Synthesis and Performance Optimization of a Switching Nano-crossbar Computer

Marie Skłodowska-Curie grant agreement No 691178
(European Union's Horizon 2020 research and innovation programme)

Dan Alexandrescu[1], Mustafa Altun[2], Lorena Anghel[3], Anna Bernasconi[4], Valentina Ciriani[5], *Luca Frontini*[5], Mehdi Tahoori[6]

[1]IROC Technologies, Grenoble, France, dan.alexandrescu@iroctech.com

[2]Dept. of Electronics and Communication Engineering, Istanbul Technical University, Turkey, altunmus@itu.edu.tr

[3]TIMA laboratory Grenoble-Alpes University, France, lorena.anghel@imag.fr

[4]Dipartimento di Informatica, Università di Pisa, Italy, anna.bernasconi@unipi.it

[5]Dipartimento di Informatica, Università degli Studi di Milano, Italy, {valentina.ciriani, luca.frontini}@unimi.it

[6]Karlsruhe Institute of Technology, Karlsruhe, Germany, tahoori@ira.uka.de

Introduction

CMOS technology

- Transistor size has shrunk for decades
- The trend reached a critical point

The Moore's Law era is coming to an end

New emerging technologies

- Biotechnologies, molecular-scale self-assembled systems
- Graphene structures
- Switching lattices arrays

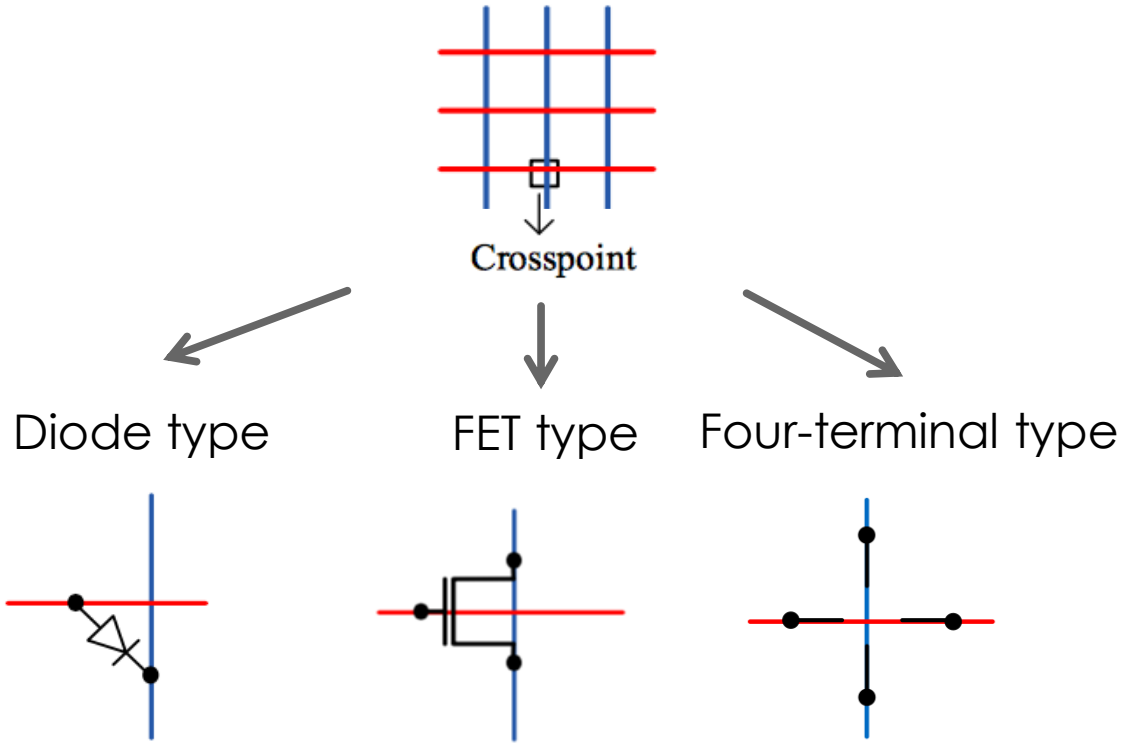
These technologies are in an early state

A novel approach to synthesis, design and testing is necessary, focused on the properties of the devices
that are the main factors for a future technology choice

Our main goal is to design an emerging nanocomputer

Crossbar-type switches

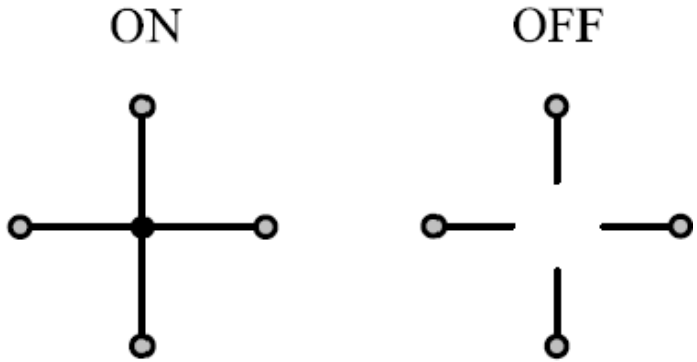
Models based on diodes, FETs, and four-terminal switches

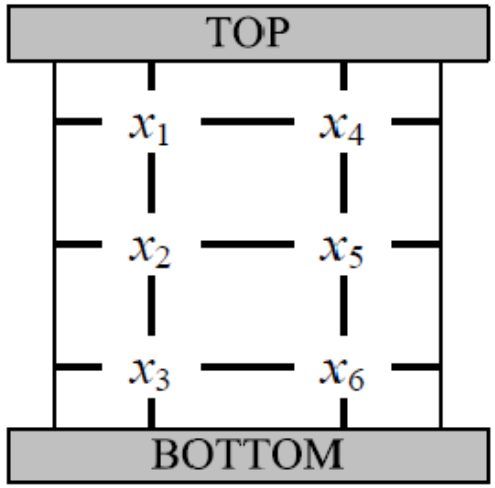


The Switching Lattices

Switching Lattices are **two-dimensional** array of **four-terminal** switches

- When switches are **ON** all terminals are connected, when **OFF** all terminals are disconnected
- Each switch is controlled by a boolean literal, **1** or **0**
- The boolean function f is the SOP of the literals along each path from **top** to **bottom**
- The function synthesized by the lattice is:
  $f = x_1x_2x_3 + x_1x_2x_5x_6 +$
  $+x_4x_5x_2x_3 + x_4x_5x_6$

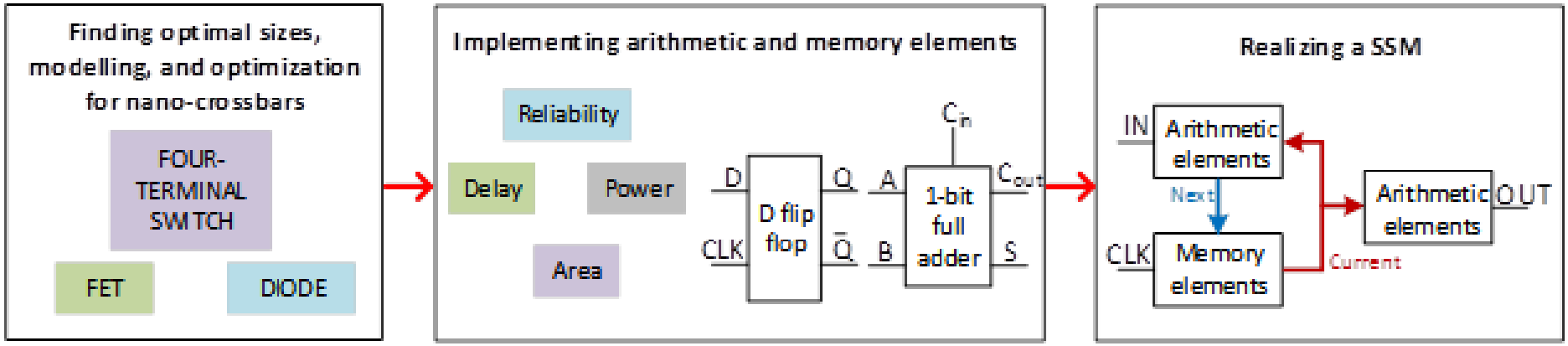




Research Objectives

Main objective: realizing a SSM with switching nanoarrays

Long term vision: being a foundational methodology for future computing devices replacing CMOS



1. **Crossbar model** to build arithmetic and memory elements, with focus on:
   - size and power consumption
   - delay and reliability
2. **Arithmetic** and **memory elements** as building blocks of a computer:
   - adders and multiplexers
   - flip-flop & registers
3. Realize a nano-crossbar based **synchronous state machine** (SSM)
   - implementing arithmetic and logic elements
   - using technology parameters (area, delay, power, reliability)
   - the SSM uses a complete logic flow and clocked control over state registration

Contribution to State of Art

Our model target: nanowire/nanotubes crossbar arrays, magnetic switch-based structures and crossbar memories

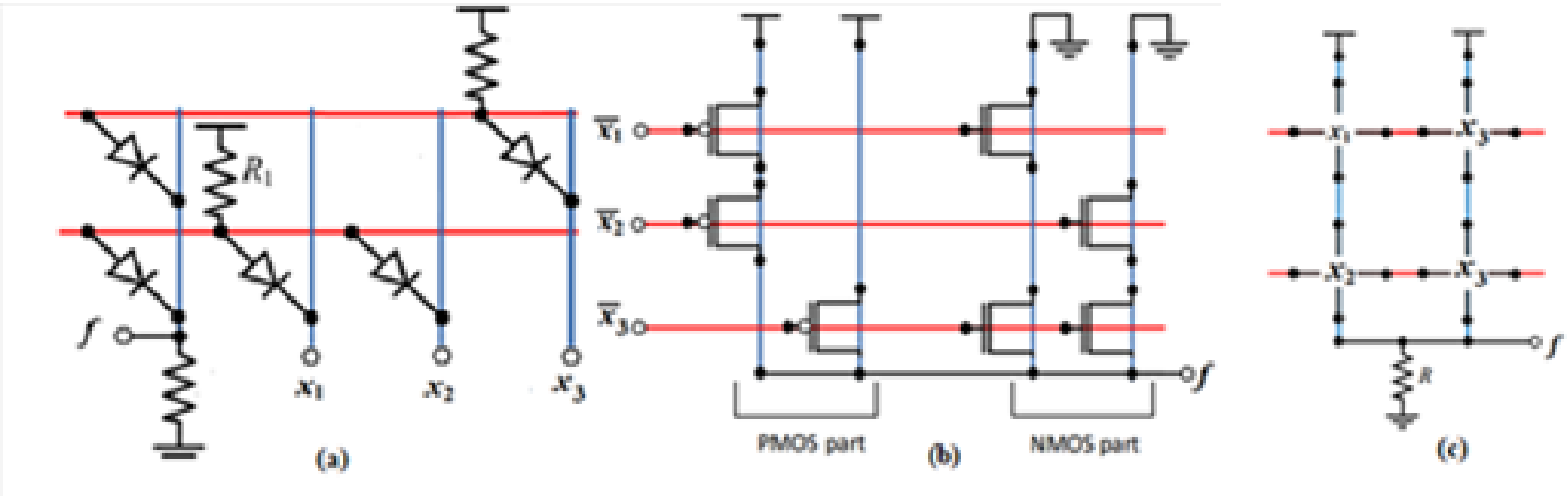
1. **Models for switching nano-crossbars**
   - the previous models are developed for **basic logic operations**
   - we **generalize the model** to be applicable to any Boolean function
   - we improve the model for diode/transistor-based crossbar
   - we develop a **new model based on four-terminal switches**
2. **Performance optimization**
   - previous crossbar circuit implementation considers only **area** and **reliability**
   - we consider also **delay** and **power dissipation**
3. Switching **nano-crossbars circuit implementation**
   - previous implementation performs **simple arithmetic operations**
   - this project will implement **complex arithmetic synthesis and memory elements**

Research Methodology and Approach

- **Technological part:** model electrical and physical characteristics of the technologies
- **Computational part:** use of logic synthesis, graph theory, probability theory, CAD

1. Finding **optimal crossbar sizes**, modeling and optimization
   - graph theory and circuit complexity techniques
   - the problem is likely NP-complete
   - identify the problem as a Boolean satisfiability problem and try heuristic approaches
2. **Implementing arithmetic and memory elements**
   - compare results with CMOS
   - determine the parameters of the technologies
   - design a comprehensive optimization software package
3. Realizing a **Synchronous State Machine**
   - programmable multi-array architecture
   - composed only by four-terminal switches

Logic Synthesis: diode, FET, 4-terminal switches



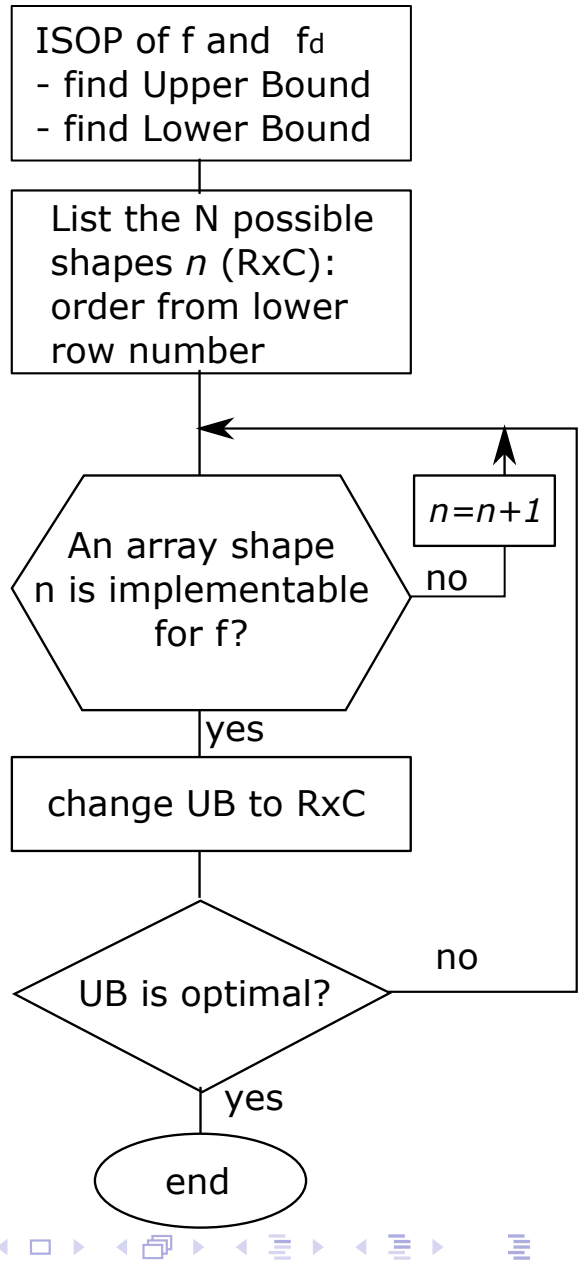
- **Diode:** (number of products in f) $\times$ (number of literals in f + 1)
- **FET:** (number of literals in f) $\times$ (number of products in f + number of products in f^D)
- **4-terminal:**(number of products in f) $\times$ (number of products in f^D)

For example, $f = x_1\overline{x_2} + \overline{x_1}x_2$ size is:

- Diode: 2 $\times$ 5
- FET: 4 $\times$ 4
- 4-terminal: 2 $\times$ 2

Logic synthesis of a 4-terminal array

- the **4-switch dimension**: (number of products in f) $\times$ (number of products in f^D) is not necessarily optimal.
- to **find the optimal solution** is necessary to enumerate all top-to-bottom paths
- this task **grows exponentially** with the array size.
- for these reasons we propose this, preliminary, brute-force **algorithm**
- we are now studying more efficient heuristics



Simulations results:

- each output of benchmark circuit is considered as a separate function
- the number of products of f and f^D are obtained using *Espresso*
- there is always the same sequence: $A_{CMOS} \geq A_{diode} \geq A_{4-terminal} \geq A_{opt.4-terminal}$

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**

Lattices and decomposition methods

- **decompose** a function into some sub-functions
- **implement** the decomposed blocks with
  - separate lattices
  - separate regions in a single lattice
- we focus on an extended form of Shannon co-factoring: **P-circuits**

A *P-circuit* of a completely specified function f is the circuit $P(f)$ denoted by the expression:

$$Pcircuit(f) = (\overline{x}_i \oplus p)\, f^{=} + (x_i \oplus p)\, f^{\neq} + f^{I}$$

where:

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. $\emptyset \subseteq f^{I} \subseteq I$

P-circuits

The sub-functions $f^{=}$, $f^{\neq}$ and f^{I}:

- depends on $n-1$ variables
- have a smaller on-set
- the synthesized lattice can be smaller

the lattice for f built composing the lattices of $f^{=}$, $f^{\neq}$ and f^{I} could be smaller.

The experiments shows that:

- in the 30% of cases the decomposed lattice area is **smaller**
- the **average gain** is at least 20%

Future works:

- more complex types of decomposition
- with more expressive projection p
- using *D-reducible* functions
  - $f = \chi_A \cdot f_A$
  - A is the affine space
  - χ_A is the charateristic function
  - f_A is the projection of f onto A

Defect and fault-tolerant techniques

- Nano-wire construction methods are high sensitive to design variation, defects and environmental factors
- structures with **higher defect rates** than CMOS (15% faulty components)

State-of-art defect and fault-tolerant techniques are not suitable because are based on **small defect rates**

We will investigate on **reconfiguration and redundancy approaches**

- combination of temporal and structural redundancy
- built-in repair circuitry
- system-level adaptation techniques

Built-in variation, defect and fault tolerance

Self assembly process reduce costs but at the expense of the control of the process

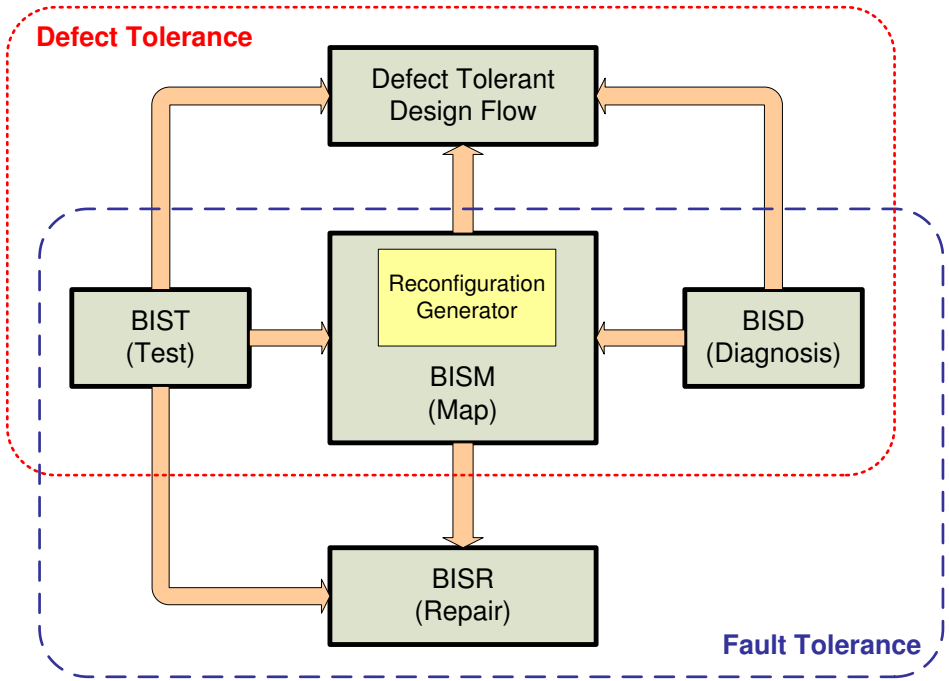
We propose to integrate:

- **defect tolerance** → improve the yield
- **fault tolerance** → lifetime reliability
- **variation tolerance** → predictability & performance

Application-dependent test and diagnosis techniques:

- **BIST:** built-in-self-test
- **BISD:** built-in-self-diagnosis
- **BISM:** built-in-self-mapping
- **BISR:** built-in-self-repair

The goal is to bypass defective resources using test and diagnosis information.



Application-Independent Defect Tolerant Flow

application-dependent

- Defective elements are stored in defect map and avoided
- Not suitable for high-volume production of nano-chips

defect-unaware design flow

- defect tolerance is **performed once**
- **same set** of resources are used for **all applications**
- all design steps are **independent from** the defects **location**

- **universal** defect-free subset of resources
- defect-free chip subset called **design view**
- defect-free subset are **application independent**
- final **mapping phase** with very low complexity

Defect-unaware flow and defect-free subsets

The goal

Find **defect-free** $k \times k$ crossbars inside the original **partially-defective** $n \times n$

1. Choose **size of** k: is correlated to the the **manufacturing yield**
   - if a $n \times n$ crossbar **contains a** $k \times k$ **defect-free** crossbar is **usable**
2. during **physical-design** the design is mapped in $k \times k$ crossbars
3. A **defect-aware mapping** step re-map the used resources within $k \times k$ crossbar inside the $n \times n$ crossbar
4. Is stored only **the location** of the $k \times k$ crossbar, not the information about the single crosspoint
5. The **size of defect map** is reduced from $O(n^2)$ to $O(n)$

Conclusions

Integrating a **new technology** into semiconductor industry is a long road, it is necessary:

- device performances & manufacturability
- advanced research, technology development and industry-compliant implementation

Emerging nano-technologies have:

- ultimate integration **density**
- manufacturing and integration $\rightarrow$ **cost-reduction**
- reduction of **power consumption**

This project is focused on filling the gap of the emerging technologies in:

- **extending the electronic design automation** (EDA) flow
- **novel computer architecture systems**