Synthesis and Performance Optimization of a Switching Nano-crossbar Computer

Dan Alexandrescu
IROC Technologies
Grenoble, France
dan.alexandrescu@iroctech.com

Mustafa Altun
Dept. of Electronics and Communication Engineering
Istanbul Technical University, Turkey
altunmus@itu.edu.tr

Lorena Anghel
TIMA laboratory
Grenoble-Alpes University, France
lorena.anghel@imag.fr

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
anna.bernasconi@unipi.it

Valentina Ciriani Luca Frontini
Dipartimento di Informatica
Università degli Studi di Milano, Italy
{valentina.ciriani, luca.frontini}@unimi.it

Mehdi Tahoori
Karlsruhe Institute of Technology
Karlsruhe, Germany
tahoori@ira.uka.de

***Abstract*—Beyond CMOS, new technologies are emerging to extend electronic systems with features unavailable to silicon-based devices. Emerging technologies provide new logic and interconnection structures for computation, storage and communication that may require new design paradigms, and therefore trigger the development of a new generation of design automation tools. In the last decade, several emerging technologies have been proposed and the time has come for studying new ad-hoc techniques and tools for logic synthesis, physical design and testing. The main goal of this project is developing a complete synthesis and optimization methodology for switching nano-crossbar arrays that leads to the design and construction of an emerging nanocomputer. New models for diode, FET, and four-terminal switch based nanoarrays are developed. The proposed methodology implements both arithmetic and memory elements, necessitated by achieving a computer, by considering performance parameters such as area, delay, power dissipation, and reliability. With combination of arithmetic and memory elements a synchronous state machine (SSM), representation of a computer, is realized. The proposed methodology targets variety of emerging technologies including nanowire/nanotube crossbar arrays, magnetic switch-based structures, and crossbar memories. The results of this project will be a foundation of nano-crossbar based circuit design techniques and greatly contribute to the construction of emerging computers beyond CMOS. The topic of this project can be considered under the research area of "Emerging Computing Models" or "Computational Nanoelectronics", more specifically the design, modeling, and simulation of new nanoscale switches beyond CMOS.**

I. Introduction

CMOS transistor dimensions have been shrinking for decades in an almost regular manner. Nowadays this trend has reached a critical point and it is widely accepted that the trend will end in a decade [1]. Even Gordon Moore, who made the most influential prediction in 1965 about CMOS size shrinking (Moore Law), accepted that his prediction will lose it validity in near future [15]. At this point, research is shifting to novel forms of nanotechnologies including molecular-scale selfassembled systems [5], [27]. Unlike conventional CMOS that can be patterned in complex ways with lithography, self-assembled nanoscale systems generally consist of regular structures. Logical functions and memory elements are achieved with arrays of crossbar-type switches. This project targets this type of switching crossbars by using models based on diodes, FETs, and four-terminal switches [20], [6], [4], as illustrated in Figure 1. In particular, Figure 2 shows three implementations of a specific Boolean function with these models. Among these models a model based on four-terminal switches deserves a special mention.

A four-terminal switch is specifically developed for crosspoints of nanoarrays; note that each crosspoint has four neigbour crosspoints. The four terminals of the switch are all either mutually connected (ON) or disconnected (OFF). If a controlling literal takes the value of 1, the switch is ON; otherwise, it is OFF. On the other hand, diode and FET are conventional two-terminal switch based devices, i.e., their two terminals are either connected (ON) or disconnected (OFF).

In this paper we describe the Marie Skłodowska-Curie grant agreement No 691178 (European Union's Horizon 2020 research and innovation programme). Section II provides a general overview of the project. The following sections summarize the obtained results and the main objectives identified in the first part of the project. In particular, Section III investigates logic synthesis techniques for diode, FET, and four-terminal switch based nanoarrays by comparing array sizes needed to implement given Boolean functions. Section IV shows a new decomposition method for the four-terminal switch based model. Section V briefly discusses reconfiguration and redundancy approaches needed to face temporal and spatial variations of component electrical properties and hard faults. Section VI discusses the development of defect, variation and fault tolerant techniques in the presence of high defect densities and extreme parametric variations, particularly for crossbar array nano-architectures. Section VII concludes the paper.

II. Overview of the Project

A. *Research Objectives*

The main objective of this project is developing a complete synthesis methodology for nanoscale switching crossbars that leads to the design and construction of an emerging computer. To achieve this objective, we follow a roadmap, illustrated in 3, with sub-objectives listed below.

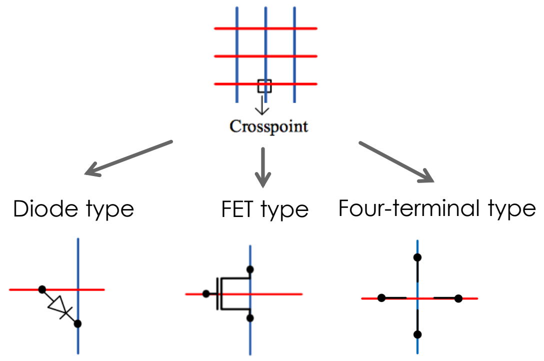

Fig. 1. Different models of switching crossbars.

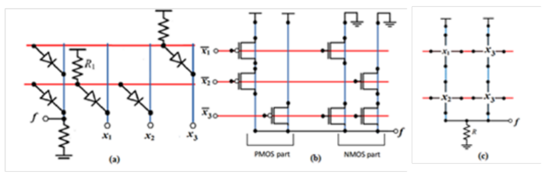

Fig. 2. Representation of a Boolean function with Diode, FET and 4-terminal models.

1) Finding optimal crossbar sizes, modeling, and optimization: Fundamentally, all building parts of a computer, namely arithmetic and memory elements, use Boolean functions for their operations. Therefore, implementing Boolean functions with optimal sizes significantly advances us toward achieving our main goal. Along with sizes, the main performance parameters power consumption, delay, and reliability values of crossbars will be achieved by developing related models. Performance optimization will be performed.
2) Implementing arithmetic and memory elements by considering reliability, area, delay, and power dissipation of the crossbars: We implement arithmetic elements such as adders and multipliers, and memory elements such as flip-flops and registers as building blocks a computer. We also perform optimization for circuit performance parameters using the specifics of applicable technologies.
3) Realizing a nano-crossbar based synchronous state machine (SSM): By integrating arithmetic and logic elements as well as using technology parameters (e.g., area, delay, power and reliability) we realize a SSM as a representation of a computer that uses a complete logic flow and clocked control over state registration.

B. State of the Art and Contributions

Researchers have been interested in models of regular arrays since the seminal paper of Akers in 1971 [3]. In recent years, this interest sees a dramatic spike with the rise of emerging technologies based on regular arrays of switches [29]. Such technologies have apparent advantages over conventional CMOS technologies, such as high density (small area and delay) and easy manufacturability due to self-assembly [27]. Our models target these emerging technologies including nanowire/nanotube crossbar arrays, magnetic switch-based structures, and crossbar memories [12], [14], [18], [19]. In this project, we will comprehensively investigate and experimentally analyze these technologies with a final aim of implementing arithmetic and memory elements with technology parameters. Using diode, FET, and four-terminal switch based models we aim to achieve a synchronous state machine at the end of this project that would be the first in the literature. As follows, we list the future contributions of this project against the state of the art.

1) Models for switching nano-crossbars: Previous models in the literature fundamentally rely on conventional approaches that were specifically developed for basic logic operations [13], [28]. In this study, we generalize them to be applicable for any given Boolean function with offering performance formulations. Along with diode and transistor based nanocrossbar models that have been extensively studies in the literature, a new model based on four-terminal switches that is wide open to new research ideas, will be considered.
2) Performance optimization: Unlike previous crossbar based circuit implementations, our method does not only consider area and reliability but it also deals with other circuit performance parameters, namely delay and power dissipation that results in a complete justification.
3) Circuit implementation with switching nano-crossbars: Previous studies have implemented arithmetic circuit elements performing simple operations [22], [24], [25]. They lack of synthesizing arithmetic and memory elements required to implement a SSM. This is considered in this project.

C. Research Methodology and Approach

The proposed research methodology involves all aspects of computer-aided circuit and system design that constitutes the “computational part” of the project. The methodology also involves electrical and physical characteristics of the applicable emerging technologies that constitutes the “technological part” of the project. This project is interdisciplinary in nature. There will be a continuous information flow between its technological and computational parts. European beneficiary organizations’ expertise is mostly on computational part and collaborations will be made for the technological part. Computational part of this project is multidisciplinary among the fields of logic synthesis and optimization, mathematical graph theory, mathematical probability theory, CAD of emerging circuits and systems. Innovation activities for each objective including limitations:

1) Finding optimal crossbar sizes, modeling, and optimization: We try a systematic approach to find optimal sizes and performance optimization. The approach exploits graph theory and circuit complexity techniques, since it is likely an NP complete problem and might be intractable. In fact, such problems are related to fundamental questions in computer science, such as the separation of the P and NP complexity classes [17]. As a contingency plan we identify the problem as a Boolean satisfiability problem and try heuristic approaches.

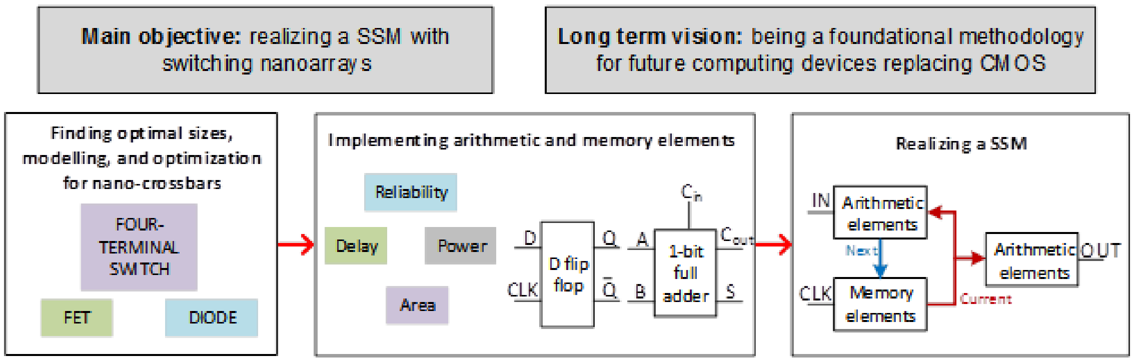
Main objective: realizing a SSM with switching nanoarrays
Long term vision: being a foundational methodology for future computing devices replacing CMOS
Finding optimal sizes, modelling, and optimization for nano-crossbars
FOUR-TERMINAL SWITCH
FET
DIODE
Implementing arithmetic and memory elements
Reliability
Delay
Power
Area
D
Q
CLK
Q̄
D flip flop
A
B
C_{in}
C_{out}
S
1-bit full adder
Realizing a SSM
IN
Arithmetic elements
Next
CLK
Memory elements
Current
Arithmetic elements
OUT

Fig. 3. Project overview.

2) Implementing arithmetic and memory elements: Our methodology considers all circuit performance parameters including reliability, area, delay, and power dissipation. This allows us to compare our results with those of CMOS circuits in a realistic and comprehensive manner. The trade-offs between parameters are first investigated. Then the specifics of applicable technologies for the performance parameters are determined. Finally, a comprehensive optimization software package for the concurrent physical and logical design of applicable technologies is revealed

3) Realizing a synchronous state machine (SSM): We implement a SSM with a programmable multi-array (tile) architecture such that each cross-point in arrays corresponds to a programmable four-terminal switch. We do not use any individual transistors and switches that causes interconnection problems and significantly worsens the density. Another potential problem is signal quality degradation that could upper-bound the number of separate arrays/crossbars in the architecture. Conventionally, this problem is solved by adding simple restoration circuits at the outputs of each circuit block (in our case a crossbar). Unfortunately, this solution would be quite costly for nano-crossbars since they are compact and hard-to-manipulate structures. Therefore, as a contingency plan, we design a single clocked programmable array that synchronously restores the signals.

D. *Originality and Innovative Aspects of the Research*

This project has a degree of novelty since it establishes a novel methodology for synthesis of switching nanoarrays. It is not a modified version of the methods originally developed for CMOS based circuits. The proposed methodology is developed by solely considering the specifics of nano-crossbar arrays with a comprehensive investigation on applicable technologies. This project has a risky and foundational character since it has an ambitious goal of implementing a high performance synchronous machine that could implement the control part of a nanocomputer that aims to be the basis of emerging nanoscale technologies based on crossbar arrays of switches. This project also has synergistic interdisciplinary approach since its computational and technological parts necessarily interfere to each other. Additionally, the project aims to innovate in the specific theoretical areas of circuit complexity, matching algorithms, and circuit performance optimization.

Type	**Array Size Formulas (Optimal)**
Diode	(number of products in ***f***) x (“number of literals in ***f*** ” + 1)
FET	(number of literals in ***f***) x (“number of products in ***f*** ” + “number of products in ***f***^{D}”)

Fig. 4. Array size formulas for diode and FET based implementations.

Type	**Array Size Formula (Non-optimal)**
Four-terminal	(number of products in ***f***) x (number of products in ***f***^{D})

Fig. 5. Array size formula for four-terminal switch based implementation.

III. Logic Synthesis Techniques for Diode, FET, and Four-Terminal Switch Based Nanoarrays

In this section, we survey logic synthesis techniques for diode, FET, and four-terminal switch based nanoarrays [21], [26]. We present experimental results on standard benchmark circuits to compare array sizes needed to implement given Boolean functions. For diode and FET based nanoarrays, Boolean functions are implemented by using conventional techniques that are diode-resistor logic and CMOS logic [21]. This has an important constraint regarding nanoarray structures. Boolean functions should be implemented in their sum-of-products (SOP) forms; other forms such as factored or BDD (Binary Decision Diagram) cannot be used since these forms require manipulation/wiring of switches that is not applicable for self-assembled nanoarrays.

Array sizes for diode and FET based nanoarrays: Given a target Boolean function f, we derive formulas of the array sizes. This is shown in Figure 4. For diode based implementations, each product of f requires a row (horizontal line), and each literal of f requires a column (vertical line) in an array. Additionally, one extra column is needed to obtain the output. For FET based implementations, each product of f and its dual, f^D, requires a column, and each literal of f requires a row in an array. As an example, consider a target function $f = x_1\overline{x_2} + \overline{x_1}x_2$ having 4 literals and 2 products; $f^D = x_1x_2 + \overline{x_1}\ \overline{x_2}$ has 2 products. This results in array sizes of 2×5 and 4×4 for diode and FET based implementations, respectively. Note that both formulas, for diode and FET, always result in optimal array sizes; no further reduction is possible.

Four-terminal switch based implementation considers each crosspoint of an array as a four-terminal switch [4]. Four terminals of the switch are all either mutually connected (ON-logic 1 applied) or disconnected (OFF-logic 0 applied). Boolean functions are implemented with top-to-bottom paths in an array by taking the sum (OR) of the product (AND) of literals along each path. This makes Boolean functions implemented in their sum-of-products (SOP) forms.

Array size for four-terminal switch based nanoarrays: Given a target Boolean function f, the array size formula can be derived by considering that each product of f and its dual, f^D, require an array column and an array row, respectively. This is shown in Figure 5. As an example, consider a target function $f = x_1\overline{x_2} + \overline{x_1}x_2$ and $f^D = x_1x_2 + \overline{x_1}\ \overline{x_2}$ both having 2 products. This results in an array size of 2×2.

Examining the array size formulas in Figure 4 and Figure 5, we see that while the formulas in Figure 4 always result in optimal sizes, the sizes derived from the formula in Figure 5, that is, for four-terminal switch based arrays, are not necessarily optimal. In the following part we present an algorithm that finds an optimal size implementation of any given target Boolean function. Finding whether a certain array with assigned literals to its switches implements a target function is the main problem in finding optimal sizes. This problem requires to check if each assignment of 0's and 1's to the switches, corresponding to a row of the target function's truth table, results in logic 1 (a top-to-bottom path of 1's exists). To check this we have to enumerate all top-to-bottom paths; the size of this task grows exponentially with the array size. This is a general statement that holds also for our algorithm described below.

Our simple brute force algorithm finds optimal array sizes to implement given target Boolean functions with arrays of four-terminal switches in four steps:

1) Obtain irredundant sum-of-products (ISOP) expressions of a given-target function f_T and its dual ${f_T}^D$. Determine the upper bound on the array size using the formula in Figure 5. The implementable lower bound (LB) values are taken from the lower bound table proposed in [4].
2) List the array shapes (R×C) and sort them regarding array sizes in ascending order. While ordering, first take the array shape which has lower number of rows. Suppose that there are total of N different shapes in the list. For Step 3, start with n=1 (1≤n≤N).
3) Consider the following statement for the n th shape. **Statement**: An array which has the shape in the n th line of the list is implementable for f_T.
   If the statement is TRUE, then change UB to the R×C (save the design); go to Step 4.
   If the statement is FALSE, then increase the number n by 1 (n=n+1).
4) Repeat Step 3 until finding UB that is the optimal size.

Simulation results: In Figure 6, we report synthesis results for standard benchmark circuits. We treat each output of a benchmark circuit as a separate target function. The number of products for each target function f_T and its dual ${f_T}^D$ are obtained through sum-of-products minimization using the program Espresso. The array size values for "Diode", "CMOS", and "4-terminal" are calculated by using the formulas in Figure 4 and Figure 5. The array size values for "Optimal 4-terminal" are obtained using the presented optimization algorithm. Examining the numbers in Figure 6, we always see the same sequence from the worst to the best result as "CMOS", "Diode", "4-terminal", and "Optimal 4-terminal". This proves that models based on four-terminal switches overwhelm those based on two-terminal switches regarding the array size. Further, the numbers obtained by our optimal synthesis method compares very favorably to the numbers obtained by previous methods.

IV. Lattices and decomposition methods

In this section we will briefly present some preliminary studies carried out for this project, that show how the cost of implementing a four-terminal switching lattice could be mitigated by exploiting Boolean function decomposition techniques. The basic idea of this approach is to first decompose a function into some subfunctions, according to a given functional decomposition scheme, and then to implement the decomposed blocks with separate lattices, or physically separated regions in a single lattice. Since the decomposed blocks usually correspond to functions depending on fewer variables and/or with a smaller on-set, their synthesis should be more feasible and should produce lattice implementations of smaller size. In the framework of switching lattices synthesis, where the available minimization tools are not yet as developed and mature as those available for CMOS technology, reducing the synthesis of a target Boolean function to the synthesis of smaller functions could represent a very beneficial approach [7], [11]. As a preliminary work for this project, we have focused on the particular decomposition method that gives rise to the bounded-level logic networks called *P-circuits* [8], [10], [11]. P-circuits are extended forms of Shannon cofactoring, where the expansion is with respect to an orthogonal basis $\overline{x}_i \oplus p$ (i.e., $x_i = p$), and $x_i \oplus p$ (i.e., $x_i \neq p$), where p is a function defined over all variables except for a critical variable x_i (e.g., the variable with more switching activity or with higher delay that should be projected away from the rest of the circuit). They can be defined as follows:

$$\text{P-circuit}(f) = (\overline{x}_i \oplus p)\, f^{=} + (x_i \oplus p)\, f^{\neq} + f^I$$

where I is the intersection of the projections of f onto the two sets $x_i = p$ and $x_i \neq p$, and

1) $(f|_{x_i=p} \setminus I) \ \subseteq\ f^{=} \ \subseteq\ f|_{x_i=p}$
2) $(f|_{x_i\neq p} \setminus I) \ \subseteq\ f^{\neq} \ \subseteq\ f|_{x_i\neq p}$
3) $\emptyset \ \subseteq\ f^I \ \subseteq\ I.$

This definition can be easily generalized to incompletely specified Boolean functions. Thus, the synthesis idea of P-circuits is to construct a network for f by appropriately choosing the sets $f^{=}$, $f^{\neq}$, and f^I as building blocks.

The same idea can be exploited in the switching lattice framework: the subfunctions $f^{=}$, $f^{\neq}$, and f^I depend on $n-1$ variables instead of n, they have a smaller on-set than f, and their lattice synthesis should produce lattices of reduced area.

Benchmark	CMOS	Diode	4-Terminal	Optimal 4-Terminal
Alu 0	30	18	6	**6**
Alu 1	30	18	6	**6**
Alu 2	30	18	6	**6**
Alu 3	30	18	6	**6**
B12 0	80	32	24	**12**
B12 1	120	70	35	**16**
B12 3	30	20	8	**8**
B12 4	42	28	8	**8**
B12 6	132	77	35	**18**
B12 7	110	66	24	**18**
B12 8	90	70	14	**14**
C17 0	36	18	9	**6**
C17 1	30	20	8	**8**
Clpl 0	64	32	16	**12**
Clpl 1	36	18	9	**9**
Clpl 2	16	8	4	**4**
Clpl 3	144	72	36	**18**
Clpl 4	100	50	25	**15**
Dc1 1	25	10	6	**6**

Benchmark	CMOS	Diode	4-Terminal	Optimal 4-Terminal
Dc1 2	72	36	16	**12**
Dc1 5	35	15	12	**6**
Dc1 6	36	18	9	**6**
Ex5 31	156	104	32	**24**
Ex5 33	110	77	21	**21**
Ex5 46	81	54	18	**18**
Ex5 49	72	54	12	**12**
Ex5 50	81	63	14	**14**
Ex5 61	64	48	12	**12**
Ex5 62	49	35	10	**10**
Misex1 1	48	16	8	**8**
Misex1 2	132	55	35	**15**
Misex1 3	156	60	40	**24**
Misex1 4	121	44	28	**16**
Misex1 5	90	45	25	**15**
Misex1 6	143	66	42	**18**
Misex1 7	81	36	20	**15**
Mp2d 4	345	75	90	**24**
Newtag	108	72	32	**18**

Fig. 6. Array sizes of three different nano-crossbar based logic families [21].

Therefore, the overall lattice for f derived composing minimal lattices for $f^{=}$, $f^{\neq}$, and f^{I}, could be smaller than the one derived for f without exploiting its P-circuits decomposition. This expectation has been confirmed by a set of experimental results, where the utility of the decomposition-based approach has been evaluated applying the two synthesis methods presented in [4] and in [16]. These results demonstrate that lattice synthesis benefits from this type of Boolean decomposition, yielding smaller circuits with an affordable computation time (even less in some cases). Indeed, in 30% of the analyzed cases the synthesis of switching lattices based on the P-circuit decomposition of the logic function allows to obtain a more compact area in the final resulting lattice, with an average gain of at least 20%.

Future work for the project includes assessing the impact of more complex types of decompositions, both within the class of P-circuits (with more expressive projection functions p) and beyond. In particular, we are currently studying a decomposition scheme that can be applied to the lattice synthesis of a special class of regular Boolean functions called *D-reducible* functions. D-reducible functions [9] are functions whose points are completely contained in an affine space strictly smaller than the whole Boolean cube $\{0,1\}^n$. A D-reducible function f can be written as $f = \chi_A \cdot f_A$, where A is its unique associated affine space, χ_A is the characteristic function of A, and f_A is the projection of f onto A. Notice that f and f_A have the same number of points, but these are now compacted in a smaller space.

The D-reducibility of a function f can be exploited in the lattice synthesis process: the idea is to independently find lattice implementations for the projection f_A and for the characteristic function χ_A of A, and then to compose them in order to construct the lattice for f. To further reduce the overall lattice area, we could exploit the peculiar structure of the function χ_A, that represents the minterms of an affine subspace of $\{0,1\}^n$. To this aim, we are currently studying and testing a method for implementing minimal lattice representations of affine spaces.

V. Defect and fault-tolerant techniques

As it has been already proved, future complex devices based on nanowire crossbars provide better density over conventional CMOS devices due to the new methods for growing and assembling. They are also a very good candidate for future high density interconnects, combinational circuits and storage parts. Some of the potential solutions are of a regular types [23], others based on memristive networks are on an irregular structures [2]. These technologies are highly sensitive to design variations, defect and intermittent faults, or susceptible to environmental factors, such as thermal stress, radiation, and so on. It may result in crossbars structures with high number of defect rates related to manufacturing or environmental constraints, much more than what are considered today in conventional CMOS technologies (15% of faulty components). These threats can be seen as major obstacles to adopting and using such architectures and technologies to build future application specific circuits or even processors.

State of the art conventional defect and fault-tolerant techniques are not suitable for these designs because they are targeting rather small defect rates. In this project we plan to investigate the reconfiguration and redundancy approaches needed to face temporal and spatial variations of component electrical properties and also hard faults. Potential solutions can be the combination of temporal and structural redundancy, built-in repair circuitry, and system-level adaptation techniques and adapt the techniques that have to be used to masks on the fly faulty components and signals caused by defects and other error sources.

Another part of of our studies during the duration of the project will target the variation and hard failures impact on nanowire networks, where we'll pick random m wires and disconnect them from the network and assess the parameters to be optimized in order to guarantee fault-free functionality.

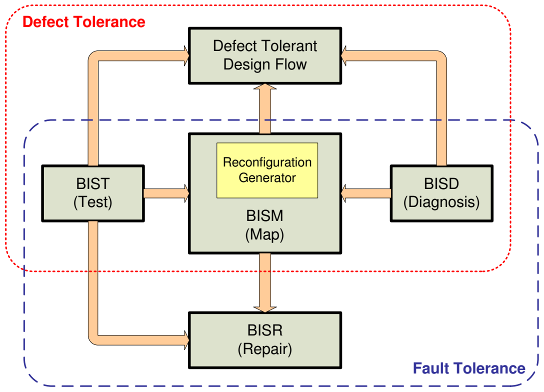

Fig. 7. Variation, defect and fault tolerance in crossbar nano-architectures.

VI. Built-in Variation, Defect, and Fault Tolerance

The key aim of this project is a set of fully integrated research activities to provide defect, variation and fault tolerance in the presence of high defect densities and extreme parametric variations in nano-crossbar architectures. Self-assembly processes promise to considerably lower manufacturing costs, but at the expense of reduced control of the exact placement of these devices. Without fine-grained control, these devices will certainly exhibit higher defect rates. Moreover, nanofabrication process yields nanowires which are a few atoms long in the diameter. For instance, the contact area between nanowires contains only a few tens of atoms. With such small cross-section and contact areas, fragility of these devices is orders of magnitude more than devices currently being fabricated using conventional lithography techniques. Therefore, extreme parametric variations and defects are assumed to be inevitable and they are believed to be two major issues for nanowire based structures. While it is impossible to eliminate all defects during nanofabrication, the design paradigms should be changed such that they can produce reliable systems in the presence of hardware defects.

One of the main focuses of this project is development of defect, variation and fault tolerant techniques in the presence of high defect densities and extreme parametric variations, particularly for crossbar array nano-architectures. To tolerate high defect rates and variations, our revolutionary approach is to integrate *defect tolerance* to improve the manufacturing yield (for fabrication defects), *fault tolerance* to ensure the lifetime reliability (for errors during normal operation), and *variation tolerance* to ensure the predictability and performance (for parametric variations), in the design methodologies for future nanotechnologies. *Adaptive* and *built-in* defect, variation and fault tolerant design flows, fundamentally different from conventional approaches, are proposed in which the objective is to ensure high manufacturing yield and runtime reliability of the circuit at extremely low costs. We plan to exploit the opportunities created by this nanotechnology such as reprogrammability and abundance of programmable resources to provide defect, variation and fault tolerance. The overview of this new methodology for crossbar nano-architectures is shown in Fig. 7, which will be detailed in the following subsections.

A. Built-in Self-testing (BIST) and Self-diagnosis (BISD)

Thorough testing and precise high-resolution location of failing resources in a defective part are keys to successful implementations of defect and fault tolerance. Thorough manufacturing testing is required to identify a defective manufactured part. During system operation, periodic testing is required to identify a defective system component with a permanent fault. Application-dependent test and diagnosis techniques are useful for defect tolerance and also for detection, location, and repair of permanent faults during normal operation of a fault-tolerant reconfigurable system. Application-independent test and diagnosis techniques are used after manufacturing, mainly for identifying defective parts and also for defect tolerance. Test and diagnosis during system operation are very complex tasks; however, they help detect permanent and transient faults and hence improve the overall system reliability. *Built-in-self-testing* (BIST) and *Built-in-self-diagnostics* (BISD) are key components for effective self-repair with minimized dependence on the external test equipment.

B. Built-in Self-mapping (BISM)

Since it is expected that all manufactured nano-chips contain a considerable percentage of defects even in a mature fabrication process, defect tolerance is inevitable. The goal of defect tolerance is to bypass defective resources using test and diagnosis information. Since defects are device specific, this part of the design flow, mapping the application and bypassing defective resources, has to be device specific as well. However, the information required for such mapping, which is obtained only after test and diagnosis, is not available at the design time. Therefore, some parts of the application mapping phase have to be postponed from *design time* to the *test time*. Nevertheless, we propose a novel design flow to minimize such per-chip customized design efforts in Sec. VI-C.

As parts of the design flow are shifted to the test time, we proposed a built-in self-mapping (BISM) approach to minimize per-chip customized mapping efforts. BISM allows the crossbar array to be configured by the on-chip interface circuitry and bypass defective resources. It also reduces physical design efforts in which detailed placement and routing will be performed on-the-fly. In other words, only global placement and routing has to be completed at the design time and detailed configuration of individual crossbars (for logic mapping or signal routing) will be determined at the configuration time by BISM.

C. Application-Independent Defect Tolerant Flow

In the conventional defect tolerant flow, the existence and the location of defective elements are identified using test and diagnosis steps and stored in *defect map*. Defect tolerance is achieved by avoiding defective resources in the physical design flow using the defect map. Particularly, placement and routing phases of the physical design use the defect map in order to map the design to the crossbar array by using only defect-free resources. However, due to prohibitively large size of defect map and per chip customization of entire design flow, this traditional approach cannot be used for high-volume production of nano-chips.

Most drawbacks of the traditional flow are due to the fact that this method is *application dependent*, i.e. defects are handled in a per-application basis. In contrast to the defect-aware design flow, we propose a *defect-unaware* design flow to tolerate defects in crossbar arrays. In this flow, defect tolerance is performed once and the same recovered (defect-free) set of resources are used for all applications. In the proposed flow, almost all design steps remain unaware of the existence and the location of defects within the nano-chip. The key idea in the proposed defect-unaware flow is to identify *universal* defect-free subsets of resources within the original partially-defective nano-chip. All design steps work with this universal defect-free subset of the chip called the *design view*. The size of these "universal" subsets is identical for all nano-chips fabricated in the same process environment (similar defect densities). Also, these universal defect-free subsets remain unchanged for different applications mapped into the same nano-chip, making this approach *application-independent*. There is a *final mapping* phase, with very low complexity, at the end of physical design flow that makes the connection between the defect-free design view and the actual *physical view* of the nano-chip which contains actual defects. This is the only defect-aware step which has to be performed per chip. This final mapping phase will be implemented as a part of the proposed BISM approach.

For the case of crossbar nano-architectures, the idea of universal defect-free subsets and the corresponding defect-unaware flow can be explained as follows. The goal of this approach is to identify defect-free $k \times k$ crossbars within the original partially-defective $n \times n$ crossbars. These defect-free subsets of the crossbars are complete, which means that every k input nanowire in the defect-free subset is connectable to every k output nanowire through a defect-free crosspoint. In general, the defect-free subset is smaller than the original crossbar ($k < n$). The size of this defect-free subset (k) has to be chosen such that for the fabricated $n \times n$ crossbars, the probability of finding such defect-free $k \times k$ crossbars, i.e., *manufacturing yield* is high enough. In order words, the concept of manufacturing yield is redefined to this probability: if a fabricated $n \times n$ crossbar contains a defect-free $k \times k$ crossbar, it is *usable*, otherwise it is considered as *unusable*. During the physical design, the original design will be mapped (placed and routed) into an array of $k \times k$ crossbars. A final defect-aware mapping step is required to re-map the used resources within $k \times k$ crossbars into the actual defect-free resources within partially-defective $n \times n$ crossbars using the defect map information. An important advantage of this approach is the reduction in the size of the defect map from $O(n^2)$ to $O(n)$, since instead of storing the actual defect information of individual crosspoints, only the information regarding the location of defect-free $k \times k$ crossbar within the $n \times n$ crossbar needs to be stored.

VII. Conclusion

Integrating a new technology into a mature industry such as the semiconductor industry is a long road in which device performances and manufacturability have to be developed jointly through a blend of advanced research, technology development and industry-compliant implementation. One of the major promises of emerging nanotechnologies for on-chip applications is ultimate integration density, manufacturing and integration cost reduction, and the reduction of power consumption. However, there is a big gap in 1) extending the existing electronic design automation (EDA) flow for emerging technologies in order to introduce them in the architecture and system design in a systematic-way, and 2) novel computer architecture systems based on emerging technologies to provide high performance and minimize power consumption at the same time. This project includes the introduction of hybrid EDA flow as well as emerging computer architectures by gathering well respected experts working in these broad fields.

VIII. Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 691178.

References

[1] "Overall Technology Roadmap Characteristics," International Technology Roadmap for Semiconductors, Tech. Rep., 2010, retrieved 2013.

[2] H. O. S. C. M.-O. M. A. A. Z. Stieg, A. V. Avizienis and J. K. Gimzewski, "Emergent criticality in complex Turing B-type atomic switch networks," *Advanced Materials*, vol. 24, no. 2, pp. 286 – 293, 2012.

[3] S. B. Akers, "A Rectangular Logic Array," in *Switching and Automata Theory, 1971., 12th Annual Symposium on*, 1971, pp. 79–90.

[4] M. Altun and M. D. Riedel, "Logic Synthesis for Switching Lattices," *IEEE Transactions on Computers*, vol. 61, no. 11, pp. 1588–1600, 2012.

[5] K. Ariga, M. V. Lee, T. Mori, X.-Y. Yu, and J. P. Hill, "Two-dimensional nanoarchitectonics based on self-assembly," *Advances in Colloid and Interface Science*, vol. 154, no. 1-2, pp. 20 – 29, 2010.

[6] P. Avouris, "Molecular Electronics with Carbon Nanotubes," *Acc. Chem. Res.*, vol. 35, no. 12, pp. 1026–1034, 2002.

[7] A. Bernasconi, V. Ciriani, R. Drechsler, and T. Villa, "Logic Minimization and Testability of 2-SPP Networks," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 27, no. 7, pp. 1190–1202, 2008.

[8] A. Bernasconi, V. Ciriani, G. Trucco, and T. Villa, "On Decomposing Boolean Functions via Extended Cofactoring," in *Design Automation and Test in Europe (DATE)*, 2009.

[9] A. Bernasconi and V. Ciriani, "Dimension-reducible boolean functions based on affine spaces," *ACM Trans. Design Autom. Electr. Syst.*, vol. 16, no. 2, p. 13, 2011.

[10] A. Bernasconi, V. Ciriani, V. Liberali, G. Trucco, and T. Villa, "Synthesis of P-Circuits for Logic Restructuring," *Integration*, vol. 45, no. 3, pp. 282–293, 2012.

[11] A. Bernasconi, V. Ciriani, G. Trucco, and T. Villa, "Using flexibility in p-circuits by boolean relations," *IEEE Trans. Computers*, vol. 64, no. 12, pp. 3605–3618, 2015.

[12] Y. C. Chen, S. Eachempati, C. Y. Wang, S. Datta, Y. Xie, and V. Narayanan, "Automated Mapping for Reconfigurable Single-electron Transistor Arrays," in *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, 2011, pp. 878–883.

[13] Z. Chen, J. Appenzeller, Y.-M. Lin, J. Sippel-Oakley, A. G. Rinzler, J. Tang, S. J. Wind, P. M. Solomon, and P. Avouris, "An Integrated Logic Circuit Assembled on a Single Carbon Nanotube," *Science*, vol. 311, no. 5768, pp. 1735–1735, 2006.

[14] A. Dehon, "Nanowire-based Programmable Architectures," *J. Emerg. Technol. Comput. Syst.*, vol. 1, no. 2, pp. 109–162, 2005.

[15] M. Dubash, "Mooreâs Law is Dead, Says Gordon Moore," *Techworld. com*, no. 13, 2005.

[16] G. Gange, H. Søndergaard, and P. J. Stuckey, "Synthesizing Optimal Switching Lattices," *ACM Trans. Design Autom. Electr. Syst.*, vol. 20, no. 1, pp. 6:1–6:14, 2014.

[17] S. Jukna, *Boolean Function Complexity: Advances and Frontiers*, Springer, Ed., 2012.

[18] A. Khitun, M. Bao, and K. L. Wang, “Spin Wave Magnetic NanoFabric: A New Approach to Spin-Based Logic Circuitry,” *IEEE Transactions on Magnetics*, vol. 44, no. 9, pp. 2141–2152, 2008.

[19] Y. Levy, J. Bruck, Y. Cassuto, E. G. Friedman, A. Kolodny, E. Yaakobi, and S. Kvatinsky, “Logic Operations in Memory Using a Memristive Akers Aarray,” *Microelectronics Journal*, vol. 45, no. 11, pp. 1429 – 1437, 2014.

[20] W. Lu and C. Lieber, “Nanoelectronics from the Bottom Up,” *Nat Mater*, vol. 6, no. 11, pp. 841–850, 2007.

[21] M. C. Morgul and M. Altun, “Synthesis and optimization of switching nanoarrays,” in *Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 161–164.

[22] A. H. Shaltoot and A. H. Madian, “Memristor Based Carry Lookahead Adder Architectures,” in *Circuits and Systems (MWSCAS), 2012 IEEE 55th International Midwest Symposium on*, 2012, pp. 298–301.

[23] G. Snider, “Computing with hysteretic resistor crossbars,” *Appl. Phys. A*, vol. 80, pp. 1165 – 1172, 2005.

[24] G. Snider, P. Kuekes, and R. S. Williams, “CMOS-like Logic in Defective, Nanoscale Crossbars,” *Nanotechnology*, vol. 15, no. 8, p. 881, 2004.

[25] G. S. Snider, P. J. Kuekes, and D. R. Stewart, “Nanoscale Latch-array Processing Engines,” Patent US 7,227,379, 06 5, 2007.

[26] O. Tunali and M. Altun, “Defect tolerance in diode, fet, and four-terminal switch based nano-crossbar arrays,” in *Nanoscale Architectures (NANOARCH), 2015 IEEE/ACM International Symposium on*. IEEE, 2015, pp. 82–87.

[27] G. M. Whitesides and B. Grzybowski, “Self-Assembly at All Scales,” *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.

[28] H. Yan, H. S. Choe, S. Nam, Y. Hu, S. Das, J. F. Klemic, J. C. Ellenbogen, and C. M. Lieber, “Programmable Nanowire Circuits for Nanoprocessors,” *Nature*, vol. 470, no. 7333, pp. 240–244, 2011.

[29] A. Y. Zomaya, *Handbook of Nature-Inspired and Innovative Computing*, Springer, Ed., 2006.