A Circuit Synthesis Flow for Controllable-Polarity Transistors

Luca Amarú, *Student Member, IEEE*, Pierre-Emmanuel Gaillardon, *Member, IEEE*, and Giovanni De Micheli, *Fellow, IEEE*

***Abstract*—*Double-gate* (DG) controllable-polarity *field-effect transistors* (FETs) are devices whose n- or p- polarity is online configurable by adjusting the second gate voltage. Such emerging transistors have been fabricated in silicon nanowires, carbon nanotubes, and graphene technologies. Thanks to their enhanced functionality, DG controllable-polarity FETs implement arithmetic functions, such as XOR and MAJ, with limited physical resources enabling compact and high-performance datapaths. In order to design digital circuits with this technology, automated design techniques are of paramount importance. In this paper, we describe a design automation framework for DG controllable-polarity transistors. First, we present a novel dedicated logic representation form capable to exploit the polarity control during logic synthesis. Then, we tackle challenges at the physical level, presenting a regular layout technique that alleviates the interconnection issue deriving from the second gate routing. We use logic and physical synthesis tools to form a complete design automation flow. Experimental results show that the proposed flow is able to reduce the area and delay of digital circuits, based on 22-nm DG controllable-polarity Silicon nanowire (SiNW) FETs, by 22% and 42%, respectively, as compared to a commercial synthesis tool. With respect to a 22-nm FinFET technology, the proposed flow produces circuits, based on 22-nm DG controllable-polarity SiNWFETs, with 2.9× smaller area-delay product.**

***Index Terms*—CAD, controllable-polarity field-effect transistors (FETs), double-gate (DG) FETs, logic synthesis, nanotechnology, silicon nanowires.**

I. Introduction

As we advance into the era of nanotechnology, semiconductor devices are scaled down to their physical limits. In this nanometer regime, most devices inherently exhibit a superposition of electrons and holes as majority charge carriers. Even though this feature is usually suppressed by technological processing steps, selecting online the device polarity, i.e., the type of majority charge carriers, is of high interest at the design level. *Double-gate* (DG) *field-effect transistors* (FETs) based on silicon [1], carbon [2], and graphene [3] nanowires, are capable to control online the device polarity by configuring the voltage at the second gate.

Manuscript received March 25, 2013; revised December 3, 2013; accepted December 3, 2013. Date of publication October 1, 2014; date of current version November 6, 2014. This research was supported by ERC-2009-AdG-246810. The review of this paper was arranged by Associate Editor T. Ernst.

The authors are with the Integrated Systems Laboratory, Ecole Polytechnique Federale de Lausanne, 1015 Lausanne, Switzerland (e-mail: luca.amaru@epfl.ch; pierre-emmanuel.gaillardon@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/TNANO.2014.2361059

Such emerging transistors with controllable polarity are intrinsically more expressive than standard unipolar transistors. Indeed, the logic functionality of a DG controllable-polarity FET is biconditional on two gates [4], while the operation of a standard unipolar transistor only depends on one gate. This feature makes possible to implement arithmetic logic with fewer physical resources as compared to standard unipolar FETs. Moreover, random control logic has the same cost as with standard unipolar FETs since fixed unipolar behavior can be emulated by biasing the voltage at the second gate. New opportunities arise for enhanced digital designs based on controllable-polarity transistors. However, challenges also appear when attempting to exploit such opportunities in an automated design flow, e.g., manipulation of new Boolean primitives, layout regularity, routing congestion related to the second gate, etc.

In this paper, we present a design automation framework aiming at unlocking the full potential of DG controllable-polarity FETs. At the logic synthesis level, we introduce *biconditional binary decision diagrams* (BBDDs), a novel canonical logic representation form driven by the biconditional expansion. The biconditional logic connective is at the basis of the operation for both BBDDs and DG controllable-polarity FETs. As a consequence, BBDDs are an appropriate logic representation form for the synthesis of circuits based on controllable-polarity transistors. At the physical design level, we present *Sea-of-Tiles* (SoTs), a novel efficient layout technique. The SoT addresses the need for layout regularity at advanced technology nodes, and at the same time, alleviate the interconnection issue deriving from the second gate routing in DG controllable-polarity FETs. To support the automated mapping of transistor netlists onto SoT structures, we present *SATSoT*, a methodology based on Boolean satisfiability (SAT) that endeavors to minimize the wiring complexity for DG controllable-polarity FETs. We use BBDDs and *SATSoT* methodologies together to obtain the first complete (from logic to physical synthesis) and dedicated design automation framework targeting DG controllable-polarity transistors. Experimental results show that digital circuits based on DG controllable-polarity FETs, in silicon nanowires 22-nm technology, designed by the proposed flow are 22% smaller and 42% faster than the same circuit synthesized by a commercial synthesis tool. Compared to FinFET 22-nm technology, the designed circuits have 2.9× smaller area-delay product.

The remainder of this paper is organized as follows. Section II first provides a technology background for controllable-polarity transistors, with an emphasis on silicon nanowires technology. Then, it provides a design automation background related to the proposed synthesis techniques. In Section III, BBDDs are

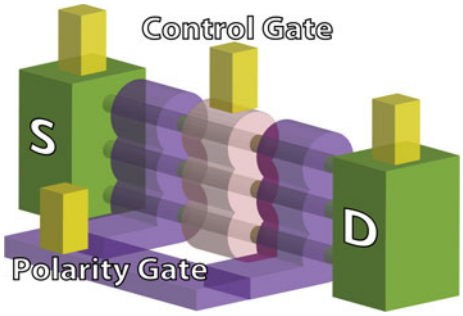

Fig. 1. Conceptual structure of a DG controllable-polarity FET with vertically stacked SiNWs [1].

presented and employed in a novel logic synthesis methodology. Section IV introduces the SoTs regular layout fabric and the associated automated transistor mapping technique. Section V first describes a complete design flow based on BBDD and SoT methodologies. Then, it presents a design case study for datapath circuits and other relevant experimental results. In Section VI, the effectiveness of the proposed design flow is discussed in light of experimental results. This paper is concluded in Section VII.

II. Background

This section first provides background on DG controllable-polarity FETs with an emphasis on *Silicon nanowire* (SiNW) technology. Then, it presents the state-of-the-art for *electronic design automation* (EDA) methodologies employed in the rest of this paper.

A. DG Controllable-Polarity SiNWFETs

The DG controllable-polarity transistors are devices whose majority charge carriers type (electrons or holes) can be configured online by adjusting the voltage at the second gate. First demonstrated by devices based on carbon nanotubes [2], such advantageous feature has been extended to graphene [3] and silicon nanowires [1] technologies. As the natural evolution of FinFET structure, SiNWs are a promising platform for DG controllable-polarity devices thanks to their high $I_{\rm on}/I_{\rm off}$ ratio and CMOS compatible fabrication process [1]. The conceptual structure for a DG controllable-polarity SiNWFET is shown in Fig. 1. It consists of three vertically stacked SiNWs and three gated regions. Note that it is also possible to realize stacks with less/more than three nanowires. The side regions are tied together to the *polarity gate* (PG), while the central region is tied to the *control gate* (CG). The PG tunes the Schottky barriers at the S/D junctions, therefore, selecting the type of majority charge carriers injected in the channel. Then, the CG modulates the amount of carriers flowing into the channel. When the PG is biased with a logic value "1" (V_{cc}) voltage, the majority carriers are electrons (n-type configuration), while when the second gate voltage corresponds to a logic value "0" (V_{ss}) the majority carriers are holes (p-type configuration).

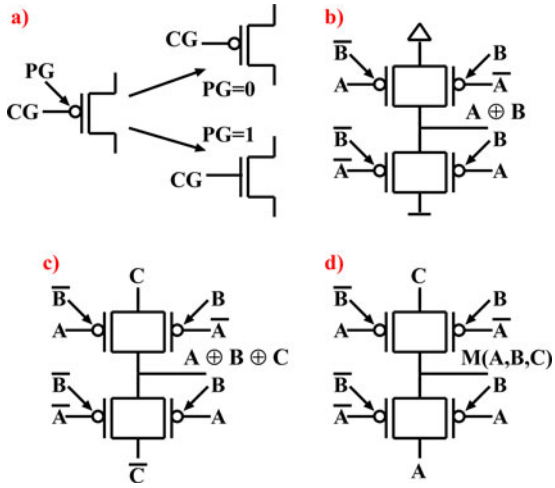

Fig. 2. (a) DG controllable-polarity FETs (re)configuration. (b) Full swing XOR-2 gate in static style, with four devices [7] . (c) Full swing XOR-3 gate in transmission gate style, with four devices [8]. (d) Full swing MAJ-3 gate in transmission gate style, with four devices [4].

The electrical symbol for a DG controllable-polarity FET with its online polarity configuration is depicted by Fig. 2(a). The enhanced functionality deriving from the polarity control, makes possible to implement complex Boolean functions with fewer resources as compared to traditional unipolar devices. Fig. 2(b) depicts a XOR-2 logic gate implemented with only four devices [7], while the static CMOS counterpart employs 2× more devices, input inverters apart [9]. In Fig. 2(c)–(d), a XOR-3 [8] and majority-3 [4] gates are shown employing only four controllable-polarity transistors each, while the static CMOS counterpart employs 3.5× more devices [11]. Since majority and XOR functions form the core of arithmetic operations, datapath circuits are expected to be efficiently realizable with DG controllable-polarity FETs. Unfortunately, current CMOS-oriented EDA methods and tools are not well suited to fully exploit such opportunity.

B. EDA

EDA is a corner stone for modern electronics supporting the exponential growth of integrated circuits complexity. We focus here on the core design flow consisting of logic synthesis and physical design tasks.

Logic synthesis transforms a behavioral description of a Boolean function into a netlist of logic cells, or gates, minimizing some objective metrics such as area, delay, and power. The effectiveness of logic synthesis heavily depends on the data representation structure employed. Driven by the advance of CMOS technology, AND/OR/MUX-oriented representation forms and related synthesis techniques [13]–[18] have been developed and highly optimized. With novel technologies and related Boolean primitives, dedicated logic representation forms and synthesis methods are desirable to fully exploit the expressiveness of a new device. In Section III, we present a novel logic representation form that shares the same core logical connective with DG controllable-polarity FETs.

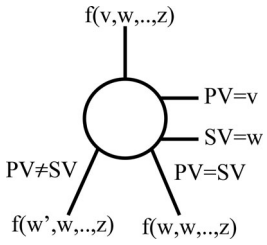

Fig. 3. BBDD internal node.

Physical design aims to place and route logic cells into a chip, minimizing again area, delay, and power. In this context, layout floorplanning is critical to maximize the chip manufacturability. As minimum feature sizes continue to shrink, regularity is one of the key features to increase the yield. Moreover, with novel devices that exhibit multiple gates, regularity is of paramount importance to keep the routing complexity under control. Hence, regular layout fabrics are crucial in the era of nanotechnology. In Section IV, we present a regular layout technique aiming to reduce the routing complexity of DG controllable-polarity FETs, while maximizing the yield.

III. BBDSs

In this section, we present BBDDs [4], a data representation structure sharing the same core logical connective with DG controllable-polarity FETs. The first part of this section is dedicated to introduce the BBDDs basic theory. The second part of this section presents a logic synthesis method based on BBDDs targeting DG controllable-polarity FETs .

A. BBDD Theory

BBDDs [4] are a canonical extension of original *binary decision diagrams* (BDDs) [19]. Fundamentals for BBDDs are given hereafter.

1) BBDD Fundamentals: A BBDD is a *direct acyclic graph* representing a Boolean function $f(v, w, ..., z)$. A BBDD is uniquely identified by its *root*, the set of *internal nodes*, the set of *edges,* and the *1/0-sink nodes*. Informally, each BBDD node has the logic functionality of a multiplexer driven by an exclusive-nor operation. More formally, each internal node (see Fig. 3) in a BBDD is labeled by two Boolean variables: v, the *primary variable* (PV), and w, the *secondary variable* (SV), and has two out-edges labeled $\mathrm{PV} \neq \mathrm{SV}$ and PV=SV. Each internal node represents the biconditional expansion with respect to Boolean variables v and w

$$f(v, w, .., z) = (v \oplus w) \cdot f(w', w, .., z) + (v \overline{\oplus} w) \cdot f(w, w, .., z) \quad (1)$$

where the symbol $\oplus$ represents the XOR operator.

The $\mathrm{PV} \neq \mathrm{SV}$ and $\mathrm{PV} = \mathrm{SV}$ edges connect to $f(w', w, ..., z)$ and $f(w, w, ..., z)$ functions, respectively. Functions of a single variable v cannot be directly decomposed by the biconditional expansion in (1). In such a condition, v is assigned to the PV and a fictitious variable $w = 1$ is introduced and assigned to the SV, collapsing the biconditional expansion into a Shannon's expansion. With this boundary condition, any Boolean function can be fully decomposed and represented with a BBDD.

2) BBDD Ordering: Constraining the order of variable appearance in every root to sink path is one of the requirements to achieve canonicity in BBDDs. For this purpose, the *chain variable order* (CVO) is introduced in [4] imposing a variable order on all root to sink paths for PVs and a rule for the adjacent SVs. Given a Boolean function f and an input variable order $\pi = (\pi_0, \pi_1, ..., \pi_{n-1})$, the CVO assigns PVs and SVs by levels as

$$\begin{cases} \mathrm{PV_i} = \pi_\mathrm{i} \\ \mathrm{SV}_i = \pi_{i+1} \end{cases} \text{with } i = 0, 1, ..., n-2; \begin{cases} \mathrm{PV}_{n-1} = \pi_{n-1} \\ \mathrm{SV}_{n-1} = 1 \end{cases}. \quad (2)$$

CVO Example: From $\pi = (a, b, c)$, the corresponding CVO ordering is obtained by the following method. First, $\mathrm{PV}_0 = a$, $\mathrm{PV}_1 = b$, and $\mathrm{SV}_0 = b$, $\mathrm{SV}_1 = c$ are assigned. Then, the final boundary conditions of (2) are applied as $\mathrm{PV}_2 = c$ and $\mathrm{SV}_2 = 1$. The consecutive ordering by couples (PV_i, SV_i) is thus $((a, b), (b, c), (c, 1))$.

Note that the particular CVO employed influences the size of the BBDD. Reordering techniques [20] are extendable from BDDs to BBDDs to search advantageous input variable orders minimizing the BBDD size.

3) BBDD Reduction and Canonicity: In order to improve the representation efficiency, BBDDs should be reduced according to a set of rules. A BBDD is said reduced when it respects the following rules.

- **R1)** It contains no two nodes, root of isomorphic subgraphs.
- **R2)** It contains no nodes with identical children.
- **R3)** It contains no empty levels.
- **R4)** Subgraphs representing single variable functions degenerates into a single node with SV=1.

BBDDs ordered and reduced are unique for a given input variable order $\pi = (\pi_0, \pi_1, ..., \pi_{n-1})$. Such feature, called canonicity, is advantageous in VLSI design and testing where intrinsic minimization and uniqueness of the data representation structure is of high interest.

Canonicity is preserved if a complement attribute is enabled only at $\mathrm{PV} \neq \mathrm{SV}$ edges. Edges with a complemented attribute, commonly referred to as complemented edges, indicate to invert the function pointed by that edge. We refer the reader to [4] for detailed proofs about BBDD canonicity.

Unless specified otherwise, we refer hereafter to BBDDs as to ordered and reduced BBDDs.

4) BBDD Examples: Fig. 4 depicts the BBDDs for some logic functions of interest in today's designs. In Fig. 4(a), a 6-bit parity function is represented with three nodes and three levels. Note that rule **R3** eliminated half of the levels originally allocated for the BBDD of the 6-bit parity function. This is thanks to the expressiveness of a BBDD node handling two variables per time. Fig. 4(b and (d) shows the BBDDs for mixed AND/OR-XOR intensive functions that frequently appear in datapath circuits. In Fig. 4(c), a three-input majority function is represented with three nodes and three levels. Together with the

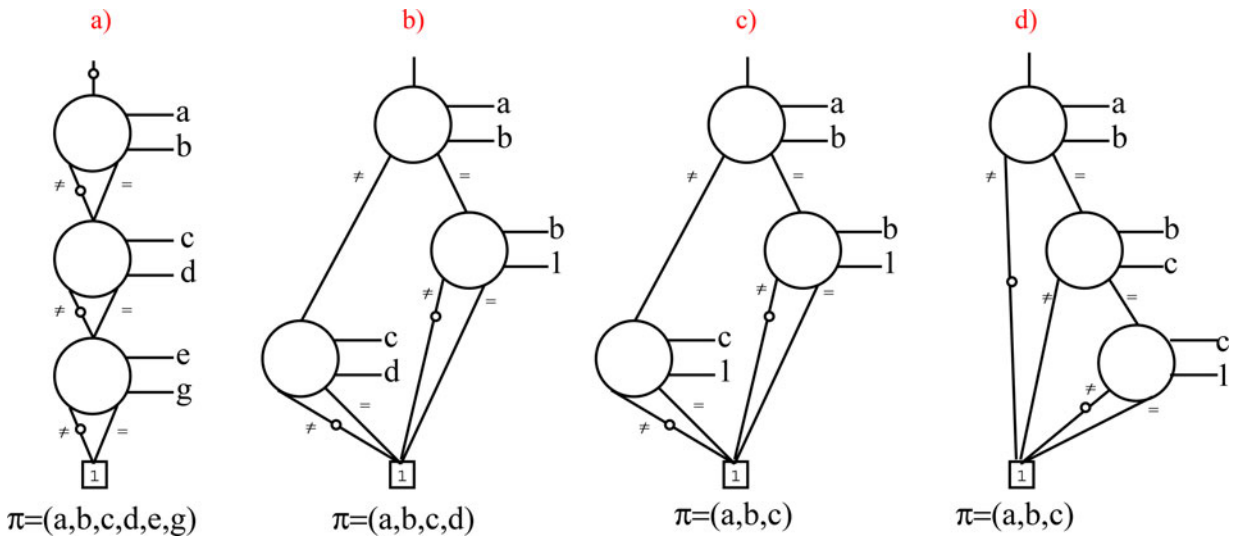

Fig. 4. Examples for BBDD representation of 3 to 6 variables logic functions. (a) Parity, (b, d) mixed AND/OR-XOR, and (c) majority functions are considered.

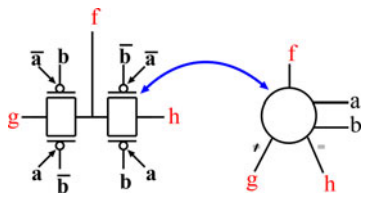

Fig. 5. Direct mapping of a single BBDD internal node onto DG controllable-polarity FETs.

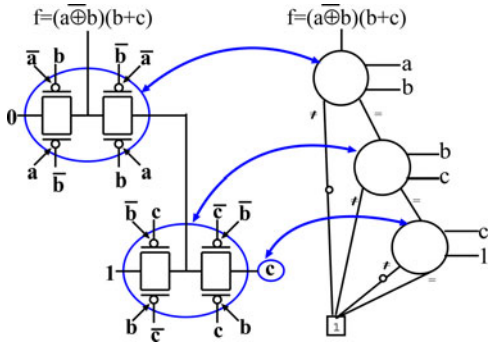

Fig. 6. Example for direct DG controllable-polarity FET mapping onto BBDDs. The target function is the same as in Fig. 4(d).

parity check, the majority function is a basis for binary addition representing the carry propagation.

For the sake of clarity, we comment on how the expression $f = (a\overline{\oplus}b)(b + c)$ in Fig. 4(d) can be obtained by parsing the biconditional expansions in the BBDD. The top node in Fig. 4(d) represents a biconditional expansion with respect to variables a and b. It connects to logic 0 on its different child and to another node, say g, on its equal child. As per (1), we can thus write $f = (a \oplus b) \cdot 0 + (a\overline{\oplus}b) \cdot g = (a\overline{\oplus}b) \cdot g$. With analogous reasoning for g, we get $g = (b \oplus c) \cdot 1 + (b\overline{\oplus}c) \cdot c = (b \oplus c) + bc = b + c$. Recalling that $f = (a\overline{\oplus}b) \cdot g$, we can finally obtain $f = (a\overline{\oplus}b) \cdot (b + c)$.

B. Logic Synthesis Based on BBDDs

Being remarkably compact and canonical, BBDDs find proper use in logic synthesis. We focus here on the application to DG controllable-polarity FET technology.

1) BBDD Nodes and DG Controllable-Polarity FETs: The functionality of a BBDD node and the operation of a DG controllable-polarity FET are strictly related. Indeed, both of them are driven by the biconditional logical connective. This feature opens up the opportunity to directly map BBDDs onto DG controllable-polarity FETs. Fig. 5 depicts an efficient mapping of a BBDD node onto only four DG controllable-polarity FETs. Nodes reduced by rule **R4** do not need any transistor to be mapped but can be simply wired to the input variable assigned to the PV. Complemented edges are mapped by a static inverter gate. Using these rules, it is possible to transpose any BBDD into its DG controllable-polarity FET implementation. Fig. 6 depicts the direct mapping for the target function $f = (a\overline{\oplus}b)(b + c)$. The top and intermediate BBDD nodes are translated into two successive four devices patterns. Each one implements a *biconditional* expansion, as per Fig. 5. The last node is subject to rule **R4**, and therefore, simply wired to its PV variable: c. Finally, edges pointing to the BBDD sink and its complement, are replaced with connections to logic 1 and logic 0, respectively. From a logic perspective, the obtained circuit is a functional clone of the original BBDD.

In order to avoid unbounded stacking of transistors, and therefore, an excessive delay, buffering is required every four consecutive mapped BBDD nodes. This limits the maximum number of stacked devices to 4, similarly as in today's standard cells libraries.

2) Bottom-up BBDD-Based Logic Synthesis: Even though BBDDs compactness is promising, monolithic BBDDs may be inappropriate to represent some large combinational design. As a consequence, the direct mapping of DG controllable-polarity

FETs onto such kind of BBDDs is not efficient. We propose a bottom-up synthesis approach to address this limitation.

Algorithm 1 Bottom-up BBDD-based Logic Synthesis for DG Controllable-Polarity FETs.

INPUT: Flattened HDL into Boolean primitives
OUTPUT: Netlist of DG controllable-polarity FETs
BBDDsize=0;
ProcessedEquations←primary inputs;
BBDDinputs←primary inputs;
BBDDroots=$\emptyset$;
while $\exists$ unsorted Boolean equations **do**
scan for eq. y whose inputs $\in$ ProcessedEquations;
build BBDD for y via APPLY operator;
if size-metric(BBDDsize,BBDDinputs)$\geq$threshold, **then**
BBDDinputs← y;
BBDDroots← y;
end if
if $y \in$ primary outputs **then**
BBDDroots← y;
end if
ProcessedEquations← y;
end while
netlist=$\emptyset$;
for all BBDDinputs α **do**
netlist←input inverter for α;
end for
for all BBDDroots ω **do**
netlist←output buffer for ω;
netlist←recursively map ω and its children (insert a buffer every four consecutive nodes);
end for

As for traditional binary decision diagrams, also BBDDs can be built bottom up using an APPLY operator. The APPLY operator builds $f \otimes g$, with $\otimes$ being any Boolean function of two arguments and f and g two given BBDDs (ordered by the same CVO). Thanks to the representation canonicity, the APPLY operator can be efficiently implemented in software via recursion, having a worst-case complexity of $O(|f| \cdot |g|)$, where $|f|$ and $|g|$ are the number of nodes of the BBDDs of f and g, respectively.

The rationale behind the bottom-up BBDD-based logic synthesis is the following: BBDDs are built depth-first from inputs to outputs of the logic network using an APPLY operator and decomposition points are inserted whenever a BBDD size metric is exceeded. Such size metric describes a polynomial relation between the number of BBDD inputs and the number of BBDD nodes (BBDD size). An adequate degree for the polynomial and a proper threshold value must be heuristically determined. This ensures that none of the individual BBDDs are disproportionately large with respect to the represented function. Consequently, the direct mapping of DG controllable-polarity FETs onto such BBDDs remains efficient. Note, however, that this approach yet remains an heuristic, i.e., with no guarantees on the decomposition optimality.

Algorithm 1 shows the pseudocode for the proposed synthesis methodology. The input of the algorithm is an HDL description flattened onto Boolean primitives, e.g., two operands Boolean operators. The output is a netlist of interconnected DG controllable-polarity FETs. The first *while* loop builds bottom-up BBDDs parsing the input HDL. If during the BBDD construction the logic network size exceeds a threshold, then a decomposition point is inserted. At the end of the *while* loop, a pool of interconnected BBDDs has been built for the target design. The successive phase consists of mapping BBDDs onto DG controllable-polarity FETs. This task is accomplished using the direct mapping strategy previously presented. Additionally, buffers are inserted every four consecutive BBDD nodes mapped to limit the maximum number of stacked devices.

We report a complete example of HDL to DG controllable-polarity FET synthesis. In the following listing, a Verilog description for the function $f = (a \overline{\oplus} b)(b + c)$ (see Fig. 4(d), 6) is shown. This is the input to our synthesis methodology.

```
module HDL (
    a, b, c,
    f  );
  input  a, b, c;
  output f;
  wire n4, n5;
  assign n4 =  ~a ^ b;
  assign n5 = b | c;
  assign f = n4 & n5;
endmodule
```

The synthesis process starts by building the BBDD representation for each Boolean equation, in topological order (bottom-up). Equation n4 is represented with one BBDD node. Equation n5is represented with two BBDD nodes. The final equation f counts three BBDD nodes [see Fig. 4(d)]. Since this objective function is efficiently represented by BBDDs, no size-threshold is exceeded and no decomposition is applied. However, decomposition is likely to happen for harder designs. For example, in the synthesis of a multiplier decomposition points are often inserted, as the BBDD size tends to increase rapidly. After the bottom-up construction, DG controllable-polarity FET are directly mapped onto the BBDD for f. The resulting transistor-level netlist is depicted by the succeeding listing.

```
module DGFETnetlist (
    a, b, c,
    f  );
  input  a, b, c;
  output f;
  wire ab, bb, cb, n1, n2;
  supply0 g;
  supply1 v;
  dgfet u0(.s(g),.cg(a),.pg(g),.
d(ab));//inv a
  dgfet u1(.s(v),.cg(a),.pg(v),.
d(ab));//inv a
  dgfet u2(.s(g),.cg(b),.pg(g),.
d(bb));//inv b
  dgfet u3(.s(v),.cg(b),.pg(v),.
d(bb));//inv b
```

```
  dgfet u4(.s(g),.cg(c),.pg(g),.
d(cb));//inv c
  dgfet u5(.s(v),.cg(c),.pg(v),.
d(cb));//inv c
  dgfet u6(.s(cb),.cg(b),.pg(c),.
d(n1));//node1
  dgfet u7(.s(cb),.cg(bb),.pg(cb),.
d(n1));//node1
  dgfet u8(.s(g),.cg(bb),.pg(c),.
d(n1));//node1
  dgfet u9(.s(g),.cg(b),.pg(cb),.
d(n1));//node1
  dgfet u10(.s(n1),.cg(a),.pg(b),.
d(n2));//node2
  dgfet u11(.s(n1),.cg(ab),.pg(bb),.
d(n2));//node2
  dgfet u12(.s(v),.cg(ab),.pg(b),.
d(n2));//node2
  dgfet u13(.s(v),.cg(a),.pg(bb),.
d(n2));//node2
  dgfet u14(.s(g),.cg(n2),.pg(g),.
d(f));//inv out
  dgfet u15(.s(v),.cg(n2),.pg(v),.
d(f));//inv out
endmodule
```

First, static inverters for *a, b, c* inputs are instantiated, to provide the required complementary signals. Then, BBDD nodes are transposed into the four-device pattern presented in Section III-B-1). Buffering is applied at the output. Note that, as most mapping tools automatically do, the logic implementation polarity is chosen to minimize the inverter stages, and thus, the number of logic levels.

The obtained DG controllable-polarity FET implementation counts 16 devices and 3 levels of logic. Instead, its standard CMOS counterpart features 22 devices and 4 levels of logic.

So far, we presented a logic synthesis methodology capable to natively harness the enhanced functionality of DG controllable-polarity FETs. However, in order to preserve the benefits enabled by logic synthesis, a proper physical design technique is needed to handle technological issues related to DG controllable-polarity FETs.

IV. SoTs

In this section, we sketch a novel regular layout fabric for DG controllable-polarity FETs, called SoTs [5]. The first part of this section introduces the concept of an SoT describing a logic tile and its corresponding layout symbol. Later, an automated method to map netlist of transistors onto the SoT is presented.

A. SoT and Symbolic Layouts

Layout regularity is one of the key features required to increase the yield of integrated circuits at advanced technology nodes. The SoT is a configurable architecture, in which an array of logic tiles are uniformly spread across the chip [5].

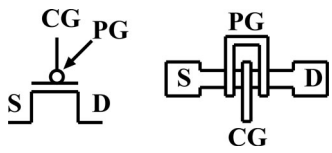

Fig. 7. DG controllable-polarity FET and its symbolic layout.

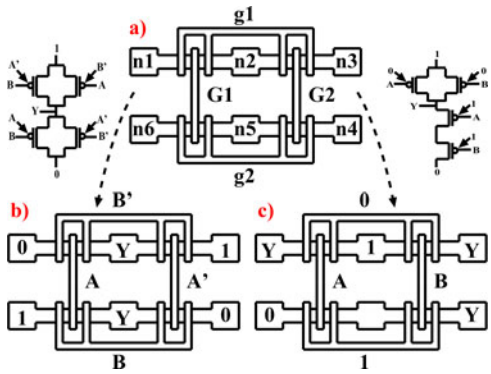

Fig. 8. (a) Tile_{G_2} with DG controllable-polarity FET (a). (b) XOR-2 configuration and (c) NAND-2 configuration [5].

1) Logic Tile: A logic tile is an array of DG controllable-polarity FETs transistors. The symbolic layout for a DG controllable-polarity FET is depicted by Fig. 7.

Two DG controllable-polarity FETs are said grouped if they are adjacent and share their polarity gates. They are said paired if they share the control gates. In a logic tile, DG controllable-polarity FETs are paired and grouped together. Different sizes are possible for a logic tile, e.g., a Tile_{G_n} is an array of n transistor pairs grouped together. In this paper, we focus on the Tile_{G_2}, depicted by Fig. 8(a), since it has the best area efficiency as compared to other tile sizes [5].

Various logic gates can be realized by configuring Tile_{G_2}. A Tile_{G_2} configuration consists of connecting the nodes (n1–n6) and gates (g1, g2, G1, and G2) to appropriate inputs. For example, XOR-2 and a NAND-2 configurations are depicted by Fig. 8(b) and (c), respectively. While the complexity of logic gates realized within a single Tile_{G_2} is limited, any complex logic function can be obtained by considering an array of Tile_{G_2}. Note that also sequential logic elements, e.g., flip–flops, can be implemented by an array of Tile_{G_2} [10].

2) SoTs: In this study, we define an SoT as a regular arrangement of Tile_{G_2} on a chip. The size of an SoT is the number of rows times the number of columns of elementary Tile_{G_2}. By floorplanning the SoT into different islands, each one performing a specific function, it is possible to realize complex digital systems with a very regular layout. For the sake of illustration, Fig. 9 shows a SoT floorplanned, for example, into six top logic units: an arithmetic logic unit, random control logic, read only memory, parity check circuitry, input/output logic, and sequential logic (FF). After high-level floorplanning, the logic functionality desired for each unit has to be physically mapped by connecting tile's nodes and gates. While the floorplanning task can be efficiently carried out with state-of-art techniques,

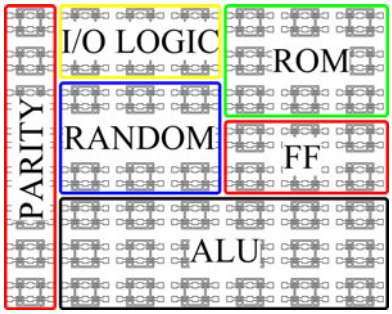

Fig. 9. Example SoT based on Tile_{G_2}. After floorplanning, the SoT consists of different logic units.

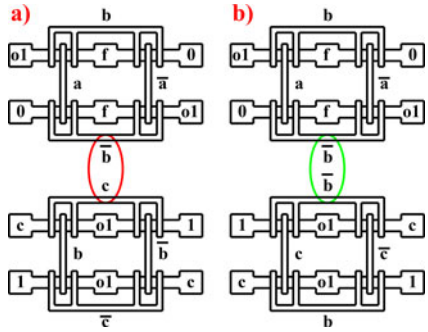

Fig. 10. Mapping the transistor netlist in Fig. 6 onto a 1×2 SoT: (a) mapping with no gate swap allowed, (b) mapping with gate swap allowed.

the physical configuration of each Tile_{G_2} needs more attention due to the routing congestion related to the second gate. For large designs, automated methods are needed to address this task.

Originally, the SoT concept has been introduced for the SiNW technology [5]. However, it can be extended to other controllable-polarity technologies, such as carbon or graphene nanotubes. Indeed, in all technologies, controllable-polarity transistors have four terminals, i.e., two independent gates, a source, and a drain. Therefore, the symbolic layout shown in Fig. 7, and its related layout techniques, remain valid regardless of the technological implementation. In this study, we do not deal with specific physical issues, such as misaligned carbon nanotubes or nanowires diameter fluctuations, to maintain the design flow technology agnostic. Note that design techniques addressing such specific problems can be successively integrated.

B. Transistor Mapping onto SoT

The automated mapping of a previously synthesized transistor netlist onto an SoT is key to keep the wiring complexity under control while implementing the desired functionality. To address this task, we introduce here a transistor mapping technique based on the SAT [6], and its related tool, called *SATSoT*.

The SAT consists of determining whether there exists or not an interpretation of a Boolean formula, in conjunctive normal form, that evaluates to *true*. While SAT is in general a difficult problem, it benefits from a large panel of solvers [21]–[24] that keep on being improved.

The rationale behind *SATSoT* is the following. A first set of Boolean formulae are built to impose a valid transistor placement (respecting the desired logic functionality) of the netlist into the SoT. A second set of Boolean formulae are built to minimize the wiring complexity by enforcing that pins assigned to the same nets are close to each other. Then, an SAT solver tool is employed to search a solution, if any, to all Boolean formulae AND-ed together. If no solution is found, the wiring complexity constraints are relaxed and a new SAT instance is run. *SATSoT* is set to start with a strong set of constraints and progressively relax them until a satisfiable instance is found. We refer the reader to [6] for details about the SAT instance formulation in *SATSoT* .

SATSoT is guaranteed to find a valid transistor mapping only if it is given in input a large enough SoT, i.e., an $I \times J$ grid of Tile_{G_2}. The minimal sizes I and J are determined by the number of DG controllable-polarity FETs to map. *SATSoT* is intended to map netlist with a limited number of devices, therefore, some prior logic decomposition technique must generate a transistor netlist of proper granularity. It is worth noticing that SATSoT achieves obliquely a wiring complexity reduction, by enforcing a maximum sharing between terminals. Standard P&R tools can then be used to get a global layout [29].

V. Design Flow: BBDD-SoT

A logic synthesis and a physical design flow for DG controllable-polarity FETs have been presented in the previous sections. By combining these two approaches, we obtain the first complete design flow targeting controllable-polarity transistors.

A. BBDD-SoT

The BBDD-SoT design flow consists of BBDD-based logic synthesis followed by physical synthesis onto a SoT grid. The linking between the BBDD and SoT is described hereafter.

1) Linking BBDD and SoT: The BBDD-based synthesis procedure in Algorithm 1 receives as input a flattened HDL description and gives in output a netlist of DG controllable-polarity FETs directly derived by BBDD structures. Given the number of devices in the netlist, a proper size for the SoT grid is determined and floorplanning partitions the chip area into the major functional blocks in the design. *SATSoT* is then used to map each set of DG controllable-polarity FETs, deriving from distinct BBDDs, onto the SoT. While in its original conception *SATSoT* is not allowed to modify the transistor netlist interconnection, in the BBDD-SoT flow this is a desirable feature. In particular, when *SATSoT* assigns the four DG controllable-polarity FETs configuration of Fig. 5 to a Tile_{G_2}, the PG and CG signals can be always swapped without affecting the functionality of the circuit. Such operation reveals to be useful for adjacent tiles sharing some nets connections. For the sake of illustration, an example is given in Fig. 10. Starting from the mapped BBDD in Fig. 6 and its corresponding transistor netlist, an 1×2 SoT is instantiated, input inverters apart. In Fig. 10(a), the automatically mapped SoT is depicted, with no gate swap

TABLE I
AUTOMATED DESIGN OF 64-BIT DATAPATH CIRCUITS

Datapaths	DG Controllable -Polarity SiNWFET				FinFET	
	BBDD-SoT		Commercial Tool		Commercial Tool	
	μm^2	ns	μm^2	ns	μm^2	ns
adder	108.19	1.02	122.32	3.70	93.02	7.01
equality	30.92	0.37	44.80	0.12	36.18	0.20
magnitude	60.85	1.02	71.93	1.13	44.17	2.30
barrel	206.38	0.71	281.35	0.30	180.50	0.58
multiplier	1.45k	1.42	1.91k	0.99	1.41k	1.78
average	371.26	0.90	486.08	1.24	352.60	2.37

TABLE II
AUTOMATED DESIGN OF MCNC CIRCUITS

Benchmarks	DG Contr. Pol. SiNWFET				FinFET	
	BBDD-SoT		Commercial Tool		Commercial Tool	
	μm^2	ns	μm^2	ns	μm^2	ns
majority19	69.52	0.31	80.12	1.10	65.25	2.16
misex1	28.55	0.15	29.04	0.41	22.34	0.79
cordic	69.45	0.34	63.64	0.64	50.91	1.11
9symml	25.41	0.22	28.07	0.49	23.39	0.93
f51m	35.81	0.16	62.43	0.71	49.94	1.10
rd73	18.39	0.17	19.11	0.41	15.05	0.63
clip	49.85	0.21	58.08	0.53	47.61	0.88
average	42.42	0.22	48.64	0.61	39.21	0.94

allowed. Adjacent gate signals are assigned to different nets. In Fig. 10(b), gate swap is enabled and internal polarity gates in the SoT are assigned to the same net. In the latter case, the circuit layout is further optimized as the routing complexity is decreased, and therefore, the impact on performance is also reduced. In our simplified design flow, no delay variation is assumed from gate-swap operations. The actual delay is then evaluated postsynthesis.

2) Methodology: The BBDD-SoT is a PERL wrapper for a BBDD manipulation package, written in C language, and *SATSoT* software, written in C++ language. To determine an appropriate size threshold for Algorithm 1, we run several synthesis experiments. We use $\mathrm{BBDD}_{\mathrm{threshold-size}} = 4 \cdot |\mathrm{inputs}|^2$ as it produces the best results on average. In order to evaluate the advantage, in terms of absolute area and delay, of the BBDD-SoT with respect to a traditional design flow, we consider a SiNW implementation for DG controllable-polarity FETs. Additionally, we provide also a comparison to standard unipolar FinFET technology. A 22-nm technology node [30] is used to characterize the devices performance, physical occupation, and routing cost. The traditional flow is a standard-cell-based approach with a commercial synthesis tool fed with a library consisting of primitive XOR-2, NAND-2, NOR-2, and INV gates.

B. Case Study: Automated Design of Datapath Circuits

Datapaths, based on arithmetic circuits, are critical components in today's integrated circuits. Double gate transistors with controllable-polarity enable compact realizations for arithmetic operations thanks to the enhanced device functionality. We study here the automated design of datapath circuits. The bit width considered is 64 bit. Adder, comparators, shifter, and multiplier circuits are written in *Verilog* language. Table I shows the synthesis results for the BBDD-SoT and the considered commercial synthesis tool. BBDDs for adder, equality, and magnitude benchmarks do not require any decomposition, as their size is just linear with respect to the number of inputs. Barrel and multiplier benchmarks instead require decomposition points to keep bounded the implementation cost. Especially in the multiplier case, a monolithic BBDD is too large to be built and clearly unfeasible for direct mapping. The decomposition capability of the BBDD-SoT is indeed crucial in this scenario.

On an average, the BBDD-SoT produces the fastest datapaths with 0.90 ns, being about 37% faster than the counterparts designed by the commercial synthesis tool. Moreover, a BBDD-SoT requires about 30% less area than the commercial synthesis flow results. With respect to unipolar FinFET realizations, datapaths based on DG controllable-polarity FETs are 2.6× faster with a limited area occupation overhead of 5%.

C. Experimental Results

In addition to custom datapaths, we test BBDD-SoT with XOR- and MAJ-intensive benchmarks from the MCNC suite. Table II shows synthesis results for the MCNC circuits designed by the BBDD-SoT and the considered commercial synthesis tool. On average, a BBDD-SoT produces circuits about 3× faster and, at the same time, 12% smaller than the counterparts designed by the commercial synthesis tool. With respect to unipolar FinFET realizations, MCNC circuits based on DG controllable-polarity FETs are 4× faster with a limited area occupation overhead of 8%, mainly deriving from the second gate area of DG SiNWFETs. The considerable logic speed up in the BBDD-SoT is possible thanks to the beneficial BBDD properties. Indeed, most of the circuits in Table II are efficiently represented with a single BBDD, with no decompositions, having a small number of nodes and levels. This is because XOR- and MAJ-intensive circuits are efficiently decomposed by the *biconditional* expansion, and thus, compactly represented by BBDDs. Such compactness is directly transposed into the designed circuit.

VI. DISCUSSIONS

The controllable-polarity feature in double-gate transistors enables an unprecedented design flexibility. The BBDD-SoT design flow demonstrated to further exploit such opportunity as compared to other traditional flows. Employed in the synthesis of datapaths, a BBDD-SoT intrinsically highlights XOR and MAJ operators, which are the basis of arithmetic circuits, and preserves their efficient implementation with DG controllable-polarity FETs. Thanks to the regular layout technique in the BBDD-SoT, the second gate routing has a limited impact on the performance as compared to single gate unipolar technologies. Also for general benchmarks, the efficiency of the BBDD-SoT remains quite marked, as a result of the BBDD representation efficiency and its native correspondence with controllable-polarity transistors. Indeed, a traditional design flow is natively

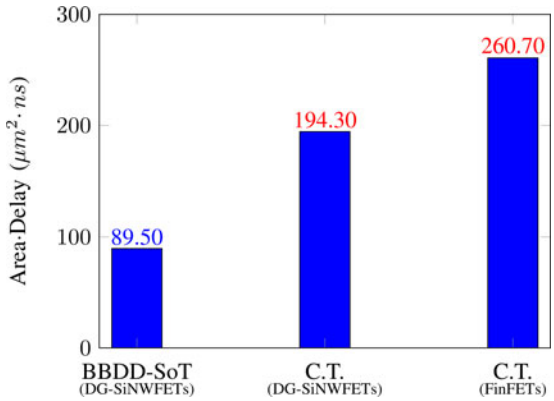

Fig. 11. Area-delay product bar chart. Data for BBDD-SoT, Commercial Tool (C.T.) for DG-SiNWFETs and FinFETs are shown.

intended for CMOS technology and can miss some optimization opportunities deriving from the novel controllable-polarity feature. To test the efficiency of the BBDD-SoT with respect to a traditional design flow we considered a commercial synthesis tool as reference. The comparison metric employed is the area-delay product. Fig. 11 shows the results for the BBDD-SoT, fed with DG controllable-polarity SiNWFETs, and a reference commercial synthesis tool, fed with both DG controllable-polarity SiNWFETs and unipolar FinFETs, all in 22-nm technology.

The BBDD-SoT flow produces circuit realizations with an area-delay product of 89.50 ($\mu\mathrm{m}^2 \cdot \mathrm{ns}$), being 2.17× smaller than the commercial tool. Moreover, unipolar FinFET-based circuits synthesized by the commercial synthesis flow have an area-delay product of 260.70 ($\mu\mathrm{m}^2 \cdot \mathrm{ns}$), which is 2.9× larger than the BBDD-SoT, confirming the notable advantage deriving from controllable polarity.

Future work for the BBDD-SoT focuses on 1) dedicated floorplanning techniques for automated partitioning of the SoT and 2) novel routing algorithms exploiting the properties of BBDD structures mapped onto the SoT.

VII. Conclusion

Advanced nanotechnologies, such as SiNWs, carbon and graphene nanotubes, present the opportunity to control online the polarity of double-gate transistors. To fully harness such new design flexibility, novel logical and physical synthesis techniques are essential. In order to address this need, we presented a BBDD-SoT, a complete design flow comprising innovative synthesis methodologies explicitly efficient for DG controllable-polarity FETs. At the logical level, the BBDD-SoT employs biconditional binary decision diagrams, a logic representation form sharing the same core functionality of controllable-polarity devices. At the physical level, the BBDD-SoT uses a regular layout technique, called SoTs, where arrays of prefabricated devices are uniformly spread across the chip to minimize the routing complexity. Compared to a commercial synthesis flow, a BBDD-SoT is able to reduce the area and delay of digital circuits, based on 22-nm DG controllable-polarity SiNWFETs, by 22% and 42%, respectively. With respect to a 22-nm FinFET technology, the BBDD-SoT produces circuits, based on 22-nm DG controllable-polarity SiNWFETs, with 2.9× smaller area-delay product.

References

[1] M. De Marchi *et al.*, "Polarity control in double-gate, gate-all-around vertically stacked silicon nanowire FETs," in *Proc. IEEE Int. Elect. Devices Meeting*, 2012, pp. 8.4-1–8.4-4.
[2] Y. Lin *et al.*, "High-performance carbon nanotube field-effect transistor with tunable polarities," *IEEE Trans. Nanotechnol.*, vol. 4, no. 5, pp. 481–489, Sep. 2005.
[3] N. Harada *et al.*, "A polarity-controllable graphene inverter," *Appl. Phys. Lett.*, vol. 96, no. 1, pp. 012102-1–012102-3, 2010.
[4] L. Amaru, P.-E. Gaillardon, and G. De Micheli, "Biconditional BDD: A novel canonical representation form targeting the synthesis of XOR-rich circuits," in *Proc. Conf. Des. Autom. Test Eur.* , 2013, pp. 1014–1017.
[5] S. Bobba, M. De Marchi, Y. Leblebici, G. De Micheli *et al.*, "Physical synthesis onto a Sea-of-Tiles with double-gate silicon nanowire transistors, in *Proc. Des. Autom. Conf.*, 2012, pp. 42–47.
[6] C. Gasnier, P.-E. Gaillardon, and G. De Micheli, "SATSoT: A Methodology to map controllable-polarity devices on a regular fabric using SAT," in *Proc IEEE/ACM Int. Symp. Nanoscale Architectures*, New York, NY, USA, Jul. 15–17, 2013, pp. 46–51.
[7] M. H. Ben-Jamaa *et al.*, "An efficient gate library for ambipolar CNTFET logic," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 2, pp. 242–255, Feb. 2011.
[8] A. Zukoski, X. Yang, and K. Mohanram, Universal logic modules based on DG carbon nanotube transistors, in *Proc. Des. Autom. Conf.*, 2011, pp. 884–889.
[9] J. M. Rabaey *et al.*, *Digital Integrated Circuits*. Englewood Cliffs, NJ, USA: Prentice Hall, 2003
[10] P.-E. Gaillardon, S. Bobba, L. Amru, M. De Marchi, D. Sacchetto *et al.*, "Vertically stacked double gate nanowires FETs with controllable polarity: From devices to regular ASICs," in *Proc. Conf. Des. Autom. Test Eur.*, 2013, pp. 625–630.
[11] O. Turkyilmaz, L. Amaru, F. Clermidy, P.-E. Gaillardon, and G. De Micheli, "Self-checking ripple-carry adder with ambipolar silicon nanowire FET," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2013, pp. 2127–2130.
[12] G. De Micheli, *Synthesis and Optimization of Digital Circuits*. New York, NY, USA: McGraw-Hill, 1994.
[13] R. L. Rudell and A. Sangiovanni-Vincentelli, "Multiple-valued minimization for PLA optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. CAD-6, no. 5, pp. 727–750, Sep. 1987.
[14] R. K. Brayton *et al.*, "MIS: A multiple-level logic optimization system," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. CAD-6, no. 6, pp. 1062–1081, Nov. 1987.
[15] E. Sentovich *et al.*, "SIS: A system for sequential circuit synthesis," ERL, Dept. of Electr. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, UCB/ERL M92/41, 1992.
[16] R. K. Brayton and C. Mc Mullen, "The decomposition and factorization of Boolean expressions," presented at the Int. Symp. Circuits and Systems, Rome, Italy, 1982.
[17] C. Yang and M. Ciesielski, "BDS: A BDD-based logic optimization system," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 7, pp. 866–876, Jul. 2002.
[18] R. Brayton and A. Mishchenko, "ABC: An academic industrial-strength verification tool," in *Computer Aided Verification* (Lecture Notes in Computer Science 6174), Berlin, Germany: Springer, pp. 24–40.
[19] R.E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.*, vol. C-35, no. 8, pp. 677–691, Aug. 1986.
[20] I. Wegener, Branching programs and binary decision diagrams: Theory and applications. Philadelphia, PA, USA: SIAM, 2000.
[21] J. P. Marques-Silva and K. A. Sakallah, "GRASP: A search algorithm for propositional satisfiability," *IEEE Trans. Comput.*, vol. 48, no. 5, pp. 506–521, May 1999.
[22] [Online]. Available: http://www.lri.fr/simon/page=glucose
[23] [Online]. Available: http://minisat.se/Main.html
[24] [Online]. Available: http://www.cril.univ-artois.fr/roussel/ppfolio
[25] S. Devadas, "Optimal layout via Boolean satisfiability," in *Proc. Int. Conf. Comput.-Aided Des.*, 1989, pp. 294–297.
[26] N. Ryzhenko and S. Burns, "Standard cell routing via Boolean satisfiability," in *Proc. Des. Autom. Conf.*, 2012, pp. 603–612.

[27] http://www.satcompetition.org/
[28] A. Biere, M. Heule, H. Van Maaren, and T. Walsh *Handbook of Satisfiability*. Amsterdam, The Netherlands: IOS Press, 2009.
[29] O. Zografos, P.-E. Gaillardon, G. De Micheli, "Novel grid-based power routing scheme for regular controllable-polarity FET arrangements," in *Proc. IEEE Int. Symp. Circuits Syst.*, Melbourne, Australia, Jun. 1–5 , 2014, pp. 1416–1419.
[30] PTM models. [Online]. Available: http://ptm.asu.edu/

Luca Amarú (S'13) received the B.S. degree in electronic engineering from Politecnico di Torino, Torino, Italy, in 2009 and the jointed M.S. degree in electronic engineering from Politecnico di Torino and Politecnico di Milano, Milano, Italy, in 2011. He is currently working toward the Ph.D. degree in computer and communication sciences at Integrated Systems Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.

His research interests include electronic design automation, beyond CMOS technologies, information and communication theory, and computer science in general.

Mr. Amarú received the Best Presentation Award at FETCH 2013 conference and a Best Paper Award Nomination at ASP-DAC 2013 conference.

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

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

Dr. Gaillardon received the C-Innov 2011 Best Thesis Award and the Nanoarch 2012 Best Paper Award. He has been serving as TPC member for Nanoarch 2012, 2013 conferences and is a Reviewer for several journals (*Applied Physics Letters*, IEEE TRANSACTIONS ON NANOTECHNOLOGY, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, *ACM Journal on Emerging Technologies in Computing Systems*), conferences (IEEE International Conference on Electronics, Circuits, and Systems 2012, IEEE International Symposium on Circuits and System 2013), and funding agencies (ANR, Chairs of Excellence program of Nanosciences Foundation).

Giovanni De Micheli (F'94) received the Nuclear Eng. degree from Politecnico di Milano, Milano, Italy, in 1979, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California at Berkeley, CA, USA, in 1980 and 1983, respectively.

He is a Professor and the Director of the Institute of Electrical Engineering and of the Integrated Systems Centre at EPF Lausanne, Lausanne, Switzerland. He is a Program Leader of the Nano-Tera.ch program. His 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. He is also interested in heterogeneous platform design including electrical components and biosensors, as well as in data processing of biomedical information. He is an author of Synthesis and Optimization of Digital Circuits (New York, NY, USA: McGraw-Hill, 1994), coauthor and/or coeditor of eight other books and of more than 500 technical articles. His citation h-index is 84 according to Google Scholar.

Prof. Micheli is a Fellow of ACM and a member of the Academia Europaea. He is a member of the Scientific Advisory Board of IMEC and STMicroelectronics. He received the 2012 IEEE/CAS Mac Van Valkenburg award for contributions to theory, practice, and experimentation in design methods and tools and of the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He received also the Golden Jubilee Medal for outstanding contributions to the IEEE Circuits and Systems Society in 2000, the D. Pederson Award for the best paper on the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN (CAD)/INTEGRATED CIRCUITS AND SYSTEMS (ICAS), in 1987, and several Best Paper Awards, including Design Automation Conference (DAC) in 1983 and 1993, Design, Automation, and Test in Europe (DATE) in 2005, and Nanoscale Architectures in 2010 and 2012. He has been serving IEEE in several capacities, namely: Division 1 Director during 2008–2009, the Cofounder and President Elect of the IEEE Council on Electronic Design Automation during 2005–2007, the President of the IEEE CAS Society in 2003, an Editor-in-Chief of the IEEE TRANSACTIONS ON CAD/ICAS during 1987–2001. He has been the Chair of several conferences, including DATE in 2010, Public Health Conferences in 2006, IEEE International Conference on Very Large Scale Integration in 2006, DAC in 2000, and IEEE International Conference on Computer Design in 1989.