The Journal of Supercomputing (2022) 78:17689–17720
https://doi.org/10.1007/s11227-022-04569-2

Development and Analysis of Novel Mesh of Tree-based embedded FPGA

Hajer Saidi[1,2] · Mariem Turki[1] · Zied Marrakchi[3] · Mohamed Abid[1,2] · Abdulfattah Obeid[4]

Accepted: 27 April 2022 / Published online: 22 May 2022

Abstract

The eFPGA IPs are made up of logic components connected by a routing network. The target architecture is a key feature of eFPGA development. There have been two main families of architecture: matrix and hierarchical topologies. The mesh architecture is distinguished by its genericity and regularity, but approximately 90% of the area is used by the routing network and just 10% by the logic blocks. Hierarchical architecture reduces this effect by on average 56% but increases the size of the critical path and causes the scalability problem. The architecture proposed in this paper will mix the benefits of the two existing architectures. This paper, therefore, proposes a Mesh of Tree architecture that maintains a strong balance between area density and layout scalability. To the best of our knowledge, this is the first eFPGA circuit with a mixing matrix and hierarchical architectures in a new eFPGA architecture. We compared the proposed eFPGA by Tree-based and Mesh of Cluster eFPGA in terms of area, power dissipation, performance and frequency. Mesh of Tree eFPGA imposes an area overhead but has a straightforward advantage in terms of performance for architectures with a size greater than 64 LUTs. The results of the experiments demonstrate that the proposed Mesh of Tree architecture has strong physical scalability: Once the layout of the nodes is generated, it can be used to create matrix layouts of the target size and shape factor.

Keywords eFPGA · Re-configurable · Hierarchical · Mesh of tree · Architecture

✉ Hajer Saidi
hajer.saidi@enis.tn

[1] CES Research Laboratory, National Engineering School of Sfax, Sfax, Tunisia

[2] Digital Research Center of Sfax, CRNS, Sfax, Tunisia

[3] Mentor Graphics, A Siemens Business, Tunis, Tunisia

[4] National Electronics and Photonics Technology Center, King Abdulaziz City for Science and Technology, KACST, Riyadh, Saudi Arabia

1 Introduction

Nowadays, designers may include an increasing amount of functionalities due to the excessive growth of silicon technology. Thus, the growth in nonrecurring engineering (NRE) cost related to complexity becomes a significant factor in system-on-a-chip (SoC) design, potentially limiting scaling prospects [14, 15]. As a result, adopting design solutions capable of reducing such expenses while maintaining high performance causes an issue. IP-reuse methods are a prevalent method of reducing design and verification costs by reusing predesigned and preverified synthesizable IPs. This enables designers to improve the implementation process of the IP for the particular goal in terms of area, speed, and power, while also providing good time-to-market and risk reduction.

The integration of embedded configurable modules has the potential to enlarge the SoC market by broadening its application scenario and extending its lifetime. In all situations, the flexibility enables the spreading of NRE expenses among an increasing number of products.

In this context, the use of field-programmable devices in the form of embedded field-programmable gate arrays (eFPGA) cores is an interesting alternative approach that can offer bit-level optimization for applications that benefit from synthesis [4, 12]. eFPGA cores must be a modest but effective portion of the larger system that adds actual value to a specific area of the system.

eFPGA cores are often delivered as fixed-size hard macros optimized by customized designs for types of applications, as reported in the majority of studies [1, 22]. The common downside of the studied eFPGA is that the circuit specifications cannot be changed after fabricating in terms of architecture, number of wires, number of input and output pads, and in particular the number of LUTs, since the proposed eFPGAs are hardcore devices or fixed soft IPs. In addition, almost all of these eFPGAs are mesh-based architectures, which means that the occupation of Logic Networks is very small relatively to the total size of the core. As a result, this affects power, performance, and the area. Our previous recent works presented in [26, 28] propose a hierarchy re-configurable co-processor, with fine-grained architecture and which can implement several algorithms in specific times. This co-processor accelerates the execution time with an acceptable trade-off between several constraints, especially, flexibility, time-to-market, power, area, and performance. The number of look-up-tables (LUTs), registers, and the number of I/O ports can be controlled when designing an eFPGA to make trade-offs between power and performance [9, 20]. Tree architecture has better area density than the traditional VPR-Style clustered Mesh. Tree-based architecture efficiency in terms of area, performance, and static power can be managed by Interconnect Rent parameters, cluster arity, and LUT size. In this context, this architecture can be modified and adapted to specific applications to accommodate different trade-offs. However, this tree-based topology may be penalized in terms of the generation of physical layouts and does not allow scalability since it does not have a flat chip structure, particularly for large circuits [22].

In summary, each one of the various embedded FPGA architectures introduced in the state of the art has its advantages and drawbacks. The matrix architecture is characterized by its genericity that enables the use of eFPGA generators, but 90% of the area is used by the interconnection network [8] and only 10% for logic network. The tree design reduces this effect by 56% but increases the size of the critical path and penalizes the scalability of the circuit [30]. The architecture proposed in this paper will have to combine the advantages of the two previous architectures in order to obtain the best performance (i.e., the genericity of a matrix architecture by reducing the interconnection utilization rate compared to logic thanks to a tree structure).

The Mesh of Tree architecture was chosen to incorporate both the theoretical aspect of the state-of-the-art analysis especially for mesh architecture and the functional respect of architectural exploration of Tree-based topology obtained through the implementation of software tool. Trying to combine the advantages offered by all of the mentioned topologies, the architectural choice considered in this analysis will be discussed in the following sections. Usually, the efficiency of the topology is determined by both the architecture and the routing algorithm. However, in this study, we concentrate on architecture performance.

The main contributions of this paper are the following:

1. Propose a clustering method for the novel Mesh of Trees eFPGA. Besides, a decoder model that loads the bitstream description file to SRAM cells is also proposed. With these two contributions the development of softcore Mesh of Tree eFPGA is ensured.
2. An exploration and evaluation of the proposed architecture (Mesh of Tree eFPGA) is performed.
3. First, a full comparison between Mesh of Tree, Mesh of Cluster, and Tree-based architectures in terms of hardware utilization, regularity, power, performance, and latency is provided. Then, existing eFPGAs in both industrial and literature have been compared to the proposed eFPGAs in this research.

The remainder of the paper is organized as follows: Section2 presents an overview of embedded FPGA architecture and describes the related works. Section 3 details the proposed embedded FPGA and the configuration decoder. Section 4 presents firstly the hardware implementation of 32 LUTs Tree-based embedded FPGA, as well as the 32 LUTs of Mesh of Tree eFPGA architecture and an alternate architecture based on Mesh of Cluster topology. A wide comparison between the mentioned architectures is made. Then, the implementations and experimental results are presented. Finally, Section 5 concludes the paper.

2 Related works and background

Several works are interested to the eFPGA development and integration. They aim to design the programmable devices with the best characteristics in order to integrate the IP into system which makes the ASIC more flexible. In this section, we outline the different existing solution's of eFPGA development.

2.1 Classic mesh architecture

Mesh-based FPGA are also called island-style FPGA. As shown in Figure 1, logic blocks are typically arranged in a grid and are surrounded by horizontal and vertical routing channels [16]. Mesh architecture is most common among academic and commercial FPGAs. FPGA is defined as an integrated circuit including a network of programmable cells. Each cell can perform a function. The interconnections are also re-configurable. Most FPGA components are mainly based on SRAM technology. The logic blocks in FPGAs are based on LUTs to implement any Boolean function with 4, 5, 6 inputs with one or two outputs.

Configurable logic block (CLB): It is a logical configurable part in the FPGA; the logical component of a user circuit is implemented in these blocks [19]. The output of a logic block depends on the logic function implemented into the block, but also on the inputs of this block. Since FPGAs must be flexible enough to implement any user design, FPGA logic blocks must be able to implement a wide range of logical functions. **Switch block (SB):** A switch box, it is a set of switches used to connect the routing channels. It is located at each intersection between horizontal and vertical routing channels and defines all possible

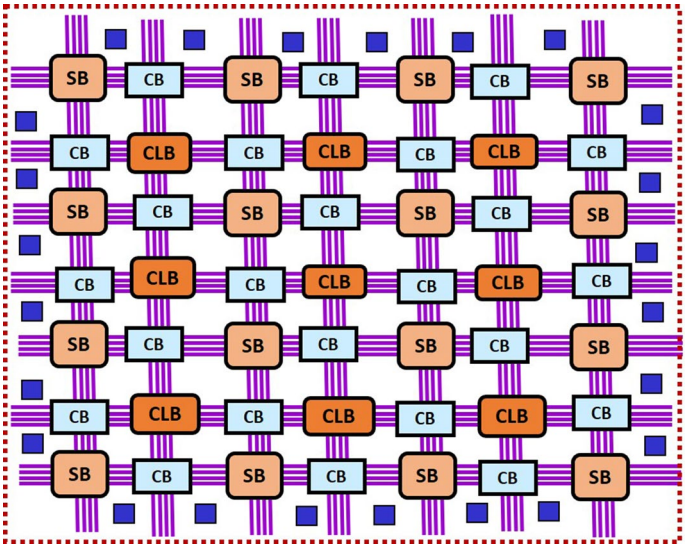


Fig. 1 Mesh-based architecture

connections between these channels. Three different typologies for Switch Box can be distinguished, but the most used is the Disjoint Switch Box [19]. In the Disjoint Switch Box case, each path can be connected to all paths of the same index. This limits routing channel where they can follow only specific paths. **Connection block (CB):** A connection block is a set of switches used to connect a logic block to an adjacent routing channel. These blocks surround CLBs to ensure the connection between its inputs/outputs with routing paths [19]. Unlike the switch boxes, a connection block gives less flexibility to routing channels but also can connect near CLBs with fewer wires and switches.

2.2 Tree-based architecture

As outlined in our previous works [24, 26], the Tree-based eFPGA architecture [20], **Logic Blocks (LBs)** are organized into clusters and each cluster includes a switch block for connecting local LBs. The switch block is divided into Mini-Switch Blocks (MSBs). The Tree-based FPGA architecture incorporates two unidirectional upstream and downstream interconnection networks by using the **BFT (Butterfly fat-tree Topology)** topology to link **Downward Mini Switch Box (DMSBs)** and **Upward Mini Switch Box (UMSBs)** to LB inputs and outputs.

As shown in Figure 2, UMSBs are used to allow LB outputs to reach a large number of DMSBs and to decrease the fanout on feedback lines. UMSBs are arranged in such a way as to allow LBs belonging to the same owner cluster to reach precisely the same group of DMSBs at each level. Thus, positions within the same cluster are identical, and LBs can coordinate with their siblings for the use of a greater number of DMSBs based on their fanout sizes. As seen in Figure 2, the programmable interconnect of the Tree-based FPGA architecture is structured in a multi-level network with switch blocks located at various tree levels using the BFT network topology.

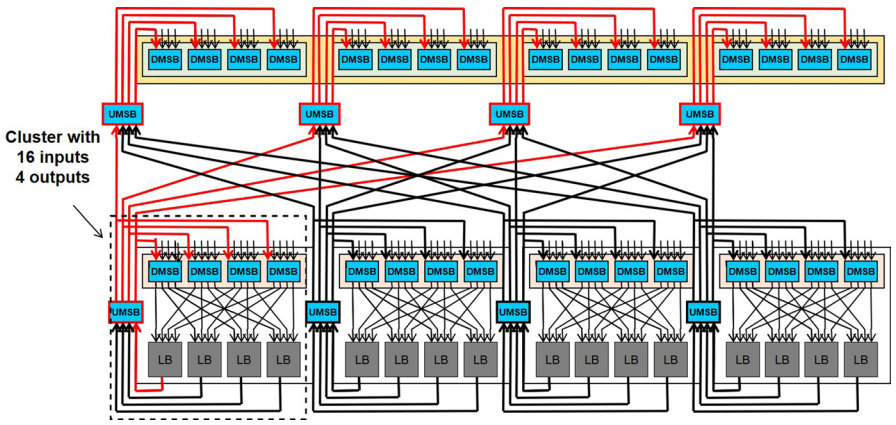


Fig. 2 Two-level tree-based architecture

2.3 Mesh of cluster architecture

The Mesh of Clusters architecture is composed of a 2D matrix of tiles, and each tile is composed of a cluster containing the logical elements and multi-level interconnection blocks (switch boxes) [3, 6, 27], as illustrated in Figure 3.

This architecture allows firstly to take advantage of the regularity of modern mesh architecture to connect the clusters to the interconnection blocks and then to gain the area in the interconnection of the logical elements of the clusters to switch boxes interconnection using a multi-level topology. Contrary to mesh architecture, this architecture does not use connection blocks. Each switch box is directly connected to the clusters and the neighboring switch boxes in a specific order.

2.4 Research trends and proposed approach

The interconnecting arrangement of the Mesh-based FPGA typically aimed to improve the use of logic. Hierarchical architectures belong to the family of FPGA designs that are built to maximize interconnect usage at the expense of logical utilization.

The theory behind hierarchical architectures is to maximize the use of silicon by effective use of the interconnected structure (which can account for 80% to 90% of the total area of Mesh-based FPGAs). The Tree-based FPGA architecture uses two unidirectional interconnecting networks to connect LB inputs and outputs to SBs. Research shows that using the Tree-based FPGA architecture reduces the number of switches used by 56% when compared to Mesh-based FPGA architectures. The key problems with the Tree-based interconnect structure are that the path delay increases exponentially as the Tree grows to higher levels. In addition, the concept behind the Mesh of Clusters architecture, the architecture presented in [27], has been strengthened in terms of power consumption and area relative to the reference VPR FPGA cluster-based mesh. Based on our previous experimentations [27], a gain of 30% and 32%, respectively, in terms of power consumption and area is noticed. Most of the reductions in power and area were attained by a reduction of 24% of the necessary buffer switches to efficiently route various circuit benchmarks [7]. Nonetheless,

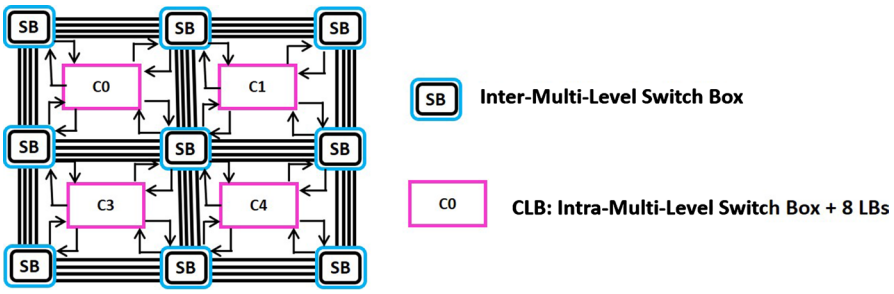


Fig. 3 Mesh of cluster architecture

Mesh of Cluster architecture had an average performance degradation of 12% compared to the VPR [3].

However, in this paper, we proposed a different embedded FPGA architecture with updated interconnect architecture and matrix node based on trees. The solutions to improve eFPGA scalability are addressed with a view to maintain better logic utilization and better layout generation using the new topology mentioned in the following sections.

3 Mesh of tree topology

The interconnect arrangement on typical eFPGA architectures is commonly designed to optimize the use of logic. A completely filled routing interconnect is easy and offers high flexibility at the expense of power and area overhead. In fact, the consumption rate of interconnecting switches is remarkably low. Therefore, in order to make embedded FPGAs more effective, we wish to investigate a new eFPGA architecture based on the matrix of Trees. Inspired by [3, 10, 21], we suggest the use of a multi-stage switch box for inter- and intra-cluster connections, based on the use of the Butter-Fat-Tree (BFT) geometry inspired by the Tree-Based structure itself [7]. The proposed Mesh of Tree architecture is a clustered matrix positioned in a standard 2D grid connected to a unidirectional routing network. Each cluster is often a tree that can be constructed from one level to several levels.

The Mesh of Tree architecture consists of a 2D tile matrix. Every tile consists of a tree containing the logical elements and the interconnection network (DMSBs + UMSBs) (illustrated in Figure 4).

This architecture allows, on the one hand, to take advantage of the regularity of the mesh to connect the trees and interconnection blocks to each other (each Tree is uniformly connected to 4 Switch Boxes, which are themselves linked to each other by the Channel Width (CW) in the form of rows and columns) and, on the other hand, to gain in the area at the level of the interconnection of the Trees and in the interconnection of the Multi-Level Switch Boxes, by using a tree architecture.

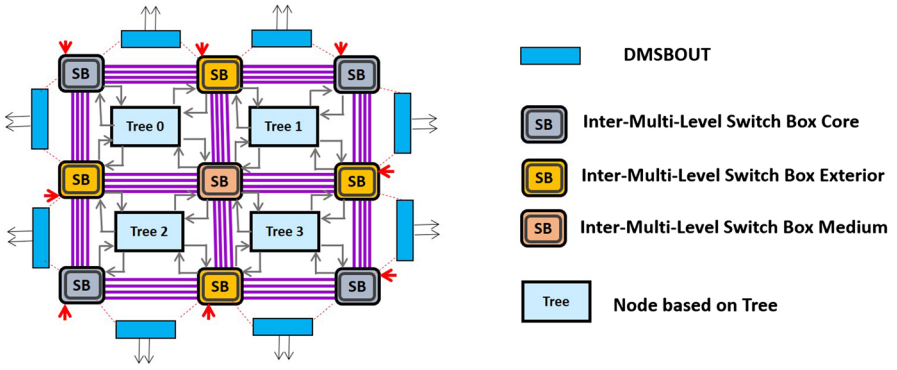


Fig. 4 Mesh of tree architecture

Unlike the Mesh architecture, we do not use Connection Blocks. Each Switch Box is directly connected to clusters and neighboring Switch boxes in a specific order. As shown in Figure 4, the inputs/outputs of a Switch Box come from the 4 adjacent Switch Boxes as well as from the 4 adjacent Trees. Thanks to this configuration, each Tree can be connected to the 8 adjacent Trees using its adjacent Switch boxes.

3.1 Tree local interconnect

Mesh of Trees are formed of Logic Blocks (LBs) that communicate within a programmable local interconnect. The intra-cluster interconnect is arranged as a Tree and based on the topology previously described in Section 2. The Logic Blocks are clustered into clusters located at the lowest level in a tree-based FPGA architecture [21]. Each cluster includes a switch block for linking local LBs. Switch blocks are composed of Mini Switch Blocks (MSBs). As shown in Figures 5 and 6, the Tree-based FPGA architecture unifies two unidirectional down and up interconnecting networks that use the Butterfly Fat-Tree topology to relate Downward MSBs (DMSBs) and Upward MSBs (UMSBs) to LB inputs/outputs and upwards MSBs (UMSBs).

1. **Downward interconnect network:** The downward network is influenced by SPIN [11]. It is built on the Buttery Fat-Tree (BFT) style of interconnecting [18] with linearly populated and unidirectional switch boxes. Tree leaves are consistent with

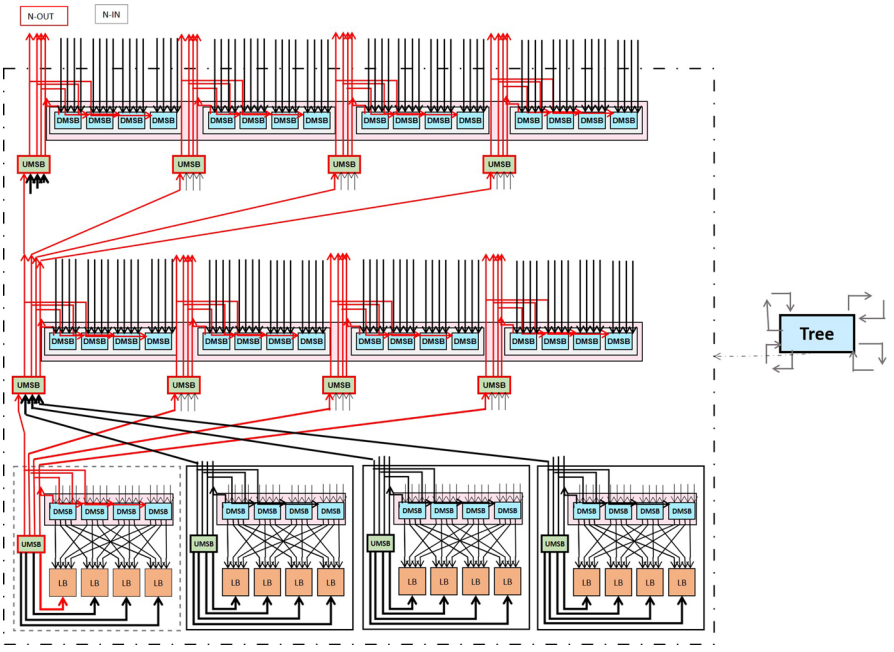


Fig. 5 Tree architecture: an example

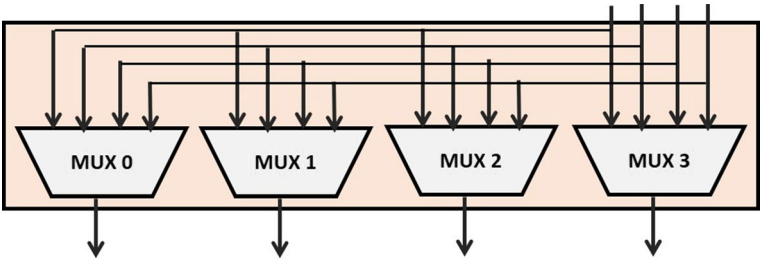


Fig. 6 MSB Architecture

logic blocks. Every DMSB joins every LB in one input, and thus, the number of DMSBs is represented by Equation 4.1 and the number of DMSB outputs is calculated by Equation 4.2.

$$nbDMSB(l) = Nin(l-1). \tag{1}$$

$$nboutDMSB(l) = Nslaves(l). \tag{2}$$

where

- l : Level number
- Nslaves: Number of slaves
- nbDMSB: DMSB number
- nboutDMSB: Output number of DMSB

2. **Upward interconnect network:** The upward network links the outputs of the logic blocks and the input Pads to the various levels of the tree structure. As a result, LBs within the same cluster are identical and their arranging does not affect the routing efficiency. The number of UMSB and inputs of UMSB is given by Equations 4.3 and 4.4 .

$$nbUMSB(l) = Nin(l-1). \tag{3}$$

$$InUMSB(l) = Nslaves(l). \tag{4}$$

where

- l : Level number
- Nslaves: Number of slaves
- nbUMSB: UMSB number
- InUMSB: Number of UMSB's input

3.2 Routing strategy: mesh interconnection

This section is dedicated to detail the routing network of the proposed Mesh of Tree (MoT) architecture.

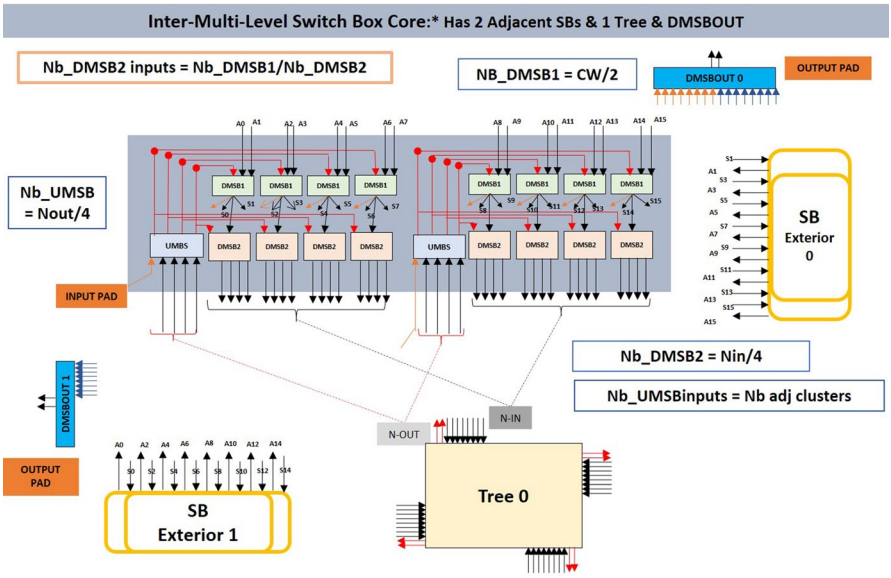


Fig. 7 eFPGA switch box core interconnection

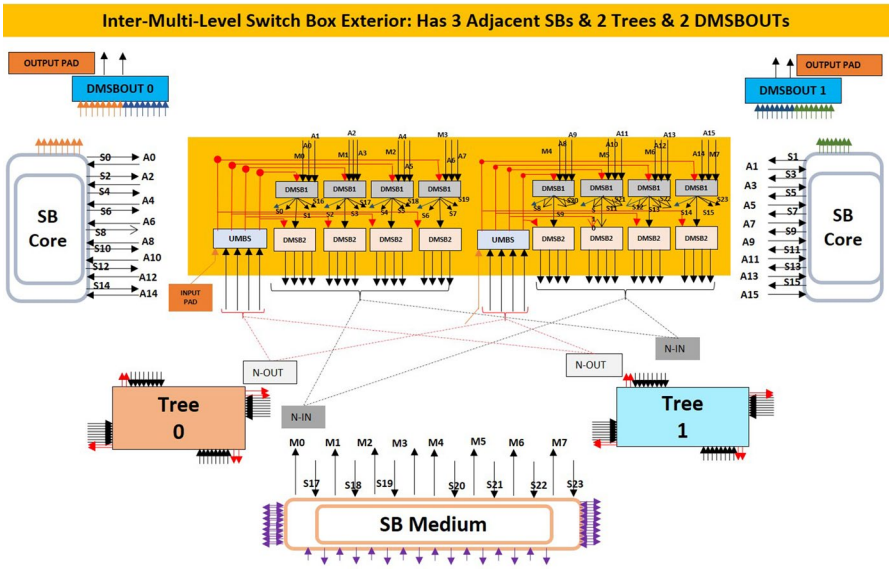


Fig. 8 eFPGA switch box exterior interconnection

Multi-level switch boxes architecture: The most essential element of the routing network of the proposed embedded FPGA is the Switch Box (Figures 7, 8, and 9). A Switch Box is composed of 2 specific blocks: an Up Mini Switch Box (UMSB) and 2 levels of Down Mini Switch Box (DMSB). These interconnection blocks are

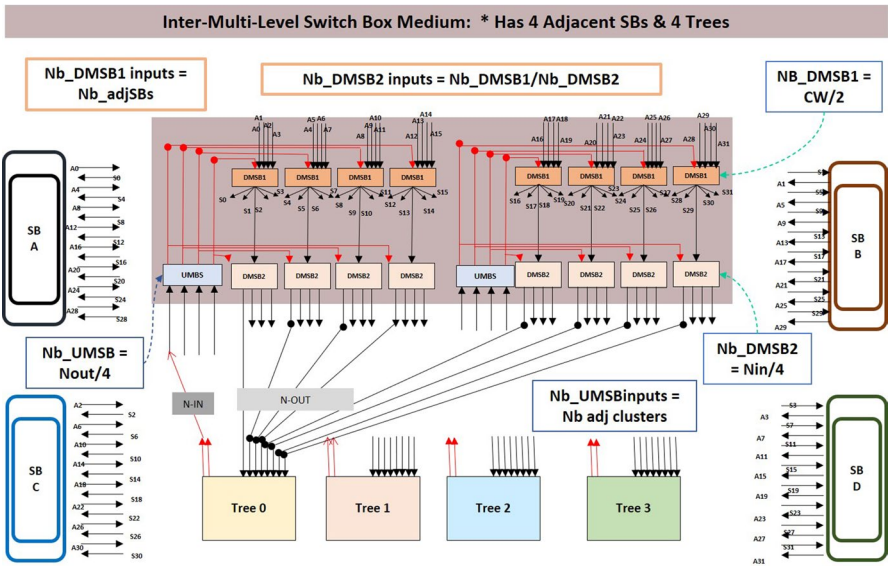


Fig. 9 eFPGA switch box medium interconnection

composed of multiplexers. The DMSBs are used to connect a Switch Box to (a) its four neighbors, (b) to DMSBOUT which is connected to outside in the case of SB-Core and SB-Exterior and to (c) a DMSB within the same structure. The number of DMSBs at each level is independent but depends on the architecture parameters. To allow the connection between 2 clusters using the same Switch Box, a UMSB is used. The interconnection blocks (DMSBs) are linked together in a specific order. The first (leftmost) block of the 1st level is connected to the input of the first DMSB of the 2nd level, the next block to the input of the second DMSB of the 2nd level, etc. Finally, the interconnection network follows a BFT topology, which allows to connect all the inputs to all the outputs of the switch box as shown in the following equations. The number of each DMSB/UMSB or inputs/outputs is calculated from the architecture parameters :

$$NB\_DMSB1 = CW/2 \tag{5}$$

$$Nb\_DMSB2 = Nin/4 \tag{6}$$

$$Nb\_DMSB1\_inputs = Nb\_adj\_SBs \tag{7}$$

$$Nb\_DMSB2\_inputs = Nb\_DMSB1/Nb\_DMSB2 \tag{8}$$

$$Nb\_UMSB = Nout/4 \tag{9}$$

$$Nb\_UMSB_inputs = Nb\_adj\_clusters \tag{10}$$

where

- Nb_DMSBi is the number of DMSBs at level i.
- Nb_DMSBi_inputs is the number of inputs for a DMSB at level i.
- CW (Channel Width) is the size of the routing channel.
- Nb_adj_SBs is the number of adjacent switch boxes.
- Nin is the number of inputs per cluster.
- Nout is the number of outputs per cluster.
- Nb_adj_clusters is the number of adjacent clusters.
- Nb_UMSB is the number of UMSBs per switch box.

Figures 7, 8, and 9 provide a detailed view of the Multi-Level SB Topology linkage where neighboring SBs are duplicated in order to clearly present the hierarchical topology and the interconnection of the multiple SB interconnection measures. Compared to the SB in the existing FPGA architecture, we incorporate a multi-stage SB that links each DMSB1 to a separate tree node in order to increase flexibility and to break the dependency between the channel width and the tree input parameters.

Figure 7 provides a clear view of the SB interconnect located in one of the four corners of the eFPGA. Figure 7 shows the two neighboring SBs (Right SB Exterior and down SB Exterior), two DMSBOUTs that ensure contact with the outside of the eFPGA, and one adjacent Tree0 (outlined in Figure 5). Interconnections within the SB are ensured by two stages of a downward network and one stage of upward interconnection. Figure 8 provides a view of the Multi-Level SB Exterior interconnection topology, which is surrounded by 2 SBs, 2 Trees, SB Medium, and 2 DMSBOUTs. The input and output pads are located perimetrically. In fact, the input pads are attached to the UMSBs in each of the SB Exterior and SB located in the corner. The output pads of the DMSBOUTs are connected to the adjacent switch boxes. Thus, both input and output pads will reach any LB of the Tree cluster. Figure 9 details the multi-stage SB Medium, which requires a big number of interconnectors since it is located between all of the eFPGA components. Despite this, there is no outside connection involved. SB Medium lanked by 4 SBs of the exterior and 4 Trees.

3.3 Loader configuration

This part describes a configuration loader. This model aims to configure SRAM cells of the Mesh of Trees eFPGA components. The configuration model receives the "address + data" from external (can be a processor or external flash memory). It selects firstly the component address, and then, it transfers data to the targeted component. The component can be a DMSB, an UMSB, or an LB of the tree architecture. Threading technology and parallelism processes are used to apply the address selection to accomplish the configuration word transfer.

The proposed architecture has a matrix which is the highest level in RTL design. The main role of the decoder is to attend an element by firstly selecting the top component where it is located. In fact, the configuration begins by component selection for example: Tree "000," SB-exterior "010," SB-core "001," SB-Medium "011," or DMSBOUT "100," as presented in Figure 10. The address passes through the top-level loader and the strobes are emitted to the boxes or Tree. The address is divided into two parts MSB (Most Significant Bit) which indicates the element index and the LSB (Less Significant Bit) which indicates the number ID of the targeted element. If the selector of the loader is equal to 1, the address components are decoded and it is determined to which component it corresponds.

An example is presented in Figure 10 to show the configuration of DMSB-Level 0 in node 1 (Tree has 3 levels, each level has 4 slaves). This DMSB will be configured with the data word "01000111." The configuration word contains two parts: the address that specifies the target element and the configuration data.

– **Address**: "0000100001000" can be explained as follows (Beginning from left to right):
  – 000: suggests the selection of the trees.
  – 01: identifies the number of the selected node (second tree).
  – 00: defines the first slave at level=1.
  – 00: specifies the first slave at level=0.
  – 10: identifies the DMSB components of slave.
  – 00: means that the first DMSB in Slave 0 at level 0 in a tree with 3 levels of interconnections in the second stage of the matrix nodes.

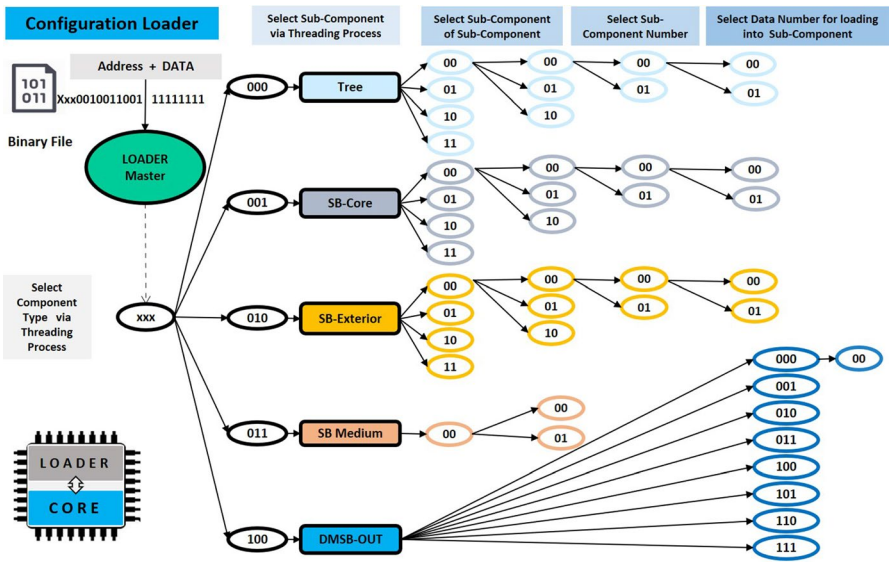


Fig. 10 Mesh of tree decoder addressing

– **Data**: "01000111," these bits are used to configure the SRAMs of the addressed DMSB.

For this function, this section first provides a topological definition of the selected Mesh of Tree architecture and then describes the established soft-core examples of the proposed eFPGA.

4 Experiments and results

This section is basically divided into two parts. First, we explore the effect of both matrix and tree sizes on Mesh of Trees eFPGA. For each exploration, the proposed eFPGA is implemented and developed for the same number of LUTs. Details of these eFPGA circuits are shown in Tables 1 and 2. Results of Mesh of Trees eFPGA architecture are given in the two following subsections.

Secondly, a wide comparison of the proposed Mesh of Tree topology with the tree-based FPGA and Mesh of Clusters eFPGA is presented. The experiments are performed to determine the effect of combining both Mesh and Tree-based typologies on a new eFPGA architecture. It is very important to determine the best balance topology to reduce the gap between regularity and layout scalability. Finally, a comparison between proposed eFPGAs with the existing eFPGA in both academia and industry is illustrated.

4.1 Mesh of tree architectural analysis

To explore the interconnect topology of the proposed Mesh of Tree embedded FPGA architecture, we aim to vary, on the one hand, the matrix size and, on the second hand, the number of level of Tree node. For all experimentation, results of the interconnect area and the average power consumption and frequency of the developed eFPGAs are provided by Xilinx Vivado tools.

Table 1 eFPGA Description for Mesh Exploration

Circuit number	Topology parameters				
	Matrix	CW	Tree		
			Level number	Cluster arity	LUTs number
C0	4*4	16	2	Level 0–2 Level 1–2	4
C1	2*2	16	3	Level 0–2 Level 1–4 Level 2–2	16

4.1.1 Mesh of tree: mesh analysis

The purpose of this section is to explore and examine how the structure and architecture parameters of the Matrix topology affect the eFPGA performance and how it can be wired to meet various specific application requirements and qualitative factors such as static power consumption, area and performance.

Table 1 presents two separate eFPGA architectures used here. These eFPGAs have both 64 LUTs. In the first case (circuit 0 : C0) in the 4 * 4 matrix contains 16 nodes. Each node contains 4 LBs connected using 2 levels of intra-connection wiring based on BFT topology. The second eFPGA (C1) contains 4 nodes (2*2), and each one includes 16 LBs that communicate through a 3-level intra-connection network. The used LUT has 4 inputs. Interconnection is based on multi-level switch boxes that communicate with the node using a channel width equal to 16.

Figure 11 shows the variation of the static power, the total area, and frequency as well as the configuration time. Each color corresponds to a size matrix design parameter. We notice that for C1, which has the smallest matrix size, the eFPGA has the lowest power which is correlated with the area. Nevertheless, C0 has less configuration time than C1, which offers the best performance to C0 and consequently the highest frequency. This is explained by the fact that the tree node in the C0 circuit has less levels (2) compared to C1 (3 levels). So, the loader has to spend less time to configure and to address the components of the tree node. For the area results, the C1 has a reduced chip size compared to C0. Indeed, the tree architecture is characterized by the area overhead, and the C0 circuit has more tree nodes (16) compared to C1 (4).

For the hardware utilization, the main exploration is provided in Figure 12. We note that the configuration component (Loader) has a large area in the different cases which is explained by the big switch number of the logic and the routing network. From Figure 13, we note that C1 has a lower routing network size compared to C0, which can be explained by a smaller matrix size despite having a higher local interconnection size.

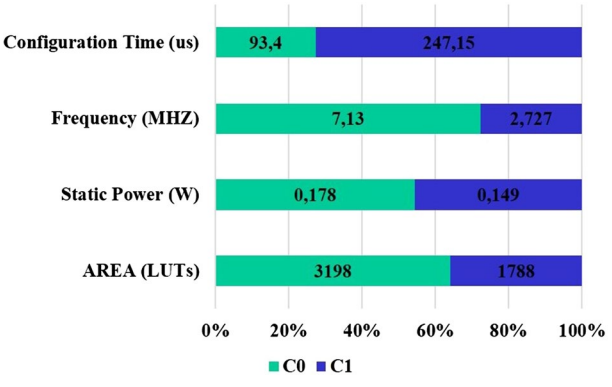


Fig. 11 Matrix exploration in mesh of tree architecture

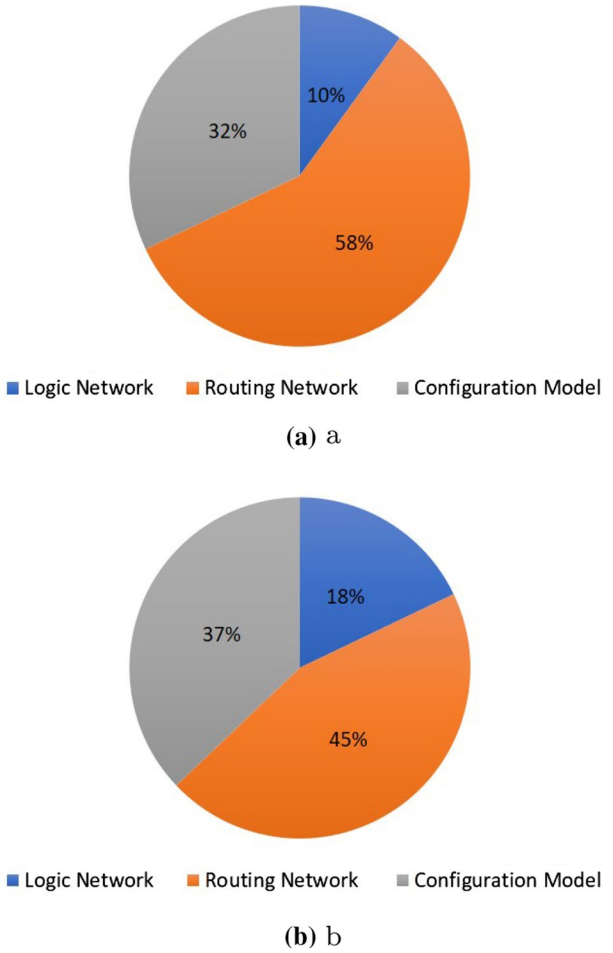


Fig. 12 Hardware utilization comparison in mesh exploration:C0 (a), C1 (b)

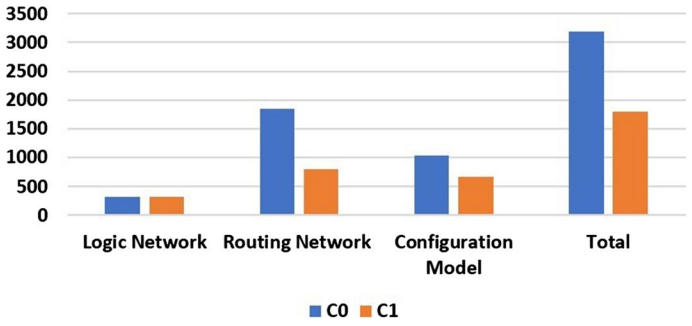


Fig. 13 Area comparison

Table 2 eFPGA Description for Tree Exploration

Circuit number	Topology parameters				
	Matrix	CW	Tree		
			Level number	Cluster arity	LUTs number
C0	2*2	16	3	Level 0 - 4 Level 1 - 2 Level 2 - 2	16
C1	2*2	16	3	Level 0 - 2 Level 1 - 4 Level 2 - 2	16

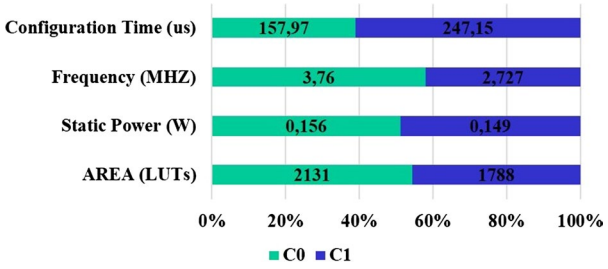
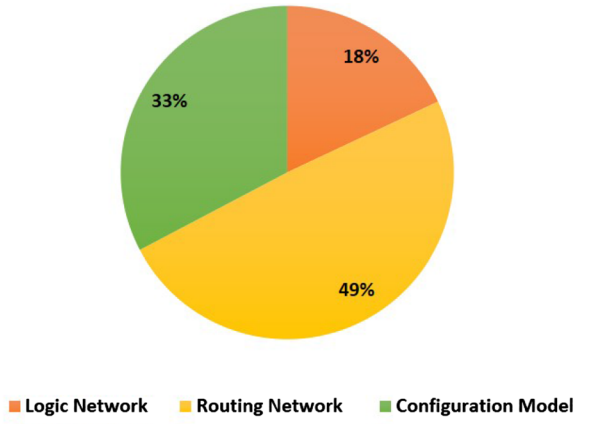


Fig. 14 Tree exploration in mesh of tree architecture

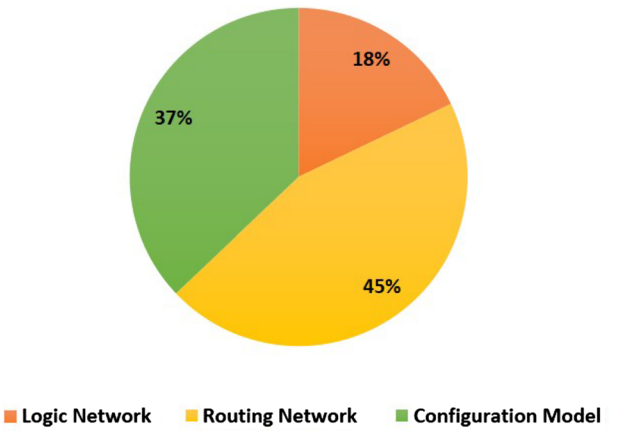
4.1.2 Mesh of tree: tree analysis

In this section, we present and discuss the node size effect on power, area, and performance. Table 2 illustrates the variation of the eFPGA tree size with the different level numbers. The interconnection is based on multi-level switch boxes that communicate with the node using a channel width equal to 16. Based on Equations presented in Section 2, each node has 32 inputs and 8 outputs. The aim of this section is to explore the effect of cluster arity and the level number in the eFPGA performances. Two cases are described in Table 2. Each eFPGA has 64 LUTs collected in 2*2 matrix (4 nodes). Each node has 16 LBs connected using 3 levels of intra-connection wiring based on BFT topology. C0 has cluster arity equal to 4 in the fewer level, and C1 has cluster arity equal to 2 in the fewer level. The results of the Vivado tool are presented in Figures 14, 15, and 16.

Figure 14 illustrates the variations of the static power, the total area, and frequency but also the configuration time. Each color correlates to the design parameter of the cluster size. We note that for C1, which has the smallest cluster size at the lower level, the eFPGA has the lowest area and power. However, C0 has less configuration time than C1, which is explained by the fact that C0 has the best frequency compared to C1. Indeed, in the case of C0, more LUTs are grouped together in the same cluster, so the routing delay is less and the communication between these LUTs is faster. On the other side, for C1, each of 2 LUTs is grouped together



(a) a



(b) b

Fig. 15 Tree exploration: hardware utilization comparison C0 (a), C1 (b)

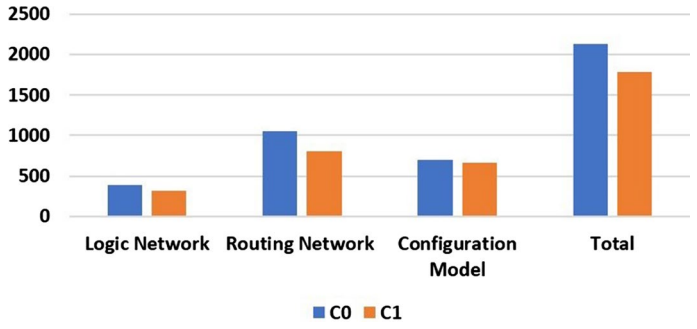


Fig. 16 Tree exploration

in the same cluster, and if these LUTs need to communicate with other 2 LUTs in a different cluster, the signal has to cross the higher interconnect level which penalizes the design frequency.

For the hardware utilization exploration given in Figure 15, we noted that the logic component in both C0 and C1 circuits occupies 18 % of the total area. However, the configuration model in C1 is 4 % larger than the one in C0. As shown in Figure 16, C1 has only 45 % for routing connections compared to C0. In fact, the biggest challenge in the design of eFPGA is to reduce the size of the interconnection network area; based on this strategy, we note that eFPGA in C1 is better than C0.

4.1.3 Data width effect

This section reveals that the Mesh of Tree eFPGA can be optimized for different metrics: performance, power, and area by adjusting the configuration data width. The data width is the size of the data to be implemented in the addressed component. We considered a test case including 16 LUTs. This test case was carried out by varying the target data width from 4 to 17 according to the same methodology: the area, the static power, and the configuration time results were evaluated in each case which are presented in Table 3.

Curves presented in Figure 17 demonstrate that, when the size of the configuration word is greater than or equal to the overall size of the SRAM, the configuration time decreases. On the other hand, we notice that the data width equal to 8 has the best area, correlated with the power consumption, that decreases by reducing the configuration time and preserving the usage of all data bits. Therefore, we find that data width 8 has the best balance of silicon exchange.

4.2 Mesh of tree VS mesh of cluster VS tree-based eFPGA

To evaluate the proposed embedded FPGA based on Mesh of Tree architecture, we chose a recent embedded FPGA [26] that is based on multilevel structure. For our experiments, three eFPGAs presented in Table 3 are evaluated. All mentioned architectures include 32 LUTs.Table 4

The general description of these eFPGA is as follows:

1. Tree-Based embedded FPGA has 3 levels and uses 4-inputs LUTs.

Table 3 Data width effect for 16-LUTs mesh of tree eFPGA

Data width	power (W)	Configuration words number	Area (LUTs)	Frequency	Configuration time (us)
4	0,135	498	1470	11,95	41,67
8	0,123	338	1129	10,58	31,94
12	0,131	274	1246	10,58	25,89
17	0,131	266	1228	12,04	22,09

Fig. 17 Data width exploration

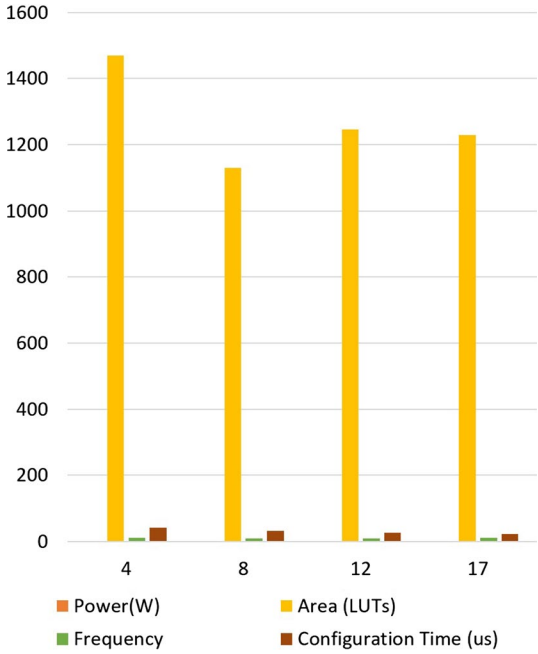


Table 4 eFPGA description for different topologies

Architecture	Circuit characteristics					
	CW	LUTs number	Level number	LUTs number / CLB	Matrix	Cluster arity
Mesh of clusters	16	32	-	8	2*2	-
Tree	-	32	3	4	-	Level 0 - 4 Level 1 - 4 Level 2 - 2
Mesh of trees	16	32	3	8	2*2	Level 0 - 2 Level 1 - 2 Level 2 - 2

2. Mesh of Clusters eFPGA is implemented with a matrix 2*2 nodes. Each node includes a single tree level with 8 clusters of LBs. The channel width of matrix topology is equal to 16.
3. Embedded FPGA hardware implementation based on a mesh of Tree topology contains 4 nodes organized with 2*2 matrix. Each node is a tree-based with 8 LBs connected via 3 levels. The channel width is equal to 16.

The experimental results for the described embedded FPGA are summarized in Figures 19, 18, and 20.

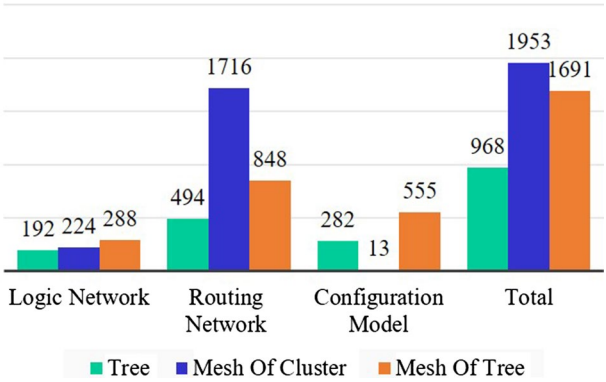
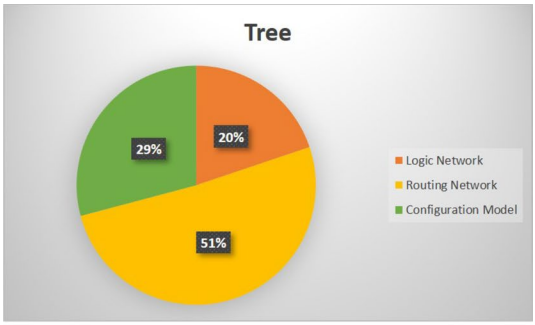


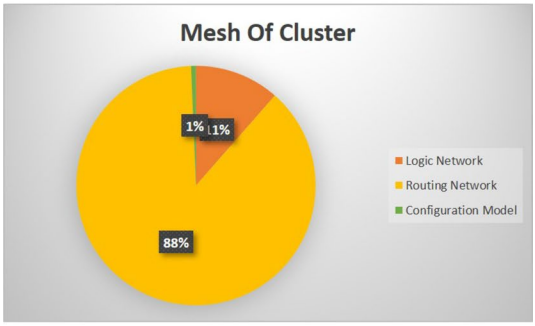
Fig. 18 Hardware Utilization Comparison: Mesh VS mesh of tree VS tree eFPGA

Figures 18 and 19 show the area results of the implementation of the tree-based and mesh of cluster eFPGAs as well as the proposed eFPGA. First, the configuration model (loader) of the mesh of cluster eFPGA occupies 1% of the total area of eFPGA IP. The most amount is dedicated to the routing network which occupies 88%. Finally, the logic blocks occupy 11% of the total area of the mesh of cluster architecture. On the other side, the tree-based architecture presents the following results: The configuration model, the logic blocks, and the routing network occupy 29%, 20%, 59%, respectively, of the total area. We notice that the configuration model of the tree-based eFPGA is bigger than the one of the mesh of cluster architecture. Indeed, in the first architecture, a sub-loader is dedicated to each level, and each sub-loader is dedicated to only one level. Each sub-loader ensures the configuration of the components of this level. In the mesh of cluster eFPGA, only one loader is used since the cluster contains only one level. Moreover, the tree-based eFPGA is 2 times smaller than the mesh of cluster architecture. The results corresponding to the Mesh of trees show that configuration part occupies 17 % of the total area. This result is considered as acceptable relatively to the compared architectures. The configuration component occupies 33% from the total area that can be explained by the big number of loaders and sub_loaders existing in the synthesized eFPGA. At the end, the routing networks of the present eFPGA occupy only 50% from the total area. Indeed, we can consider Mesh of Tree eFPGA as the best eFPGA in terms of area since it has the lowest area for routing network.

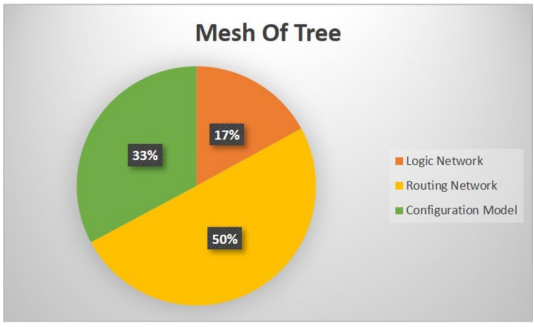
In Figure 20, we analyze the energy efficiency, performance, and frequency of the proposed eFPGA compared to the recent Tree-based and mesh of clusters eFPGAs [26]. Tree-based eFPGA has the best results in terms of performance since it has the highest frequency compared to Mesh of cluster and Mesh of Tree eFPGA. Even if the proposed eFPGA is not optimized in terms of performance, area, and power consumption, but it is considered as an intermediate solution between classical island-style architecture that has good physical scalability and hierarchical architecture that is very disadvantaged in terms of physical layout



(a) a



(b) b



(c) c

Fig. 19 Hardware Utilization Comparison: Tree-based Architecture (a), Mesh of Cluster Architecture (b), Mesh of Tree Architecture (c)

generation (it does not support scalability and does not suit the planar chip structure, particularly for large circuits).

The Mesh of Tree has strong physical scalability: Once the cluster layout is created, it can be used to generate Mesh layouts with the required size. The suggested Mesh of Tree architecture has an acceptable trade-off between area density and layout scalability.

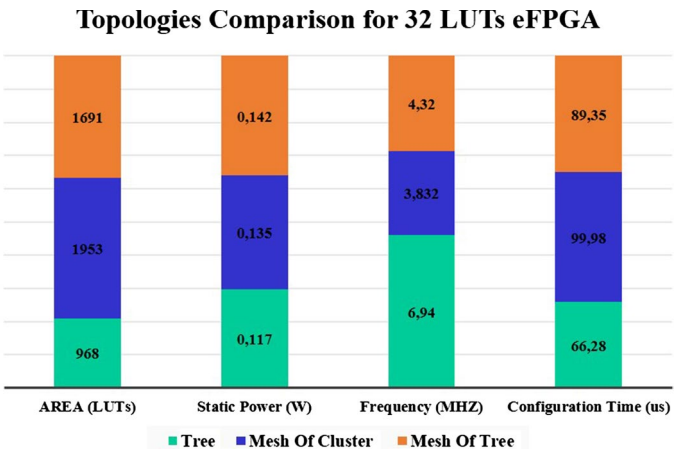


Fig. 20 Mesh VS mesh of tree VS tree eFPGA

4.3 Analysis of eFPGA size progress

This section provides a comparison between the proposed eFPGA and the previous proposed Tree-based eFPGA in terms of size growth from 16 LUTs to 128 LUTs. We have selected the only Tree based on the comparison, as the Mesh of Cluster eFPGA can be viewed as a particularly special case of the proposed Mesh of Tree eFPGA.

The various implementations are detailed in Table 5; we have selected the same cluster size equal to 2 in both topologies: Tree and Mesh of Tree are shown in Figure 21. We also set the size of the data width to 8. The only parameter that has been modified is the number of levels and the matrix size in the evaluated topology.

To evaluate the impact of the multiple sizes of tiles having the same number of LUTs in each Mesh of Tree and Tree-based eFPGA, we run several experiments. The results are reported in Figure 22 and Table 6. From the curves of Figure 22, we notice the same results as the previous section. Indeed, the Tree-based eFPGA has the best results in terms of frequency and power consumption which is correlated with the

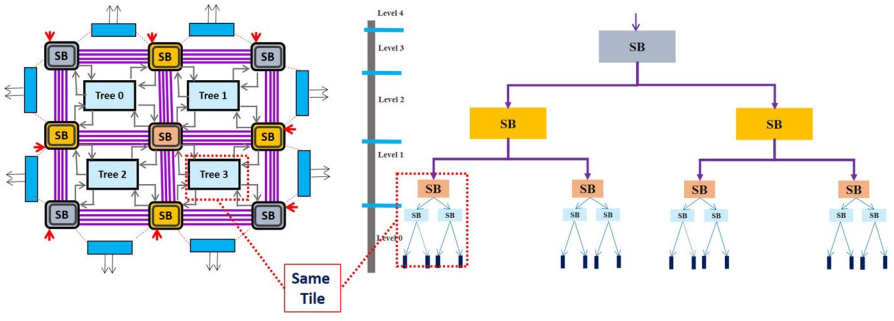
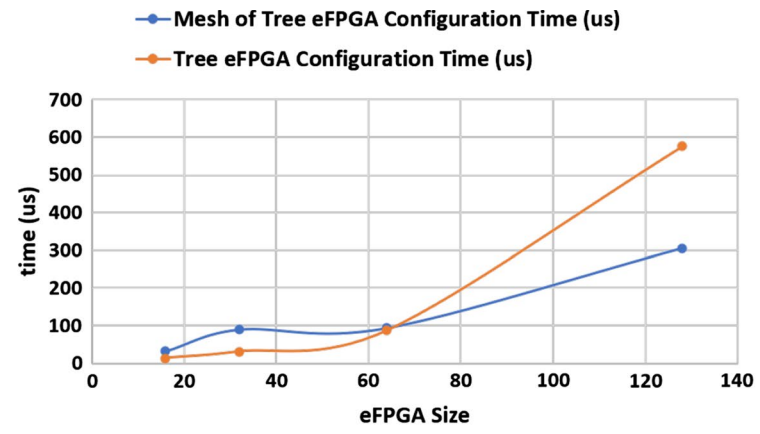


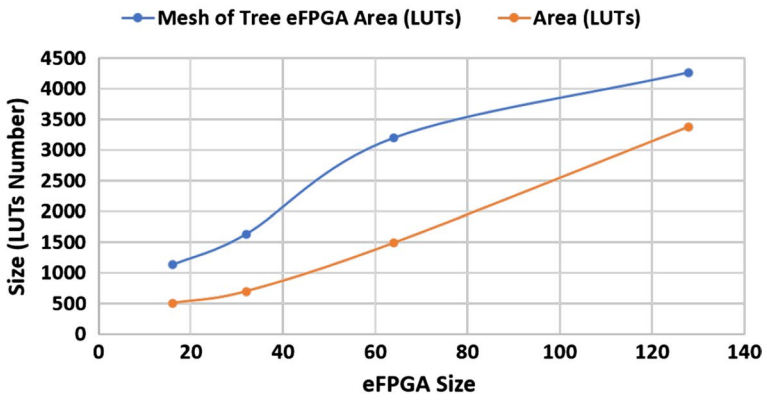
Fig. 21 Mesh of tree vs tree eFPGA

Table 5 Experimented eFPGAs Details

Circuit number	eFPGA size	Architecture	Circuit characteristics				
			CW	Level number	LUTs number / CLB	Matrix	Cluster arity
C0	16	Tree	-	4	4	-	Level0 – 2 Level1 – 2 Level2 – 2
C1		Mesh Of Tree	16	2	4	2*2	Level0 – 2 Level1 – 2
C2	32	Tree	-	5	4	-	Level0 – 2 Level1 – 2 Level2 – 2 Level3 – 2 Level4 – 2
C3		Mesh Of Tree	16	3	8	2*2	Level0 - 2 Level1 - 2 Level2 -2
C4	64	Tree	-	6	4	-	Level0 – 2 Level1 – 2 Level2 – 2 Level3 – 2 Level4 – 2 Level5 – 2
C5		Mesh Of Tree	16	2	4	4*4	Level0 – 2 Level1 – 2
C6	128	Tree	-	7	4	-	Level0 – 2 ...~ Level6 – 2
C7		Mesh Of Tree	16	5	32	2*2	Level0 – 2...Level4 – 2



(a) a



(b) b

Fig. 22 Tree VS mesh of tree eFPGA in Terms of Progress size Comparison: Performance Comparison**a** , Area Comparison **b**

area and in particular to the number of switches. However, when the number of LUTs increases (up to 128), the Mesh of Tree configuration time becomes better compared to the tree-based eFPGA. Indeed, for big eFPGA, the tree topology used a big number of levels which makes the configuration process more complex and needs many instruction cycles.

Table 6 Mesh of Tree eFPGA VS Tree-based eFPGA

Mesh of tree eFPGAs				Tree eFPGAs			
Circuit	eFPGA size	Static power(W)	Frequency (MHz)	Circuit	eFPGA size	Static power(W)	Frequency (MHz)
C1	16	0,123	10,58	C0	16	0,106	12,04
C3	32	0,142	4,32	C2	32	0,111	12,075
C5	64	0,178	7,13	C4	64	0,131	7,13
C7	128	0.211	1.656	C6	128	0.172	3.23

5 eFPGA comparison

To compare the evaluated eFPGA architectures (Tree-based eFPGA; Mesh of Cluster eFPGA; and Mesh of Tree eFPGA) to previous works in the state of the art, Table 7 provides full research addressing the different emerging trends presented in the literature for reconfigurable hardware. The main downside of the work presented in [13] is that, after manufacturing, the target eFPGA parameter's cannot be modified in terms of architecture, number of wires, number of input and output pads, and especially the number of Look-Up-Tables (LUTs) because the presented eFPGA is a hardcore device or a fixed softcore. In most previous works, the topology of the eFPGA is the island-style architecture with limit LB number in cluster node. Thus, the Versatile Place and Route (VPR) [23], an academic open-source tool for FPGA, is used to ensure the placement and routing tasks. The main drawback of the mesh architecture is that the occupancy of Logic Networks is smaller compared to the total size of the core, which affects the power, performance, and the total area of the embedded FPGA.

The major challenge of the eFPGA architecture is to have full flexibility and at the same time ensuring minimum area costs. Mesh-based eFPGAs are composed of logic blocks usually arranged in a grid and surrounded by horizontal and vertical routing networks. Mesh architecture is the most popular of all academic and commercial FPGAs. Besides, the Mesh eFPGA area is dominated by the interconnection network, with routing resources accounting for 80-90 % of the total eFPGA area, while logic blocks occupy only 10-20% of the total area. Reducing the number of programmable switches on the routing path could increase the speed of the circuit and reduce the eFPGA area. The mesh architecture has a higher routing delay because the number of Switch Boxes depends linearly on the Manhattan distance d [29]. However, Tree-based architectures have primarily been proposed for this purpose. Centered on the Manhattan distance rule, logical blocks are gathered in clusters in the hierarchical architecture. These clusters are then recursively clustered to form higher-level hierarchical clusters (one cluster and its slaves).

Based on the results of the experiments presented in this study, we prove that the architecture suggested combines the advantages of the special features of both Mesh and Tree (Figure 23). Unifying all architectures by building clusters with Tree-based local connections and linking these clusters to Mesh-based multi-stage switch boxes, we have shown that the resulting architecture represents a good balance between layout scalability and area density. The proposed architecture has the best result in terms of total hardware utilization compared to Tree-Based eFPGA.

6 Conclusions

In this paper, we have implemented the first Mesh of Tree embedded FPGA. We have evaluated the area utilization, frequency, and static power. We have developed the interface to load the configuration file. The proposed eFPGA is fully

Table 7 Embedded FPGA Comparison

eFPGA	Date	Reference	Architecture Mesh / Hierarchical	Input-LUT	Soft/ Hard Core	Coarse/ Fine Grain	Research Team	Hardware Flow	Software Flow	Generation Layout Scalability	Regularity	Logic Hardware Utilization
FASE	2007	[5]	Mesh	4-input -1	Soft	Fine	GET Telecom Paris, CNRS - LTCI France	–	VPR	+	+	–
ESL	2009 - 2011	[1, 2]	Mesh	4-input-1	Soft	Fine	UM/CRNS France Menta Company	–	Origami	+	+	–
ACLBs	2019	[22]	Mesh	-	Soft	Coarse Fine	Valbonne - Sophia	–	Own Tool	+	+	–
Flex tiles	2015	[13]	Mesh	-	Hard	Fine	Embedded Systems Ruhr-University Bochum, Germany	–	VPR	+	-	–
Overlay	2018	[17]	Mesh	-	Soft - Hard	Fine	ENSTA Bretagne Lab-STICC UMR	-	VPR	+	+	–

Table 7 (continued)

eFPGA	Date	Reference	Architecture Mesh / Hierarchical	Input-LUT	Soft/ Hard Core	Coarse/ Fine Grain	Research Team	Hardware Flow	Software Flow	Generation Layout Scalability	Regularity	Logic Hardware Utilization
Proposed eFPGA: Tree-based eFPGA	2019 -2020	[25, 26, 28]	Hierarchical	4-input-1	Soft	Fine	LIP6, France CES Lab, ENIS, Tunisia	VHDL - Verilog Generator	LIP6 Tool	-	-	+
Proposed eFPGA: Mesh of Custer eFPGA	2020	[7, 27]	Mesh	4-input-1	Soft	Fine	CESLab, ENI Tunisia LIP6, France	–	LIP6Tool	+	+	-
Proposed eFPGA: Mesh of Tree eFPGA	2020-2021	-	Mesh Tree	4-input-1	Soft	Fine	CES Lab, ENIS Tunisia Mentor Graphics, Tunisia	–	CES Tool	+	+	+

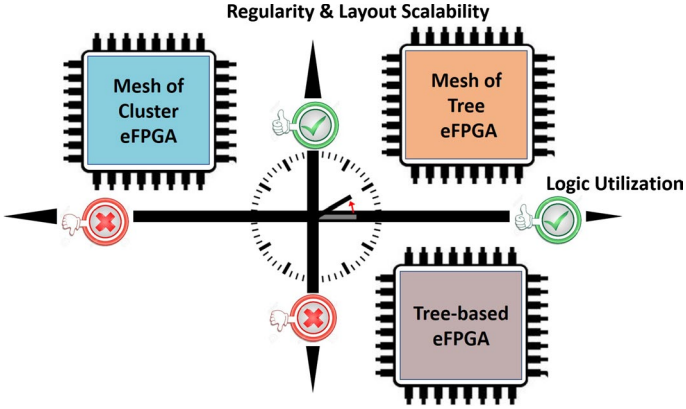


Fig. 23 eFPGA evolution's

synthesizable. To the best of our knowledge, the proposed eFPGA topology and the addressing decoder are the first complete IP dedicated to be integrated into ASIC. We compared the proposed core to 32 LUTs tree-based embedded FPGA in terms of area, power dissipation, performance, and frequency. We have demonstrated that Mesh of Tree eFPGA is a good trade-off between area density and layout scalability.

This paper discussed and contributed to a range of aspects to the investigation of the potentials and difficulties of eFPGAs. There may be some future outlines based on the information acquired and the experiments performed in this study. The Mesh of Tree has good structure scalability but is punishing in terms of the area relative to a stand-alone Tree structure. These points deserve to be more explored the Channel Width has an impact on hardware utilization. Channel Width variation affects the interaction between Mesh and Tree interconnectors. Also, as perspective, we will develop a VHDL generator can automate the generation of useful eFPGAs, facilitate and speed up the development process, and finally provide a ready-to-use eFPGA that can be incorporated into the SoC design.

Acknowledgements This work was supported by CES Research Laboratory, National School of Engineering of Sfax, Mentor Graphics Tunisia, King Abdulaziz City for Science and Technology (KACST), and the Digital Research Center of Sfax (CRNS) under a research grant (project no. 35/1012)

Data availability statement The manuscript has no associated data.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Ahmed SZ, Eydoux J, Rougé L, Cuelle JB, Sassatelli G, Torres L (2009) Exploration of power reduction and performance enhancement in leon3 processor with esl reprogrammable efpga in

processor pipeline and as a co-processor. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp. 184–189. European Design and Automation Association
2. Ahmed SZ, Fernandez M, Sassatelli G, Torres L(2009) efpga architecture explorations: Cad & silicon analysis of beyond 90nm technologies to investigate new dimensions of future innovations
3. Amouri E, Blanchardon A, Chotin-Avot R, Mehrez H, Marrakchi Z (2013) Efficient multilevel interconnect topology for cluster-based mesh fpga architecture. In: 2013 International Conference on Reconfigurable Computing and FPGAs (ReConFig), pp. 1–6. IEEE
4. Bollengier T, Lagadec L, Najem M, Le Lann J.C, Guilloux P (2017) Soft timing closure for soft programmable logic cores: The argen approach. In: International Symposium on Applied Reconfigurable Computing, pp. 93–105. Springer
5. Chaudhuri S, Danger JL, Guilley S (2007) Efficient modeling and floorplanning of embedded-fpga fabric. In: 2007 International Conference on Field Programmable Logic and Applications, pp. 665–669. IEEE
6. Chtourou S, Abid M, Pangracious V, Amouri E, Marrakchi Z, Mehrez H (2014) Three-dimensional mesh of clusters: An alternative unified high performance interconnect architecture for 3d-fpga implementation. In: 2014 International 3D Systems Integration Conference (3DIC), pp. 1–7. IEEE
7. Chtourou S, Marrakchi Z, Amouri E, Pangracious V, Abid M, Mehrez H (2016) Improvement of cluster-based mesh fpga architecture using novel hierarchical interconnect topology and long routing wires. Microproc Microsyst 40:16–26
8. Farooq U, Marrakchi Z, Mehrez H (2012) Fpga architectures: An overview. Tree-based heterogeneous FPGA architectures. Springer, London, pp 7–48
9. Farooq U, Marrakchi Z, Mehrez H (2012) Tree-based heterogeneous FPGA architectures: application specific exploration and optimization. Springer Science & Business Media, USA
10. Farooq U, Marrakchi Z, Mrabet H, Mehrez H (2008) The effect of lut and cluster size on a tree based fpga architecture. In: 2008 International Conference on Reconfigurable Computing and FPGAs, pp. 115–120. IEEE
11. Guerrier P, Greiner A (2008) A generic architecture for on-chip packet-switched interconnections. In: Design, Automation, and Test in Europe, pp. 111–123. Springer
12. Hsiung PA, Santambrogio MD, Huang CH (2018) Reconfigurable system design and verification. CRC Press, London
13. Janßen B, Schwiegelshohn F, Koedam M, Duhem F, Masing L, Werner S, Huriaux C, Courtay A, Wheatley E, Goossens K., et al (2015)Designing applications for heterogeneous many-core architectures with the flextiles platform. In: 2015 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), pp. 254–261. IEEE
14. Kamar S, Fouda A, Zekry A, El-Mahdy A (2017) Fpga implementation of rs codec with interleaver in dvb-t using vhdl. Int J Eng & Technol 6(4):171–180
15. Kim JK, Oh JH, Hwang GB, Gwon OS, Lee SE (2020) Design of low-power soc for wearable healthcare device. J Circuit Syst Comput 29(06):2050085
16. Kuon I, Tessier R, Rose J, et al (2008) Fpga architecture: Survey and challenges. Foundations and trends® in electronic design automation **2**(2): 135–253.
17. Le Lann JC, Bollengier T, Najem M, Lagadec L (2018) An integrated toolchain for overlay-centric system-on-chip. In: 2018 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), pp. 1–8. IEEE
18. Leiserson CE (1985) Fat-trees: universal networks for hardware-efficient supercomputing. IEEE Trans Comput 100(10):892–901
19. Lemieux G, Lewis D (2002) Circuit design of routing switches. In: Proceedings of the 2002 ACM/SIGDA tenth International Symposium on Field-Programmable Gate Arrays, pp. 19–28
20. Marrakchi Z, Mrabet H, Farooq U, Mehrez H (2009) Fpga interconnect topologies exploration. Int J Reconfigurable Comput 2009:6
21. Marrakchi Z, Mrabet H, Mehrez H (2005) Hierarchical fpga clustering to improve routability. In: research in microelectronics and electronics, 2005 PhD, vol. 1, pp. 165–168. IEEE
22. Martin G, Cavalli D, Firmin F (2019) Embedded fpga with multiple configurable flexible logic blocks instantiated and interconnected by abutment . US Patent App. 16/175,671
23. Rose J, Luu J, Yu C.W, Densmore O, Goeders J, Somerville A, Kent K.B, Jamieson P, Anderson J (2012) 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, pp. 77–86
24. Saidi H, Turki M, Marrakchi Z, Abid M, Obeid A (2021) Soft-core embedded fpga based system on chip. Analog Integr Circ Sig Proc 109(3):517–533

25. Saidi H, Turki M, Marrakchi Z, Ahmed HB, Obeid A, Abid M (2021) Exploration of word width and cluster size effects on tree-based embedded fpga using an automation framework. J Circuits Syst Comput p. 2150241
26. Saidi H, Turki M, Marrakchi Z, Obeid A, Abid M (2020) Implementation of reed solomon encoder on low-latency embedded fpga in flexible soc based on arm processor. In: 2020 International Wireless Communications and Mobile Computing (IWCMC), pp. 1347–1352. IEEE
27. Saidi H, Turki M, Marrakchi Z, Obeid A, Abid M (2020) Novel synthesizable efpga based on island network with multilevel switch boxes. In: 2020 IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA), pp. 1–6. IEEE
28. Saidi H, Turki M, Marrakchi Z, Saleh M.B, Abid M (2019) New cad tools to configure tree-based embedded fpga. In: 2019 International Conference on High Performance Computing & Simulation (HPCS), pp. 643–649. IEEE
29. Sinwar D, Kaushik R (2014) Study of euclidean and manhattan distance metrics using simple k-means clustering. Int J Res Appl Sci Eng Technol 2(5):270–274
30. Zied M, Hayder M, Emna A, Habib M (2008) Efficient tree topology for fpga interconnect network. In: Proceedings of the 18th ACM Great Lakes symposium on VLSI, pp. 321–326