This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A Novel FPGA Architecture Based on Ultrafine Grain Reconfigurable Logic Cells

Pierre-Emmanuel Gaillardon, *Member, IEEE*, Xifan Tang, Gain Kim, and Giovanni De Micheli, *Fellow, IEEE*

***Abstract*— In this paper, we investigate the opportunity brought by controllable-polarity transistors to design efficient reconfigurable circuits. Controllable-polarity transistors are devices whose polarity can be electrostatically programmed to be either n- or p-type. Such devices are used to build ultrafine grain computation cells. These cells are arranged into regular matrices, called *MClusters*, with a fixed and incomplete interconnection pattern, employed to minimize the reconfigurable interconnection overhead. We subsequently use them into field-programmable gate arrays (FPGAs). To assess this architectural scheme in an efficient and objective manner, we present a complete benchmarking tool flow and focus on the packing algorithm developed to handle the architecture. We finally perform the evaluation with widely used benchmark circuits. Leveraging the ultrafine grain cells compactness from a system-level perspective, we show that FPGAs exploiting MClusters demonstrate average savings of 43% and 23% in area and delay, respectively, as compared with the CMOS lookup table FPGA counterpart at 22-nm technological node.**

***Index Terms*— Controllable-polarity devices, field-programmable gate arrays (FPGAs), packing tools, ultrafine grain logic, vertically stacked nanowires.**

I. Introduction

IN THE quest to push further the Moore's scaling laws, the semiconductor industry introduced the fin-based field effect transistors (FinFETs) to provide an alternative to planar CMOS transistors at the 22-nm technology node [1]. Following the trend toward 1-D structures, several devices are currently investigated. Among them, carbon nanotubes FETs [2] and vertically stacked silicon nanowires FETs (SiNWFETs) [3] are promising extensions to current tri-gate FinFETs technology. These devices exploit the 1-D properties of their channels to exhibit superior performances. In addition, the gate-all-around (GAA) structure improves the electrostatic control of the channel and leads to a higher $I_{\mathrm{ON}}/I_{\mathrm{OFF}}$ ratio and reduced leakage current [4].

At advanced technology nodes, more and more devices are affected by Schottky contacts at the source and drain interfaces. Hence, devices face an ambipolar behavior, i.e., the device exhibits n- and p-type characteristics simultaneously. While technologists target to suppress the ambipolar behavior of the devices through additional process steps, new design methodologies [5]–[7] showed that it is of high interest to control the ambipolar phenomenon through programmable polarity devices.

By engineering the source and drain contacts and by constructing independent double-gate structures, the device polarity can be electrostatically programmed to be either n- or p-type. The reconfiguration at the device level is called here ultrafine grain programmability. While such devices were already demonstrated using silicon [8], [9] and carbon electronics [10], [11], we focus, in this paper, on a double-gate SiNWFET (DG-SiNWFET), built using a top-down fabrication flow [12].

The enhanced functionality of such a device has been shown to be well suited to build ultrafine grain reconfigurable logic blocks [5]. Indeed, thanks to this technology shift, a reconfigurable logic cell (LC), able to perform many different two-input Boolean operations can be realized with only seven transistors [5]. The cell compactness makes them appealing to rethink the standard reconfigurable architectures, such as the traditional field-programmable gate arrays (FPGAs) architectural scheme. In reconfigurable logic, less than 15% of the area is dedicated to the logic computation, while the other resources are used for the structure reconfigurability [13]. Such a large imbalance leads to a large cost in terms of area, routing delay, and power consumption.

In this paper, we exploit ultrafine grain LCs to improve the FPGA architecture. In particular, we introduce 1) a novel architectural organization where the standard lookup tables (LUTs) are replaced by ultrafine-grain LCs. While it seems counterintuitive to further reduce the size of the logic blocks, therefore worsening the logic/routing imbalance, we will see that the ultrafine grain elements can be organized in compact matrix arrangements of LCs, called MClusters that become competitive as compared with the typical combinational elements of the FPGAs. To prevent the reconfigurable interconnection overload within the MClusters, we interconnect the cells through a layered, fixed and incomplete interconnection pattern, i.e., that an output on a given layer is permanently connected to a subset of inputs on the next layer. To support the novel architecture, we propose 2) a complete benchmarking tool flow suited to this organization. In particular, we focus on the packing step of cells into MClusters. A preliminary version of this paper has

Manuscript received November 8, 2013; revised March 17, 2014 and July 10, 2014; accepted September 1, 2014. This work was supported in part by the French National Research Agency under Grant ANR-08-SEGI-12 through the NANOGRAIN Project, in part by the European Research Council under Grant ERC-2009-AdG-246810 through the NANOSYS Project, and in part by the Swiss National Science Foundation under Grant 200021_146600.

The authors are with the Integrated Systems Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne 1015, Switzerland (e-mail: pierre-emmanuel.gaillardon@epfl.ch; xifan.tang@epfl.ch; gain.kim@epfl.ch; giovanni.demicheli@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2014.2359385

1063-8210 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

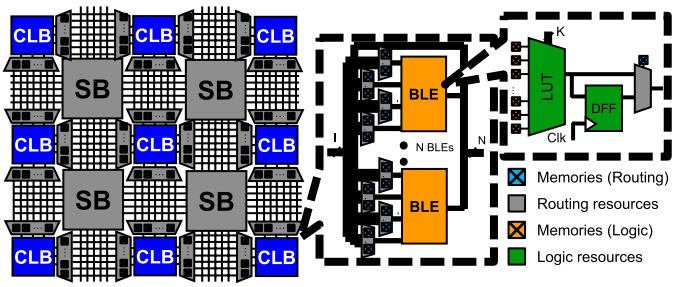


Fig. 1. Baseline FPGAs architecture [13].

been presented in [14]. This paper extends the initial work by integrating MClusters with a state-of-the-art FPGA flow supporting area, delay, and power consumption of the structure estimations. It also provides a complete evaluation of the approach for a 22-nm technology node. Simulation results show that an FPGA structure using 2 by 2 MClusters in replacement of the traditional LUTs gives an average area and delay reduction of 43% and 23%, respectively, compared with a conventional CMOS FPGA at 22-nm technology node. This large performance gains are directly accounted from the efficient use of ultrafine grain LCs. In addition, we observe that the proposed approach opens a promising path to correct two intrinsic limitations of the FPGA circuits, with an 8% reduction in the logic/routing imbalance and a better control in the distribution of the routing delay.

The rest of this paper is organized as follows. Section II surveys the necessary background on FPGA architecture, controllable-polarity devices, and ultrafine grain computation. Section III presents the novel architectural scheme for ultrafine grain reconfigurable computing. It highlights the concept and shows the organization at each architectural level. Section IV introduces the specific benchmarking tool, with an emphasis on the MCluster packing. Section V evaluates the performance of the targeted architecture, in terms of granularity and compares them with its CMOS FPGA counterpart. Section VI concludes this paper.

II. Background

In this section, we introduce the required background on FPGA architecture, controllable-polarity devices, and ultrafine grain computation.

A. FPGA Architecture

FPGAs are regular circuits typically consisting of several identical configurable logic blocks (CLBs) surrounded by reconfigurable interconnect lines [13]. As shown in Fig. 1, every CLB is formed by a set of N basic logic elements (BLEs). A BLE is a K-input LUT whose output can be routed to any other LUT input with or without an intermediate registration phase through a flip-flop. Every CLB has I inputs coming from other CLB outputs.

All design parameters N, K, and I can be set by the FPGA architect depending on the targeted system granularity. The routing part of the FPGA is formed by large channels of width W, interleaved between the CLBs. The channels cross

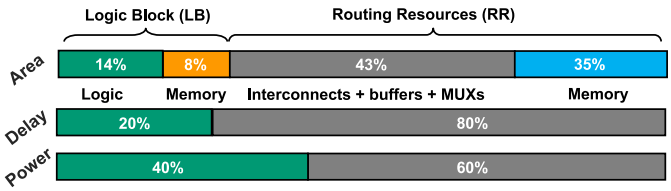


Fig. 2. FPGAs area/delay/power repartition per block [15].

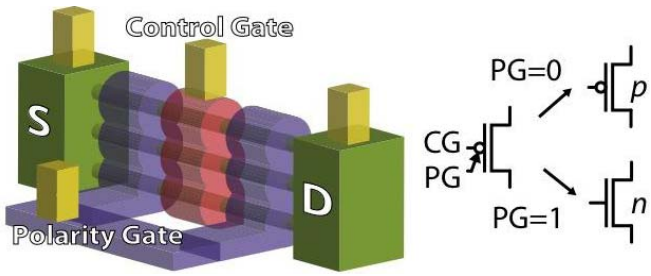


Fig. 3. 3-D sketch of the SiNWFETs featuring two independent gates and its associated symbol [12].

each other and the signals can be routed within the FPGA using switchboxes. A switchbox is a matrix at the intersection of channels, which is made of reconfigurable switches.

Fig. 2 shows the area/delay/power breakdown of the various components of a baseline static random access memory (SRAM)-based island style FPGA. Note that the configuration memories occupy roughly half of the area in both the logic blocks and the routing resources, and that only 14% of the total area is used for actual computation. In addition to consuming most of the die area, routing resources significantly contributes to FPGAs hurdles. In [15], interconnect delays are reported to account for roughly 80% of the total path delay. It also contributes to the high-power consumption of FPGAs, with more than 60% of the total dynamic power consumption.

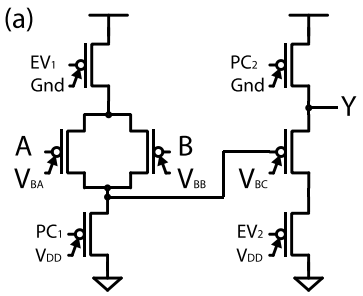
In this paper, we focus on the use of novel logic block structures, to reduce the imbalance between logic and routing. For this purpose, we will exploit the reconfigurability at the device level, offered by novel device technologies.

B. Transistors With Controllable Polarity

In several nanoscale FET devices (45-nm and below), the superposition of n-type and p-type carriers is observable under normal bias conditions. The phenomenon, called ambipolarity, occurs in many different materials, including silicon [16], carbon nanotubes [17], and graphene [18]. The control of this ambipolarity allows us to adjust the device polarity online. Transistors with a controllable polarity have been experimentally fabricated in several novel technologies, such as carbon nanotubes [10], graphene [11], and silicon nanowires (SiNWs) [8], [9]. The operation of these FETs is enabled by the regulation of Schottky barriers at the source/drain (S/D) junctions thanks to an additional gate.

In particular, we consider here vertically stacked SiNW FETs, featuring two GAA electrodes [12], as shown in Fig. 3. Vertically stacked GAA SiNWs represent a natural evolution of FinFET structures, providing better electrostatic

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(b)

V_{BA}	V_{BB}	V_{BC}	Y
0	0	0	$\overline{A.B}$
0	0	1	A.B
0	1	0	$\overline{A}+B$
0	1	1	$A.\overline{B}$
1	0	0	$A+\overline{B}$
1	0	1	$\overline{A}.B$
1	1	0	A+B
1	1	1	$\overline{A+B}$

Fig. 4. (a) Ultrafine grain reconfigurable cell schematic [5] and (b) associated configurations.

control over the channel and consequently superior scalability properties [12].

In the device, one gate electrode, the *control gate* acts conventionally by turning ON and OFF the device. The other electrode, the polarity gate (PG), acts on the side regions of the device, in proximity of the S/D Schottky junctions, switching the device polarity dynamically between n- and p-type. The input and output voltage levels are compatible, resulting in directly cascadable logic gates [12]. For a complete review on the design opportunities brought by these transistors, we refer the reader to [6].

C. Ultrafine Grain Reconfigurable Logic Gates

This property of in-field reconfigurability has been used in [5] to build a compact reconfigurable cell. The cell, as shown in Fig. 4(a), can realize any of the eight Boolean logic functions Y of the two inputs A and B, reported in Fig. 4(b). The cell is built with only seven transistors arranged in two dynamic logic stages: logic function and follower/inverter. Signals PC_1, EV_1, PC_2, and EV_2 are, respectively, the global precharge and evaluation signals of the two stages. The reconfiguration of the cell depends on the signals applied to the PGs V_{BA}, V_{BB}, and V_{BC}. Each of these signals is biased with either V_{DD} or Gnd. This results in configuring the related transistors to either n- or p-type, thereby customizing the gate internal circuit. For a detailed description of the circuit operation, we refer the reader to [5].

Note that, in this paper, we consider only binary configuration signals for practical reasons. This restriction limits the gate reconfigurability to eight functions. In [5], ternary signals were considered, increasing the logic capability of the cell to 14 functions. We qualify the circuit as operating at ultrafine grain. The terminology of ultrafine grain is twofold. First, it covers the size of the cell, which is highly compact, as opposed to its CMOS equivalent. Second, it also describes the granularity of the covered logic functions, as compared with a traditional four-input LUT.

III. Leveraging the Ultrafine Granularity at the Architectural Level

In the previous section, we saw how the new class of controllable-polarity devices leads to compact reconfigurable LCs. In this section, we will move toward the architectural level and study the impact of the blocks onto the FPGA architecture.

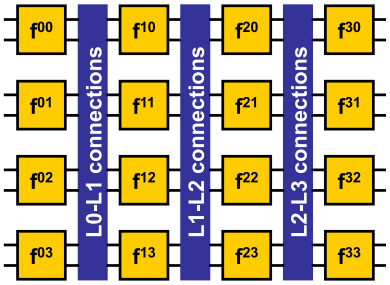


Fig. 5. MClusters_4_4 arrangement for reconfigurable architectures.

A. Fine-Grain Logic Integration

We replace the traditional LUTs by an *ultrafine grain LC*. Such a modification, in a conventional FPGA architecture, will require each cell to be directly connected to a full connectivity unit. In [5], the typical cell size, which is less than 10 transistors, is comparable with a 1-bit switchbox. This projection would, therefore, result in a large overhead in terms of connections and would worsen the already significant imbalance between routing and logic resources at the FPGA level.

To correct the granularity issues, we propose to pack the cells into an intermediate structure, compatible in terms of size with the traditional levels of FPGAs, and called *MClusters* for Matrix *Clusters*.

B. MCluster Organization

The MCluster organization is defined by the LC organization and by a specific interconnect.

1) MClusters: Adding a Logic Layer: To increase the logic coverage of the structure, we propose a 2-D matrix assembly of ultrafine grain logic. The LCs are arranged in a layered structure, with connections only existing between adjacent layers. This avoids long connections and maximizes local connectivity. The layers are simple levels of logic with no pipeline or signal restoration. This organization is shown in Fig. 5. In the following, MClusters will be defined by the d (depth) and w (width) parameters and named as: *MClusters_d_w*. In Fig. 5, the single output Y of the LC [Fig. 4(a)] is represented with a fan-out of two to have the same number of the input and output terminals per layers. The matrix patterns exhibit a fundamental structural regularity that is easily transposable to regular fabrics [6]. Note that the architectural concept is not limited to the presented architecture. Indeed, it is possible to consider diamond-shaped or triangular matrices, and to have interconnects crossing different layers. However, to keep the focus of this paper, only squared matrices, i.e., $d \times w$ LCs and $d = w$, and layer-to-layer connections are considered.

2) MClusters: Simplifying the Interconnect Overhead: For the intramatrix interconnect, any topology giving a full interconnectivity is prohibited, due to the associated wiring complexity, and the additional area requirements. Instead, and

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

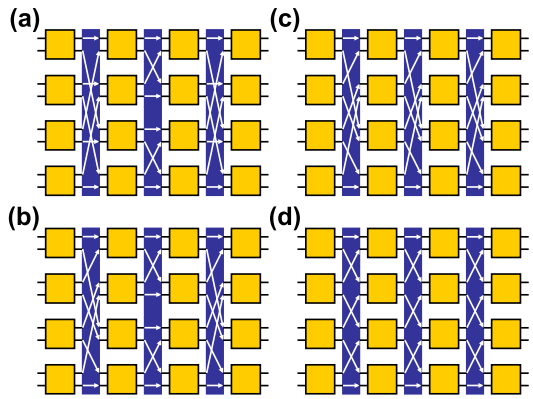


Fig. 6. Matrix of 16 reconfigurable gates with fixed interconnect topology. (a) Banyan. (b) Baseline. (c) Flip. (d) Modified Omega.

through analogy to computer networks, our approach is to adapt incomplete interconnection sets to the matrix architecture. Multistage interconnection networks (MINs) are designed to interconnect computing layers in an efficient way and can be applied in this context. In computer engineering, MINs are used to interconnect layers of switchboxes to route information packets only. This concept has been reported many times in interconnect strategies for VLSI [19] and FPGAs to reduce the complexity of wiring between the logic blocks [22]. Note that the main difference, with respect to the network context, is that switchboxes have been replaced by LCs, thus introducing computing within the network itself. Furthermore, the use of very local MIN-style interconnect has a drastic impact on the size and the wire length is reduced accordingly. There are many MIN topologies and combinations [20], [21]. Four typical permutations are reported in Fig. 6: Banyan [Fig. 6(a)], Baseline [Fig. 6(b)], Flip [Fig. 6(c)], and Modified Omega [Fig. 6(d)], where the modifications to standard Omega maximize the shuffling between the layers. Since the interconnect topology is fixed and static, the choice of topology is made by the designer. In the context of matrix-based logic, the performance of the different topologies has been investigated in [23]. It has been shown that a Modified Omega topology has the best performance in terms of mapping efficiency. Therefore, in the following, we consider only Modified Omega interconnect topology.

3) BLE and CLB Organization: MClusters perform combinational logic functions and are good replacement candidates for the original LUTs. Fig. 7 shows the organization of the logic blocks at the CLB level. Each BLE consists of an *MClusters_d_w*, whose outputs can be individually latched on demand. The CLB consists in a collection of N MCluster-based BLEs. All the BLE outputs, i.e., $w \times N$ signals, are connected to the global reconfigurable interconnect and are also fed back to the inputs of the CLB, thus achieving a full connectivity pattern.

At the top architectural level, we consider a standard FPGA organization, where the CLBs are interconnected by a complete interconnect set, but limited in terms of resources, i.e., all the connections cannot be realized simultaneously.

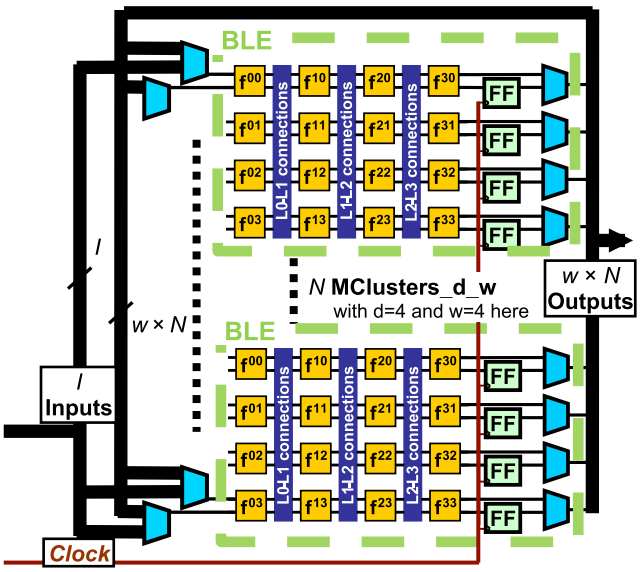


Fig. 7. MCluster-based CLB proposal.

IV. MCluster CAD Flow

In this section, we describe the tool suite supporting the MClusters architecture. To analyze the performance gain from an application perspective and to compare it to existing approaches, the benchmarking of standard circuits has to be carried out. We propose a complete benchmarking tool flow that is designed on top of the well-known verilog-to-routing (VTR) flow [25]. We first discuss the benchmarking flow strategy and the integration of a novel packer, called MPack, specifically designed to handle the MCluster packing step with fixed interconnect topologies.

A. General Overview of the Flow

Evaluations of an FPGA-like architecture can be conducted with traditional FPGA tool flows [25]. In an FPGA flow, a set of logic circuits, called benchmarks, are first synthesized using the ABC synthesis tool. Subsequently, the logic packing of the synthesized circuit into CLBs is performed, using AAPACK. Finally, the placement and routing are carried out using versatile place and route (VPR). To enable power estimation, ACE2 is used to feed VPR with the net activities.

However, the MCluster architecture introduces new concepts, not handled by traditional FPGA tools. In particular, the layered structure of the interconnect topology makes the use of traditional packing tools impossible. Therefore, a specific tool, called MPack, is added to the flow. The whole considered flow is shown in Fig. 8, MPack handles the specific interconnect topologies, as well as the associated buffering generation required by the layered structure. The tool is further described in the following section.

B. MPack: The Matrix Packer

The Matrix Packer, *MPack*, has been designed to pack netlists of LCs into MClusters. Fig. 9 shows the internal organization of MPack. The tool consists of two principal algorithms: the mapper and the clusterer. The mapper

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

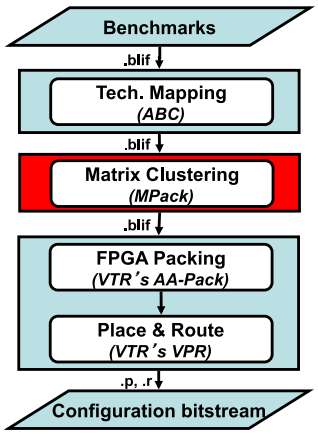


Fig. 8. MCluster-compatible benchmarking flow diagram.

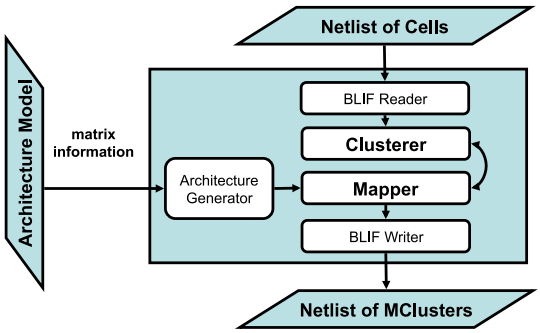


Fig. 9. MPack model flow.

maps simple logic functions on a unique MCluster. The clusterer splits the large initial logic network into subgraphs that will be subsequently mapped on individual MClusters thanks to the mapper. To ensure the correct integration into the whole flow, netlists are read and exported using the Berkeley Logic Interchange Format (BLIF) file format.

The targeted MClusters shape is parameterized in terms of the size and topology scheme. The Architecture Generator generates the MClusters template. An example of a Modified Omega topology with $d = w = 3$ is shown in Fig. 10(a). In this figure, the nodes f^{ij} are uniquely addressed by the indices i and j with i indicating the row and j the column [see Fig. 10(a) for an example]. In the tool internal representation, the connectivity between nodes is represented by the adjacency matrix of the graph. In such matrices, a *1* at the position (i, j) means that the node i is connected to the node j. We also define the local adjacency matrices X_{nm} between the LC stages at depth n to m as the individual cross-connectivity matrices. For instance, X_{01} and X_{12} are shown in Fig. 10(b). If we consider the first row of the matrix X_{01} that shows the connectivity of the cell f^{10}, we can say that f^{10} is connected to f^{00} and f^{01}.

C. Matrix Mapping Algorithm

The mapping algorithm aims to fit a netlist of LCs onto an MClusters architecture. The module is split into two parts. First, an architecture optimization is performed to adapt the netlist to the physical architectural scheme.

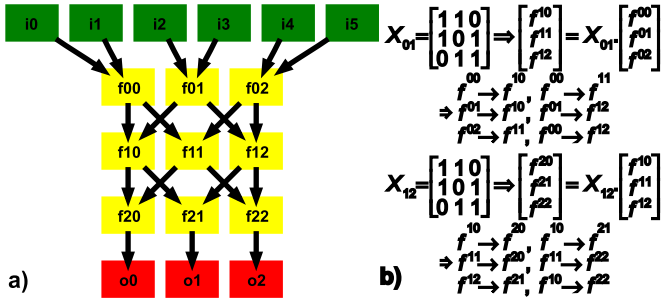


Fig. 10. (a) MClusters_3_3 with a Modified Omega topology (LCs are in yellow, virtual input nodes are in green, and virtual output nodes are in red) and (b) associated cross-connectivity matrices.

Then, the logic function netlist is mapped to the architecture netlist.

1) Architectural Optimization: The architectural optimization is first performed to adapt the netlist to the physical architectural scheme. Due to the layered structure, the logic can be seen as pipelined. The input netlist has to be processed by adapting its connections to conform to the physical topology. For instance, in a typical postsynthesis netlist, any logic gate output can be connected to several gate inputs. However, MClusters have a fixed fan-in and fan-out. Therefore, the netlist is made MClusters-compatible by adding buffering nodes to limit the fan-out. An optimization example is given in Fig. 11. The netlist is processed as follows.

a) Jump correction: A connection which jumps at least one logic layer is not allowed by the layered topology. Then, a connection between two layers must pass through a LC. Consequently, the creation of a path (by the addition of a buffer cell) instead of a jump is necessary to handle such a situation, as shown in Fig. 11(a).

b) Multiple output correction: In the MClusters organization, the LCs are connected to a limited number of other cells, i.e., with a fan-in/fan-out limitation. Larger fan-in/fan-out requires duplicating the signals using buffers. Fig. 11(a) shows an example case, where node f0 drives three different nodes. In this example, it is not possible to obtain more than two outputs per logic node. Hence, a buffer is added to reach the constraint [Fig. 11(b)].

c) Output layer correction: In this final optimization step, all the outputs of the logic functions are placed on the physical output layer. In Fig. 11(b), we considered a logic function where the outputs are on layer L3. To meet the constraints of the architecture (here a matrix of depth 3), buffers are added to place the outputs on the layer L4 [Fig. 11(c)].

2) Mapping Algorithm: After the preprocessing steps, the logic function netlist is mapped to the architecture netlist, using the recursive algorithm reported in Fig. 13.

First, the adapted netlist is analyzed in depth, meaning that for each node in the structure, child branches are identified and recursively explored. This depth exploration is performed without any consideration of the edge orientation in the graph. This allows us to identify the connections between the nodes and to initialize of the mapping sequence.

Second, logic nodes are assigned to cells, with respects to the physical interconnections. Each layer's connections

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

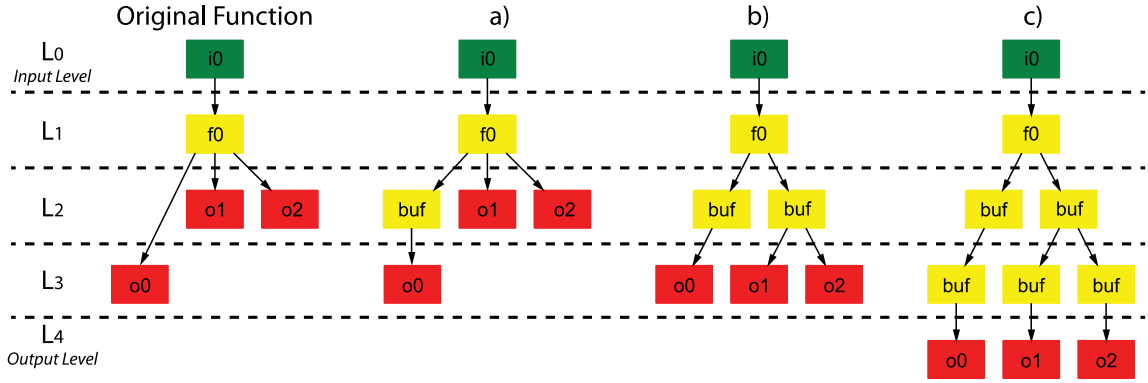


Fig. 11. Architectural optimization of the function graph. Original function before formatting. (a) Jump correction. (b) Fan-in/fan-out correction. (c) Output layer correction by buffer insertion.

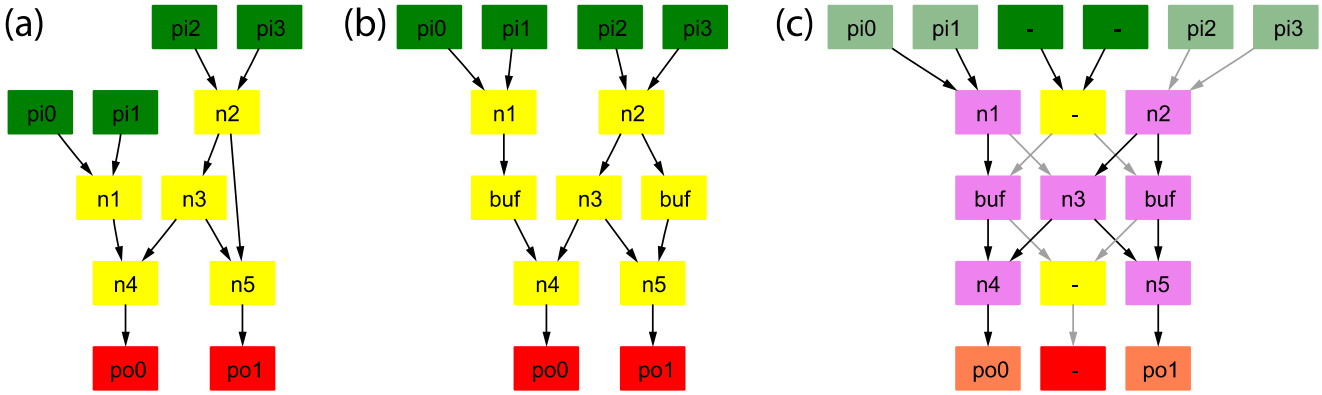


Fig. 12. Mapping example. (a) Original function graph to map onto an MClusters_3_3 with a Modified Omega interconnect topology. (b) Function graph after correction step. (c) Mapped function.

are compared with the relevant interlayer architectural connectivity—allowing (or not) the assignment of functions to cells. Branching, i.e., the exploration of the immediately preceding alternative, is used when the arbitrary choice leads to a dead-end. The process is repeated until all functional nodes are assigned to physical cells.

For instance, let us consider a matrix which is three-cell deep and three-cell wide (MClusters_3_3) using a Modified Omega interconnect topology [Fig. 10(a)]. A simple function to map is shown in Fig. 12(a). With the previously described adaptation methodology, the original function graph is corrected to maximize the matching with the targeted MCluster. Hence, the position of the inputs/outputs and interlayer jumps will be corrected. The corrected graph is shown in Fig. 12(b). The graph exploration is then triggered and the following sequence (*buf* represents synchronization nodes) is obtained:

$$pi0\text{-}n1\text{-}pi1\text{-}buf\text{-}n4\text{-}n3\text{-}n2\text{-}pi2\text{-}pi3\text{-}buf\text{-}n5\text{-}po1\text{-}po0.$$

Subsequently, the graph assignment is performed. In the example, the first point *n1* is assigned to the cell f^{00}. According to the path defined in the previous step, a buffer is the next node to assign to a cell in the matrix. Since f^{00} is physically connected to f^{10} and f^{12}, the cell with lower index (here f^{10}) is arbitrarily chosen for the buffer assignment, and the other possibility is memorized. In our example, the final programmed matrix is shown in Fig. 12(c). In this figure, we can see the nodes of the logic function graph and the nodes added for synchronization purposes correctly placed on the cell matrix.

D. Clustering Algorithm

The clustering algorithm splits the initial logic network into subgraphs that will subsequently be mapped onto MClusters. Such functionality is widely used in computer-aided design (CAD) tools. In the MPack tool, we used algorithms derived from VPACK [13]. The tool constructs each MCluster sequentially, where the algorithm greedily packs the cells into MClusters. The pseudocode for the algorithm is shown in Fig. 14.

It begins by choosing a seed cell for the current MCluster. As described by [13], the best way to choose the seed is to select the unclustered cell with the most used inputs. This approach prioritizes early placement of cells using the largest number of cluster inputs, which are a scarce resource. Next, the algorithm selects the cell with the highest attraction to the current MCluster, and checks if it could legally be added to the current cluster. The previously described mapping algorithm performs the evaluation of legality. In effect, we check if the cell added to the current MCluster leads to a valid and mappable MCluster. If the cluster is mappable, then the cell

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.


```
FUNCTION Mapping_procedure(current node)
 IF (all children/parents of current node placed)
 THEN
   FOR (empty cell connected to children or parents)
      Store the potential positions
   END FOR
 ELSE Store all the positions on the current level
 END IF

 IF (No positions have been found)
 THEN
    MAPPING FAILURE
 END IF

 FOR (Each potential position)
    Place the current node at the position
    Mapping_procedure(Next node)
    IF (Return is MAPPING FAILURE)
    THEN
      Unplace the current node from the position
    END IF
    IF (Return is MAPPING SUCCESS AND No more nodes)
    THEN
      MAPPING SUCCESS
    END IF
 END FOR

 IF (No potential positions have been correct)
 THEN
   MAPPING FAILURE
 END IF
END FUNCTION
```

Fig. 13. MPack mapping algorithm (pseudocode).

```
Remain_Cells: set of unclustered logic cells
C: set of cells packed in the current MCluster
MClusters: set of MClusters

MClusters = NULL
WHILE (Remaining_Cells != NULL)
// More Cells to cluster
 C = Seed (Remaining_Cells)
 // Most Used Inputs and legal Cell
 WHILE (|C| < (MCluster_w.MCluster_d))
  // Cluster is not full
  Best_Cell = MaxAttractionLegalCell(C,Remain_Cells)

  IF (Best_Cell == NULL)
     BREAK
  END IF

  Remain_Cells = Remain_Cells - Best_Cell
  C = C ∪ Best_Cell
 END WHILE
 MClusters = MClusters ∪ C
END WHILE
```

Fig. 14. Clustering algorithm (pseudocode).

is definitively added to the current cluster. The attraction between a LC and the current MCluster (MC) is, as described in [13], the number of the inputs and outputs they have in common

$$\text{Attraction (LC)} = |\text{Nets (LC)} \cap \text{Nets (MC)}|.$$

This greedy procedure continues until either: 1) the MCluster is full or 2) adding any additional cell would make the current cluster illegal. If the cluster is full, a new seed is selected and the packing starts for another MCluster.

V. Experimental Results

In this section, we evaluate the architecture introduced in Section III, thanks to the tool flow presented in Section IV. After describing the methodology, we perform an architectural exploration to identify the best MClusters sizing. Then, we compare the approach with its traditional CMOS-based counterpart.

A. Methodology

In this paper, we study the impact of replacing standard LUTs by MClusters-based structures. To keep our study simple, we evaluate the performances of a set of logic circuits taken from the Microelectronics Center of North Carolina (MCNC) and ISCAS'89 mapped on simple homogeneous architectures. Our baseline FPGA architecture corresponds to a fully homogeneous FPGA architecture. The CMOS reference architecture is composed of four-input nonfractionable LUTs with a CLB organization of $N = 10$ BLEs and $I = 22$ external inputs. This architecture is optimal for homogeneous FPGAs [26]. We then proceed to a one-to-one replacement of the LUTs by MClusters according to Fig. 7, i.e., keeping $N = 10$. MClusters_1_1 to MClusters_4_4 are evaluated. The number of BLEs inputs and outputs grows with the size of the MCluster. Therefore, to ensure a proper routability of the logic cluster, we set the number of CLBs external inputs to

$$I = \frac{(N+1) \times nb_\text{MCluster_inputs}}{2}.$$

This relation typically ensures that 98% BLEs are utilized on average [26]. As an example, $I = 22$ and $I = 44$ are used when considering MClusters_2_2 and MClusters_4_4, respectively. For these two examples, the local routing circuits consist of a set of 42-bit and 84-bit multiplexers, respectively. SRAM circuits are used to store the configuration information for both LUTs and MClusters.

The evaluations are performed using the benchmarking flow described in Section IV and shown in Fig. 8. The physical parameters of the different architectures are extracted for a 22-nm technological node [1]. MClusters and LUTs area evaluation estimates the size of the blocks as a function of the elementary transistors area [1], [12] and the transistor sizing. For instance, a 2 by 2 cluster costs an area of 2.22 μm^2 while a four-input LUT costs 5.45 μm^2. Electrical performances of the elementary MClusters and LUTs are electrically characterized using HSPICE (FinFET high performance at 22-nm model took from [27] and simple table-based model for DG-SiNWFETs took from [12]) to extract their basic average delay and power consumption numbers.

The architectural evaluation considers, as metrics, the area, the critical path delay, the dynamic power consumption, and the leakage power. These metrics are computed during the place and route iterations of the flow. The area corresponds to the sum of the logic area, i.e., the area of used CLBs, and the routing area, i.e., the area of the used routing resources. The critical path delay corresponds to the most constrained delay through the implemented structures. Finally, the power consumptions include both the contribution of logic blocks

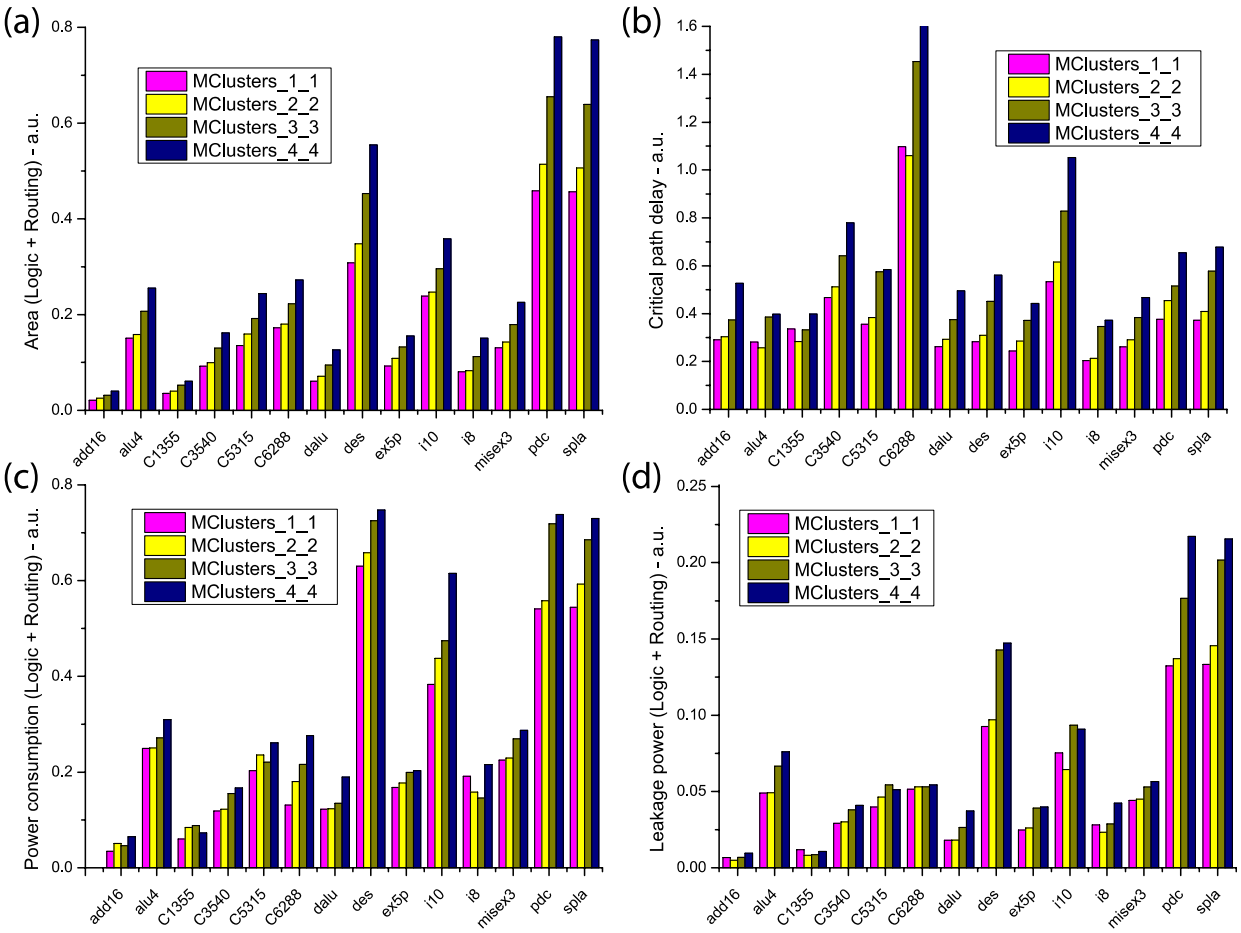


Fig. 15. Impact of various granularities on MCluster-based FPGAs. (a) Global area. (b) Critical path delay. (c) Dynamic power consumption. (d) Leakage power consumption. MClusters are sized as squares ranging from 1 to 4 reconfigurable cells.

and the contribution of routing structures. All the metrics are normalized with respect to the most constrained CMOS design.

B. Impact of the Granularity

We evaluate the impact of MCluster sizes by studying the area, the delay, and the power consumptions of standard benchmarks. In this paper, we vary the dimensions of squared-shaped MClusters from MClusters_1_1 to MClusters_4_4. Note that averaged over the different benchmarks and different cluster sizes, the utilization rate of the MClusters, i.e., the ratio between the employed cells and the total number of cells, reaches 90%, thereby indicating a good level of performances of MPack.

Fig. 15(a) shows the area estimation of an FPGA, using the proposed architectural scheme. The best results are obtained for small MClusters, i.e., MClusters_1_1 and MClusters_2_2. Note that, for a higher granularity, the area increases drastically. For large clusters, this is the direct impact of their internal routing. Indeed, in such cases, the incomplete interconnectivity becomes a major hurdle, because of the large number of wasted cells used only for buffering purposes.

Fig. 15(b)–(d) show the critical path delay, the dynamic power consumption, and the leakage power consumption, respectively, in the same conditions than for the previous study. We note that the best results are obtained for the small granularities, i.e., MClusters_1_1 and MClusters_2_2. Indeed,

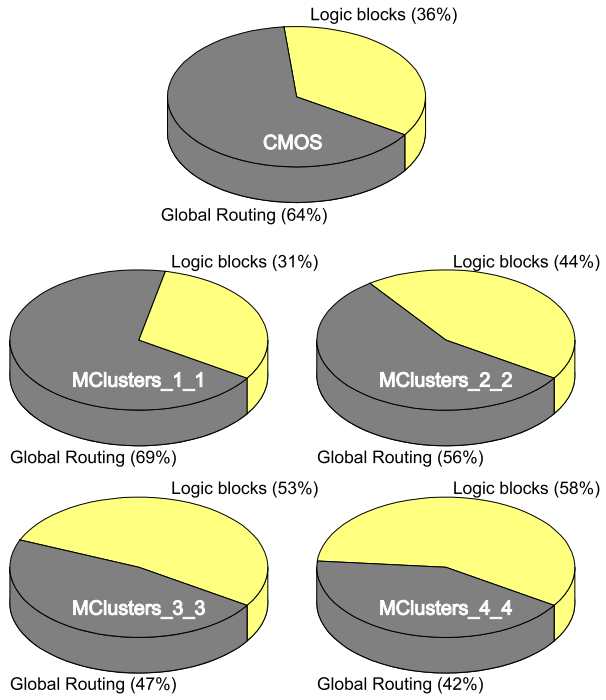


Fig. 16. Area breakdown of an interconnection intensive benchmark (i10) for 2 by 2 MCluster-based and CMOS-based FPGAs.

the delay and the power consumption are almost linearly dependent on the size of the MClusters. Large MClusters mean

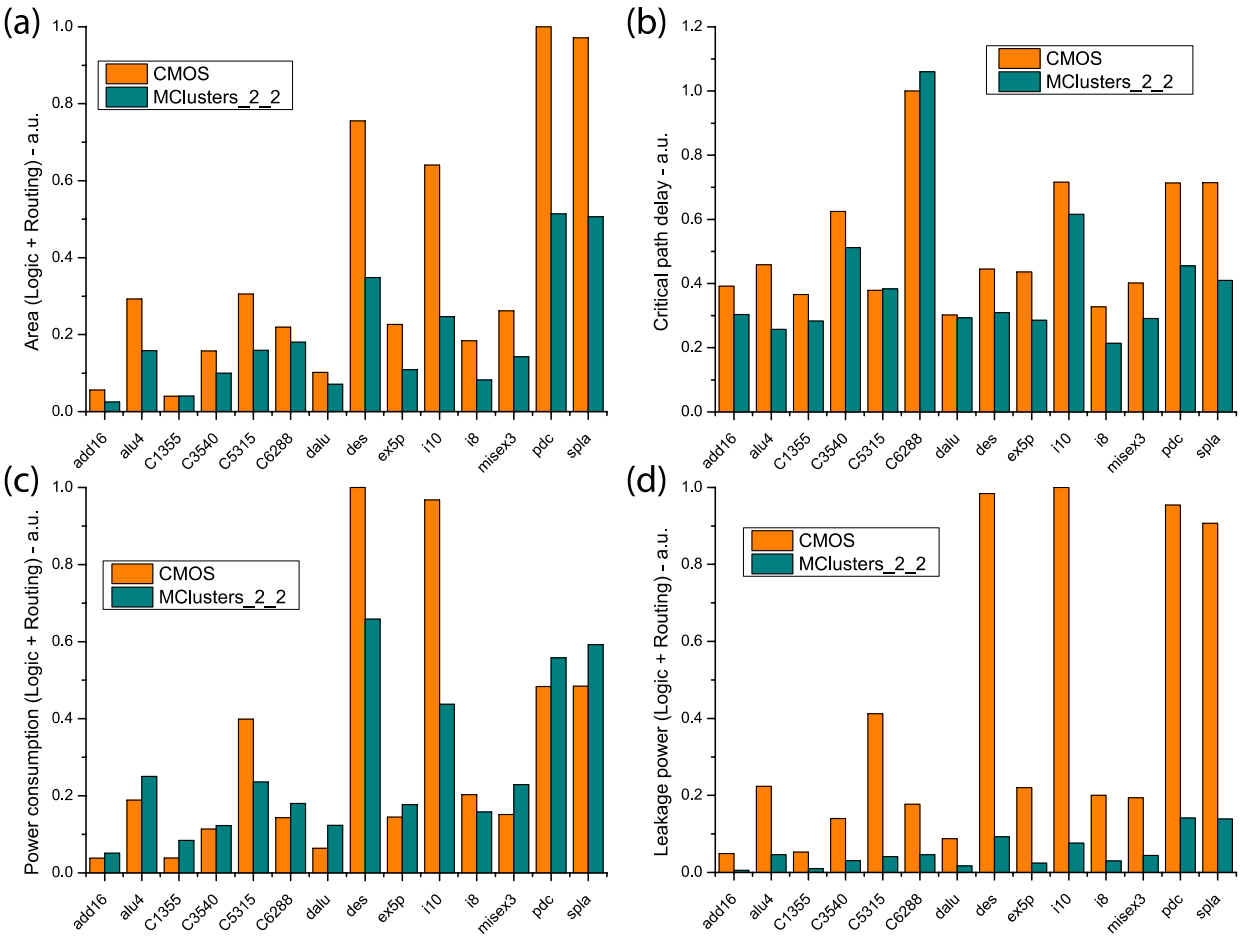


Fig. 17. Performance comparisons between CMOS-based and MClusters_2_2-based FPGAs. (a) Global area. (b) Critical path delay. (c) Dynamic power consumption. (d) Leakage power consumption.

deeper logic pipeline and more cells only used as buffers, therefore directly reducing the performances and increasing the power consumption. MClusters_1_1 overperforms slightly MClusters_2_2 with 5%, 4%, 12%, and 1% gains in the area, the delay, the dynamic power, and the leakage power, respectively. However, we also consider in Fig. 16 the impact of the different MCluster granularities on the area breakdown between the logic blocks and global routing for an interconnection intensive benchmark (i10). As already mentioned, a fundamental drawback of the traditional FPGA architecture is its strong imbalance between the routing peripherals and the logic. In Fig. 16, we can see that the use of MClusters_1_1 worsen the natural imbalance, confirming the initial hypothesis of this paper, where the direct transposition of an FPGA architecture to ultrafine grain would lead to an interconnection overhead. Conversely, large clusters can increase the logic share, but they will also strongly limit the performances of the architecture, as shown previously.

In the following, we will select MClusters_2_2 as a good tradeoff between performances (Fig. 15) and logic/routing balance (Fig. 16).

C. Performance Comparison With CMOS

Fig. 17(a) shows the area estimation for MClusters_2_2-based FPGA and compares it to its CMOS counterpart. The benchmarks show an area reduction of up to 61%, with an average of 43%. This can be accounted: 1) to the performance of a logic-gate-based computation (as compared to the LUT approach) and 2) to the low area impact of ultrafine grain LCs, compared with the rather larger area required by a CMOS LUT. As mentioned previously, a 2 by 2 cluster is 2× smaller than a four-input LUT. Furthermore, the functionality of MClusters_2_2 is higher than its equivalent LUT. A four-input LUT computes a single output signal depending on four inputs. While MClusters_2_2 can only realize a subset of the functions reachable by a LUT, they are capable to produce up to two outputs that are functions of the same four inputs. Thus, MClusters can potentially output 2× more results for roughly 2× less area. Correlated to the efficiency of the packing tool for matrix clustering, this demonstrates a clear advantage of our proposal as compared with the CMOS approach.

These results still hold for the delay [Fig. 17(b)], the dynamic power consumption [Fig. 17(c)], and the leakage power consumption [Fig. 17(d)]. With regards to the standard FPGA counterpart, the critical path delay is improved by up to 44% with an average value of 23%. The performance improvement can be accounted to the higher performance of the block structure as compared with LUTs. A dynamic power consumption reduction of up to 54% is observed, though, on average, we observe an increase of 19% in the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

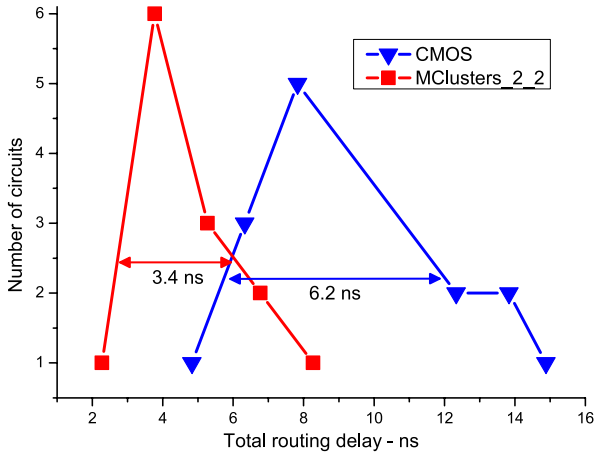


Fig. 18. Critical net routing delay repartition for CMOS-based and 2 by 2 MCluster-based FPGAs.

dissipated power. Nevertheless, we should remark that large dynamic power consumptions reductions are observed on the larger benchmarks, showing a promising trend toward the power reduction of larger circuits. Finally, leakage power can be reduced by up to 94% with an average reduction of 83%. This large reduction can be accounted to the ultralow power capabilities of the controllable-polarity transistors, coming from the gate-all-around structure, coupled to the resources reduction coming from the MCluster approach. Note that even if the granularity chosen in the previous section is not optimal for delay and power numbers, the improvements as compared with traditional solutions are significant and can be further pushed by choosing an appropriate granularity.

In addition to the previous conclusions, we extend the study to the critical net routing delay. This metric strongly relates to the FPGA architecture efficiency as well. We observe from the distribution of the critical routing delay given by Fig. 18 that, in addition to improve the average routing delay by 43%, the use of MClusters allows us to reduce the standard deviation of the delay distribution by 45%. We can remark that, while the CMOS distribution is quite large, the use of a MClusters_2_2 implies a lowering of the extremes, and tends to globally improve the performance of the mapped circuits. This behavior can be explained through the global impact of ultrafine granularity on the benchmarking toolflow. The ultrafine granularity induces a predominance of local inter-CLB interconnect instead of long wire connections, thus leading to the reduction of long critical paths. Such results are interesting from an architectural perspective as it tends to homogenize the results of mapped circuits and makes them less dependent to the route phase.

VI. Conclusion

This paper presents a new architectural scheme for FPGAs, which is based on the use of controllable-polarity transistors. Thanks to the introduced reconfigurability at the transistor level, ultrafine grain reconfigurable logic gates can be introduced. We propose to use the matrices of logic gates interconnected through a fixed interconnection pattern to replace LUTs in FPGAs. To evaluate this approach objectively, we developed a complete tool flow, able to pack logic circuits onto the considered architecture. We evaluated the potential of the architecture and compared it to a regular CMOS FPGA. We showed that 2 by 2 clusters give an average improvement of 43% and 23% area and delay, respectively, with respect to CMOS LUT-based FPGAs at 22-nm technology node. Finally, we observed that the proposed approach opens a promising path to correct two intrinsic limitations of the FPGA circuits, with a reduction in the logic/routing imbalance, and a better control in the distribution of the routing delay.

Acknowledgment

The authors would like to thank Pr. I. O'Connor (INL, Lyon, France) and Dr. F. Clermidy (CEA-LETI, Minatec Campus, Grenoble, France) for the fruitful discussions.

References

[1] C. Auth *et al.*, "A 22 nm high performance and low-power CMOS technology featuring fully-depleted tri-gate transistors, self-aligned contacts and high density MIM capacitors," in *Proc. Symp. VLSI Technol.*, Jun. 2012, pp. 131–132.

[2] H.-S. P. Wong *et al.*, "Carbon nanotube electronics—Materials, devices, circuits, design, modeling, and performance projection," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, Dec. 2011, pp. 23.1.1–23.1.4.

[3] S. D. Suk *et al.*, "High performance 5 nm radius twin silicon nanowire MOSFET (TSNWFET): Fabrication on bulk si wafer, characteristics, and reliability," in *Proc. IEEE Int. Electron Devices Meeting (IEDM)*, Dec. 2005, pp. 717–720.

[4] S. Bangsaruntip *et al.*, "High performance and highly uniform gate-all-around silicon nanowire MOSFETs with wire size dependent scaling," in *Proc. Int. Electron Devices Meeting (IEDM)*, Dec. 2009, pp. 1–4.

[5] I. O'Connor *et al.*, "CNTFET modeling and reconfigurable logic-circuit design," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 11, pp. 2365–2379, Nov. 2007.

[6] P.-E. Gaillardon, L. G. Amarù, S. Bobba, M. De Marchi, D. Sacchetto, and G. De Micheli, "Nanowire systems: Technology and design," *Philosoph. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 372, no. 2012, p. 20130102, Mar. 2014.

[7] M. H. B. Jamaa *et al.*, "FPGA design with double-gate carbon nanotube transistors," *Electroch. Soc. Trans.*, vol. 34, no. 1, pp. 1005–1010, 2011.

[8] J. Appenzeller, J. Knoch, E. Tutuc, M. Reuter, and S. Guha, "Dual-gate silicon nanowire transistors with nickel silicide contacts," in *Proc. Int. Electron Devices Meeting (IEDM)*, Dec. 2006, pp. 1–4.

[9] A. Heinzig, S. Slesazeck, F. Kreupl, T. Mikolajick, and W. M. Weber, "Reconfigurable silicon nanowire transistors," *Nano Lett.*, vol. 12, no. 1, pp. 119–124, 2011.

[10] Y.-M. Lin, J. Appenzeller, J. Knoch, and P. Avouris, "High-performance carbon nanotube field-effect transistor with tunable polarities," *IEEE Trans. Nanotechnol.*, vol. 4, no. 5, pp. 481–489, Sep. 2005.

[11] N. Harada, K. Yagi, S. Sato, and N. Yokoyama, "A polarity-controllable graphene inverter," *Appl. Phys. Lett.*, vol. 96, no. 1, pp. 012102-1–012102-3, Jan. 2010.

[12] M. De Marchi *et al.*, "Polarity control in double-gate, gate-all-around vertically stacked silicon nanowire FETs," in *Proc. IEEE Int. Electron Devices Meeting*, Dec. 2012, pp. 8.4.1–8.4.1.

[13] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. New York, NY, USA: Kluwer, 1999.

[14] P.-E. Gaillardon, M. H. Ben-Jamaa, F. Clermidy, I. O'Connor, "Ultra-fine grain FPGAs: A granularity study," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NanoArch)*, Jun. 2011, pp. 9–15.

[15] M. Lin, A. El Gamal, Y.-C. Lu, and S. Wong, "Performance benefits of monolithically stacked 3-D FPGA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 2, pp. 216–229, Feb. 2007.

[16] A. Colli, S. Pisana, A. Fasoli, J. Robertson, and A. C. Ferrari, "Electronic transport in ambipolar silicon nanowires," *Phys. Status Solidi B*, vol. 244, no. 11, pp. 4161–4164, 2007.

[17] R. Martel *et al.*, "Ambipolar electrical transport in semiconducting single-wall carbon nanotubes," *Phys. Rev. Lett.*, vol. 87, no. 25, p. 256805, Dec. 2001.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[18] A. K. Geim and K. S. Novoselov, "The rise of graphene," *Nature Mater.*, vol. 6, no. 3, pp. 183–191, 2007.

[19] D. L. Lewis, S. Yalamanchili, and H.-H. S. Lee, "High performance non-blocking switch design in 3D die-stacking technology," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, May 2009, pp. 25–30.

[20] C.-L. Wu and T.-Y. Feng, "On a class of multistage interconnection networks," *IEEE Trans. Comput.*, vol. C-29, no. 8, pp. 694–702, Aug. 1980.

[21] G. B. Adams, III, D. P. Agrawal, and H. J. Seigel, "A survey and comparision of fault-tolerant multistage interconnection networks," *Computers*, vol. 20, no. 6, pp. 14–27, Jun. 1987.

[22] Z. Marrakchi, H. Mrabet, C. Masson, and H. Mehrez, "Efficient mesh of tree interconnect for FPGA architecture," in *Proc. Int. Conf. Field-Program. Technol. (ICFPT)*, Dec. 2007, pp. 269–272.

[23] P.-E. Gaillardon, I. O'Connor, J. Liu, and F. Clermidy, "Interconnection scheme and associated mapping method of reconfigurable cell matrices based on nanoscale devices," in *Proc. IEEE/ACM Int. Symp. Nanosc. Archit. (NANOARCH)*, San Francisco, CA, USA, Jul. 2009, pp. 69–74.

[24] P.-E. Gaillardon, I. O'Connor, M. Amadou, J. Liu, F. Clermidy, and G. Nicolescu, "Matrix nanodevice-based logic architectures and associated functional mapping method," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 7, no. 1, pp. 3:1–3:23, Jan. 2011.

[25] J. Rose *et al.*, "The VTR project: Architecture and CAD for FPGAs from Verilog to routing," *Int. Symp. Field Program. Gate Arrays*, 2012, pp. 77–86.

[26] E. Ahmed, "The effect of logic block granularity on deep-submicron fpga performance and density," M.S. thesis, Dept. Elect. Comput. Eng., Univ. Toronto, Toronto, ON, Canada, 2001.

[27] S. Sinha, G. Yeric, V. Chandra, B. Cline, and Y. Cao, "Exploring sub-20 nm FinFET design with predictive technology models," in *Proc. 49th ACM/EDAC/IEEE Design Autom. Conf.*, Jun. 2012, pp. 283–288.

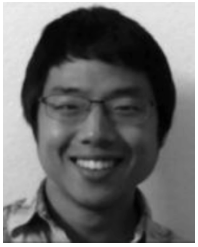
Pierre-Emmanuel Gaillardon (S'10–M'11) received the Degree in electrical engineering from the École Supérieure de Chimie Physique Électronique de Lyon, Lyon, France, in 2008, the M.Sc. degree from the Institut National des Sciences Appliquées de Lyon, Lyon, in 2008, and the Ph.D. degree in electrical engineering from the University of Lyon, Lyon, in 2011.

He was a Research Assistant with CEA-LETI, Grenoble, France, where he was involved in the Nanosys Project. He is currently with the Laboratory of Integrated Systems, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, as a Research Associate. His current research interests include emerging nanoscale devices and their use in digital circuits and architectures.

Dr. Gaillardon was a recipient of the 2011 C-Innov Best Thesis Award and the 2012 Nanoarch Best Paper Award. He has served as the TPC member for 2012 Nanoarch and 2013 conferences. He is a reviewer for several journals, including the AIP *Applied Physics Letters*, the IEEE TRANSACTIONS ON NANOTECHNOLOGY, the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION, and ACM *Journal on Emerging Technologies in Computing Systems*, the conferences including the 2012 IEEE International Conference on Electronics, Circuits, and Systems and the 2013 IEEE International Symposium on Circuits and Systems 2013, and funding agencies such as ANR and the Chairs of Excellence Program of the Nanosciences Foundation.

Xifan Tang received the B.Sc. degree in microelectronics from Fudan University, Shanghai, China, in 2011, and the M.Sc. degree in electrical engineering from the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2013, where he is currently pursuing the Ph.D. degree.

His current research interests include computer-aided design for programmable architecture and emerging technologies.



Gain Kim received the B.Sc. degree in electrical engineering from the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2013, where he is currently pursuing the master's degree.

He was with the Laboratory of Integrated Systems, École Polytechnique Fédérale de Lausanne, as an Undergraduate Research Assistant in 2013.

Giovanni De Micheli (F'94) is currently a Professor and the Director of the Institute of Electrical Engineering and the Integrated Systems Centre with the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. He is also a Program Leader of the Nano-Tera.ch Program. He is interested in heterogeneous platform design, including electrical components and biosensors, and in data processing of biomedical information. He is a member of the Scientific Advisory Board of imec, Leuven, Belgium, and STMicroelectronics, Geneva, Switzerland. He is an author of a book entitled *Synthesis and Optimization of Digital Circuits* (McGraw-Hill, 1994), and co-author and co-editor of eight other books and over 500 technical articles. His citation h-index is 81 according to the Google Scholar. His current research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis for emerging technologies, networks-on-chips, and 3-D integration.

Prof. De Micheli is a fellow of the Association for Computing Machinery, and a member of the Academia Europaea. He was a recipient of the 2012 IEEE Circuits and Systems Society (CAS) Mac Van Valkenburg Award for contributions to theory, practice and experimentation in design methods and tools, and the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He was also a recipient of the Golden Jubilee Medal for Outstanding Contributions to the IEEE CAS in 2000, the D. Pederson Award for the Best Paper on the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN and the IEEE International Conference on Autonomic and Autonomous Systems (ICAS) in 1987, and several Best Paper Awards, including the Design Automation Conference (DAC) in 1983 and 1993, Design Automation and Test in Europe (DATE) Conference in 2005, and Nanoarch in 2010 and 2012. He has served the IEEE in several capacities, namely, the Division 1 Director from 2008 to 2009, the Co-Founder and President Elect of the IEEE Council on Electronic Design Automation from 2005 to 2007, the President of the IEEE CAS Society in 2003, the Editor-in-Chief of the IEEE TRANSACTIONS ON TRANSACTIONS ON COMPUTER-AIDED DESIGN and the IEEE ICAS from 1987 to 2001. He has been the Chair of several conferences, including DATE in 2010, pHealth in 2006, the International Conference on Very Large Scale Integration in 2006, DAC in 2000, and the IEEE International Conference on Computer Design in 1989.