 2011 (pp. 369-75). IEEE Press.

[10] Cong J, Huang H, Yuan X. Technology mapping and architecture evalution for k/m-macrocell-based FPGAs. ACM transactions on design automation of electronic systems (TODAES). 2005; 10(1):3-23.

[11] Hu Y, Das S, Trimberger S, He L. Design, synthesis and evaluation of heterogeneous FPGA with mixed LUTs and macro-gates. In Proceedings of the IEEE/ACM international conference on Computer-aided design 2007 (pp. 188-93). IEEE Press.

[12] Kuon I, Rose J. Measuring the gap between FPGAs and ASICs. IEEE Transactions on computer-aided design of integrated circuits and systems. 2007; 26(2):203-15.