Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 2044 – 2054

doi:10.3906/elk-1506-51

Performance analysis and optimization of cluster-based mesh FPGA architectures: design methodology and CAD tool support

Sonda CHTOUROU[1,*], Zied MARRAKCHI[2], Emna AMOURI[3], Vinod PANGRACIOUS[3], Mohamed ABID[1], Habib MEHREZ[3]

[1]CES Laboratory, National Engineering School of Sfax, University of Sfax, Sfax, Tunisia
[2]Flexras Technologies SAS, Romainville, France
[3]LIP6 Laboratory, Universite Pierre et Marie Curie, Paris, France

Received: 05.06.2015 • **Accepted/Published Online:** 18.08.2016 • **Final Version:** 29.05.2017

Abstract: Field programmable gate arrays (FPGAs) have become an attractive implementation medium for digital circuits. FPGA design's big challenge is to find a good trade-off between flexibility and performance in terms of power dissipation, area density, and delay. This paper presents a new cluster-based FPGA architecture combining mesh and hierarchical interconnect topologies. Based on experimental method and benchmarks circuit implementation, this work provides a detailed exploration and analyses of the effect of cluster functionality on the proposed cluster-based FPGA in terms of power dissipation, area density, and delay. The exploration results showed that architecture with high cluster size provides high speed performance and low power dissipation. We noted also that architecture with small cluster size is more efficient in terms of area. Look-up-table (LUT) exploration showed that using architecture with 4-input LUT offers the best trade-off between power dissipation, area density, and delay.

Key words: Field programmable gate arrays, mesh of clusters architecture, computer-aided design tools, architecture exploration, performance analysis

1. Introduction

Field programmable gate arrays (FPGA) are integrated circuits that can be configured to implement arbitrary logic functions after manufacturing. FPGAs have become one of the most attractive platforms since they offer cost-effective circuits and enable rapid prototyping of design alternatives and then lead to a fast design cycle. Nevertheless, FPGAs have significant inefficiency in terms of power dissipation, area density, and delay. In fact, the ratio of power dissipation, area density, and delay, from FPGA to application specific integrated circuit (ASIC), is respectively 7–14, 18–35, and 3–4 times [1]. This happens because a large part of the FPGA is dedicated to routing interconnect [2]. To make FPGAs more efficient, recent works investigated improvements of computer-aided design (CAD) algorithms used in each step of mapping a logic circuit into FPGAs. For example, improving the logic synthesis algorithm step can decrease the amount and depth of needed logic and power dissipation [3]. In [4], the authors propose to revisit partitioning, placement, and clustering steps. They show that the proposed algorithms can reduce the use of routing interconnect and improve the performance by shortening connections. In [5], the authors explore new efficient clustering and placement techniques on area-delay trade-off and power dissipation. They showed that the proposed technique can provide a better

*Correspondence: sonda.chtourou@ceslab.org

spatial uniformity distribution of the placed design and then improve the overall performance. We find in the literature other work exploring how to design a low power system on FPGAs. For example, in [6] the authors investigated design optimization to improve FPGA performance. They developed a typical design for video zoom-in processing with low power dissipation. Implementation on FPGAs showed that the proposed design provides an important gain in power without degrading frequency and area.

Another way to improve FPGA performance is to explore new FPGA architectures. Defining an FPGA architecture is a challenge of fixing logic and routing resources so that these algorithms produce the most efficient possible results. FPGA designers can investigate two architectural factors: logic block granularity and routing interconnect topology. The first factor consists of exploring new logic block structures or exploring multigranularity architectures that can enhance FPGA functionality and performance [2]. The second factor consists of exploring new interconnect topologies. According to [2], FPGA routing architecture is more critical than logic architecture since it denotes 60% of the power dissipation, 80% of the area density, and 80% of the delay. Therefore, FPGA routing architecture is the bottleneck behind FPGA performance inefficiency and that is why we will tackle interconnect exploration in this paper.

Today, modern academic and industrial FPGA architectures are composed of sets of clusters grouping a number of look-up-tables (LUTs). Previous work [7] demonstrates that these cluster-based FPGA architectures provide better performance. Researchers have investigated and explored cluster-based FPGA architectures. In [8], the authors explored versatile place and route (VPR) [4] FPGA architecture. VPR architecture is characterized by a fully populated intracluster interconnect. This kind of interconnect ensures a high flexibility; however, it penalizes area efficiency. Moreover, VPR routing interconnects use bidirectional routing networks that dissipate an important amount of the leakage power [9]. In [10], the authors proposed VPR intracluster crossbar depopulation that saves 18% of the area. All architectures mentioned above consider the connection blocks (CBs) and the intracluster crossbar separately. The CBs interconnect assures interconnection between clusters inputs/outputs and adjacent routing tracks. In [11], the authors proposed to merge CB and intracluster interconnect levels. The new proposed interconnect saves 28% of the total area. However, they used a full crossbar to connect feedbacks to logic block inputs, which can be very penalizing.

In this paper, we present a new efficient way to design interconnection structures for cluster-based mesh architectures. The proposed architecture presents a depopulated routing interconnect based on butterfly-fat-tree (BFT) topology. We merge CBs interconnect level in switch blocks (SBs) and we propose a new hierarchical SB interconnect to assure all intercluster routing interconnects. To optimize the proposed cluster-based FPGA architecture, we focus on investigating the cluster functionality in terms of power dissipation, area density, and delay of an FPGA. In particular, we look at the effect of LUT size (number of inputs per LUT) and cluster size (number of LBs per cluster). The effect of varying these architecture parameters is unpredictable [7] and we need to conduct real experimentation to identity how architecture parameters should be tuned to provide the best trade-offs.

This paper is organized as follows: Section 2 gives an overview of the proposed cluster-based FPGA architecture. Section 3 presents power, density, and timing models designed for target FPGA architecture. Finally, Section 4 details exploration results.

2. Proposed cluster-based architecture with optimized switch blocks

In this section, we present an advanced cluster-based architecture with optimized SBs interconnect. As shown in Figure 1, this architecture is composed of clusters placed in a 2D grid.

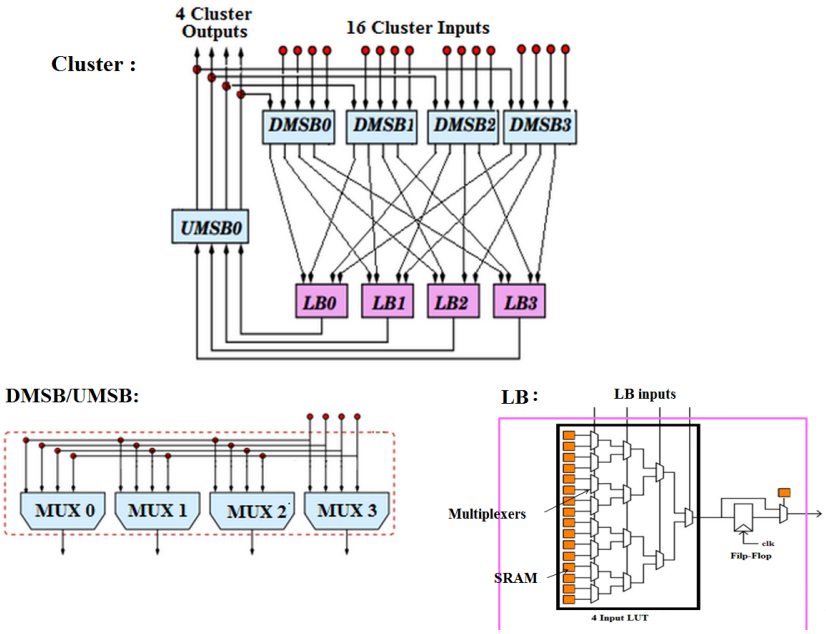

Figure 1. Proposed FPGA architecture.

2.1. Cluster architecture

Each cluster groups a number of LBs connected with an intracluster interconnect. Figure 2 illustrates an example of a cluster with 4 LBs (cluster size = 4). The intracluster crossbar is a depopulated interconnect. It is divided into mini switch blocks (MSBs) and is composed of two networks: a downward network and an upward network. The downward network is based on the BFT topology connecting downward MSBs (DMSBs) outputs to LBs inputs. The upward network connects LB outputs to an upward MSB (UMSB) and allows all LBs outputs to reach all DMSBs and cluster outputs.

Each LB is composed of a LUT followed by a flip-flop (FF). A LUT can be implemented using a multiplexer, as illustrated in Figure 2 for an example of an LB with a 4-input LUT. The 4 inputs of the LUT are used to control which SRAM value the output will take. FFs are used to implement sequential circuits in the FPGA. In general, a k-input LUT contains 2^k configuration bits and it can implement any Boolean function with k variables. Increasing either LUT or cluster size increases the functionality of the cluster, which has two positive effects. In fact, it decreases the total number of clusters required to implement a user circuit in the FPGA and decreases the number of clusters on the critical path and then improves performance. Nevertheless, the resulted logic area grows large with the increase in LUT and cluster size [2].

2.2. Routing architecture

Routing architectures of cluster-based mesh FPGAs are generally composed of CB and SB interconnect levels. CBs are used to connect clusters inputs/outputs to adjacent routing channels and SBs assure interconnection between horizontal and vertical routing channels. In the proposed architecture, we merge CBs within an SB and we propose a new class of SB interconnect that assures all intercluster interconnections. Figure 3 illustrates a view of the SB interconnect and a global view of the 4 adjacent SBs (A, B, C, D) and the 4 adjacent clusters (A, B, C, D) highlighted in Figure 1. To assure different intercluster interconnections, the SB has a multilevel topology including 3 main boxes (Box 1, Box 2, and Box 3).

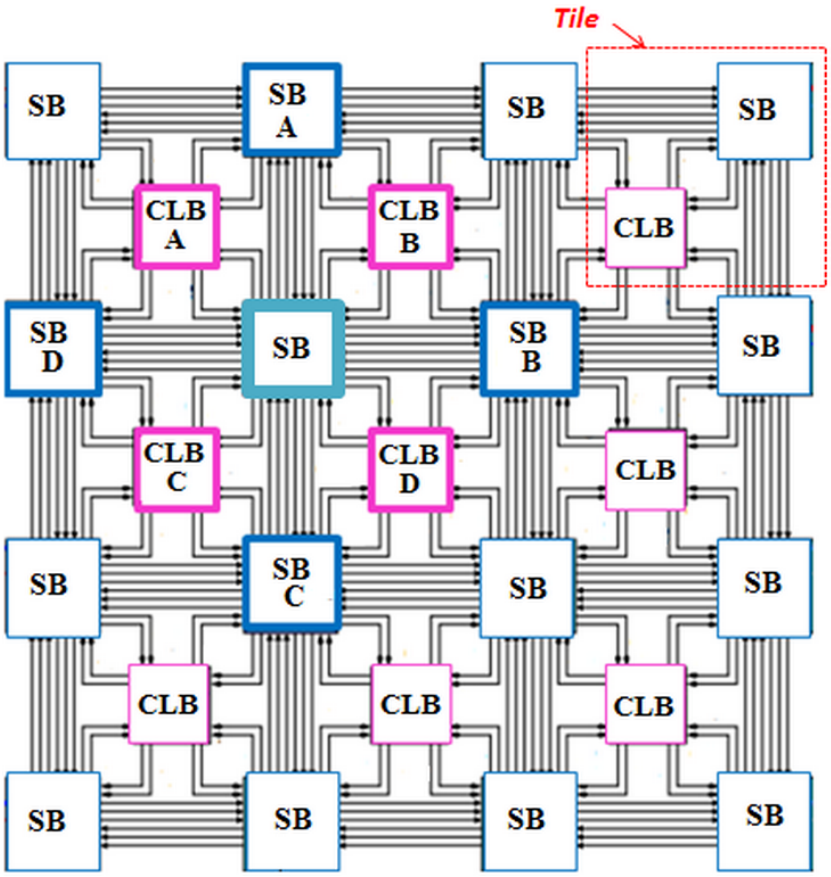

Figure 2. Cluster and logic block architectures.

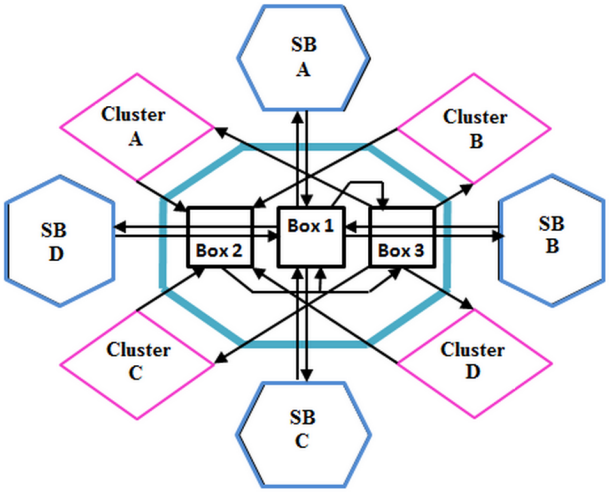

Figure 3. Multilevel SB interconnect.

3. Quality metrics models

In the following subsections, we present the proposed power, area, and timing models. These models are designed to work closely with the target FPGA architecture.

3.1. Power model

To estimate the power dissipation, we use 3 modules: activity estimation, architecture generation, and low-level power estimation. The activity estimation module determines the activity information of all nets in the design. This activity information consists of two values: the signal probability and the switching activity. We used the ACE-2.0 activity estimation tool [12], which combines both probabilistic and simulation techniques to address these weaknesses in previous activity estimation tools and to provide better accuracy results. The architecture generation module decomposes the entire cluster-based FPGA circuit into low-level components (inverters, simple multiplexers, and wires). As detailed in Section 2, the proposed cluster-based FPGA architecture is based on MSBs, LBs, and buffers:

- MSBs are a fully populated crossbar. An MSB with m-input and n-output contains n instances of m-to-1 multiplexers. Each multiplexer is implemented with 2-to-1 multiplexers. Figure 4 presents how a 4-to-1 multiplexer is implemented with 2-to-1 multiplexers.
- LBs are composed of a LUT followed by an FF. As illustrated in Figure 5, LUTs are built with 2-input multiplexers. Level restoring buffers are inserted to reconstruct data [13]. FFs are implemented with 2 inverters and a 2-input multiplexer [14].
- Buffers are implemented as multiple stages of cascaded inverters (see Figure 6). The size of the final buffer stage S is determined using Eq. (1) derived from the logical effort model [14]:

$$S = \frac{1}{4} \times \frac{C_L}{C_{inv}}, \tag{1}$$

where C_L is the load capacitance, which includes the wire length and fanouts, and C_{inv} is the input capacitance of a minimum-sized inverter.

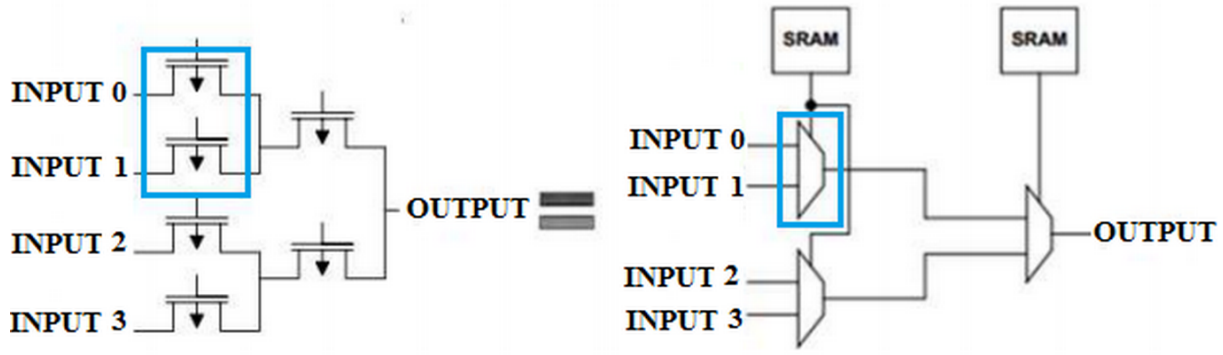

Figure 4. Low-level modeling of a multiplexer with 4 inputs.

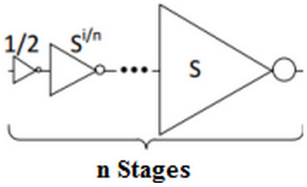

Figure 5. Low-level modeling of a LUT with 4 inputs.

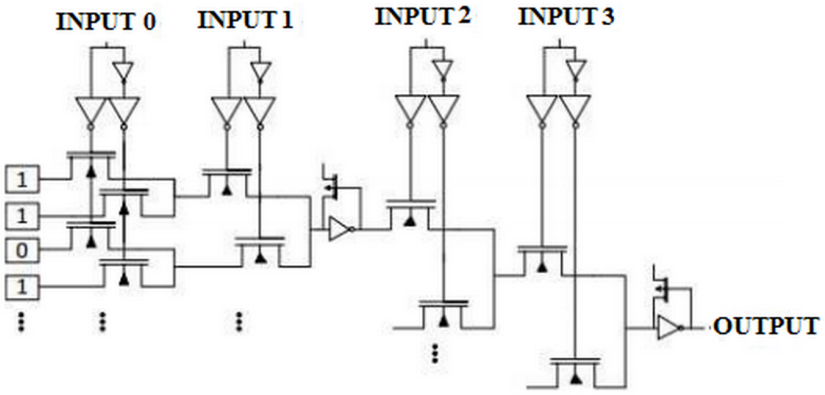

Figure 6. Low-level modeling of a buffer.

Once FPGA components are decomposed into low-level components, the low-level power estimation module estimates the power dissipation at transistor level using equations and rules defined in the latest mesh FPGA power model called VersaPower [15]. The total power dissipation comes from two sources: dynamic and static powers. Dynamic power is dissipated every time a signal switches due to the charging and discharging of load and parasitic capacitances. Dynamic power is calculated for every low-level component using Eq. (2), where f is the clock frequency of the circuit and $Vswing$ is the voltage swing. Vdd is the supply voltage, C is the node capacitance being charged and discharged during each transition, and α is the individual transition density or switching activity at each node.

$$P_{dyn} = 0.5 * C * Vswing * Vdd * \alpha * f \tag{2}$$

Static power is estimated as the sum of subthreshold and gate leakages [14]. Subthreshold leakage is computed using this equation: $Vdd * P * I_{st}$, where I_{st} is the subthreshold current of the transistor and P is the signal probability. Gate leakage power is computed using the equation $Vdd * (1 - P) * I_g$, where I_g is the gate leakage current of the transistor and P is the signal probability.

3.2. Area model

This section describes the area model used to compute the density of the FPGA architectures under investigation. Discussions with FPGA vendors have revealed that routing area is transistor-dominant and not wiring-dominant [4]. In this work, we develop a new area model that accurately computes the total number of transistors. By looping through all nodes in the routing resource graph, we can identify the type of switches (buffer and/or m-to-1 multiplexers inside a crossbar), type of wires (local or global), and primitives (LUTs and FFs within LBs) used for that particular connection. Then we extract from the node record stores all information needed for the decomposition of switches or primitives at transistor level by using the same architecture assumptions presented in Section 3.1. The area is expressed as a function of λ equal to the half of the minimum distance between the source and drain of the transistor.

3.3. Delay model

The performance (system frequency) of an implemented circuit is estimated by calculating the delay of circuit elements along its slowest path (critical path). Wire length and switch delays depend respectively on physical layout and cell library

characteristics. The delay of these elements can be precharacterized since they are independent of placement and routing phases. The process begins with the RTL description of the target FPGA generated using an HDL generator. Then we use a cadence design compiler to compile VHDL into Verilog. The compiled Verilog is used as input into Cadence Encounter to perform physical design layout generation and to extract layout parameters (wires length, capacitances ...). The physical design experiments are performed using the layout generated using ST Micro's 130 nm technology node. Finally, the ELDO circuit simulator is used to obtain highly accurate wires and switch delays estimation based on extracted layout parameters.

4. Exploration results

The best way to choose the FPGA cluster granularity (LUT and cluster size) is to experimentally implement benchmarks on the target architecture with different LUT and cluster sizes and compare the resulting performance. To conduct this exploration, we use Microelectronics Center of North Carolina (MCNC) [16] benchmarks. These circuits cover various application types with several sizes and in/out pads number. Figure 7 illustrates the experimental block diagram to implement design circuits on the proposed FPGA. First, we use the T-VPack tool [17] to group LBs of given circuit design into clusters according to given cluster inputs and cluster size values. Then we implement the simulated annealing placement algorithm [4] to determine the physical locations of the circuit's clusters on the FPGA's clusters. Then we implement the PathFinder routing algorithm [18] to determine how LBs' connections are realized within the prefabricated routing interconnect in a manner such that no routing resource is shared by more than one net. In experimentation, we used the MCNC benchmark. For each circuit, we determine the minimum required channel width (*Wmin*) to implement the circuit. Once a circuit is implemented on the FPGA, for a given cluster and LUT size, we determine the resulting power dissipation, area density, and delay using quality metric models described in Section 4.

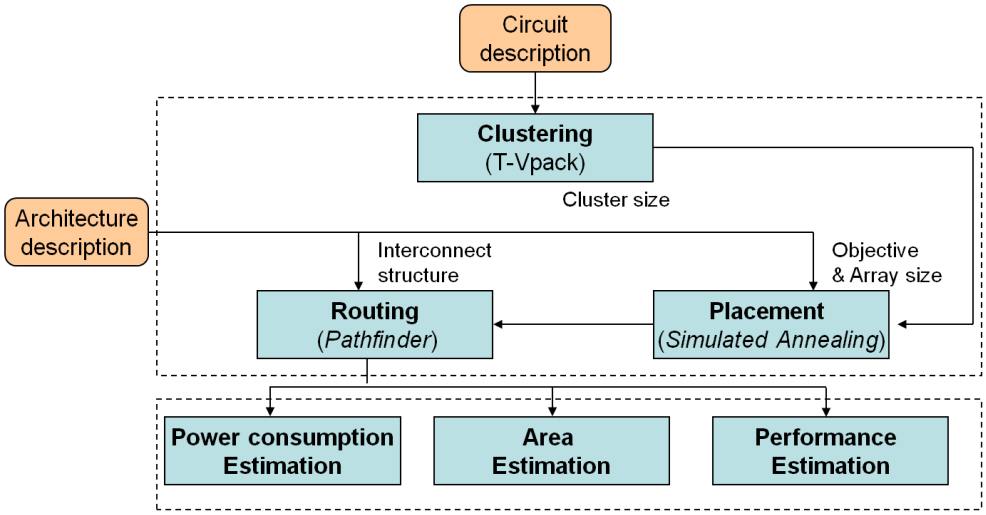

Figure 7. Block diagram of circuit implementation on proposed FPGA architecture.

4.1. Cluster size

In this section, we present and discuss cluster size effect on power dissipation, area density, and delay. Table 1 illustrates the variation in FPGA size, channel width (*Wmin*), and total buffer with cluster size. By increasing the cluster size, the total clusters number decreases and as consequence a smaller FPGA size is required to

implement circuits. Nevertheless, a higher *Wmin* is required to successfully route the circuit within the smaller FPGA. Consequently, the total intercluster interconnection is reduced, which reduces the total buffers number.

Table 1. FPGA size, channel width (Wmin), and total buffer.

Cluster size	FPGA size	Channel width (Wmin)	Total buffer
4	49 × 49	30	147000
8	36 × 36	46	122544
16	28 × 28	60	97440

Figure 8 shows the variation in the total, routing, and logic powers with cluster sizes. We remarked that the total power dissipation is reduced with the increase in cluster size. As shown in Figure 8, clusters of size 8 and 16 are more power efficient than 4. This result is due to the fact that the total number of buffers is reduced (see Table 1). In fact, buffers denote one of the major factors behind power dissipation [15]. We note also that total powers of cluster size 8 and 16 are quite close. To analyze this behavior, we need to analyze routing and logic powers variation. By increasing cluster size, we increase cluster multiplexer number to connect LBs and consequently increase the total logic power. That is why total logic power of cluster size 16 is higher than that of cluster size 8. However, with larger cluster size, we can absorb a larger number of nets and communication becomes local and then the total number of routing interconnects decreases. Consequently, the total routing power of cluster size 16 is lower than that of cluster size 8. These two opposite effects make the total power of cluster size 8 and 16 total quite close.

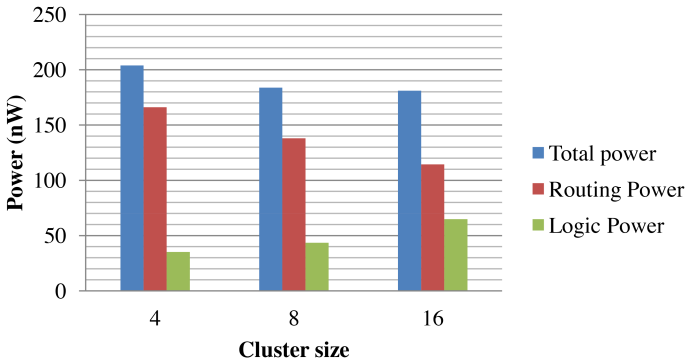

Figure 8. Power consumption for different cluster sizes.

Figure 9 illustrates the variation in the total, routing, and logic areas with cluster sizes. By increasing the cluster size, the area efficiency goes down. As shown in Figure 9, clusters of size 4 and 8 are more area efficient than 16. In fact, when we increase cluster size, the number of multiplexers used within intracluster interconnects grows greatly and then switch number increases and affects the area efficiency. We note also that total areas of cluster size 8 and 16 are quite close. In fact by increasing cluster size, the total number of 2-to-1 multiplexers and buffers used in routing interconnects decreases and then total routing area decreases. Thus, the routing area of cluster size 8 is lower than that of cluster size 4 (see Figure 9). However, when we increase cluster size, we increase the number of LBs and then the total number of multiplexers within a cluster grows larger and consequently the logic area efficiency goes down. Therefore, total logic power of cluster size 8 is higher than that of cluster size 4 (see Figure 9). These two opposite effects mean that clusters 4 and 8 have close resulting numbers of switches and then total area.

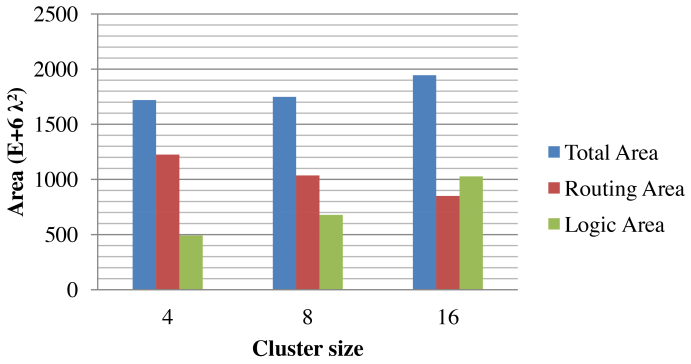

Figure 9. Area for different cluster sizes.

In terms of performance, Table 2 shows the variation in critical path delay with cluster sizes. The critical path delay decreases when we increase cluster size. In fact, using larger cluster size allows reduced external communications and then reduces the number of crossed switches in the critical path.

Table 2. Critical path delay vs. cluster size.

Cluster Size	Delay (ns)
4	62.504
8	55.255
16	46.633

4.2. LUT size exploration

In this section, we explore the effect of LUT size of the proposed cluster-based FPGA. We implemented circuits on architecture with LUT size varying between 3 and 7 and cluster size varying between 4 and 8. Figures 10 to 12 show respectively the variation in power, area, and delay for different cluster and LUT sizes.

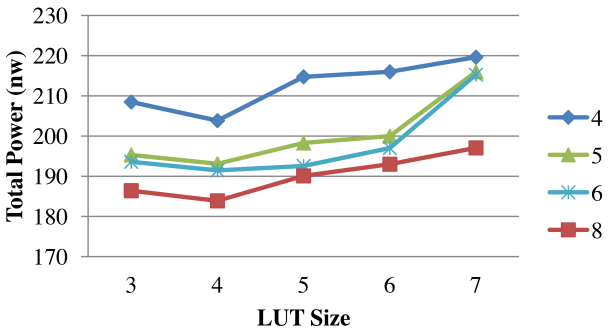

Figure 10. Power for different LUT and cluster sizes.

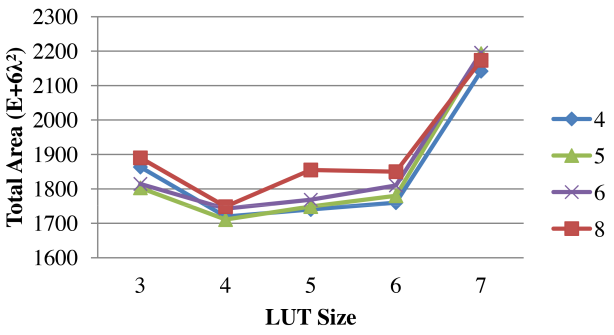

Figure 11. Area for different LUT and cluster sizes.

In term of power dissipation, we note from Figure 10 an improvement in power dissipation with the increase in cluster size, which confirms the results and discussions in Section 4.1. We note also that, compared to LUT size 3, architecture with LUT size 4 offers better results in term of power. Afterwards, starting from LUT size 4, increasing the LUT size increases the power dissipation. In fact, with small LUT size, the logic power denotes lower than 30% of the total power; however, with high LUT size the total logic power becomes an important factor and as a consequence we get the upward trend in Figure 10. According to conducted experimentations, the best low-power solution is an architecture with cluster size 8 and LUT size 4.

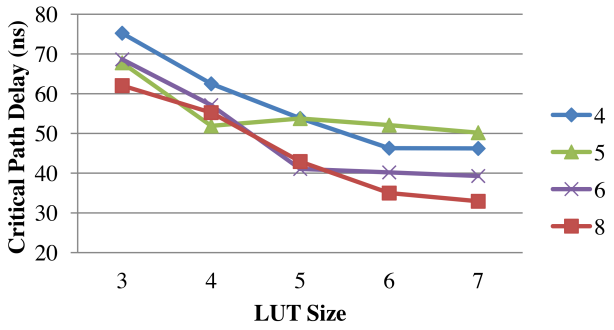

Figure 12. Delay for different LUT and cluster sizes.

We note from Figure 11 that, for all LUT sizes, varying cluster size from 4 to 8 gives close results in term of area, which confirms the results and discussions in Section 4.1 (about area for cluster size 4 and 8). We note that logic area increases exponentially as a function of LUT size. In fact, logic area increases exponentially with the increase in LUT size since the number of multiplexers within LUTs grows greatly. At the same time, a high LUT size permits us to implement more logic functions in each LUT, which reduces the required LUT number. Nevertheless, the decrease in LUT number is slower than the increase in LUT area and as consequence we get the upward trend in Figure 11.

The third key metric is the critical path delay. Figure 12 shows that increasing cluster and LUT size improves the performance of the FPGA. For example, compared to architecture with cluster size 4 and LUT size 3, we get 70% performance improvement with architecture with cluster size 8 and LUT size 7. In fact, the increase in LUT or cluster size allows connecting clusters with lower number of switches, which reduces the number of switches in the critical path and improves performance.

5. Conclusion

In this paper, we propose a new advanced way to design interconnection structures for cluster-based mesh FPGA architectures. We demonstrated that the efficiency of the proposed FPGA in terms of power dissipation, area density, and delay can be controlled by interconnect cluster size and LUT size. In this way, the proposed architecture can be tuned and adapted to satisfy different application constraints and trade-offs. For applications requiring high speed performance and low power dissipation, it is recommended to use clusters with high size (8–16). If we need to reduce silicon area, using small cluster arities seems to be more efficient. We note from experimentation that cluster size 8 presents the best trade-off between area and power compared to cluster sizes 4 and 16. This is achieved thanks to the equitable sharing of resources between logic and routing. Moreover, we have discovered that LUT size of 4 is the most efficient in terms of area and power dissipation. In future work, we plan to introduce heterogeneous logic block heterogeneity for logic function optimization. In fact, modern commercial FPGAs include a large number of hard macro blocks that complement the LUT-based logic blocks. These macro blocks can include embedded memories, adders, multipliers, or DSP blocks and they are designed to perform specific operations with high performance. Using such heterogeneous FPGA architectures may reduce flexibility but improve performance and density and hence it deserves exploration.

References

[1] Kuon I. Measuring and navigating the gap between FPGAs and ASICs. PhD, University of Toronto, Toronto, ON, Canada, 2008.

[2] Gaillardon P, Tang X, Kim G, De Micheli G. A novel FPGA architecture based on ultrafine grain reconfigurable logic cells. IEEE T VLSI Syst 2015; 23: 2187-2197.

[3] Chen D, Cong J, Fan Y. Low-power high-level synthesis for FPGA architectures. In: The International Symposium on Low Power Electronics and Design; 25–27 August 2003; Seoul, South Korea. New York, NY, USA: IEEE. pp 134-139.

[4] Betz V, Rose J, Marquardt A. Architecture and CAD for Deep-Submicron FPGAs. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[5] Singh A, Marek-Sadowska M. Efficient circuit clustering for area and power reduction in FPGAs. ACM T Des Automat El 2002; 7: 643-663.

[6] Touil L, Kechiche L, Ouni B. Design of low power system on programmable chip for video zoom-in processing. Turk J Elec Eng & Comp Sci 2015; 24: 3405-3418.

[7] Ahmed E, Rose J. The effect of LUT and cluster size on deep-submicron FPGA performance and density. IEEE T VLSI Syst 2004; 12: 1063-8210.

[8] Li F, Chen D, He L, Cong J. Architecture evaluation for power-efficient FPGAs. In: The International Symposium on Field Programmable Gate Arrays; 23–25 February 2003; Monterey, CA, USA. New York, NY, USA: ACM. pp. 12-20.

[9] Lemieux G, Lewis D. Directional and Single-Driver Wires in FPGA Interconnect. In: International Conference on Field Programmable Technology; 6–8 December 2004; Brisbane, Australia. New York, NY, USA: IEEE. pp 41-48.

[10] Lemieux G, Lewis D. Design of Interconnection Networks for Programmable Logic. Norwell, MA, USA: Kluwer Academic Publishers, 2004.

[11] Feng W, Kaptanoglu S. Designing efficient input interconnect blocks for LUT clusters using counting and entropy. In: The International Symposium on Field Programmable Gate Arrays; 18–20 February 2007; Monterey, CA, USA. New York, NY, USA: ACM. pp. 23-32.

[12] Lamoureux J, Wilton S. Activity estimation for field programmable gate arrays. In: International Conference on Field Programmable Logic and Applications; 28–30 August 2006; Madrid, Spain. New York, NY, USA: IEEE. pp 1-8.

[13] Hung E, Wilton S, Yu H, Chau T, Leong P. A Detailed Delay Path Model for FPGAs. In: International Conference on Field Programmable Technology; 9–11 December 2009; Sydney, NSW, Australia. New York, NY, USA: IEEE. pp 96-103.

[14] Weste N, Harris D. CMOS VLSI Design: a Circuits and Systems Perspective. New York, NY, USA: Addison Wesley, 2010.

[15] Goeders J, Wilton S. VersaPower: power estimation for diverse FPGA architectures. In: International Conference on Field Programmable Technology; 10–12 December 2012; Seoul, Korea. New York, NY, USA: IEEE. pp. 229-234.

[16] Yang S. Logic Synthesis and Optimization Benchmarks User Guide. Raleigh, NC, USA: Microelectronics Center of North Carolina (MCNC), 1991.

[17] Marquart A, Betz V, Rose J. Using cluster-based logic block and timing-driven packing to improve FPGA speed and density. In: The International Symposium on Field Programmable Gate Arrays; 21–23 February 1999; Monterey, CA, USA. New York, NY, USA: ACM. pp. 37-46.

[18] McMurchie L, Ebeling C. Pathfinder: a negotiation-based performance-driven router for FPGAs. In: The International Symposium on Field Programmable Gate Arrays; 12–14 February 1995; Monterey, CA, USA. New York, NY, USA: ACM. pp. 111-117.