POLITECNICO DI TORINO

Master of Science in Electronics Engineering

Master Thesis

Design of standard cell libraries for molecular field-coupled nanocomputing design automation

Supervisors
Dr. Yuri Ardesi
Dr. Federico Ravera
Dr. Marcel Walter

Candidate
Danilo Quinci

12/12/2025

Compiled with LaTeX on November 25, 2025.

Abstract

Molecular Field-Coupled Nanocomputing (MolFCN) represents one of the most promising post-CMOS technologies for ultra-low power computing. This technology encodes information in the charge distribution of molecules and processes it through electrostatic interactions, using molecular patterns that implement logic gates and signal routing. Although physical simulation is possible through tools like SCERPA (Self-Consistent Electrostatic Potential Algorithm), the design process remains entirely manual. No automated design tools exist for molecular circuits, severely limiting the technology's practical adoption. The main contribution of this work is to bridge the critical gap between molecular-level physical simulation and automated circuit design, by the development and validation of the first physically simulated standard cell library for MolFCN. The library comprises seven fundamental devices (bus, inverter, L-wire, fan-out, majority voter, AND and OR gates) implemented on a standardized 10×10 grid. Each device was characterized through SCERPA simulations, validating correct functional behavior under realistic physical conditions. Through a collaboration between the Politecnico di Torino and the Technical University of Munich, the developed library, named SIM(7), was integrated into Fiction framework, enabling automatic synthesis of MolFCN circuits from Verilog specifications to molecular implementations. Simulation-based validation demonstrated 100% functional correctness for all library cells and synthesized benchmark circuits, including XOR gates, half-adders, comparators and c17. The work establishes a reproducible methodology for standard library creation in field-coupled technologies, complete with detailed design rules for cell standardization, clocking zones, and EDA compatibility. Through these contributions, MolFCN has transitioned from purely theoretical research into a tangible design platform, opening new possibilities for the scientific community.

Sommario

Il Molecular Field-Coupled Nanocomputing (MolFCN) rappresenta una delle tecnologie post-CMOS più promettenti per il calcolo a bassissima potenza. Questa tecnologia codifica l'informazione nella distribuzione di carica delle molecole e la elabora attraverso interazioni elettrostatiche, utilizzando pattern molecolari che implementano porte logiche e instradamento dei segnali. Sebbene la simulazione fisica sia possibile attraverso strumenti come SCERPA (Self-Consistent Electrostatic Potential Algorithm), il processo di progettazione rimane completamente manuale. Non esistono strumenti di progettazione automatizzata per circuiti molecolari, limitando gravemente l'adozione pratica della tecnologia. Il contributo principale di questo lavoro consiste nel colmare il gap critico tra la simulazione fisica a livello molecolare e la progettazione automatizzata di circuiti, attraverso lo sviluppo e la validazione della prima libreria di celle standard fisicamente simulata per MolFCN. La libreria comprende sette dispositivi fondamentali (bus, inverter, L-wire, fan-out, majority voter, porte AND e OR) implementati su una griglia standardizzata 10×10. Ogni dispositivo è stato caratterizzato mediante simulazioni SCERPA, validando il corretto comportamento funzionale in condizioni fisiche realistiche. Attraverso una collaborazione tra il Politecnico di Torino e l'Università Tecnica di Monaco, la libreria sviluppata, denominata SIM(7), è stata integrata nel framework Fiction, consentendo la sintesi automatica di circuiti MolFCN da specifiche Verilog a implementazioni molecolari. La validazione basata su simulazioni ha dimostrato una correttezza funzionale del 100% per tutte le celle della libreria e i circuiti benchmark sintetizzati, inclusi porte XOR, half-adder, comparatori e c17. Il lavoro stabilisce una metodologia riproducibile per la creazione di librerie standard nelle tecnologie field-coupled, completa di regole di progettazione dettagliate per la standardizzazione delle celle, le zone di clock e la compatibilità EDA. Attraverso questi contributi, il MolFCN è passato da ricerca puramente teorica a una piattaforma di progettazione concreta, aprendo nuove possibilità per la comunità scientifica.

A chi ha visto il meglio anche nelle mie versioni peggiori.

Acknowledgements

I wish to express my deepest gratitude to my supervisors, Dr. Yuri Ardesi, Dr. Federico Ravera, and Dr. Marcel Walter, for their constant guidance, scientific support, and availability throughout this thesis. Their contribution has been fundamental to the development of this research.
Special thanks go to the QNANO research group at Politecnico di Torino for making this project possible and for providing the expertise, resources, and tools necessary for its realization. I am grateful to the research group at the Technical University of Munich for the fruitful collaboration.
Heartfelt thanks go to all members of the research groups who contributed, directly or indirectly, to this achievement.

Contents

List of Tables 11

List of Figures 13

List of acronyms and abbreviations 16

1 Introduction 18

1.1 Limitations of current EDA flows . . . 19

1.2 Objectives . . . 20

1.3 Original contributions . . . 21

1.4 Structure . . . 22

2 Theoretical background and employed tools 25

2.1 Field-Coupled Nanocomputing . . . 25

2.1.1 Quantum-dot Cellular Automata (QCA) . . . 25

2.1.2 In-plane Nano-Magnetic Logic (iNML) . . . 27

2.1.3 Synchronisation and clocking system . . . 28

2.2 Molecular Field-Coupled Nanocomputing . . . 29

2.3 Simulation and characterisation tools . . . 33

2.3.1 SCERPA (Self-Consistent ElectRostatic Potential Algorithm) . . . 33

2.3.2 BBchar: Block-Based Characterization . . . 36

2.3.3 MagCAD . . . 39

2.4 Fiction . . . 41

2.4.1 Supported FCN Technologies . . . 41

2.4.2 Workflow . . . 42

2.4.3 The need for a Standard-Cell Library for MolFCN . . . 46

3 Development of the molFCN device library 49

3.1 Analysis and simulation of current devices . . . 49

3.1.1 Classification of devices . . . 49

3.2 Simulation Methodology . . . 51

3.3 Simulation Results . . . 53

4 Fiction framework alignment 57
4.1 Fiction Requirements . . . 57
4.1.1 Geometric constraints . . . 57
4.1.2 Clocking System and Global Synchronization . . . 57
4.1.3 Compatibility with Fiction Data Structures . . . 58
4.2 Development of the SIM(7) Library . . . 58
4.2.1 Selecting devices from the extended library . . . 59
4.2.2 Tile Design . . . 59
4.2.3 Orientational Variants . . . 60
4.3 Simulation and validation . . . 62
4.3.1 Simulation Methodology . . . 62
4.3.2 Results . . . 63
4.4 Validation of benchmark circuits . . . 75
4.4.1 AND3 . . . 75
4.4.2 NAND2 . . . 77
4.4.3 XOR2 . . . 78
4.4.4 MUX21 . . . 80
4.4.5 HA (Half-Adder) . . . 81
4.4.6 1-bit Comparator . . . 83
4.4.7 C17 . . . 84

5 Conclusion 87
5.1 Future perspective . . . 88

Bibliography 91

A Tiles layout 95

B Tiles Waveplots 97

C MATLAB script for SCERPA simulation 99

D MATLAB script for BBChar simulation 101

List of Tables

2.1 Parameters for the configuration of the termination circuit in BBchar . . 38
2.2 Main parameters for clock signal configuration in BBchar 39
2.3 Operating modes of drivers in BBchar 39

3.1 Interconnection devices 51
3.2 Basic logic devices 52
3.3 Complex logic devices 52
3.4 Complete Library of Validated molFCN Devices 55

4.1 Devices of the library 59
4.2 AND3 truth table 76
4.3 NAND2 truth table 78
4.4 XOR2 truth table 79
4.5 MUX21 truth table 81
4.6 Half-Adder truth table 82
4.7 1-bit Comparator truth table 84

A.1 Tiles - all orientations (Part I) 95
A.2 Tiles - all orientations (Part II) 96

B.1 Waveplot results (Part I) 97
B.2 Waveplot results (Part II) 98

List of Figures

1.1 IC scaling roadmap (More Moore, More than Moore and Beyond CMOS) 18

2.1 Fundamental QCA cell and logic encoding. (a) Basic cell composed of four quantum dots. (b) Two state configurations encoding “0” ($P = -1$) and “1” ($P = +1$). 26
2.2 Fundamental iNML cell and logic encoding 27
2.3 Fundamental molFCN cell and logic encoding 29
2.4 Bisferrocene structure 30
2.5 Information propagation mechanism in molFCN through electrostatic coupling between driver molecules and adjacent cells 31
2.6 Clock field (Eck) 31
2.7 Clock and Phase 33
2.8 SCERPA architecture 34
2.9 Flowchart of the BBchar program to simulate and characterize the DUT. 37
2.10 Edge effect on information propagation: (a) Propagation in a simulated wire in isolation with SCERPA, showing data decay at the end of the wire. The graph below represents the electrostatic potential on each molecule. (b) Propagation in the same wire connected to a termination block consisting of 4 cells, showing data recovery at the output. 38
2.11 XOR layout designed in MagCAD 40
2.12 Field-Coupled Nanocomputing technologies supported by fiction: (a) Quantum-dot Cellular Automata, (b) in-plane Nanomagnet Logic, (c) Silicon Dangling Bonds 42
2.13 Complete design flow in fiction: from Verilog specification to the generation of the physical cell-level layout, verifiable through external simulators. . . 43
2.14 Regular clocking patterns integrated into fiction 45

3.1 Representative validation results from SCERPA simulations 54

4.1 Physical layouts of the devices in the library SIM(7) 61
4.2 The four orientation variants of the double-branch inverter: (a) INV_UP - 0° rotation, (b) INV_DW - 90° rotation, (c) INV_LH - 180° rotation, (d) INV_RH - 270° rotation 62
4.3 Bus potential plot 63
4.4 Bus waveplot 64

4.5 Inverter potential plot . . . 64
4.6 Inverter waveplot . . . 65
4.7 L_DXDW potential plot . . . 65
4.8 L_DXUP potential plot . . . 66
4.9 L_DXDW and L_DXUP waveplots . . . 66
4.10 T_DXDW potential plot . . . 67
4.11 T_DXUP potential plot . . . 67
4.12 T_UPDW potential plot . . . 67
4.13 T_DXDW, T_DXUP and T_UPDW waveplots . . . 68
4.14 Majority Voter potential plot . . . 69
4.15 Majority Voter waveplot . . . 70
4.16 AND_UP potential plot) . . . 71
4.17 AND_DW potential plot . . . 71
4.18 AND_LH potential plot . . . 72
4.19 AND_UP, AND_DW and AND_LH waveplots . . . 72
4.20 OR_UP potential plot . . . 73
4.21 OR_DW potential plot . . . 74
4.22 OR_LH potential plot . . . 74
4.23 OR_UP, OR_DW and OR_LH waveplots . . . 75
4.24 AND3 layout . . . 76
4.25 AND3 waveform characterization . . . 77
4.26 NAND2 layout . . . 77
4.27 NAND2 waveform characterization . . . 78
4.28 XOR2 layout . . . 79
4.29 XOR2 waveform characterization . . . 80
4.30 MUX21 layout . . . 80
4.31 MUX21 waveform characterization . . . 81
4.32 Half adder layout . . . 82
4.33 Half-Adder waveform characterization . . . 83
4.34 Comparator layout . . . 83
4.35 1-bit Comparator waveform characterization . . . 84
4.36 C17 layout . . . 85
4.37 C17 waveform characterization . . . 86

List of acronyms and abbreviations

AC	Aggregated Charge
AIG	And-Inverter Graph
AR	Active Region
BBchar	Block-Based Characterization
CAD	Computer-Aided Design
CLI	Command-Line Interface
CMOS	Complementary Metal-Oxide-Semiconductor
DAG	Directed Acyclic Graph
DFT	Density Functional Theory
DRC	Design Rule Check
DRV	Design Rule Violation
EDA	Electronic Design Automation
FCN	Field-Coupled Nanocomputing
FIB	Focused Ion Beam
HDL	Hardware Description Language
iNML	in-plane Nanomagnet Logic
IR	Interaction Radius
ISCAS	International Symposium on Circuits and Systems
ITRS	International Technology Roadmap for Semiconductors
LUT	Look-Up Table
MNT	Munich Nanotech Toolkit
molFCN	Molecular Field-Coupled Nanocomputing
MoSQuiTo	MOlecular Simulator QCA TOrino
OGD	Orthogonal Graph Drawing
QCA	Quantum-dot Cellular Automata
QLL	QCA Layout Language
SAM	Self-Assembled Monolayers
SAT	Boolean Satisfiability Problem
SCERPA	Self-Consistent ElectRostatic Potential Algorithm
SET	Single-Electron Transistor
SiDB	Silicon Dangling Bonds
STM	Scanning Tunneling Microscopy
VACT	Vin-to-Aggregated Charge Transcharacteristic

Chapter 1

Introduction

Molecular Field-Coupled Nanocomputing (molFCN) represents one of the most promising technologies for the future of digital electronics beyond CMOS. CMOS technology, despite having dominated the semiconductor industry for decades thanks to the continuous scaling predicted by Moore's law, is rapidly approaching its fundamental physical limits. Current technological nodes are, in fact, very close to atomic dimensions, making it virtually impossible to further satisfy Moore's law by simply reducing the size of transistors [1]. In this context, the scientific community has identified several strategies for the advancement of electronics: "More Moore", which pushes scaling to the limits of industrial possibilities; "More than Moore", which improves performance through heterogeneous integration; and "Beyond CMOS", which introduces devices based on alternative physical principles. Fig.1.1 shows the semiconductor technology roadmap from 1960 to 2080, representing the evolution of characteristic dimensions through these three main trajectories: CMOS miniaturization, functional diversification, and emerging beyond-CMOS technologies. The molFCN falls into the latter category.

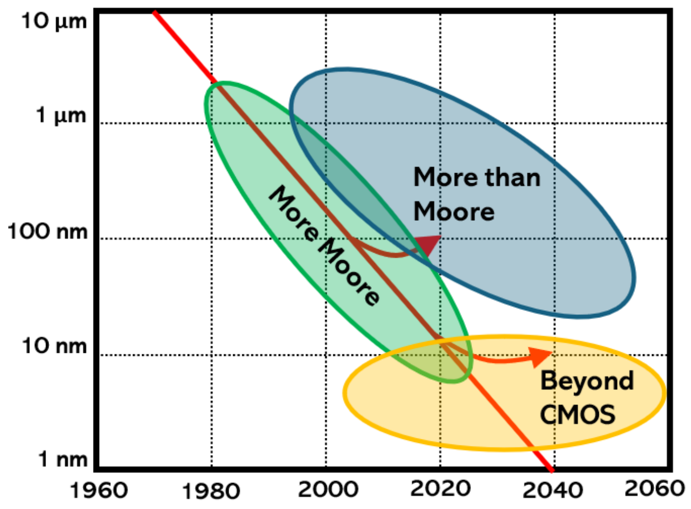


Figure 1.1: IC scaling roadmap (More Moore, More than Moore and Beyond CMOS)

Based on the Quantum-dot Cellular Automata (QCA) paradigm proposed by Lent et al. in 1993 [2], molFCN encodes digital information in the charge distribution of molecules and propagates it through electrostatic coupling between adjacent molecules. This fundamental feature virtually eliminates current flow during logical operations, enabling ultra-low power consumption [3, 14], which is crucial for high-density applications where thermal management is a fundamental limitation.

Despite promising results in the molFCN field, a key challenge remains: bridging the gap between physical simulation of molecular devices and their effective integration into automated design flows for complex digital circuits. Traditional EDA tools, developed for CMOS technologies, rely on established electrical models that do not account for the unique physical characteristics of molecular technologies.

1.1 Limitations of current EDA flows

Traditional electronic design tools have several fundamental limitations when applied to emerging molecular technologies such as molFCN. Conventional EDA flows are based on an established paradigm that separates the physical, logical, and architectural levels, using standardised electrical models developed specifically for CMOS transistors. These models are based on concepts of current, voltage, and resistance that define the behaviour of traditional integrated circuits.

In the molFCN context, this separation is problematic because the behaviour of the device is intrinsically linked to molecular physics and electrostatic interactions [12]. Existing EDA tools lack the granularity necessary to accurately model the quantum effects and electrostatic properties specific to the molecules used in molFCN devices. Specialised tools such as QCADesigner [47], although developed for the QCA paradigm, implement simplified models (such as the Two-State Approximation) that do not consider specific molecular physics, leading to simulation results that can diverge significantly from the actual behaviour of physically implemented devices [4].

The limitations of traditional EDA flows discussed above highlight a critical need: to bridge the gap between molecular physics simulation and automated circuit design, it is necessary to develop standard molFCN device libraries that serve as a bridge between these two worlds. The strategic importance of such a development is clearly evident when looking at the Fiction framework [30, 31], developed by the Technical University of Munich, which currently represents the most mature environment for the automated design of Field-Coupled Nanocomputing circuits. Fiction already supports three FCN technologies through fully characterised standard libraries: QCA via QCA ONE [41], iNML via ToPoliNano [10, 11], and SiDB with Bestagon [42]. The success of these integrations concretely demonstrates the value of standard libraries in enabling the automated design of complex FCN circuits, allowing for the synthesis, placement, and automatic routing of architectures that would be impossible to design manually.

The absence of standard molFCN libraries creates a vicious circle that hinders the development of the technology: without physically characterised standard libraries, EDA tools such as Fiction cannot be used effectively to design complex molFCN circuits; without the ability to design and validate complex circuits, it is difficult to demonstrate the real potential of the technology compared to existing alternatives. This vicious circle currently confines molFCN research to the manual design of simple circuits, preventing the exploration of complex architectures that could demonstrate the real competitive advantages of the technology.

The complexity of creating standard libraries for molFCN presents unique challenges compared to other technologies. A molFCN standard library must include not only fundamental logic devices, but also technology-specific interconnect elements and complex devices necessary for information routing. The field-coupled nature of the technology introduces particularly challenging characterisation requirements[19]: each element must be characterised through physically accurate simulations that consider the specific molecule used, the precise geometry of the device, electrostatic interactions with adjacent devices, and the effects of the external clock system[17]. molFCN devices require computationally intensive simulations based on self-consistent electrostatic calculations at the molecular level [21], where small geometric variations or electrostatic perturbations can compromise device operation.

The creation of molFCN standard libraries requires the development of a systematic methodology that links physical simulation at the molecular level with the parameters required by EDA tools (timing, area, geometric, and layout constraints). This characterisation must be sufficiently detailed and accurate to ensure that automatically synthesised circuits maintain functional correctness when implemented at the physical level. The validation of this methodology through integration with Fiction is therefore the strategic objective to demonstrate that physically accurate molFCN standard libraries can effectively enable the automated design of complex molecular circuits, definitively bridging the gap between physical simulation and design automation.

1.2 Objectives

The primary objective of this thesis is to develop a comprehensive library of standard cells for molFCN technology, characterised through physically accurate simulations, in order to bridge the critical gap identified in the previous section. The library includes all the fundamental devices necessary for the automatic synthesis of complex digital circuits: from basic logic devices to technology-specific interconnection elements, to the complex components required for information routing.

The development process is based on the use of the SCERPA (Self-Consistent Electrostatic Potential Algorithm) simulator[21] for the rigorous physical characterisation of each device. SCERPA, developed at the Polytechnic University of Turin, is an advanced tool for simulating molFCN circuits that implements a self-consistent approach for calculating electrostatic potential, accurately considering molecular interactions, the effects of the external clock system, and electrostatic perturbations between adjacent devices, critical aspects that traditional simulators are unable to capture.

Characterisation is carried out using a standardised ideal molecule, representing a methodological evolution compared to previous approaches based on specific molecules such as bis-ferrocene [48]. The ideal molecule, while retaining the essential characteristics for QCA operation, offers optimised parameters to maximise electrostatic coupling, thermal stability and reproducibility. This approach eliminates dependencies on variations in the physicochemical properties of specific molecules, facilitating the development of standard methodologies for device characterisation and optimisation that can be applied systematically.

The final library provides comprehensive information for each device necessary for integration with EDA tools: implemented logic function, number and position of inputs and outputs, area occupied, physical dimensions, number of cells and molecules used.

A strategic objective of the thesis is to ensure full compatibility of the developed library with existing automated design tools, in particular with the Fiction framework [30, 31]. This thesis is part of a strategic collaboration between the Politecnico di Torino and the Technical University of Munich, which aims to create a complete ecosystem for molFCN circuit design. The collaboration combines the complementary expertise of the two research groups: the POLITO group has developed SCERPA, an advanced tool for the accurate physical simulation of molecular devices, while the TUM group has developed Fiction, the most mature framework for the design automation of FCN technologies. Fiction already supports QCA, iNML and SiDB through dedicated libraries that provide the specific physical implementations for each technology. However, the absence of a molFCN library currently prevents the use of Fiction's capabilities for this technology: while the logic synthesis, placement, routing and clocking algorithms work correctly, generating abstract gate-level layouts, the crucial technology mapping phase is impossible due to the lack of standardised molecular implementations.

The aim of this thesis is to bridge this gap through the development and integration of the first molFCN standard library in Fiction, representing an example of vertical integration covering the entire design chain, from the physical characterisation of molecules with SCERPA to the automatic synthesis of complete architectures with Fiction. The standard library developed is the missing link that effectively connects these two levels of abstraction, enabling for the first time automatic technology mapping from abstract gate-level layouts to physically simulatable cell-level molecular layouts [49]. This will demonstrate the feasibility of end-to-end automatic synthesis of complex molFCN circuits starting from high-level Verilog descriptions, thus validating the entire design chain and bringing molFCN from a purely theoretical technology to a design platform that can be used in practice by the scientific community.

1.3 Original contributions

The first original contribution of this thesis consists of the systematic and comprehensive characterisation of a library of molFCN devices using physically accurate simulations based on SCERPA. Unlike previous work that focused on individual isolated devices or simplified models, this work provides a consistent and rigorous characterisation of the entire set of fundamental devices.

The process involved 24 devices of varying complexity, organised into a comprehensive functional taxonomy: interconnection devices (elementary wires, shaped wires, ramified connections), basic logic devices (inverters, majority voters, fundamental gates), and complex logic devices (multiplexers, XOR/XNOR gates). For each device, a complete functional analysis was conducted through multi-timestep simulations that consider electrostatic interactions with adjacent devices and the effects of the clock system, identifying any critical devices that require optimisation. The results constitute the first fully characterised and physically validated molFCN library available to the scientific community.

A second contribution is the development of a systematic and reproducible methodology for creating standard libraries for emerging field-coupling technologies. The methodology covers the entire flow from physical simulation with SCERPA to the generation of Fiction compatible library files, providing a template that can be applied to other FCN technologies.

The main components include: rigorous criteria for device selection based on functional completeness and EDA compatibility; geometric standardisation rules to ensure integrability with automatic placement algorithms; functional validation procedures through multi-timestep simulations; protocols for automatic extraction of EDA parameters; and systematic management of device rotations and orientations. Detailed documentation of all steps and decision criteria ensures reproducibility and facilitates future extensions.

The third original contribution concerns the adaptation of the library to the strict architectural and layout constraints imposed by Fiction to ensure compatibility with design automation algorithms [30]. The adaptation required: standardisation of cell shapes to ensure alignment during placement; definition of fixed positions for inputs and outputs to facilitate automatic routing; management of clock zones for correct synchronisation; and creation of all geometric rotations for each device .

The reduced SIM(7) library was derived from the complete inventory following criteria of functional completeness (universal logical completeness), compatibility with Fiction (compliance with all geometric and clocking constraints), and implementation complexity (preference for compact layouts). Validation through synthesis and physical simulation of benchmark circuits demonstrated for the first time the feasibility of automated design for physically realistic molFCN circuits [49]. The developed export interface allows closed-loop validation: circuits synthesised by Fiction can be exported in SCERPA format and simulated to verify functional correctness at the molecular physics level.

1.4 Structure

The thesis is organised into five chapters that systematically cover all aspects of the work carried out, from reviewing the state of the art to experimentally validating the results obtained.

Chapter 2 provides a comprehensive review of the theoretical background in molFCN technologies, clearly identifying the technological gap that this thesis aims to bridge. The chapter also includes a detailed description of the tools used (SCERPA, Fiction, Bbchar, and MagCad) and their capabilities.

Chapter 3 describes the development of the molFCN device library, starting from the transition from the bis-ferrocene molecule to the ideal molecule used in this work. The chapter presents a comprehensive analysis and simulation of 24 devices from the initial extended library. All devices are simulated and validated using SCERPA. This chapter establishes the foundation for the subsequent selection and standardisation process.

Chapter 4 focuses on the alignment of the library with Fiction framework requirements and the development of the final SIM(7) standard cell library. The chapter details the selection criteria, tile standardisation, and validation through both individual cell testing and benchmark circuit synthesis (AND3, NAND2, XOR2, MUX21, half-adder, comparator, C17).

Chapter 5 discusses technological challenges toward functional prototypes and summarises the achieved results, emphasising how molFCN has transitioned from theoretical research to a practical design platform.

Appendices provide complete technical documentation: tiles layouts (Appendix A), tiles waveplots (Appendix B), SCERPA simulation scripts (Appendix C), and BBchar characterisation scripts (Appendix D), ensuring reproducibility and facilitating future extensions.

Chapter 2

Theoretical background and employed tools

2.1 Field-Coupled Nanocomputing

The Field-Coupled Nanocomputing (FCN) paradigm represents an innovative approach to nanoscale computing that uses electric or magnetic field-based control to guide the behaviour of nanoscopic components [5]. One of the main advantages of FCN lies in its potential for significant power reduction, thanks to the use of coupling fields instead of electric currents, eliminating net charge transport between adjacent cells.
All FCN implementations share some common fundamental principles. First, the basic element is a nanoscopic cell capable of assuming distinct states corresponding to binary logic values. Second, the cells interact with each other via local fields, without the need for direct physical connections, allowing information to propagate through the coupling of adjacent cells. Thirdly, to ensure correct propagation and avoid metastable conditions, a synchronisation system based on clock zones controlled by phase-shifted signals is required to guide the switching of cells in a quasi-adiabatic manner. The following paragraphs discuss the leading implementations of the FCN paradigm, which are the Quantum-dot Cellular Automata (QCA) [2] and the In-plane Nano Magnetic Logic (iNML) [10]. After describing the specific implementations, the synchronisation system common to both technologies will be presented.

2.1.1 Quantum-dot Cellular Automata (QCA)

Quantum-dot Cellular Automata (QCA) is one of the most studied implementations of FCN. The fundamental QCA cell consists of a square architecture of four quantum dots that interact via tunnelling, as shown in Figure 2.1(a).

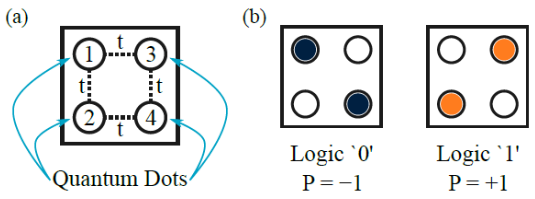


Figure 2.1: Fundamental QCA cell and logic encoding. (a) Basic cell composed of four quantum dots. (b) Two state configurations encoding "0" ($P = -1$) and "1" ($P = +1$).

Energy tunnelling between adjacent sites allows electrons to move within the cell [2]. When two additional electrons are introduced into the cell, Coulomb repulsion forces them to occupy dots positioned at opposite corners, generating two energetically equivalent state configurations that encode the logical values "0" and "1", as illustrated in Figure 2.1(b).

The polarisation of cell P is defined by the equation:

$$P = \frac{(\rho_2 + \rho_3) - (\rho_1 + \rho_4)}{\rho_1 + \rho_2 + \rho_3 + \rho_4} \tag{2.1}$$

where ρ_i represents the electron charge of dot i, and is a fundamental quantity for describing the charge distribution in the cell [2]. The cases corresponding to the two logical values are associated with $P = -1$ for the logical state "0" and $P = +1$ for the logical state "1". For an isolated cell, the fundamental state is an equivalent combination of these two states and therefore has a net polarisation of zero.

The QCA paradigm can be implemented using different technological platforms, each with specific advantages and limitations. Implementations with **metallic quantum dots** use aluminium quantum dots with aluminium oxide tunnel junctions, which have experimentally demonstrated the possibility of cell switching. Metal quantum dots, typically on the order of tens of nanometres, offer high manufacturing yield and consistent electrical behaviour. However, they require operation at cryogenic temperatures, representing one of the main implementation challenges. **Semiconductor implementations**, on the other hand, use materials such as silicon or gallium arsenide, offering compatibility with existing manufacturing processes and potential integration with current electronic devices.

In a multi-cell QCA system, interaction occurs exclusively through Coulomb electrostatic coupling, without electron transfer between distinct cells. This constraint is physically achieved through sufficiently high potential barriers between different cells, which prevent intercellular tunnelling while allowing intracellular tunnelling [2, 4]. Information propagation is based on the principle that, in a two-cell system, the minimum energy configuration corresponds to the situation in which both cells encode the same logical information. If the polarisation of the first cell (driver cell) is fixed, the Coulomb repulsion between electrons in adjacent cells forces the second cell (receiver cell) to assume the same polarisation, thereby propagating the logical value [2]. By placing several cells in a row, the value forced at the beginning propagates through all the cells by means of electrostatic coupling, creating a binary wire. By rotating the cells 45° relative to the propagation axis, neighbouring cells assume opposite configurations, creating a wire of inverting cells that performs the NOT logic function. Since the interaction is based on electrostatic coupling, the state of a cell depends on the superposition of the effects of all the surrounding cells, allowing the implementation of complex logic gates such as the three-input majority voter, which is the fundamental logic element of QCA architecture.

2.1.2 In-plane Nano-Magnetic Logic (iNML)

Nano-Magnetic Logic (NML), and in particular its In-plane NML (iNML) variant, represents another significant implementation of the FCN paradigm, where information is encoded through the magnetisation of single-domain nanomagnets rather than through electrical charges [8, 7]. This technology uses rectangular nanomagnets as carriers of binary information. The nanomagnets are characterised by stable magnetisations parallel or antiparallel to the easy axis (longest axis), encoding the logical states “1” and “0” respectively as shown in Figure 2.2.

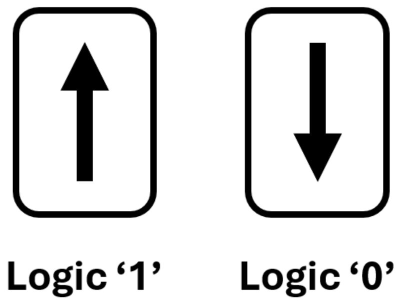


Figure 2.2: Fundamental iNML cell and logic encoding

iNML technology exploits the shape anisotropy of rectangular nanomagnets, typically measuring a few tens of nanometres. The rectangular shape defines two magnetic axes with distinct characteristics: the easy axis, corresponding to the larger dimension, along which magnetisation spontaneously aligns with minimal energy, and the hard axis, corresponding to the smaller dimension, along which magnetic orientation requires additional energy. This geometric anisotropy creates a sufficiently high energy barrier between the two stable magnetic states (parallel or antiparallel magnetisation along the easy axis), ensuring information retention without power consumption in standby mode. The magnets can therefore operate simultaneously as memory and logic elements [8], offering intrinsic non-volatility, which is a significant advantage over QCA implementations. An iNML magnetic wire is constructed by arranging an even number of nanomagnets in cascade. Information propagates through dipolar magnetic coupling between adjacent magnets, the nature of which depends on their mutual geometric orientation. When the magnets are placed side by side along their easy axes, antiferromagnetic coupling occurs: adjacent magnets assume opposite magnetisations to minimise the total magnetostatic energy. When the magnets are aligned along their hard axes, the coupling becomes ferromagnetic and neighbouring magnets tend to assume the same direction of magnetisation [26].

By modifying the magnetisation of the initial magnet, one would expect a cascade propagation that causes all subsequent magnets to switch. However, the same energy barrier that ensures the stability of the information is too high to be overcome by the dipole coupling energy alone [7]. An external clock field is therefore necessary to temporarily reduce the energy barrier and facilitate the controlled switching of the magnets.

The creation of complex magnetic circuits is based on the ability to arrange magnets in various geometric configurations. The three-input majority voter is the universal logic element for iNML. It is also possible to create two-input AND and OR gates without using a majority voter, by using magnets with an angular cut that favours the preferential orientation of magnetisation towards the direction of the cut. The nanomagnets are defined using Focused Ion-Beam (FIB) lithography, a technique that also allows local manipulation of magnetic properties [26]. The manufacture of iNML prototypes has reached a higher level of maturity than other FCN implementations, with numerous experimental validations of logic gates and functioning circuits.

2.1.3 Synchronisation and clocking system

To ensure correct propagation of information and avoid metastable conditions that could corrupt data, all FCN circuits require a sophisticated synchronisation system based on clock domains [6]. This mechanism is common to all FCN implementations, although the physical field used may vary. In six-element FCN cells (four logic elements plus two for the NULL state), the clock dynamically modifies the barriers between elements to encourage or discourage charge movement or magnetisation rotation, thereby controlling the encoding and propagation of the logic state. The FCN circuit is divided into clock zones, each controlled by a clock signal that is phase-shifted with respect to adjacent regions [6]. The clock cycle consists of four distinct phases:

1. **Switch phase**: the barriers are slowly modified to favour the adiabatic switching of the cells from the NULL state to one of the two logical states. The charges (or magnetisation in magnets) are gradually released from the NULL elements and can localise on the logical elements under the influence of neighbouring cells. In particular:
   - In QCA: the potential barriers are increased, allowing electrons to move from the NULL dots to the logical dots
   - In iNML: the external magnetic field induces a 90 rotation of the magnetisation vector along the short axis, facilitating propagation.
2. **Hold phase**: the barriers are kept stable to preserve the encoded information. The elements remain confined to their logical positions and the polarisation of the cell is kept stable without further switching phenomena.
3. **Release phase**: the barriers are slowly returned to promote adiabatic switching to the NULL state. The elements are gradually moved from their logical positions to their NULL positions, gradually disabling intercell interaction.
4. **Reset phase**: the barriers are maintained at the level that keeps the cells in the NULL state, erasing the previous information and preparing the cells for the next

processing cycle. The cells are inhibited and do not participate in information processing.

2.2 Molecular Field-Coupled Nanocomputing

The Molecular Field-Coupled Nanocomputing (molFCN) paradigm represents one of the most promising implementations of the Quantum-dot Cellular Automata (QCA) concept[12]. This technology encodes digital information in the charge distribution within individual molecules, exploiting the electrostatic coupling between adjacent molecules for information propagation, thus eliminating the need for charge transport and consequently minimizing power dissipation[15].

The fundamental element of molFCN is the unit cell, consisting of two electrostatically coupled oxidised molecules. In the general QCA paradigm, a cell is composed of four quantum dots arranged in a square, with two mobile electrons that, due to Coulomb repulsion, occupy the antipodal dots, defining two minimum energy configurations that encode the logical states “0” and “1” (Figures 2.1). In molFCN, this concept is extended using a total of six redox centres (or “dots”): each molecule has three functional dots, and two adjacent molecules form the complete cell. The charges within the molecules are distributed across the dots to minimise the total electrostatic energy of the system, resulting in stable configurations that represent logical states. In addition to the two binary logical states, molFCN introduces a third fundamental state called ‘NULL’, in which both charges occupy the central dots of the cell (Dot3 in each molecule). Although the NULL state does not encode useful logical information, it is essential for implementing the adiabatic switching mechanism and for correctly managing the synchronisation between different regions of the circuit during information propagation[17]. Figure 2.3 clearly shows the unit cell and the charge distributions corresponding to the states “0”, “1” and NULL.

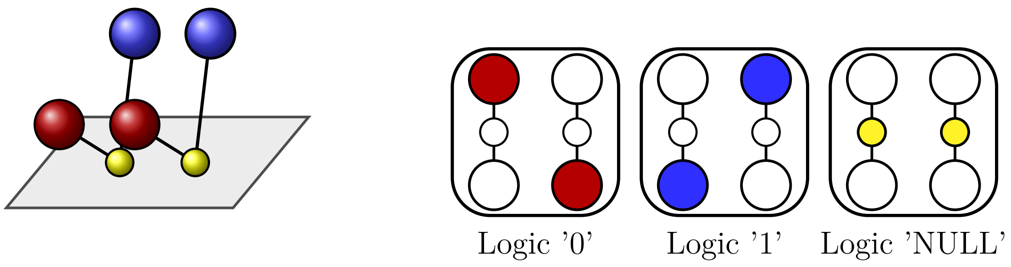


Figure 2.3: Fundamental molFCN cell and logic encoding

Among the various molecules proposed for the implementation of molFCN, bis-ferrocene has received particular attention. The molecular structure, illustrated in Figure 2.4, comprises two ferrocene units that act as active logic dots (Dot1 and Dot2), a central carbazole group that constitutes Dot3 for encoding the NULL state, and a thiol group that allows the molecule to be anchored to the gold substrate. This molecular architecture, with the functional dots separated by approximately 1 nm, allows for clear localisation of the charge in the active sites. The molecule is typically used in its oxidised form, giving it a net charge of +1e.

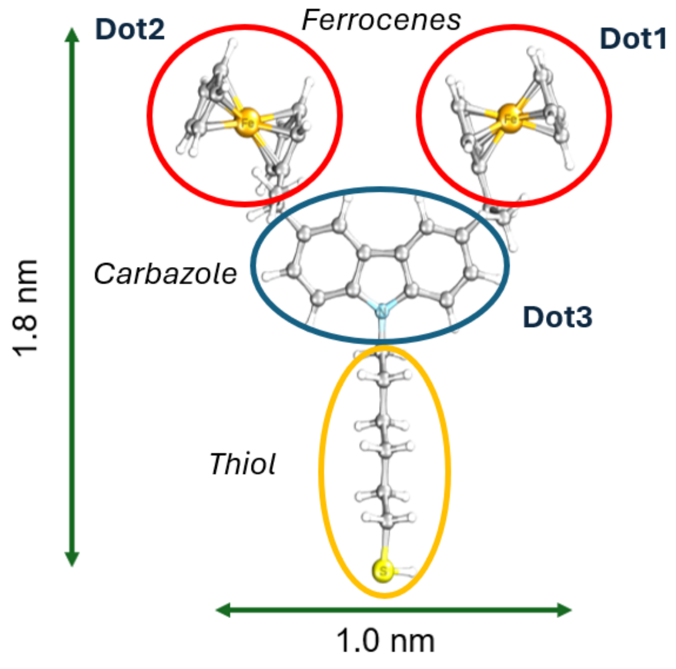


Figure 2.4: Bisferrocene structure

The fundamental mechanism governing the operation of molFCN is the electrostatic coupling between neighbouring molecules[19]. When two molecules are positioned at a nanometre distance, the charge distributions of each molecule generate electric fields that mutually influence the charge state of the other molecule. This principle of interaction can be understood by considering that each oxidised molecule, with its positive charge localised on specific dots, acts as a source of electric field. Adjacent molecules, subjected to this field, rearrange their charge distribution to minimise the total electrostatic energy of the system, respecting the principle of Coulomb repulsion. Information propagation in a molFCN system occurs through a cascade mechanism: when an input molecule (called a “driver”) has its charge distribution fixed in a defined logical state, it generates an electric field that polarises the adjacent molecule. The latter, in turn, adopts a charge configuration that minimises the interaction energy with the driver and, simultaneously, generates an electric field that influences the next molecule in the chain[18]. This process repeats itself, allowing logical information to propagate along linear structures called wires, as shown in Figure 2.5.

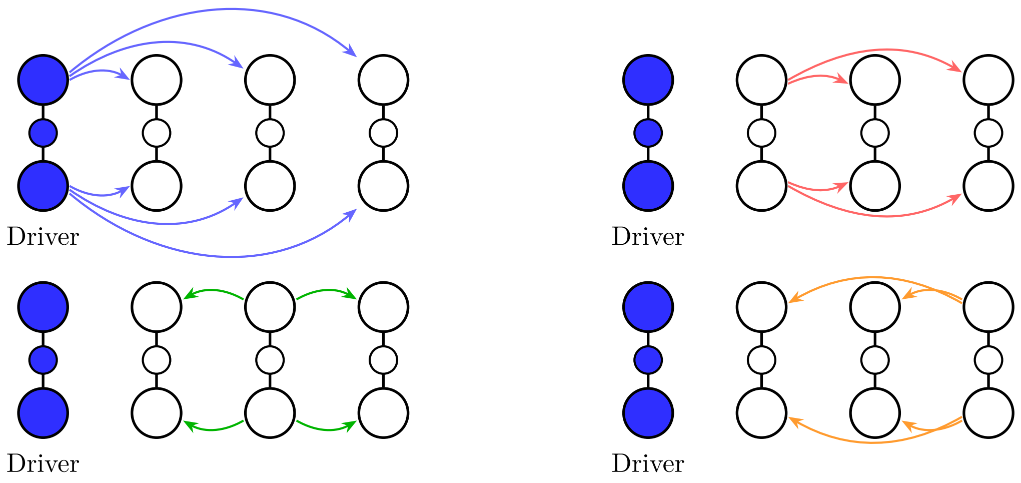


Figure 2.5: Information propagation mechanism in molFCN through electrostatic coupling between driver molecules and adjacent cells

It is important to note that the strength of electrostatic coupling critically depends on the intermolecular distance and the geometry of the system. Distances that are too great reduce the intensity of the interaction to the point where information propagation becomes impossible, while distances that are too small could cause unwanted interference. Furthermore, the electrostatic nature of the interaction implies that the field generated by a molecule extends radially into space, potentially affecting even molecules that are not directly adjacent along the desired propagation path, a phenomenon known as crosstalk that must be carefully managed in circuit design.

In the absence of control mechanisms, such propagation would quickly lead to metastable energy states or incorrect configurations, making it necessary to use a synchronisation system based on the clock field called the Eck field. This field, applied perpendicularly to the substrate on which the molecules are anchored, dynamically controls the charge state of the molecules, allowing the circuit synchronisation to be managed. The clock field can take positive or negative values: a positive field pushes the oxidation charge towards the logic dots (Dot1 and Dot2), enabling the encoding of binary information; a negative field forces the charge towards the central dot (Dot3), imposing the NULL state and erasing the previously encoded information[16] (fig.2.6).

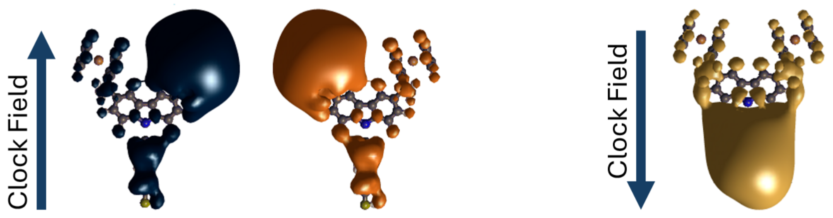


Figure 2.6: Clock field (Eck)

To implement functional molFCN circuits, layouts are divided into spatially distinct clock regions, each controlled independently by a dedicated clock signal[17]. Each clock region operates according to a complete cycle (clock cycle) consisting of four sequential phases: Switch, Hold, Release and Reset. This division allows information to be propagated in a unidirectional and controlled manner.

- During the **Switch phase**, the positive charge present in the molecule is gradually pushed towards the two logic dots (Dot1 and Dot2) by applying a ramping clock signal. During this phase, the molecules can polarise under the influence of neighbouring cells, allowing information to be encoded through electrostatic coupling.

- The **Hold phase** keeps the charge trapped in the active dots of the molecules by means of a fixed positive clock field. In this phase, cell polarisation is preserved without further switching phenomena, ensuring the stability of the encoded information.

- During the **Release phase**, the positive molecular charge is gradually returned to the central dot (Dot3) by applying a decreasing ramp clock signal. Intermolecular interaction, and therefore the propagation of information, is gradually disabled.

- During the **Reset phase**, the charge is held fixed in the dot 3 by a negative clock field. All molecules are kept in the NULL state, preparing the system for the next processing cycle.

In Figure 2.7, a wire divided into four consecutive clock regions is considered. The propagation process is divided into time phases corresponding to the different configurations of the regions, in particular:

- At time S1, region CK1 completes the Switch phase and enters the Hold phase. The molecules in CK1 are then influenced by the electrostatic field from the driver molecules and encode the logical information. Region CK2 is at the beginning of the Switch phase, while CK3 and CK4 are in the Reset state.

- At instant S2, CK1 enters the Release phase, CK2 completes the Switch and enters Hold, CK3 begins the Switch phase, and CK4 remains in Reset. In this way, the information previously encoded in CK1 is propagated to the CK2 molecules.

- At time S3, CK1 enters Reset, CK2 enters Release, CK3 completes Switch and enters Hold, while CK4 begins the Switch phase.

- Finally, at time S4, CK1 is in Reset, CK2 completes Release and enters Reset, CK3 remains in Hold, and CK4 completes the Switch and enters Hold, thus completing the propagation of information across all regions.

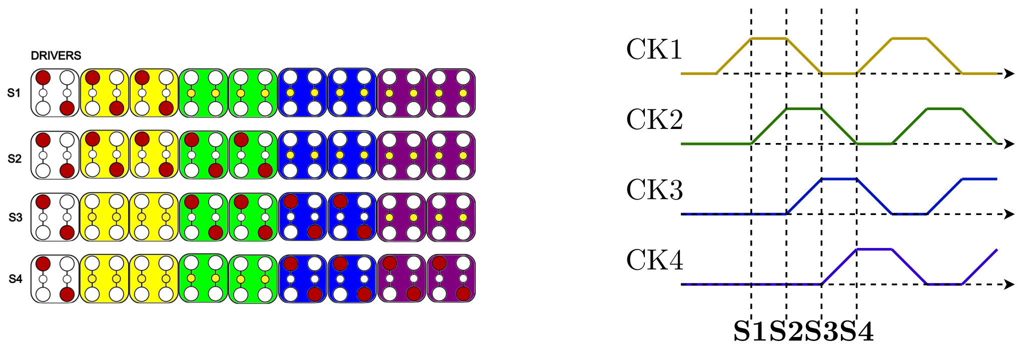


Figure 2.7: Clock and Phase

This architecture ensures unidirectional and controlled propagation of information. The sequential activation of regions, achieved by shifting the clock signals by $\pi/2$, prevents information from propagating backwards or generating logical conflicts, eliminating possible instabilities due to the bidirectional nature of electrostatic coupling[13]. Furthermore, the gradual transition between states (ramps in the Switch and Release phases) ensures quasi-adiabatic switching, ensuring that the charges within the molecules are moved gradually, minimising energy dissipation.

2.3 Simulation and characterisation tools

The design and validation of molFCN circuits require specialized simulation tools capable of accurately modeling the complex electrostatic interactions between molecules, the effects of external clock fields, and the propagation of information through coupled molecular networks. This section presents three complementary tools: SCERPA for physical simulation, BBchar for device characterization, and MagCAD for layout design.

2.3.1 SCERPA (Self-Consistent ElectRostatic Potential Algorithm)

The **Self-Consistent Electrostatic Potential Algorithm** (SCERPA) is a tool for simulating Field-Coupled Nanocomputing molecular circuits (molFCN)[21]. Implemented in MATLAB, SCERPA constitutes the third phase of the **MoSQuiTo** (MOlecular Simulator QCA TOrino) framework, a structured methodology for modelling molFCN circuits. The first two phases of MoSQuiTo involve: (i) the optimisation of molecular geometry and ab initio characterisation under different electric fields (E_{sw}, E_{ck}); (ii) extraction of **VACT** (Vin-to-Aggregated Charge Transcharacteristic), which correlate the input voltage V_{in} with the aggregated charge (AC) in the molecule[20].

In the third phase, SCERPA uses these transcharacteristics to simulate complete circuits using classical electrostatic equations, eliminating the need for ab initio calculations during simulation. The SCERPA architecture is organised into three components: **Layout**, **Algorithm**, and **Viewer** (Fig.2.8)[24].

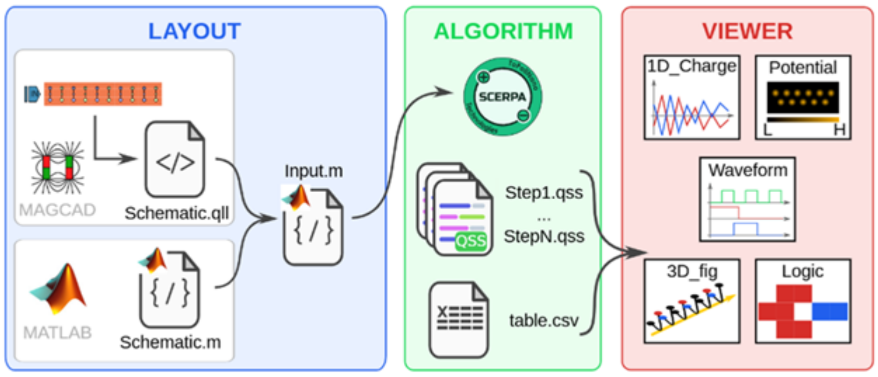


Figure 2.8: SCERPA architecture

Layout

The **Layout** module manages the geometric and topological definition of the circuit. The specification can be done by: (i) **direct definition in MATLAB**, where the user specifies molecular positions, intermolecular distances, rotations, and clock zones; (ii) **import from MagCAD**[27, 28], designing the circuit graphically and importing it via `.qll` files.

The layout still requires the definition of critical parameters, such as:

- **Driver molecules**: provide inputs with a fixed charge distribution defined by the user.
- **Output molecules**: molecules on which the logical result is evaluated.
- **Time sequence**: *Values_Dr* matrix where each row represents a driver and each column a time step.
- **Clocking system**: definable in *phase mode* (uniform field for zones, typically 3-4 phases with $E_{reset} = -2$ V/nm, $E_{hold} = +2$ V/nm) or *map mode* (complete spatial map of the field on a grid).

For each molecule, SCERPA accesses a molecular database containing pre-calculated trans-characteristics for different clock field intensities, molecular geometry, and chemical associations between sites. The modular architecture allows for expansion with new molecules.

Algorithm

The **Algorithm** module implements a self-consistent iterative procedure to solve the system of nonlinear equations of electrostatic interactions[22]. For a circuit of N molecules, the input potential on molecule i is:

$$V^{\tau}_{in,i} = V^{\tau}_{D,i} + \sum_{\substack{j=1 \\ j\neq i}}^{N} V_{j,i}(V^{\tau}_{in,j}, E^{\tau}_{clk,j}) \tag{2.2}$$

where τ denotes the timestep, $V^{\tau}_{D,i}$ is the contribution of the drivers, and $V_{j,i}$ is the electrostatic potential of molecule j on molecule i, evaluated using the **aggregate charge** model:

$$V_{j,i} = \frac{1}{4\pi\epsilon_0} \sum_{\alpha=1}^{N_{AC}} Q_\alpha(V_{in,j}, E_{clk,j}) \left(\frac{1}{|\mathbf{R}_{1,i} - \mathbf{R}_{\alpha,j}|} - \frac{1}{|\mathbf{R}_{2,i} - \mathbf{R}_{\alpha,j}|} \right) \tag{2.3}$$

where $Q_\alpha(V_{in,j}, E_{clk,j})$ is the aggregate charge on site α of molecule j obtained from the VACTs, $\mathbf{R}_{1,i}$ and $\mathbf{R}_{2,i}$ are the positions of the logical sites of molecule i, and $\mathbf{R}_{\alpha,j}$ is the position of site α of molecule j.

The system is solved iteratively using the **fixed point method**. Given an initial configuration $\{V^0_{in,1}, \dots, V^0_{in,N}\}$, the algorithm iterates[21]:

$$V^k_{in,i} = F_i(V^{k-1}_{in,1}, \dots, V^{k-1}_{in,N}) \tag{2.4}$$

where k is the iterative step and F_i evaluates Eq. (2.3). The procedure continues until convergence:

$$\max_{i=1,\dots,N} \left| V^k_{in,i} - V^{k-1}_{in,i} \right| < \epsilon_{conv} \tag{2.5}$$

To improve numerical stability, a damping factor $\xi \in [0,1)$ is applied:

$$V^k_{in,i} = \xi V^{k-1}_{in,i} + (1-\xi) F_i(V^{k-1}_{in,1}, \dots, V^{k-1}_{in,N}) \tag{2.6}$$

which reduces the Jacobian norm of the system, promoting convergence[21, 23].

Direct evaluation of interactions has a complexity of $O(N^2)$ per step, which is prohibitive for realistic circuits. SCERPA implements two optimisations that reduce the computational cost while maintaining errors of $< 1\%$:

Interaction Radius (IR) Mode Taking advantage of Coulombic decay ($V \propto 1/r$), a maximum radius d_{IR} (typically 10 nm) is defined. Only molecules j with centre-to-centre distance $d^{\{i,j\}}_{cc} < d_{IR}$ are considered in the calculation (IR list). This reduces the complexity to $O(N)$.

Active Region (AR) Mode At each step, only molecules with a potential variation $> V_{AR}$ are recalculated (AR list). Inactive molecules retain their previous potentials. In clocked circuits, this technique exploits the localised nature of propagation.

The IR+AR combination reduces computation time by more than two orders of magnitude. Once convergence is achieved, the refining mode can temporarily disable the AR mode to achieve accuracy equivalent to MODE I.

Viewer

The **Viewer** module provides tools for post-simulation visualisation and analysis[24]. It imports data from output files (`.qss` files with charge distributions for each time step, and `.txt` files with additional information) and produces various visualisations:

- **3D layout plot**: three-dimensional representation of the circuit with charge distribution on each molecule, for immediate spatial analysis.
- **1D charge plot**: time evolution of the charge on the logic sites of molecules, useful for analysing propagation dynamics.
- **Logic plots**: 2D maps of the logic polarisation $P = (Q_2 + Q_3 - Q_1 - Q_4)/(Q_1 + Q_2 + Q_3 + Q_4)$, which encodes binary information ($P \approx -1$ for "0", $P \approx +1$ for "1"), for quick verification of logical correctness.
- **Electrostatic potential plots**: 2D maps of the potential at 0.2-0.5 nm above the molecular plane, analogous to STM (Scanning Tunnelling Microscopy) measurements, providing a bridge between simulation and experimental characterisation.
- **Waveform plots**: complete time diagrams of potentials and logical values of drivers, internal molecules and outputs, essential for verifying the temporal functioning of clocked devices.

2.3.2 BBchar: Block-Based Characterization

BBchar (Block-Based Characterisation) is an automated tool developed for the characterisation and simulation of molecular Field-Coupled Nanocomputing (molFCN) circuits designed in MagCAD[25]. The tool represents a fundamental solution for standardising the process of creating molecular device libraries, significantly simplifying the design flow for complex circuits. BBchar's architecture is based on a modular approach that allows complex circuits to be broken down into elementary functional blocks. The tool integrates the SCERPA algorithm for the electrostatic simulation of molecular circuits and automates three main functions:

- automatically generates SCERPA simulation input files;
- performs automatic circuit characterization to produce look-up tables with I/O behavior, timing, and area metrics;
- computes circuit outputs for arbitrary input combinations.

The BBchar operational workflow, illustrated in Figure 2.9, consists of three main phases. First, the **initialization phase** defines paths, configures driver and clock parameters, and sets up termination circuits to address border effects. Second, the **signal construction phase** builds driver voltage matrices and trapezoidal clock waveforms, optionally adding 4-cell termination circuits to stabilize output information. Third, the

simulation and characterization phase launches SCERPA for electrostatic simulation and extracts characterization data to generate library files containing LUTs[26].

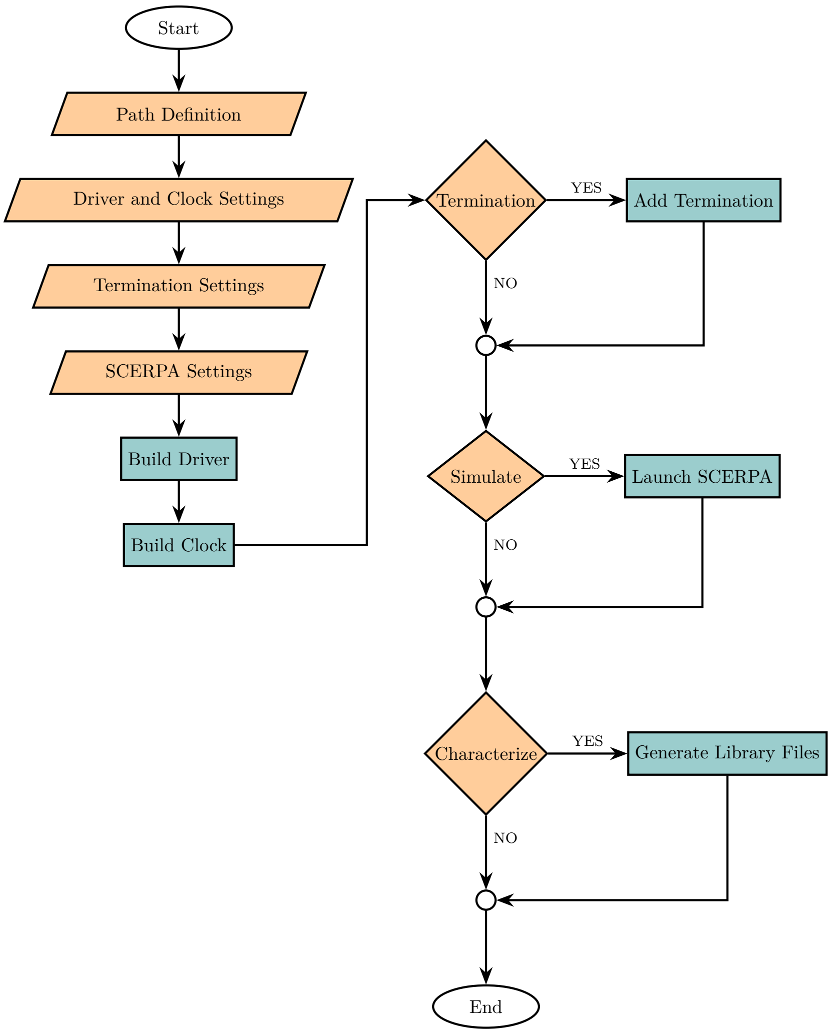
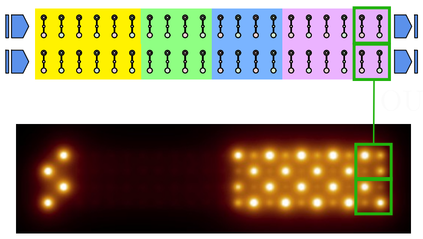


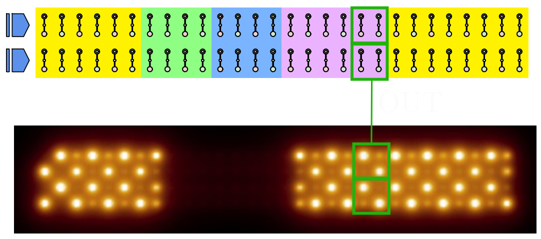
Figure 2.9: Flowchart of the BBchar program to simulate and characterize the DUT.

A critical aspect in the simulation of isolated molFCN devices concerns the stability of information at their ends. Devices may exhibit information decay at their termination, even under bistable propagation conditions. This phenomenon, known as the border effect, occurs because in an isolated device there are no molecules beyond the output that can reinforce the information, leading to a less pronounced charge separation between the dots of the final molecules.

Figure 2.10b illustrates this behaviour by comparing propagation in a single simulated wire in isolation (a) with the same structure connected to a termination block (b). In the first case, the output voltages are lower than those of the intermediate molecules due to information decay. When a second device is connected to the end of the wire, however, the information is restored thanks to the presence of the connected device.



(a) Bus without termination



(b) Bus with termination

Figure 2.10: Edge effect on information propagation: (a) Propagation in a simulated wire in isolation with SCERPA, showing data decay at the end of the wire. The graph below represents the electrostatic potential on each molecule. (b) Propagation in the same wire connected to a termination block consisting of 4 cells, showing data recovery at the output.

To accurately simulate the input-output behaviour of a device intended to be integrated as a module in a more complex design, it is essential to consider this phenomenon. When isolating and simulating each port for characterisation, it is necessary to anticipate its connection to other circuits during actual operation. Even when simulating the DUT in isolation, in steady-state operation the device will be connected to different ports that serve as termination circuits.

To account for this aspect, BBchar simulates the device in conjunction with a termination circuit that mimics the presence of a cascaded device. This approach ensures that the output voltages stored in the library are consistent with the expected behavior in a cascaded configuration, maintaining accuracy with respect to the underlying physical system. Termination circuits can take various forms. In BBchar, a short wire is used to reinforce information retention at the output cell, without introducing excessive complexity that could slow down simulations. Typically, the termination wire consists of four cells and, in the case of multi-line configurations, is replicated on both lines[26]. The termination configuration parameters are summarized in Table 2.1.

Parameter	**Description**
`enableTermination`	Flag to enable the addition of termination
`customLength`	Custom number of cells for termination (default: 4)
`busLayout`	Flag for bus layout (multi-line or single-line)

Table 2.1: Parameters for the configuration of the termination circuit in BBchar

BBchar implements trapezoidal waveforms divided into four phases (switch, hold, release, and reset), each discretized into a specific number of steps configurable by the user through the `clock_step` variable. The voltage levels are defined by the `clock_low` and `clock_high` variables. Table 2.2 summarizes the main parameters for clock configuration.

Parameter	Description	Typical value
`clock_low`	Low voltage level	-1 V
`clock_high`	High voltage level	$+1$ V
`clock_step`	Number of steps per phase	7
`NclockRegions`	Number of clock regions	4
`phasesRepetition`	Repetitions of regions	1

Table 2.2: Main parameters for clock signal configuration in BBchar

For drivers, the tool supports various operating modes, described in Table 2.3. The number of variation steps (`NsweepSteps`) determines the granularity of the characterization.

Mode	Description
`sweep`	Variation between '0' and '1' ($-V_{\max}$ to $+V_{\max}$)
`not_sweep`	Inverse variation between '1' and '0' ($+V_{\max}$ to $-V_{\max}$)
`0`	Constant logical value '0'
`1`	Constant logical value '1'

Table 2.3: Operating modes of drivers in BBchar

2.3.3 MagCAD

MagCAD is a graphical editor developed for the design of digital circuits based on Quantum-dot Cellular Automata (QCA) and, more recently, on molecular implementations [27, 28]. The tool is a fundamental component of the ToPoliNano suite, providing design capabilities through Computer-Aided Design (CAD) and Electronic Design Automation (EDA) tools specifically designed for FCN nanotechnologies.
MagCAD stands out for its simple and intuitive graphical user interface, which allows designers to create customised digital circuits using a visual approach. The tool natively supports multiple FCN technologies and can be easily extended to include new technological implementations. The building blocks for each supported technology can be inserted into the workspace using drag-and-drop functionality, greatly simplifying the design process.

The main features of MagCAD include:

- **Intuitive graphical interface**: User-friendly GUI that facilitates design even for inexperienced users;
- **Multi-technology support**: Compatibility with different FCN implementations (QCA, iNML, molFCN);
- **Multi-level layout**: Full support for 3D design with multiple layer management;
- **Hierarchical design**: Ability to reuse previously created circuits as blocks in more complex projects;

- **Project management**: Ability to save, export, and reload designs for subsequent iterations;
- **Automatic VHDL generation**: Automatic extraction of VHDL code for functional verification of the circuit;
- **Scalability**: Ability to manage a large number of elements without performance degradation;
- **Snap-to-Grid**: Automatic alignment feature to facilitate precise positioning of elements

The designer begins by selecting the target technology and graphically placing the building blocks in the workspace. As shown in Figure 2.11, the tool allows the layout to be visually divided into zones with different clock signals, facilitating the design of pipeline systems. Clock regions are represented by different colours in the interface, allowing the designer to immediately understand the temporal organisation of the circuit.

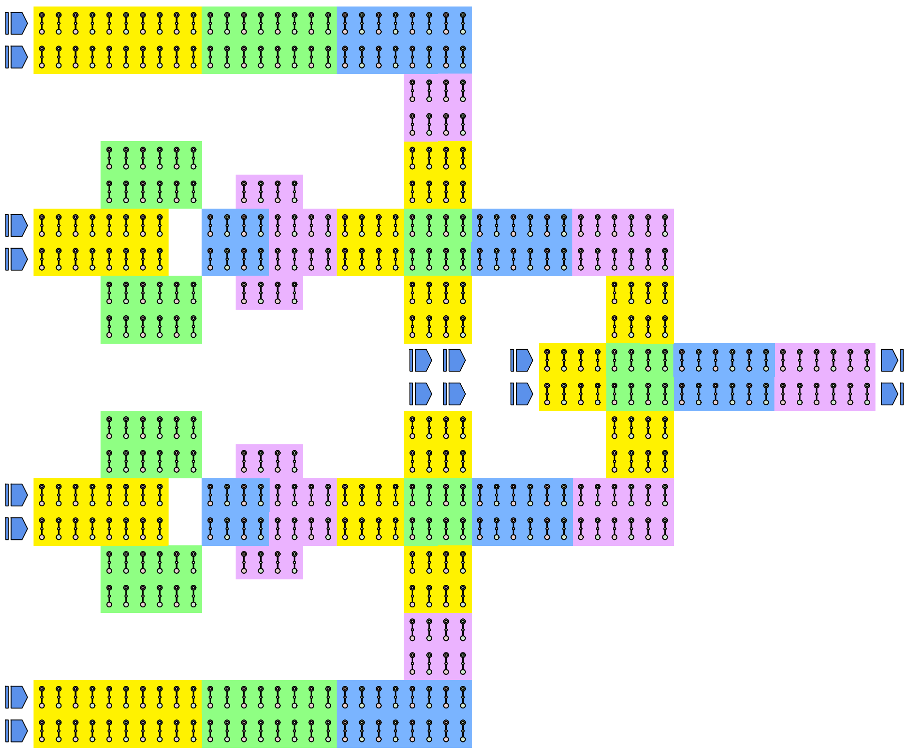


Figure 2.11: XOR layout designed in MagCAD

MagCAD is the graphical front-end of the molFCN circuit simulation ecosystem, integrating with the SCERPA and BBchar physical analysis tools. The typical workflow involves designing the layout in MagCAD, exporting it in QLL format, and then simulating it using SCERPA for physical validation of the circuit behaviour. The circuit designed graphically in MagCAD is automatically translated into the input format

required by SCERPA, preserving all information relating to molecular geometry, clock region organisation, and driver and output positioning[27]. The generated QLL file contains a complete description of the layout, including input/output pins and the precise coordinates of each element.
Despite its numerous features, MagCAD has some limitations that require attention from the user. In particular, the tool does not currently implement automatic Design Rule Check (DRC) mechanisms, leaving the designer responsible for creating circuits that comply with technological constraints.

2.4 Fiction

Fiction [30, 31] is an open source framework developed by the Chair for Design Automation at the Technical University of Munich as part of the Munich Nanotech Toolkit (MNT), specifically designed for the automation of Field-Coupled Nanocomputing (FCN) circuit design. Implemented in C++17 as a header-only library, *fiction* integrates algorithms for logic synthesis, placement, routing, clocking, verification and simulation of FCN technologies, also providing Python bindings (`mnt.pyfiction`) to facilitate its use. The signature feature of *fiction* is its technological independence: most physical design operations are performed on generic data structures that abstract from specific technologies, allowing compilation to any desired FCN technology through an extensible set of gate libraries. The command-line interface (CLI) of fiction, based on the store paradigm of alice, allows the execution of complete design flows through sequential commands.

2.4.1 Supported FCN Technologies

fiction natively supports several implementations of FCN technologies, each with specific gate libraries and output formats for external simulators. The supported technologies are:

- **Quantum-dot Cellular Automata (QCA):** QCA uses the position of electrons in quantum dots to represent binary information. The information is encoded into the cell polarization, where the position of the electrons determines the logical state. fiction supports the QCA ONE gate library [41] and provides output formats for several physical simulators: `.qca` for QCADesigner [47], `.qll` for MagCAD and SCERPA, `.fqca` for QCA-STACK, and `.svg` for visual representation.

- **in-plane Nanomagnet Logic (iNML):** iNML uses the magnetisation of nanomagnets arranged on a plane to represent logical states. The binary state is determined by the direction of magnetisation of the nanomagnets, and information propagation occurs through magnetic coupling between adjacent magnets. fiction supports the ToPoliNano gate library and provides `.qcc` and `.qll` output formats for ToPoliNano and MagCAD.

- **Silicon Dangling Bonds (SiDBs):** SiDBs represent an atomic implementation of FCN where information is encoded through dangling bonds on a hydrogen-passivated silicon surface. A SiDB cell uses only two quantum dots, forming hexagonal structures for logic gates. fiction supports the Bestagon gate library[42] and provides the `.sqd` output format for the SiQAD[29] simulator. The framework also integrates hexagonalization algorithms[40] to automatically convert Cartesian QCA layouts into SiDB-compatible hexagonal layouts.

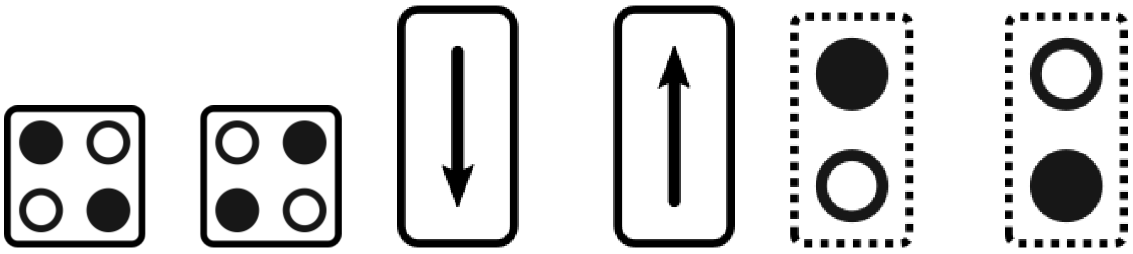


Figure 2.12: Field-Coupled Nanocomputing technologies supported by fiction: (a) Quantum-dot Cellular Automata, (b) in-plane Nanomagnet Logic, (c) Silicon Dangling Bonds

2.4.2 Workflow

The fiction framework implements a comprehensive and modular design automation flow that transforms a high-level functional specification into a physically realisable and verified FCN circuit. The workflow is organised into six main phases, described in the following sections.

Logic Synthesis

The starting point of the flow is a behavioural specification of the circuit expressed in Verilog format, AIGER or BLIF. fiction integrates established logic synthesis tools such as ABC [43] and the mockturtle library [44], which transform the functional description into an optimised logic network represented as a directed acyclic graph (DAG) or And-Inverter Graph (AIG)[32]. This intermediate representation abstracts from implementation details, facilitating subsequent optimisation and physical design stages. Input files can be generated externally or using the benchmarks included in the framework, which include standard circuits such as those from the ISCAS85 suite.

Placement & Routing

The placement and routing phase is at the heart of physical design, where the logic network is mapped onto a two-dimensional tile grid, generating the gate-level layout. fiction implements algorithms that can be classified into different categories:

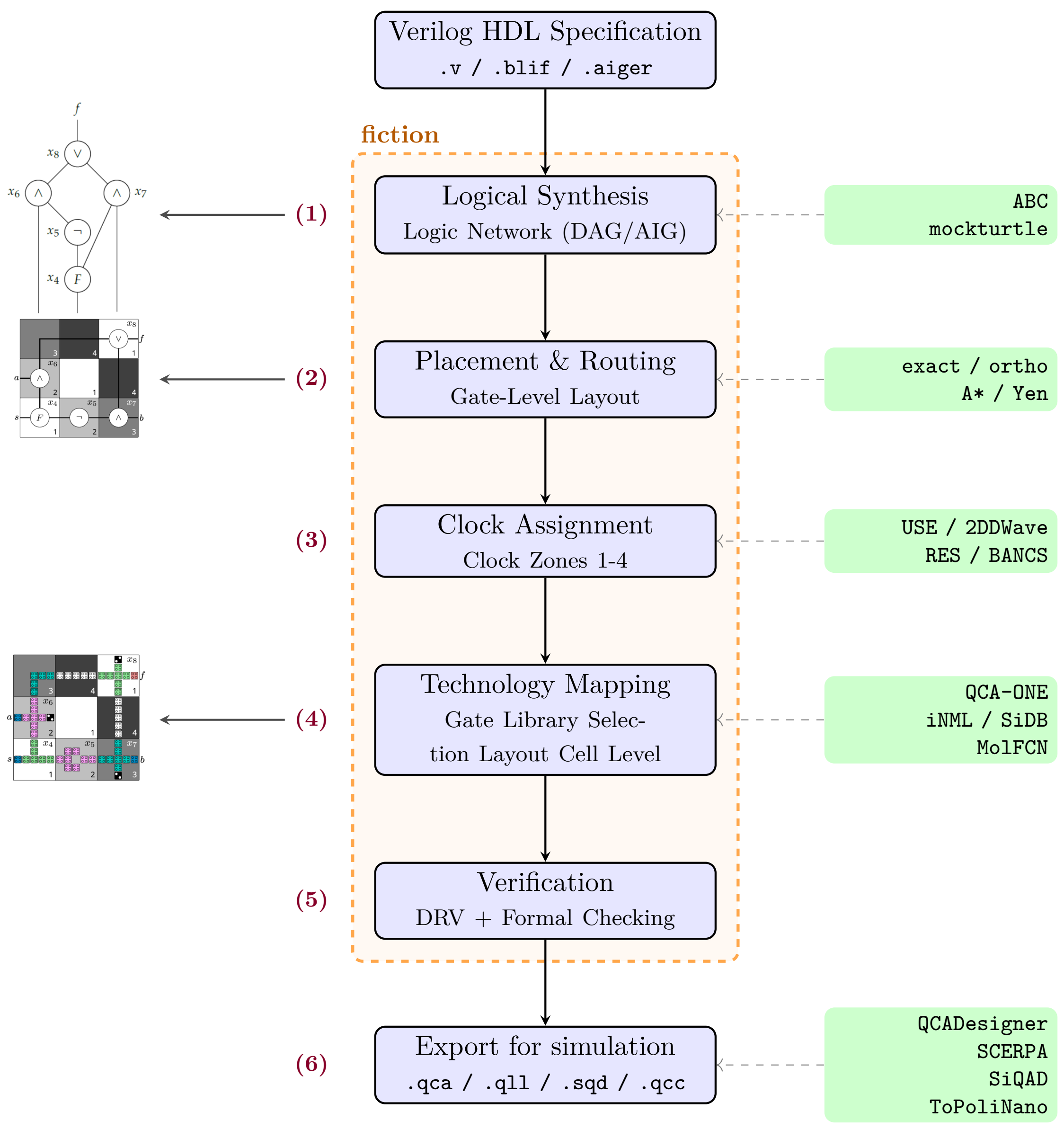


Figure 2.13: Complete design flow in fiction: from Verilog specification to the generation of the physical cell-level layout, verifiable through external simulators.

- **Exact approaches based on SMT**: The `exact` [35] algorithm uses the SMT Z3 solver to generate minimal layouts in terms of area, supporting advanced configurations such as custom clocking schemes, crossing management, and path balancing. However, computational complexity limits its applicability to relatively small circuits (typically up to 20–30 gates).

- **Scalable heuristic approaches**: the `ortho` [36] algorithm based on Orthogonal Graph Drawing (OGD) is scalable to realistic circuit sizes (up to 40,000 gates in a few seconds). While not guaranteeing optimality, ortho produces high-quality solutions in practical time, making it suitable for industrial applications.

- **One-pass synthesis**: Combines logic synthesis and physical design in a single SAT-based step, eliminating intermediate representation and directly generating minimal layouts while simultaneously considering logical and physical constraints [45].
- **Multi-path routing**: SAT-based approach for routing with multiple paths[46].
- **Graph-oriented layout design**: A* algorithm for topological optimisation [37].

Complementary path-finding algorithms, such as A* for the shortest path, support optimised routing of connections between gates. The result of this phase is a technology independent gate-level layout, where each tile in the grid contains a logic gate, a wire segment, an I/O element, or remains empty. This level of abstraction allows you to think about the design without prematurely committing to a specific FCN technology.

Clocking Schemes

Clocking is fundamental to the correct functioning of FCN circuits, as it controls the temporal propagation of signals through sequential clock domains [6]. Fiction supports a wide range of regular clocking schemes documented in the literature, including Columnar and Row, 2DDWave, USE, RES, ESR, CFE, Ripple, SRS, and BANCS. Each scheme defines a clock zone pattern that repeats on the grid, ensuring the correct sequencing of logical operations.

In addition to regular predefined schemes, fiction supports irregular clocking (open clocking), where clock numbers are dynamically assigned by placement algorithms based on circuit-specific constraints [38]. This flexibility allows clocking to be adapted to the characteristics of the specific implementation, potentially improving overall performance. The clocking assignment phase also integrates post-layout optimisation techniques that can reduce wire segments by up to 73% and overall area by up to 34%, with significant improvements on the critical path.

Technology Mapping

Technology mapping represents the crucial transition from gate-level abstraction to cell-level layout, i.e. from logical abstraction to the specific physical implementation of the chosen FCN technology. This phase uses gate libraries that define the physical implementation of each logical gate in terms of nanotechnological cells. fiction supports gate libraries for three FCN technologies[34]:

- **QCA**: the QCA-ONE library provides optimised implementations for QCA cells. Each logic gate is typically expanded into 10–20 QCA cells arranged according to specific patterns.
- **iNML**: the ToPoliNano library provides implementations for planar nanomagnets.

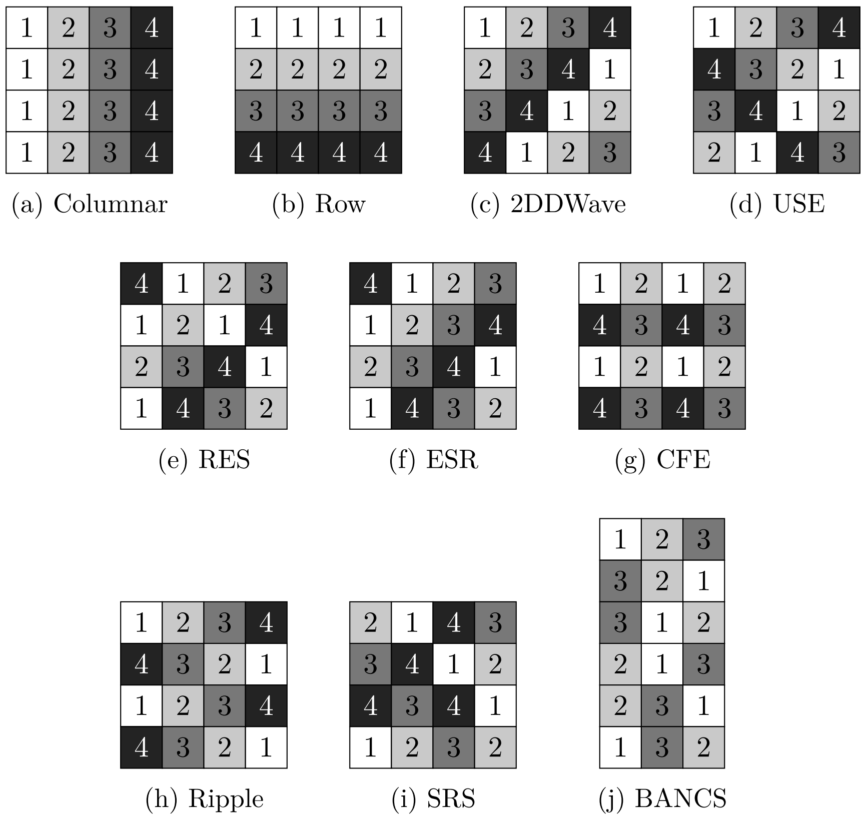


Figure 2.14: Regular clocking patterns integrated into fiction

- **SiDB (Silicon Dangling Bonds)**: the Bestagon library uses optimised hexagonal patterns.

During technology mapping, each tile of the gate-level layout containing a logical element is replaced by its specific physical implementation, significantly expanding the number of elements: a circuit of a few dozen gates can translate into hundreds or thousands of physical cells. The resulting cell-level layout contains the precise coordinates of each cell, its physical properties (polarisation, magnetisation or charge) and physical clocking zones.

Verification

Verification ensures that the generated layout is functionally correct with respect to the original specification. *fiction* implements two complementary methodologies [30]:

- **Design Rule Violation (DRV) checking**: Verify compliance with physical and technological constraints, such as minimum distances between cells, maximum wire lengths, and technology specific manufacturing constraints.
- **SAT-based formal verification**: It uses equivalence checking techniques to formally demonstrate that the layout implements the same logical function as the

original specification, taking into account the synchronisation constraints specific to FCN technologies. The approach handles layouts with millions of tiles in a reasonable amount of time (minutes), providing mathematical guarantees on the correctness of the design.

For layouts compiled at the cell level, *fiction* integrates technology-specific physical simulators that enable functional validation under realistic physical models, including thermal effects and manufacturing tolerances.

Export and Simulation

The final stage of the workflow prepares the verified layout for physical simulation or manufacturing. *fiction* supports export to multiple physical simulators, each specialised for specific FCN technologies:

- **QCA**: `.qca` format for QCADesigner, `.qll` for MagCAD and SCERPA, `.fqca` for QCA-STACK, as well as `.svg` for visualisation.
- **iNML**: `.qcc` and `.qll` formats for ToPoliNano and MagCAD.
- **SiDB**: `.sqd` format for SiQAD, as well as physical simulation capabilities integrated directly into fiction through algorithms such as QuickExact and QuickSim.

2.4.3 The need for a Standard-Cell Library for MolFCN

Although *fiction* is the most mature and comprehensive design automation tool for Field-Coupled Nanocomputing technologies, currently supporting complete and validated gate libraries for QCA (QCA-ONE), iNML (ToPoliNano) and SiDB (Bestagon), the framework does not have any libraries for Molecular Field-Coupled Nanocomputing (MolFCN) technology. This absence constitutes a critical gap that prevents the integration of MolFCN into the complete design automation flow offered by *fiction*: while the logic synthesis, placement, routing, and clocking algorithms work correctly, generating technology-independent gate-level layouts, the crucial phase of technology mapping is impossible due to the lack of standardised molecular implementations [30, 50]. The generated layouts therefore, remain unusable abstractions, impossible to convert into cell-level molecular layouts for export to physical simulators such as SCERPA for validation.
In order to create a standard library for MolFCN, each device must be characterised considering not only its logical functionality but also its underlying molecular physics, including electrostatic, thermal, and quantum effects. This characterisation requires computationally intensive simulations using SCERPA [21], which must be integrated with standard EDA flows.

There is therefore a need to design the first complete standard-cell library for MolFCN and physically validate it using SCERPA simulation to integrate it into the *fiction* framework to enable automatic technology mapping from abstract gate-level layouts to molecular cell-level layouts. The library would provide a standardised basis for comparative research, definitively bridging the gap between molecular simulation and design automation and completing the support of fiction for all major FCN technologies. This finally allows fair and reproducible technological comparisons between QCA, iNML, SiDB and MolFCN using the same benchmark circuits and physical design algorithms, taking MolFCN from a purely theoretical technology to a practical design platform for the scientific community.

Chapter 3

Development of the molFCN device library

This chapter presents the development and validation of a comprehensive molFCN device library validated through SCERPA simulations. All devices are characterized using a standardized ideal molecule model (designated "62" in the SCERPA database). This approach enables systematic and reproducible characterization across the entire device inventory.
The chapter is organized as follows. Section 3.1 presents the functional classification of devices into interconnection elements, basic logic gates, and complex logic functions. Section 3.2 describes the standardized simulation methodology implemented using MATLAB scripts interfacing with SCERPA. Section 3.3 presents the validation results, demonstrating functional correctness for all 24 devices and establishing the foundation for the Fiction-compatible SIM(7) library development in Chapter 4.

3.1 Analysis and simulation of current devices

The analysis and simulation phase of current devices represents the first critical step in the validation and migration process towards the ideal molecule. The initial inventory included 24 molFCN devices of varying complexity, each originally designed for the bisferrocene molecule and now to be validated with the new molecular model. This phase required the development of a systematic simulation methodology that could guarantee both the physical accuracy and computational efficiency necessary for large-scale characterisation.

3.1.1 Classification of devices

The initial library has been organised according to a functional taxonomy that reflects the different requirements of molFCN circuit design:

1. **Interconnection devices**: This category includes the fundamental elements for routing information.

- **Basic wires**: bus, hwire_xnor
- **L-shaped wires**: Lwire_dxdw, Lwire_dxdw_xnor, Lwire_dxup, Lwire_short_dxdw, Lwire_short_dxup.
- **T-shaped wires**: T_connection, T_dxdw, T_dxup, T_updw.

2. **Basic logic devices**: this category includes fundamental logic elements.
   - **Inverter**: inv, invDr, inv_xor2nand.
   - **Majority Voters**: MV_xor2nand, MVlongdw, Mvlongdx, Mvlongup.
   - **Logic gates**: nand_nor, nand_nor_big.
3. **Complex logic devices**: this category includes devices of greater complexity.
   - **Multiplexer**: mux21
   - **XOR/XNOR Gates**: Xor, Xor_2, Xnor

The layouts of all previously described devices are presented in Tables 3.1, 3.2, and 3.3.

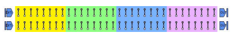
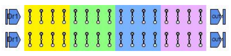
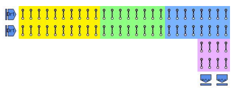
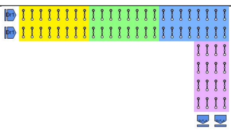
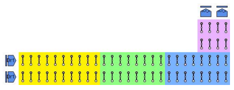
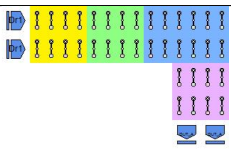
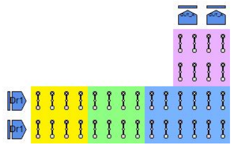
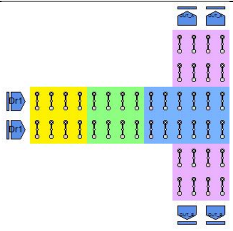
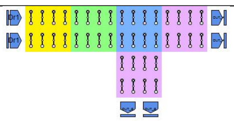
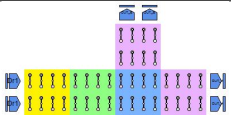
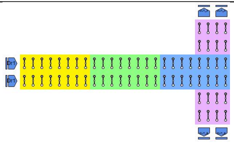
Device	**Layout**	**Device**	**Layout**
bus		hwire_xnor	
Lwire_dxdw		Lwire_dxdw_xnor	
Lwire_dxup		Lwire_short_dxdw	
Lwire_short_dxup		T_connection	
T_dxdw		T_dxup	
T_updw			

Table 3.1: Interconnection devices

3.2 Simulation Methodology

To ensure the reproducibility and accuracy of the simulations, a standardised test framework was developed and implemented using a MATLAB script in the appendix C that interfaces directly with SCERPA. This methodology allows for a complete and systematic characterisation of each device.
The clock system implemented follows the standard four-phase scheme, with the following main parameters:

- **Clock range**: $V_{clock,low} = -1$ V, $V_{clock,high} = +1$ V

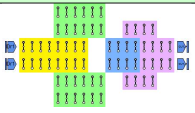
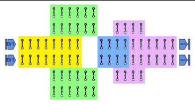
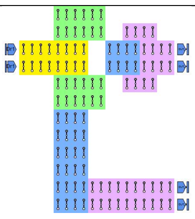
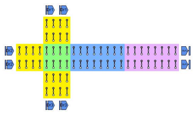
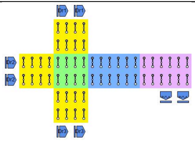
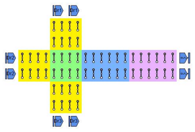
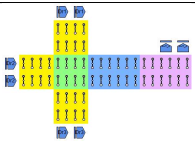
Device	Layout	Device	Layout
inv		inv_xor2nand	
invDr		MV_xor2nand	
MVlongdw		MVlongdx	
MVlongup			

Table 3.2: Basic logic devices

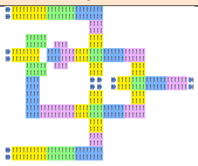
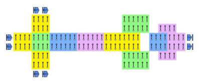
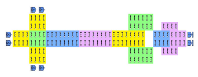
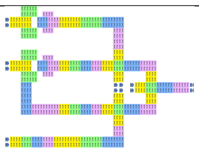
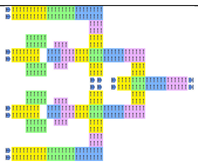
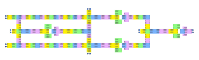
Device	Layout	Device	Layout
mux21		nand_nor	
nand_nor_big		Xnor	
Xor		Xor_2	

Table 3.3: Complex logic devices

- **Clock step**: $N_{step} = 7$ steps per phase

As regards the circuit configuration, the following geometric and technological parameters are set:

- **circuit.dist_z = 10** nm: vertical distance between molecules and substrate
- **circuit.molecule = "62"**: selection of the ideal molecule from the SCERPA database
- **circuit.magcadImporter = 1**: activation of layout import from QLL file generated with MagCAD
- **circuit.doubleMolDriverMode = 1**: modelling of drivers as a pair of molecules to realistically represent electrostatic coupling

Finally, the parameters for generating graphs are configured, allowing for detailed visual verification of switching phenomena:

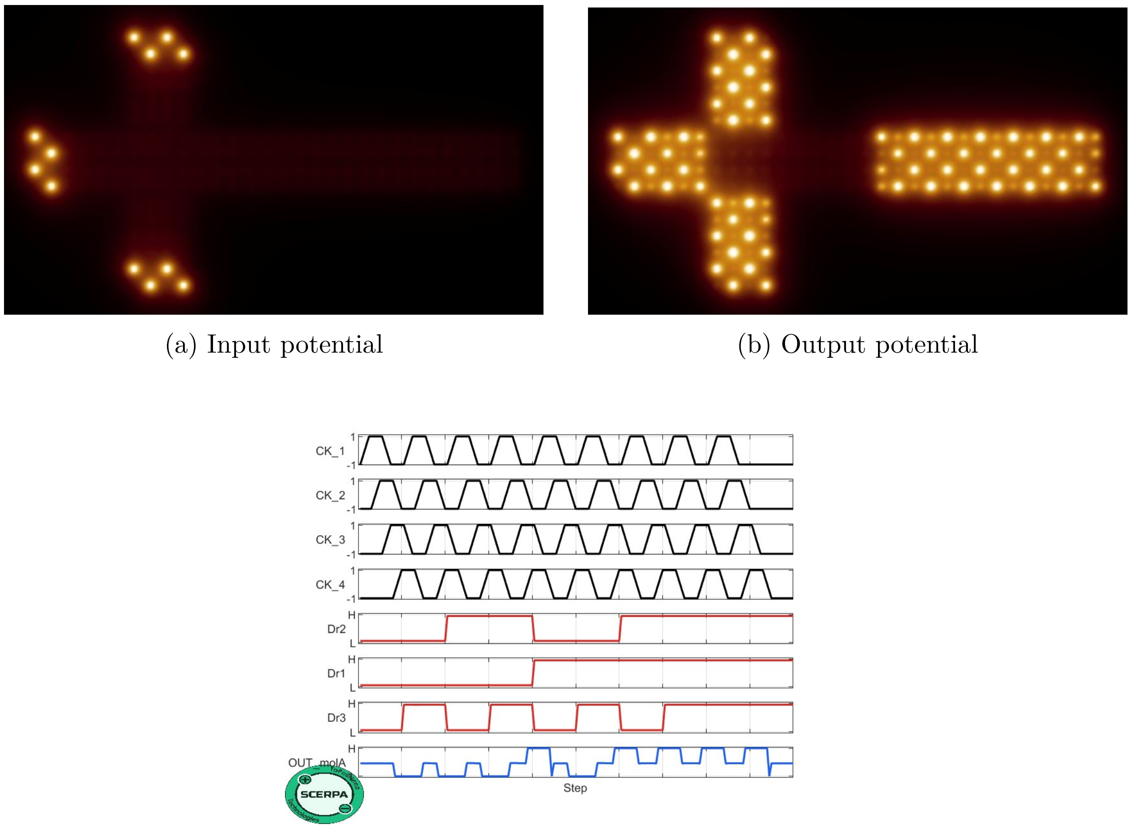
- **plot_logic = 1**: generation of the logical propagation sequence
- **plot_potential = 1**: production of 2D maps of the electrostatic potential
- **plot_waveform = 1**: acquisition of temporal waveforms on molecules
- **plot_waveform_index = 1**: enabling interactive selection of molecules to be analysed

3.3 Simulation Results

The SCERPA simulations yielded positive results, demonstrating the complete success of the transition to the ideal molecule. All 24 devices in the initial library passed all functional tests, demonstrating logical correctness, robust propagation, and effective synchronisation, confirming the validity of the adopted methodological approach.

The validation process was conducted through a comprehensive analysis combining multiple verification methodologies. For each device, both waveform characterizations and electrostatic potential plots were generated across all possible input combinations. The waveforms confirmed correct temporal behaviour, showing logic transitions during the appropriate clock phases, and maintaining stable output values during hold periods. The potential plots provided direct visualisation of the charge distribution patterns, validating that the electrostatic coupling between molecules correctly implements the intended logic functions.

Figure 3.1 illustrates representative validation results obtained for the entire device library. As an example of the waveform analysis performed on all 24 devices, Figure 3.1c shows the characterization of the MVlongdx, demonstrating correct majority function implementation across all eight input combinations with clean transitions and stable output levels. Similarly, as representative examples of the potential plot analysis conducted for the complete library, Figure 3.1a presents the input configuration and Figure 3.1b shows the corresponding output response for the MVlongdx, clearly visualizing the charge distribution through the molecular chain. These validation methodologies were systematically applied to every device in the library, confirming functional correctness across all interconnection elements, basic logic gates, and complex logic functions.



(a) Input potential

(b) Output potential

(c) Waveform

Figure 3.1: Representative validation results from SCERPA simulations

Three key aspects emerged as critical indicators of successful device operation. First, logical correctness: every device produced outputs matching the expected truth tables across all input combinations. Second, signal propagation robustness: the charge polarization patterns propagated reliably through the molecular chains, maintaining sufficient signal strength even in longer or more complex structures. Third, clock synchronization effectiveness: the four-phase clocking scheme successfully controlled information flow, ensuring proper sequencing of logic operations.

Table 3.4 shows the complete parameters of all validated devices, including geometric data, number of cells and molecules, and physical dimensions. Each device was characterized considering both the base area and the area with overhead necessary for isolation and routing. The successful validation of this extended library established a foundation for the selection and standardization process described in Chapter 4, where a reduced subset of devices would be adapted to meet the strict requirements of the Fiction framework for automated circuit synthesis.

Device name	**Number of inputs**	**Occupied area [nm²]**	**Occupied area with overhead [nm²]**	**Number of cells**	**Number of molecules**	**Number of outputs**	**Dimension [x,y,z]**
INTERCONNECTION DEVICES							
bus	4	136.00	128.00	32	64	4	[2, 32, 2]
hwire_xnor	4	72.00	64.00	16	32	4	[2, 16, 2]
Lwire_dxdw	4	128.00	208.00	30	60	4	[4, 26, 2]
Lwire_dxdw_xnor	4	136.00	288.00	32	64	4	[6, 24, 2]
Lwire_dxup	4	128.00	208.00	30	60	4	[4, 26, 2]
Lwire_short_dxdw	4	80.00	112.00	18	36	4	[4, 14, 2]
Lwire_short_dxup	4	80.00	112.00	18	36	4	[4, 14, 2]
T_connection	4	96.00	168.00	22	44	8	[6, 14, 2]
T_dxdw	4	88.00	128.00	20	40	8	[4, 16, 2]
T_dxup	4	88.00	128.00	20	40	8	[4, 16, 2]
T_updw	4	136.00	288.00	32	64	8	[6, 24, 2]
BASIC LOGIC DEVICES							
inv	4	136.00	216.00	32	64	4	[6, 18, 2]
inv_xor2nand	4	144.00	240.00	34	68	4	[6, 20, 2]
invDr	4	224.00	432.00	54	108	8	[12, 18, 2]
MV_xor2nand	12	152.00	288.00	32	64	4	[6, 24, 2]
MVlongdw	12	136.00	240.00	28	56	4	[6, 20, 2]
Mvlongdx	12	136.00	240.00	28	56	4	[6, 20, 2]
Mvlongup	12	136.00	240.00	28	56	4	[6, 20, 2]
nand_nor	12	264.00	456.00	60	120	4	[6, 38, 2]
nand_nor_big	12	280.00	504.00	64	128	4	[6, 42, 2]
COMPLEX LOGIC DEVICES							
mux21	24	840.00	2200.00	198	396	4	[22, 50, 2]
Xnor	24	1168.00	3432.00	280	560	4	[26, 66, 2]
Xor	28	888.00	2200.00	208	416	4	[22, 50, 2]
Xor_2	24	1920.00	5688.00	468	936	4	[18, 158, 2]

Table 3.4: Complete Library of Validated molFCN Devices

Note: Intermolecular distances considered: 1 nm in the x-direction and 2 nm in the y-direction.

Chapter 4

Fiction framework alignment

4.1 Fiction Requirements

The integration of a molFCN standard cell library into the Fiction framework requires compliance with specific technical requirements that ensure compatibility with the physical design algorithms implemented in the tool. These requirements concern geometric aspects, clocking, synchronization, and interfacing with the framework's data structures[38, 30].

4.1.1 Geometric constraints

Fiction imposes strict geometric constraints to ensure compatibility with placement and routing algorithms. Each standard cell must be implemented on a uniform grid with fixed dimensions, defined a priori for the entire library. This uniformity is essential to allow placement algorithms to operate on a discrete and predictable search space[38]. The cell size must be chosen considering a trade-off between occupied area and sufficient space for the physical implementation of logic devices and clock zones. Excessive dimensions lead to area waste, while insufficient dimensions can cause electrostatic crosstalk problems between adjacent cells or the impossibility of implementing the required logic. Furthermore, a fundamental requirement concerns the standardization of input and output pin positions. All cells in the library must share identical dedicated positions for pins, ensuring alignment with the global layout grid. This constraint enables direct composition of cells, where the output of one cell can be connected to the input of another without the need for additional routing.

4.1.2 Clocking System and Global Synchronization

Fiction requires that each standard cell be subdivided into clock zones to ensure correct directional signal flow and appropriate pipelining. The clocking scheme must be compatible with the mechanisms supported by the framework. Each cell must implement the 4 internal clock zones following a predetermined sequence. When cells are connected in series, the framework automatically guarantees local synchronization: each new cell starts with clock zone 1, allowing orderly signal propagation through the circuit. This

convention, known as *local synchronization*, must be strictly respected during cell design to avoid timing violations. Typically, the zones are organized in sequence from the input zone (clock 1) to the output zone (clock 4), passing through the intermediate zones necessary for signal processing.
The *global synchronization constraint* represents one of the most critical requirements imposed by Fiction[50]. This constraint establishes that all signals converging on a tile must arrive simultaneously. Compliance with this constraint is essential to ensure correct operation of multi-input logic devices. Violation of this constraint can cause malfunctions due to race conditions, where signals that should be processed together arrive at different times, producing incorrect results or metastable states. To ensure compliance with this constraint, standard cells must be designed considering internal propagation delays and providing precise timing information to placement and routing tools. In complex circuits, compliance with the global synchronization constraint often requires the insertion of delay elements (typically wires) on shorter paths to equalize signal arrival times. Although Fiction automatically handles these aspects during placement and routing through algorithms that solve the combined placement, routing, and clocking problem, cell design must facilitate this management through predictable and well-characterized delays.

4.1.3 Compatibility with Fiction Data Structures

The tiles must be representable through Fiction's native data structures.

This requires that each cell can be described through structured metadata specifying:

- Number and exact position of input and output pins
- Mapping between pins and grid coordinates
- Implemented logic function
- Internal subdivision into clock zones
- Occupied dimensions in terms of tiles

The representation format must allow technology mapping, i.e., the conversion from the gate-level abstraction to the physical cell-based representation [30]. This process uses library objects that manage the mapping between abstract logic elements and concrete physical cells, requiring a complete and unambiguous description of each cell.

4.2 Development of the SIM(7) Library

This section describes the development process of the SIM(7) standard cell library, starting from the selection of the fundamental devices to their physical validation using SCERPA simulations. The library is the first physically simulated standard-cell library for molFCN, designed to be fully compatible with the Fiction framework[49].

4.2.1 Selecting devices from the extended library

The previous chapter presented an extensive library of molFCN devices characterised using SCERPA simulations. This library includes various logic and interconnection devices (Table 3.4). The transition from this extensive library to the reduced SIM(7) library was guided by the following criteria:

1. **Functional completeness**: Selected devices must guarantee universal logical completeness, allowing the synthesis of any Boolean function
2. **Compatibility with Fiction**: Devices must strictly comply with the geometric and clocking constraints imposed by the framework
3. **Implementation complexity**: Preference for devices with simple and compact layouts that minimise the area occupied
4. **Physical robustness**: Selection of devices with stable and predictable electrostatic behaviour, validated using SCERPA
5. **Minimal set efficiency**: Reduction in the number of cells to simplify the technology mapping process and accelerate the exploration of the design space

Based on these criteria, seven fundamental devices were selected which, including essential variants, result in a total of 14 standard cells in the SIM library(7). Table 4.1 summarises all the devices implemented with their main characteristics.

Type	**Cell Name**	**# Inputs**	**# Outputs**
Wire	BUS	4	4
Inverter	INV	4	4
L-wire	L_DXDW	4	4
	L_DXUP	4	4
Fan-out	T_DXDW	4	8
	T_DXUP	4	8
	T_UPDW	4	8
Majority Voter	MV	12	4
AND	AND_UP	12	4
	AND_DW	12	4
	AND_LH	12	4
OR	OR_UP	12	4
	OR_DW	12	4
	OR_LH	12	4

Table 4.1: Devices of the library

4.2.2 Tile Design

In order to comply with the criteria described above, standard SIM(7) cells were implemented on a uniform grid of 10 × 10 molFCN cells. This size represents an optimal compromise between several conflicting factors:

- **Area occupied**: A smaller grid (e.g., 6×6 or 8×8) would reduce the available space, making it impossible to implement complex logic devices and the four clock zones required with adequate physical separation.
- **Electrostatic crosstalk**: A larger grid would reduce interference between adjacent cells due to the greater physical distance, but would result in wasted area and reduced integration density. With the 10×10 grid, the distance between molecules is sufficient to minimise unwanted parasitic couplings.
- **Clocking compatibility**: The 10×10 size allows sufficient distinct clock zones to be allocated.
- **Internal routing**: The space inside the cell must allow signals to be routed between inputs and outputs through intermediate clock areas, avoiding overly convoluted paths that would cause signal loss.

All SIM(7) tiles adopt a standardised I/O pin layout that is essential for compatibility with Fiction. Specifically, all input/output pins are aligned vertically and horizontally to the global layout grid, allowing direct connection without additional routing when cells are placed adjacent to each other. This standardisation is essential for Fiction, as it ensures that the output of one cell can be connected directly to the input of the next cell without the need for additional interconnect elements or geometric realignment. This approach drastically reduces the complexity of the routing problem and minimizes area overhead. Furthermore, each standard tile is internally divided into 4 clock zones, highlighted by different colours, which follow the clocking scheme compatible with Fiction[50]. Therefore, when two cells are connected in series, Fiction automatically ensures that clock zone 1 of the second cell is synchronised with clock zone 4 of the first, respecting the local synchronisation constraint.

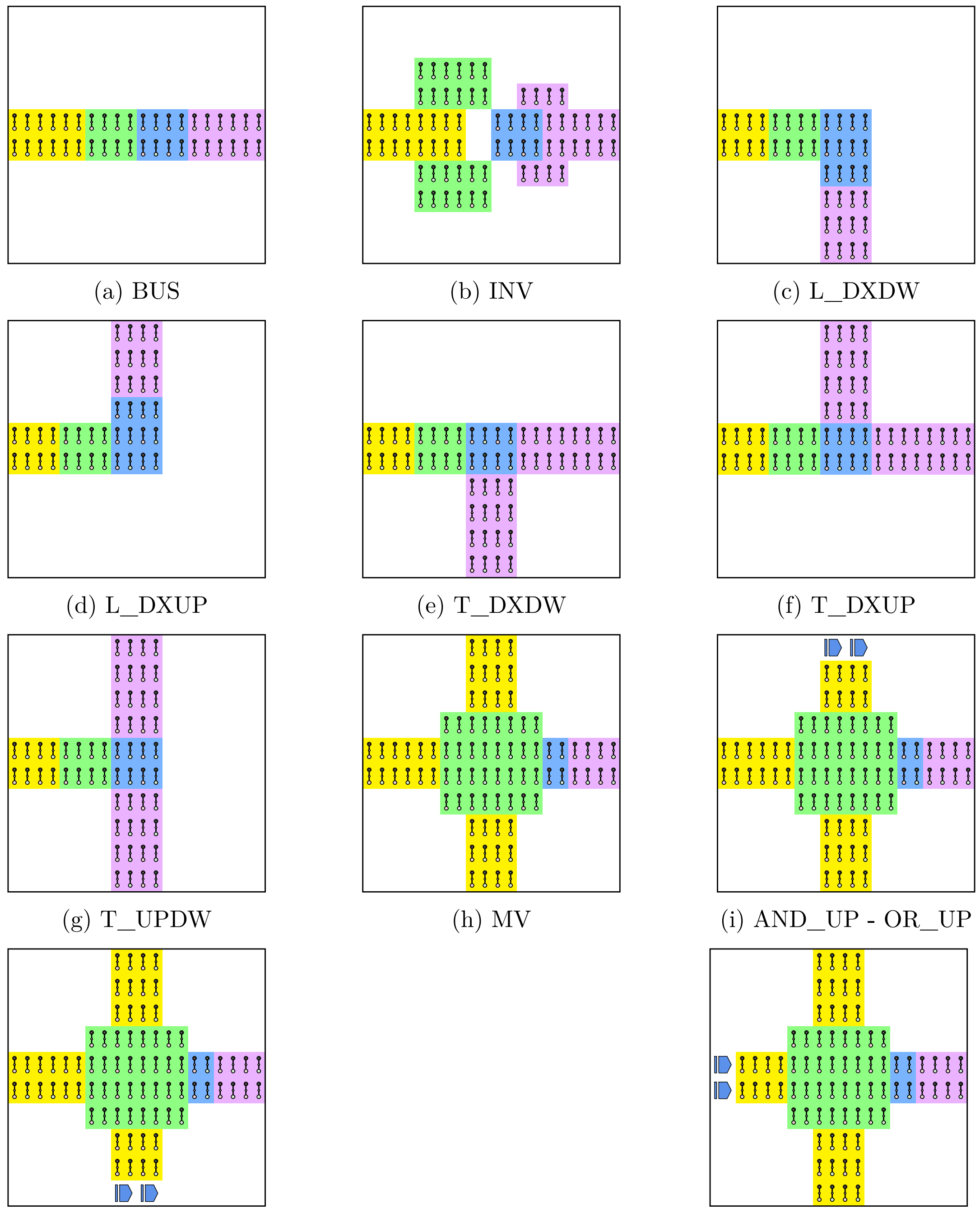
Figure 4.1 shows the complete physical layouts of all 14 devices.

4.2.3 Orientational Variants

For each device, all orientations have been implemented, allowing the device to be adapted to different routing configurations without the need for complex interconnection elements. The layouts and waveplots of all orientations are shown in the appendixes A and B.

Fiction greatly benefits from the availability of **orientational variants** of standard cells[50], offering significant advantages during the physical design process. In particular:

- **Reduction in interconnection length**: Ability to choose the optimal orientation to minimise the number of wires required
- **Improved planarity**: Facilitates the maintenance of planar layouts by avoiding crossover
- **Area optimisation**: Reduces dead space due to complex routing
- **Placement flexibility**: Increases the search space for placement algorithms, improving the quality of solutions



(a) BUS (b) INV (c) L_DXDW

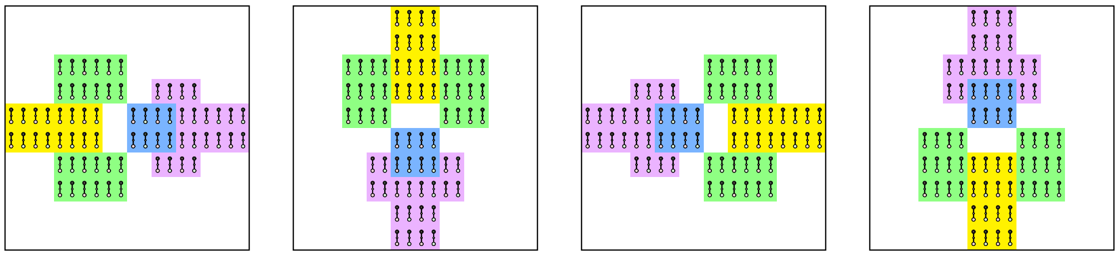
(d) L_DXUP (e) T_DXDW (f) T_DXUP

(g) T_UPDW (h) MV (i) AND_UP - OR_UP

(j) AND_DW - OR_DW (k) AND_LH - OR_LH

Figure 4.1: Physical layouts of the devices in the library SIM(7)

All orientation variants retain the same logical function and equivalent electrical properties as the original device.



(a) INV_UP (b) INV_DW (c) INV_LH (d) INV_RH

Figure 4.2: The four orientation variants of the double-branch inverter: (a) INV_UP - 0° rotation, (b) INV_DW - 90° rotation, (c) INV_LH - 180° rotation, (d) INV_RH - 270° rotation

4.3 Simulation and validation

Each device in the SIM(7) library has been validated through comprehensive physical simulations using BBchar. This validation step is crucial to ensure that the designed devices function correctly at the molecular level.

4.3.1 Simulation Methodology

Simulations of devices in the SIM(7) library were conducted using standardised parameters to ensure consistency and reproducibility of results. Appendix D provides the Matlab script used. The clock signal was configured with extreme values of -1 V and $+1$ V, discretised into 7 time steps for each phase (`clock_step = 7`). Each complete clock cycle consists of four sequential phases: switch (transition from low clock to high clock), hold (maintaining the high clock), release (transition from high clock to low clock), and reset (maintaining the low clock). This temporal structure, implemented using the BBchar framework [25], allows for accurate simulation of the adiabatic propagation of the signal through the four clock zones of each device.

The input drivers were configured in double molecule mode (`doubleMolDriver = 1`) to ensure signal robustness. For complete characterisation of the devices, all logical input combinations were simulated. The generation of voltage values for the driver sweep follows a linear distribution with a configurable number of steps (`NsweepSteps`), typically set to 10 for complete characterisation or to 1 for quick functional checks.

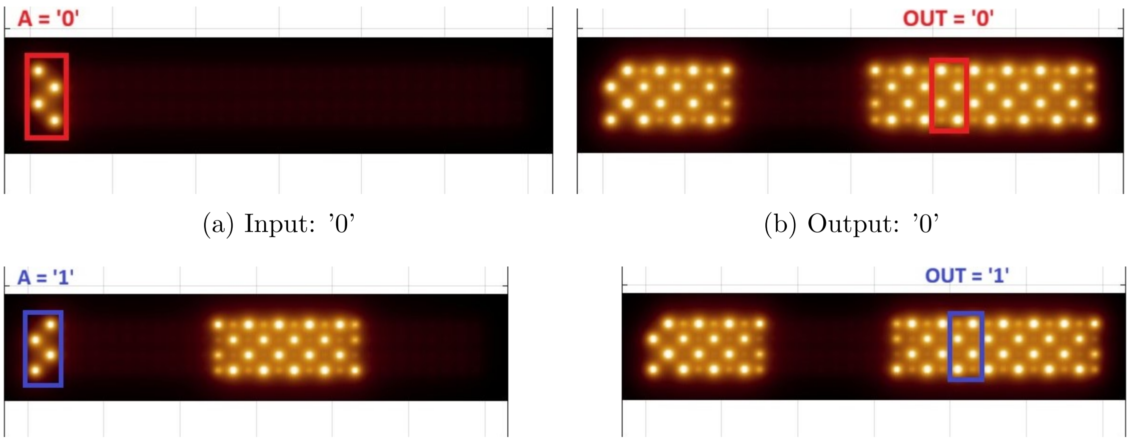
Each layout has been extended with a bistable termination to prevent edge effects and ensure information stability. The termination, implemented using 2-line bus structures (`busLayout = 1`), uses a configurable number of cells to ensure output bistability, typically 4. The SCERPA algorithm was run with a damping factor of 0.6 to ensure stable convergence, enabling the dumping of driver, clock, and output waveforms to allow detailed analysis of the temporal behaviour of each device.

4.3.2 Results

The simulation results confirmed the validity of the SIM(7) library, demonstrating 100% functional correctness for all devices across all input combinations. This section presents the comprehensive validation results for the 14 base cells through both waveform and potential plot characterizations. For each device, the potential plots are organized in pairs: one showing the input configuration and the corresponding one showing the output response. This input-output pairing allows direct visual verification of the logical function implemented by each device.

BUS Figure 4.3 presents the potential plot analysis:

- **Input '0'** (fig. 4.3a) → **Output '0'** (4.3b): The charge distribution characteristic of logic '0' propagates faithfully through the entire bus structure.
- **Input '1'** (fig. 4.3c) → **Output '1'** (4.3d): The logic '1' polarization pattern is correctly transmitted.



(a) Input: '0' (b) Output: '0'

(c) Input: '1' (d) Output: '1'

Figure 4.3: Bus potential plot

Figure 4.4 shows the waveform characterization, demonstrating proper signal propagation synchronized with the four clock phases (CLK_1 through CLK_4). The driver signal (Dr1) clearly shows the input transitions, while the output waveform (OUT_molA) faithfully reproduces the input after the appropriate propagation delay determined by the clock zones.

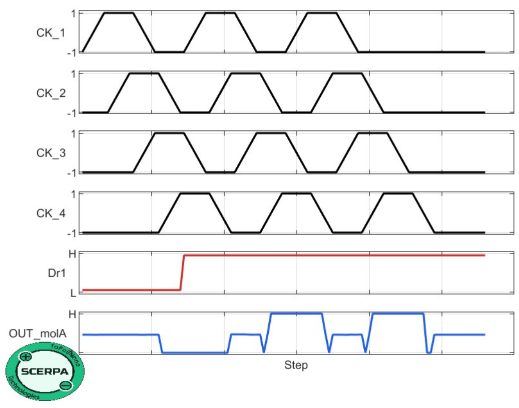
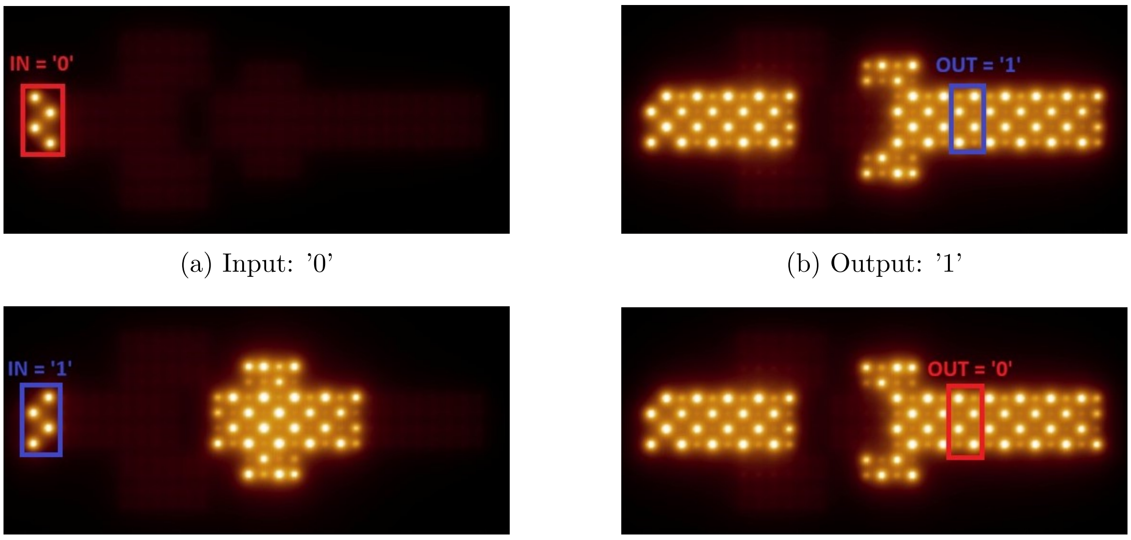


Figure 4.4: Bus waveplot

Inverter (INV) Figure 4.5 demonstrates input inversion through potential plots:

- **Input '0'** (fig. 4.5a) → **Output '1'**(fig. 4.3d): The input is correctly inverted.
- **Input '1'** (fig. 4.3c) → **Output '0'** (fig. 4.3b): Correct logical inversion confirmed by the complementary charge distribution pattern.



(a) Input: '0'

(b) Output: '1'

(c) Input: '1'

(d) Output: '0'

Figure 4.5: Inverter potential plot

Figure 4.6 presents the waveform analysis, showing **clear anti-phase behavior** between input and output signals.

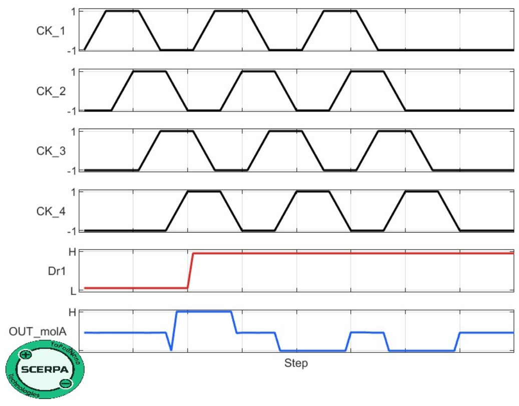
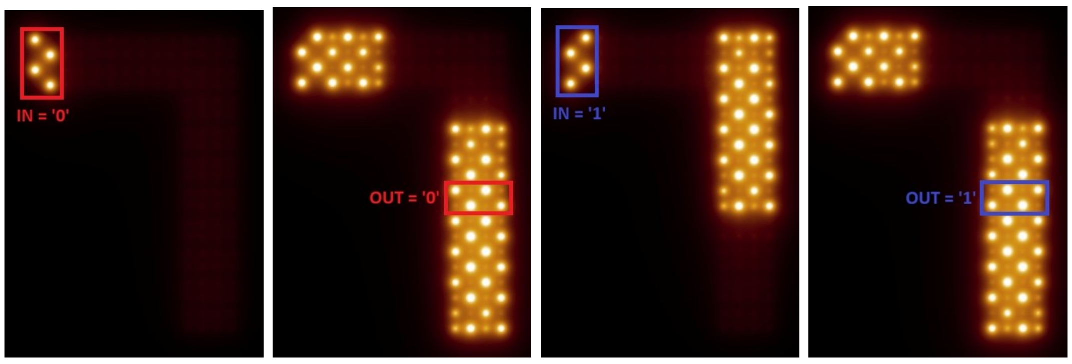


Figure 4.6: Inverter waveplot

L-Wires (L_DXDW and L_DXUP) Figures 4.7 and 4.8 show the potential plot characterization for both L-wire variants:
L_DXDW

- **Input '0'** (fig. 4.7a) → **Output '0'** (fig. 4.7b): Logic '0' successfully propagates through the 90-degree turn
- **Input '1'** (fig. 4.7c) → **Output '1'** (fig. 4.7d): Logic '1' faithfully transmitted through the corner

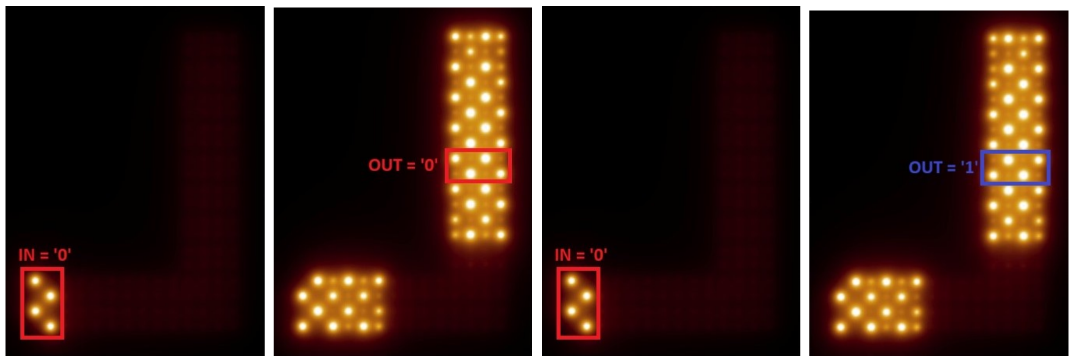


(a) Input '0' (b) Output '0' (c) Input '1' (d) Output '1'

Figure 4.7: L_DXDW potential plot

L_DXUP

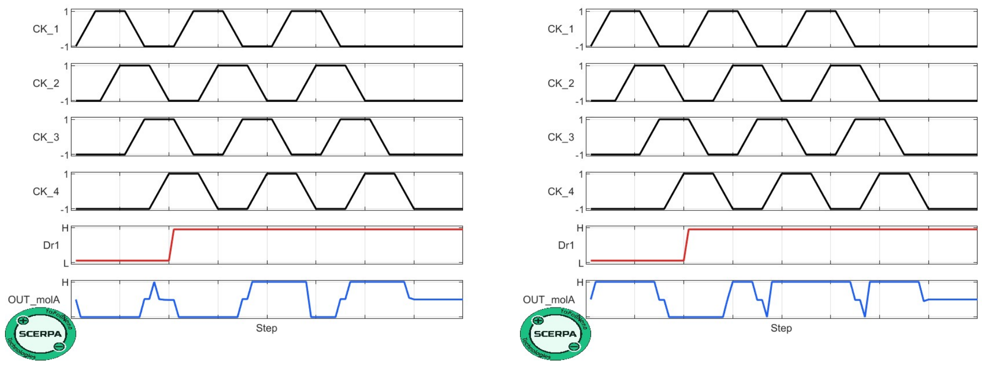
- **Input '0'** (fig. 4.8a) → **Output '0'** (fig. 4.8b): Correct logic '0' propagation through upward turn
- **Input '1'** (fig. 4.8c) → **Output '1'** (fig. 4.8d): Logic '1' polarization preserved through the routing structure



(a) Input ’0’ (b) Output ’0’ (c) Input ’1’ (d) Output ’1’

Figure 4.8: L_DXUP potential plot

Figure 4.9 presents the waveform analysis. The corner regions show **no signal degradation**, validating robust electrostatic coupling.



(a) L_DXDW waveplot (b) L_DXUP waveplot

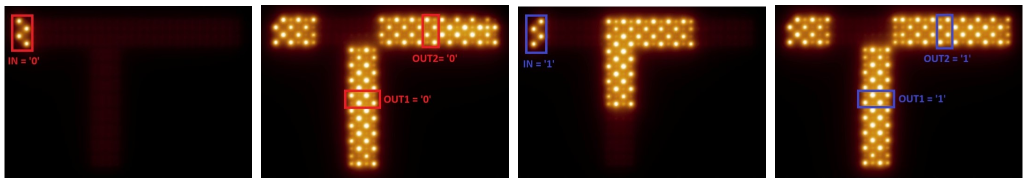
Figure 4.9: L_DXDW and L_DXUP waveplots

Fan-out Devices (T_DXDW, T_DXUP, T_UPDW) Figures 4.10, 4.11 and 4.12 present the potential plot characterization for all three T-junction variants:

T_DXDW

- **Input ’0’** (fig. 4.10a) → **Output ’00’** (fig. 4.10b): Both output branches maintain uniform logic ’0’ polarization

- **Input ’1’** (fig. 4.10c) → **Output ’11’** (fig. 4.10d): Logic ’1’ equally distributed to both outputs

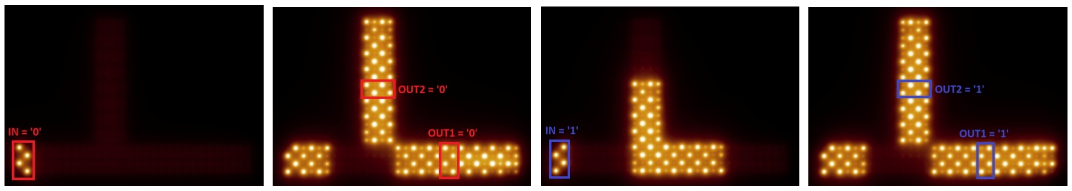


(a) Input '0' (b) Output: '00' (c) Input: '1' (d) Output: '11'

Figure 4.10: T_DXDW potential plot

T_DXUP

- **Input '0'** (fig. 4.11a) → **Output '00'** (fig. 4.11b): Stable logic '0' on both branches
- **Input '1'** (fig. 4.11c) → **Output '11'** (fig. 4.11d): Uniform logic '1' distribution

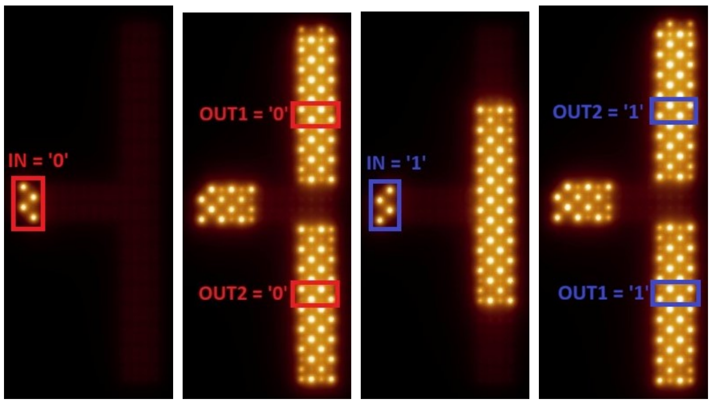


(a) Input: '0' (b) Output: '00' (c) Input: '1' (d) Output: '11'

Figure 4.11: T_DXUP potential plot

T_UPDW

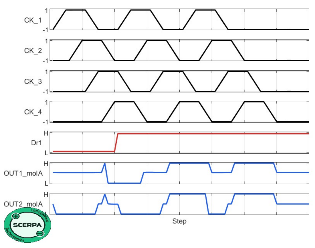
- **Input '0'** (fig. 4.12a) → **Output '00'** (fig. 4.12b): Correct logic '0' propagation to both outputs
- **Input '1'** (fig. 4.12c) → **Output '11'** (fig. 4.12d): Logic '1' successfully distributed to all branches



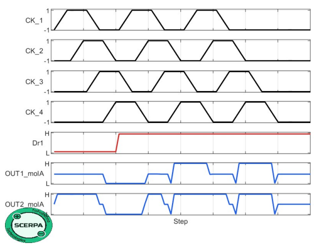
(a) In: '0' (b) Out: '00' (c) In: '1' (d) Out: '11'

Figure 4.12: T_UPDW potential plot

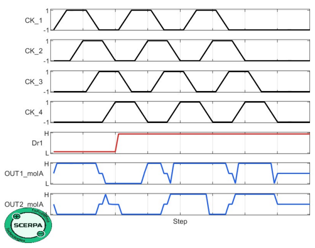
Figure 4.13 confirms proper temporal synchronization across all output branches.



(a) T_DXDW waveplot



(b) T_DXUP waveplot

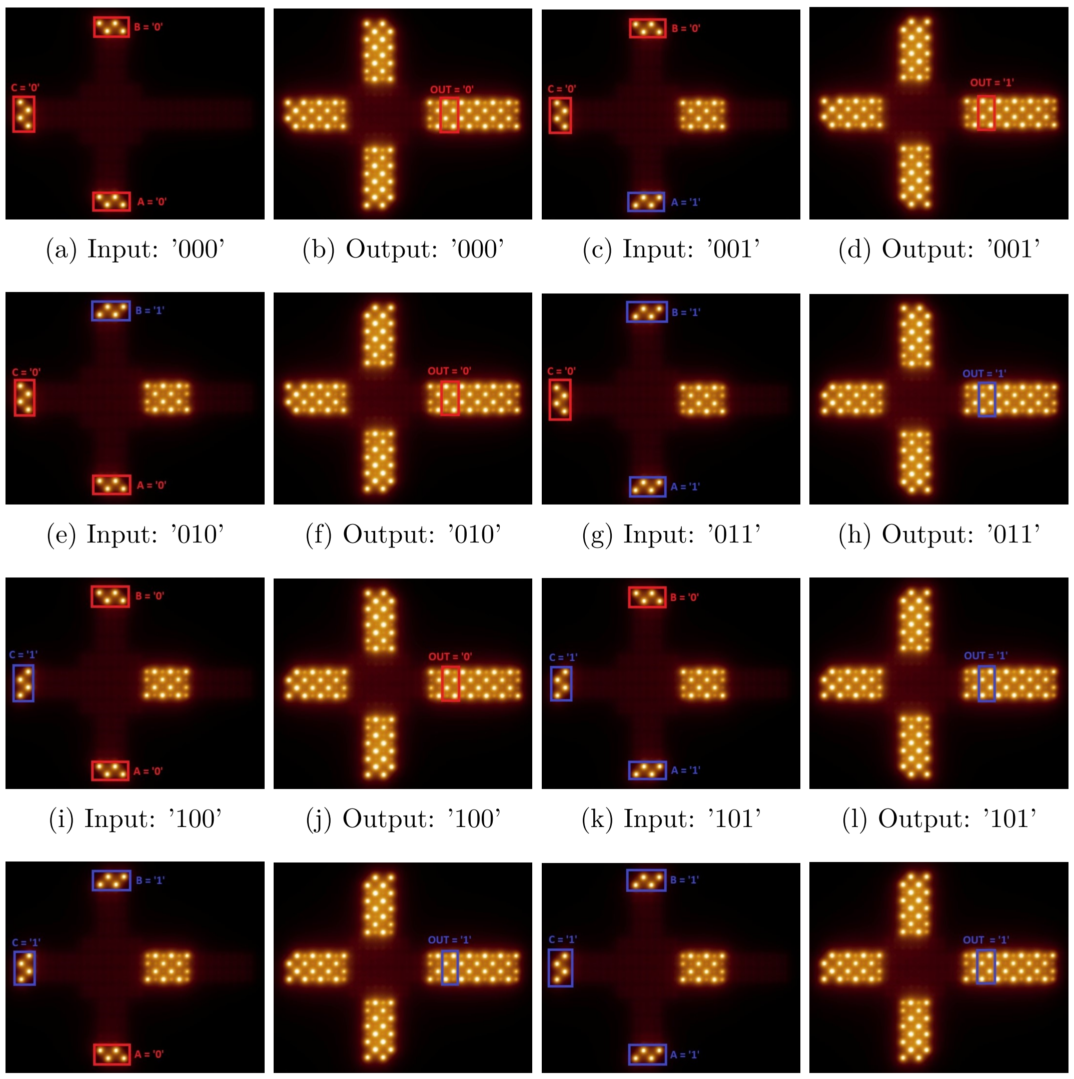


(c) T_UPDW waveplot

Figure 4.13: T_DXDW, T_DXUP and T_UPDW waveplots

Majority Voter (MV) The three-input Majority Voter implements the function OUT = MAJ(IN1, IN2, IN3), outputting logic '1' when at least two inputs are '1'. Figure 4.14 presents the complete characterization across all eight input combinations:

- **Input '000'** (fig. 4.14a) → **Output '0'** (fig. 4.14b): All three inputs at logic '0'; output correctly remains at logic '0'
- **Input '001'** (fig. 4.14c) → **Output '0'** (fig. 4.14d): Two inputs low, one high; majority voting produces logic '0' output
- **Input '010'** (fig. 4.14e) → **Output '0'** (fig. 4.14f): Different two-low configuration; output maintains logic '0'
- **Input '100'** (fig. 4.14i) → **Output '0'** (fig. 4.14j): Third two-low pattern; output correctly remains at logic '0'
- **Input '011'** (fig. 4.14g) → **Output '1'** (fig. 4.14h): Two inputs high, one low; majority voting switches output to logic '1'
- **Input '101'** (fig. 4.14k) → **Output '1'** (fig. 4.14l): Alternate two-high configuration; output at logic '1'
- **Input '110'** (fig. 4.14m) → **Output '1'** (fig. 4.14n): Third two-high pattern; output displays logic '1'
- **Input '111'** (fig. 4.14o) → **Output '1'** (fig. 4.14p): All three inputs high; unanimous vote produces strong logic '1' output



(a) Input: '000' (b) Output: '000' (c) Input: '001' (d) Output: '001'

(e) Input: '010' (f) Output: '010' (g) Input: '011' (h) Output: '011'

(i) Input: '100' (j) Output: '100' (k) Input: '101' (l) Output: '101'

(m) Input: '110' (n) Output: '110' (o) Input: '111' (p) Output: '111'

Figure 4.14: Majority Voter potential plot

Figure 4.15 shows the waveform characterization, demonstrating proper temporal behavior with the four clock phases (CLK_1 through CLK_4). The three input drivers (Dr1, Dr2, Dr3) clearly show all input transitions, while the output waveform (OUT_molA) correctly implements the majority function after appropriate propagation delay through the clock zones.

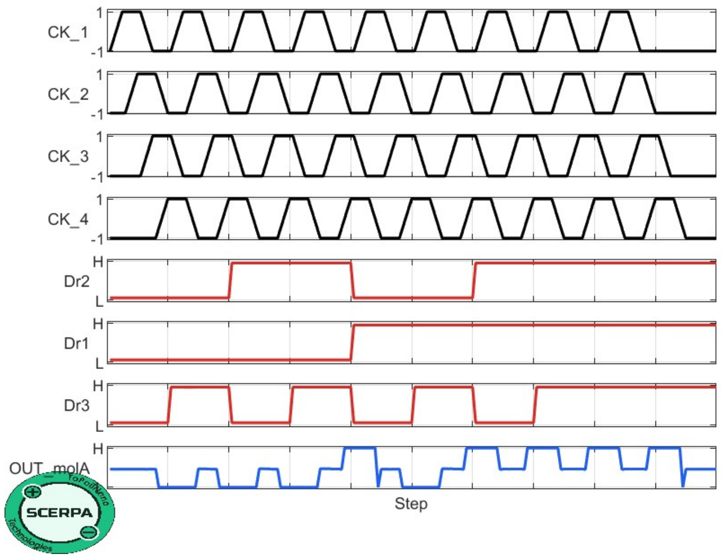


Figure 4.15: Majority Voter waveplot

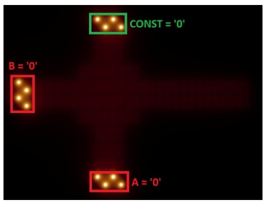
AND Gates (AND_UP, AND_DW, AND_LH) The two-input AND gates are implemented by **forcing one input of the three-input Majority Voter to logic '0'**. The figures show the complete characterization for all three AND gate variants across the four possible input combinations:
AND_UP

- **Input '00'** (fig. 4.16a) → **Output '0'** (fig. 4.16b): Both inputs at logic '0' (plus fixed '0'); output correctly at logic '0'

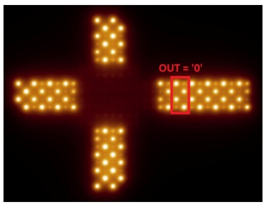
- **Input '01'** (fig. 4.16c) → **Output '0'** (fig. 4.16d): One input high, one low (plus fixed '0'); majority not reached, output remains logic '0'

- **Input '10'** (fig. 4.16e) → **Output '0'** (fig. 4.16f): Alternate single-high case (plus fixed '0'); output stays at logic '0'

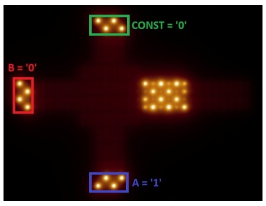
- **Input '11'** (fig. 4.16g) → **Output '1'** (fig. 4.16h): Both inputs high (plus fixed '0'); two of three inputs high, output switches to logic '1' (AND condition satisfied).



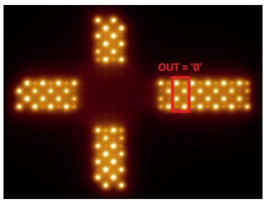
(a) Input: ’00’



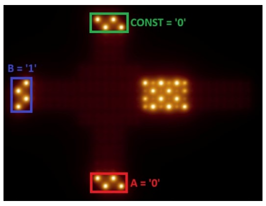
(b) Output: ’0’



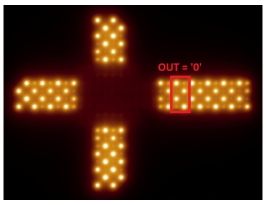
(c) Input: ’01’



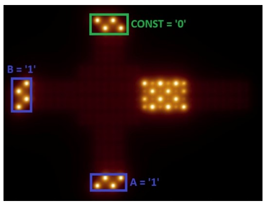
(d) Output: ’0’



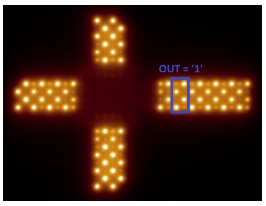
(e) Input: ’10’



(f) Output: ’0’



(g) Input: ’11’

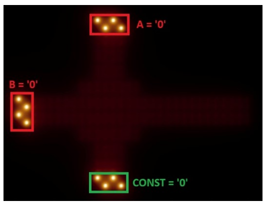


(h) Output: ’1’

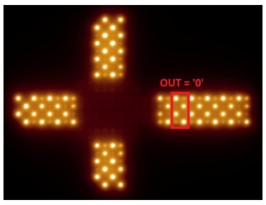
Figure 4.16: AND_UP potential plot)

AND_DW

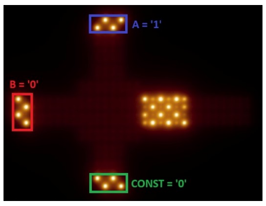
- **Input ’00’** (fig. 4.17a) → **Output ’0’** (fig. 4.17b): Output at logic ’0’
- **Input ’01’** (fig. 4.17c) → **Output ’0’** (fig. 4.17d): Output remains logic ’0’
- **Input ’10’** (fig. 4.17e) → **Output ’0’** (fig. 4.17f): Output stays at logic ’0’
- **Input ’11’** (fig. 4.17g) → **Output ’1’** (fig. 4.17h): Output achieves logic ’1’ (AND condition satisfied)



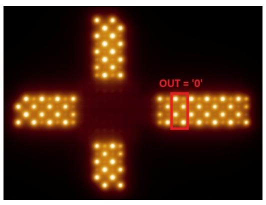
(a) Input: ’00’



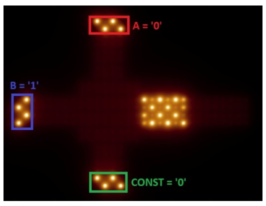
(b) Output: ’0’



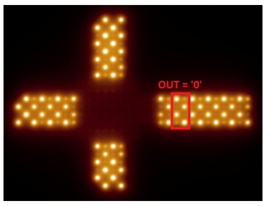
(c) Input: ’01’



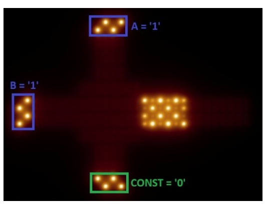
(d) Output: ’0’



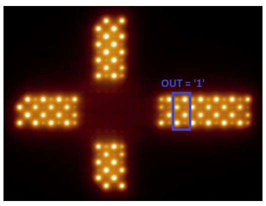
(e) Input: ’10’



(f) Output: ’0’



(g) Input: ’11’

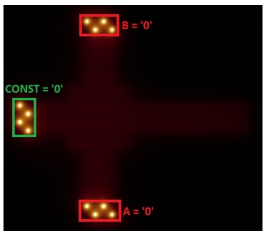


(h) Output: ’1’

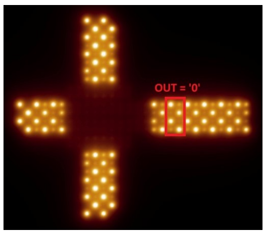
Figure 4.17: AND_DW potential plot

AND_LH

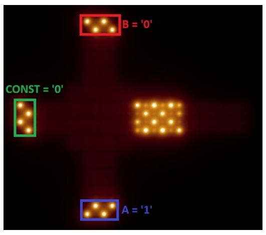
- **Input ’00’** (fig. 4.18a) → **Output ’0’** (fig. 4.18b): Output at logic ’0’
- **Input ’01’** (fig. 4.18c) → **Output ’0’** (fig. 4.18d): Output maintains logic ’0’
- **Input ’10’** (fig. 4.18e) → **Output ’0’** (fig. 4.18f): Output remains logic ’0’
- **Input ’11’** (fig. 4.18g) → **Output ’1’** (fig. 4.18h): Output displays logic ’1’ (AND condition satisfied)



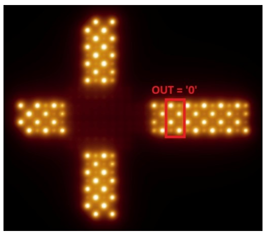
(a) Input: ’00’



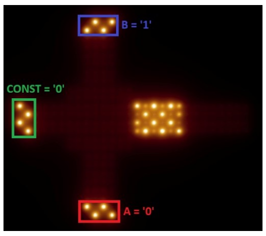
(b) Output: ’0’



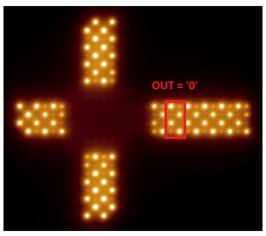
(c) Input: ’01’



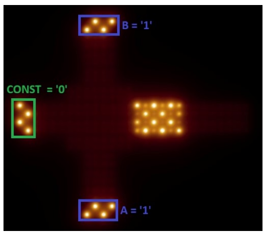
(d) Output: ’0’



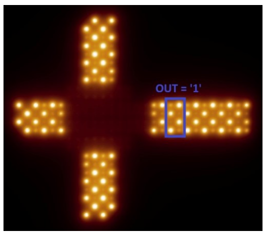
(e) Input: ’10’



(f) Output: ’0’



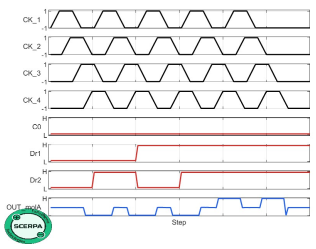
(g) Input: ’11’



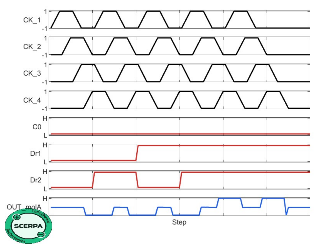
(h) Output: ’1’

Figure 4.18: AND_LH potential plot

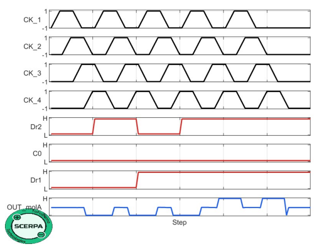
The waveform analysis (fig. 4.19a, 4.19b, 4.19c) confirms **identical behavior across all three orientations**, validating the AND implementation.



(a) AND_UP waveplot



(b) AND_DW waveplot



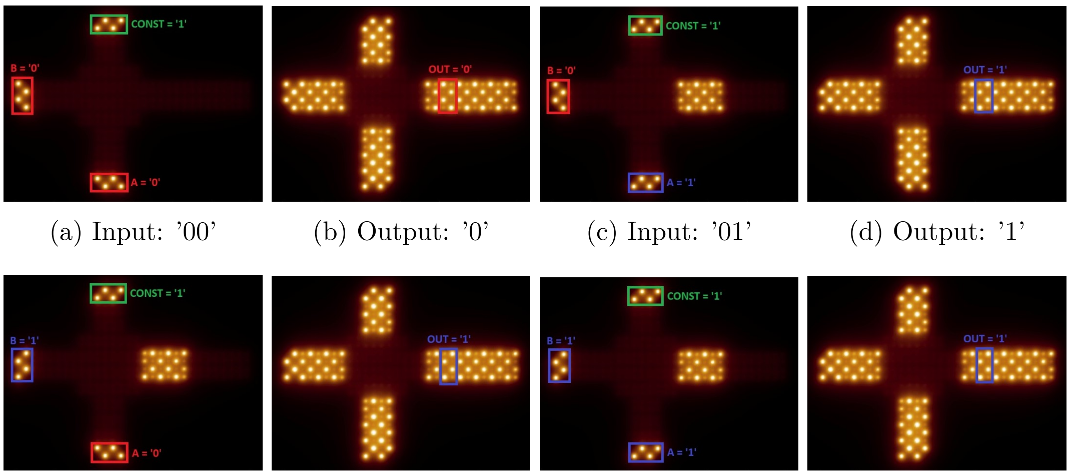
(c) AND_LH waveplot

Figure 4.19: AND_UP, AND_DW and AND_LH waveplots

OR Gates (OR_UP, OR_DW, OR_LH) The two-input OR gates are implemented by **forcing one input of the three-input Majority Voter to logic ’1’**. The figures show the complete characterization for all three OR gate orientations across the four possible input combinations:

OR_UP

- **Input ’00’** (fig. 4.20a) → **Output ’0’** (fig. 4.20b): Both inputs at logic ’0’ (plus fixed ’1’); only one of three inputs high, output at logic ’0’
- **Input ’01’** (fig. 4.20c) → **Output ’1’** (fig. 4.20d): One input high (plus fixed ’1’); two of three inputs high, output switches to logic ’1’
- **Input ’10’** (fig. 4.20e) → **Output ’1’** (fig. 4.20f): Alternate single-high case (plus fixed ’1’); majority reached, output at logic ’1’
- **Input ’11’** (fig. 4.20g) → **Output ’1’** (fig. 4.20h): Both inputs high (plus fixed ’1’); all three inputs high, strong logic ’1’ output



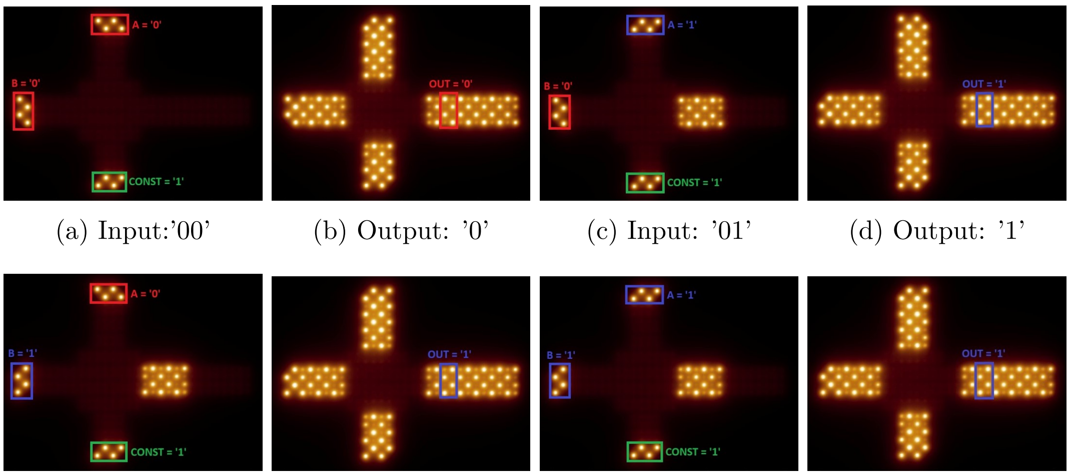
(a) Input: ’00’ (b) Output: ’0’ (c) Input: ’01’ (d) Output: ’1’

(e) Input: ’10’ (f) Output: ’1’ (g) Input: ’11’ (h) Output: ’1’

Figure 4.20: OR_UP potential plot

OR_DW

- **Input ’00’** (fig. 4.21a) → **Output ’0’** (fig. 4.21b): Output at logic ’0’ (only fixed input high)
- **Input ’01’** (fig. 4.21c) → **Output ’1’** (fig. 4.21d): Output switches to logic ’1’ (OR condition satisfied)
- **Input ’10’** (fig. 4.21e) → **Output ’1’** (fig. 4.21f): Output at logic ’1’ (OR condition satisfied)
- **Input ’11’** (fig. 4.21g) → **Output ’1’** (fig. 4.21h): Reinforced logic ’1’ output (both inputs high)



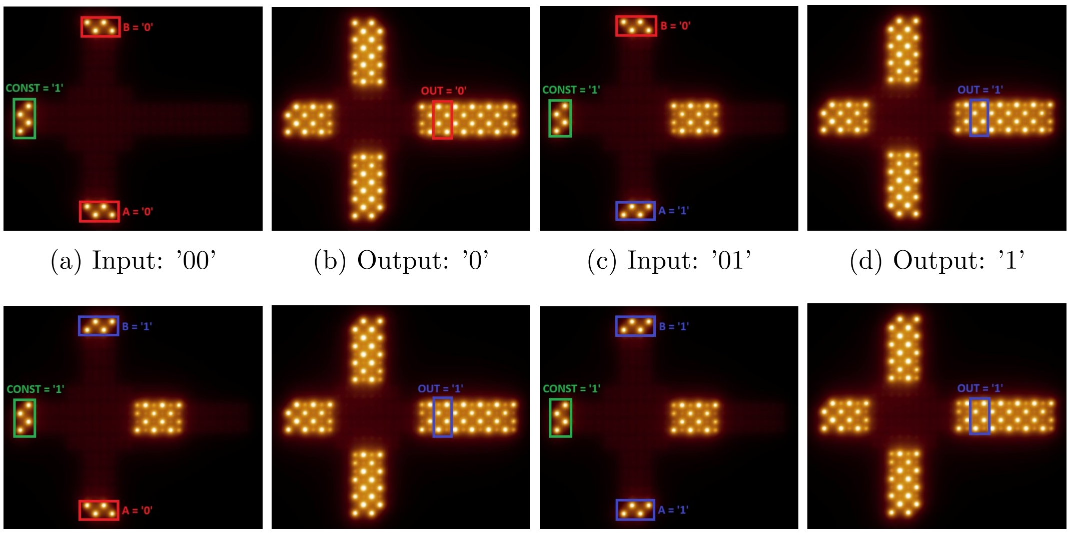
(a) Input:'00' (b) Output: '0' (c) Input: '01' (d) Output: '1'

(e) Input: '10' (f) Output: '1' (g) Input: '11' (h) Output: '1'

Figure 4.21: OR_DW potential plot

OR_LH

- **Input '00'** (fig. 4.22a) → **Output '0'** (fig. 4.22b): Output at logic '0'
- **Input '01'** (fig. 4.22c) → **Output '1'** (fig. 4.22d): Output switches to logic '1'
- **Input '10'** (fig. 4.22e) → **Output '1'** (fig. 4.22f): Output at logic '1' (one input
- **Input '11'** (fig. 4.22g) → **Output '1'** (fig. 4.22h): Strong logic '1' output

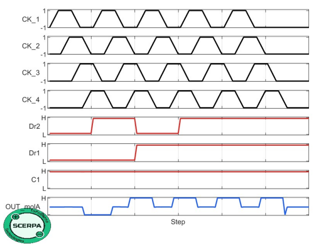


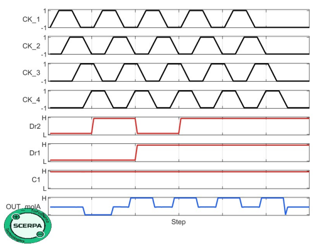
(a) Input: '00' (b) Output: '0' (c) Input: '01' (d) Output: '1'

(e) Input: '10' (f) Output: '1' (g) Input: '11' (h) Output: '1'

Figure 4.22: OR_LH potential plot

The waveform analysis (fig. 4.23a, 4.23b, 4.23c) confirms functional equivalence across all orientations, validating the OR implementation.


(a) OR_UP waveplot


(b) OR_DW waveplot

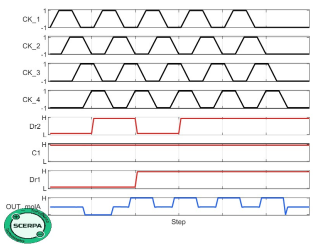

(c) OR_LH waveplot

Figure 4.23: OR_UP, OR_DW and OR_LH waveplots

4.4 Validation of benchmark circuits

Beyond the validation of individual standard cells, **multi-gate combinational circuits were synthesized and simulated** to verify the composability and cascading behavior of the SIM(7) library devices. These benchmark circuits were automatically generated using the fiction framework from high-level specifications and subsequently verified at the molecular level through SCERPA simulations.

The following benchmark circuits were implemented and validated:

- **AND3**: Three-input AND gate
- **NAND2**: Two-input NAND gate
- **XOR2**: Two-input XOR gate
- **MUX21**: 2:1 multiplexer
- **HA**: Half-adder (sum and carry outputs)
- **COMP**: 1-bit comparator (a<b, a=b, a>b outputs)
- **C17**: ISCAS'85 benchmark circuit

Each circuit demonstrates correct logical functionality across all input combinations, validating both the individual cell behavior and the inter-cell connectivity through the standardized tile interface.

4.4.1 AND3

The three-input AND gate implements the logical function OUT $= A \cdot B \cdot C$, producing a logic '1' output only when all three inputs are simultaneously at logic '1'. The circuit is composed of two AND gates and one bus. Figure 4.24 shows the complete layout of the AND3 gate.

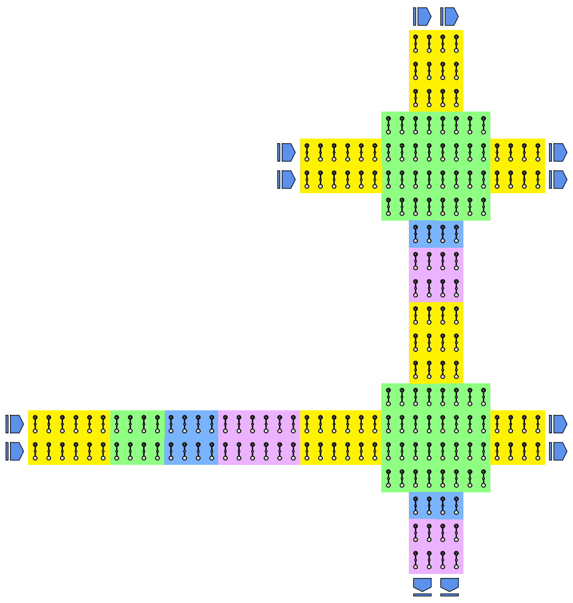


Figure 4.24: AND3 layout

Table 4.2 presents the complete truth table for the AND3 gate.

Input A	**Input B**	**Input C**	**Output**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Table 4.2: AND3 truth table

Figure 4.25 shows the waveform characterization that **validates the truth table** across all eight input combinations. The simulation demonstrates complete consistency with the expected three-input AND behavior. The waveplot clearly shows seven consecutive low output periods followed by one high period, exactly matching the expected '0', '0', '0', '0', '0', '0', '0', '1' output sequence.

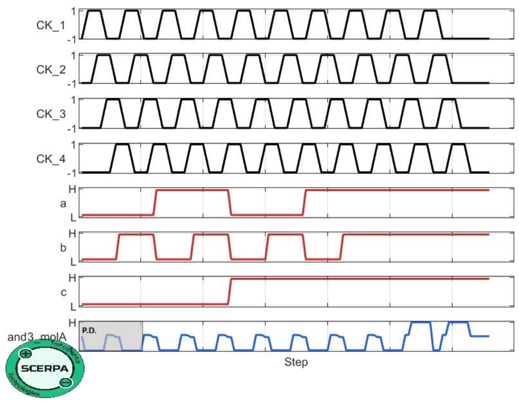


Figure 4.25: AND3 waveform characterization

4.4.2 NAND2

The two-input NAND gate implements the logical function OUT $= \overline{A \cdot B}$, producing a logic '0' output only when both inputs are at logic '1'. The NAND2 circuit is constructed by cascading a Majority Voter with one input fixed at logic '0' (implementing the AND function), followed by an inverter.

Figure 4.26 shows the complete layout of the NAND2 gate, with the four clock zones clearly visible through the color coding.

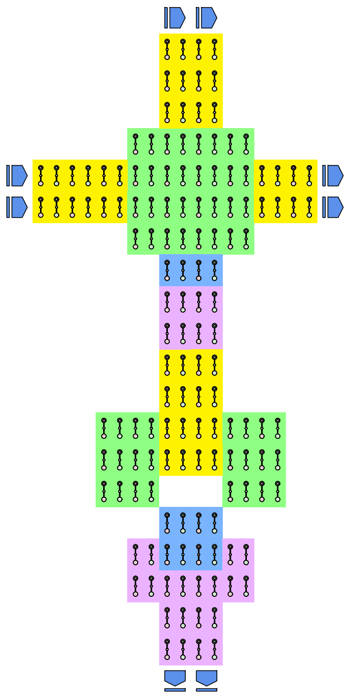


Figure 4.26: NAND2 layout

Table 4.3 presents the complete truth table for the NAND2 gate.

Input A	**Input B**	**Output**
0	0	1
0	1	1
1	0	1
1	1	0

Table 4.3: NAND2 truth table

Figure 4.27 shows the waveform characterization that **validates the truth table** across all four input combinations. The simulation demonstrates complete consistency with the expected NAND behavior:

- **Input (00)**: Both inputs at logic '0'; output correctly at logic '1'
- **Input (01)**: One input high; output remains at logic '1'
- **Input (10)**: Alternate single-high case; output maintains logic '1'
- **Input (11)**: Both inputs at logic '1'; output switches to logic '0' (NAND condition satisfied)

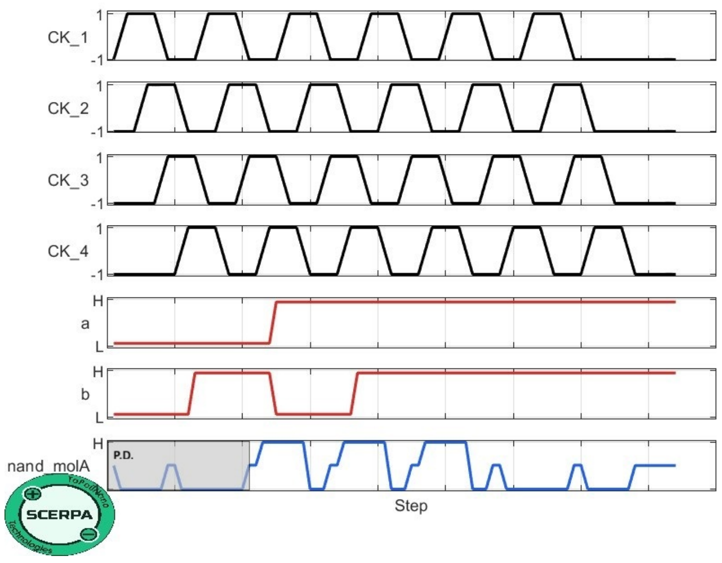


Figure 4.27: NAND2 waveform characterization

4.4.3 XOR2

The two-input XOR (Exclusive-OR) gate implements the logical function OUT $= A \oplus B$, producing a logic '1' output when the two inputs differ. The XOR2 circuit requires one inverter, two AND gates, one OR gate, one T_DXUP, two L_DXDW, one L_DXUP, and four BUS. Figure 4.28 shows the complete layout of the XOR2 gate.

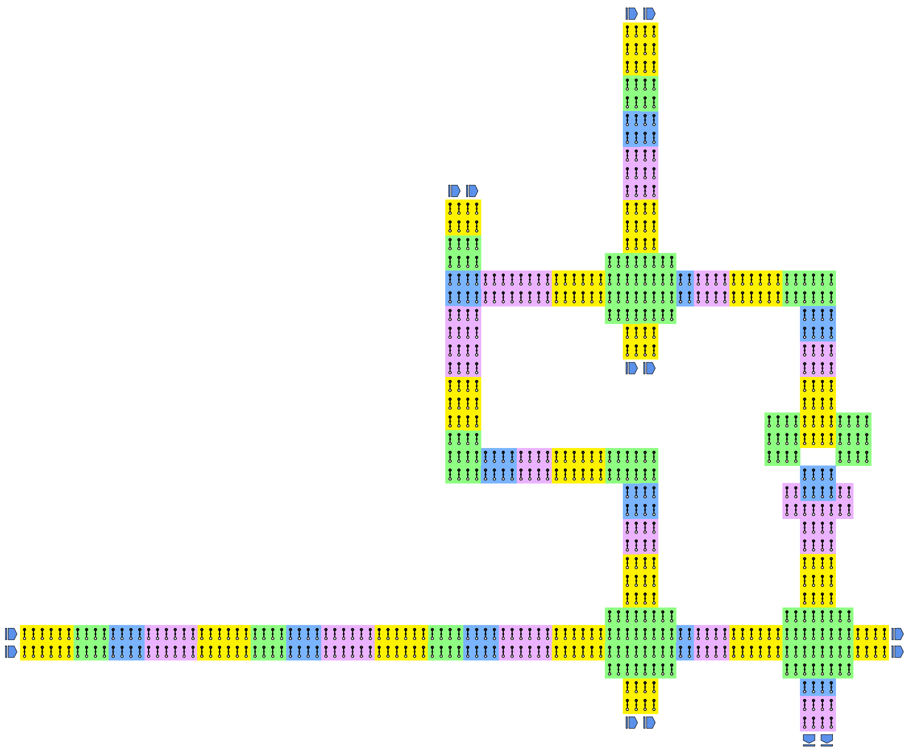


Figure 4.28: XOR2 layout

Table 4.4 presents the complete truth table for the XOR2 gate.

Input A	**Input B**	**Output**
0	0	0
0	1	1
1	0	1
1	1	0

Table 4.4: XOR2 truth table

Figure 4.29 shows the waveform characterization that **validates the truth table** across all input combinations. The simulation demonstrates complete consistency with the expected XOR behavior. The waveplot shows the distinctive ’0’, ’1’, ’1’, ’0’ output pattern characteristic of XOR gates.

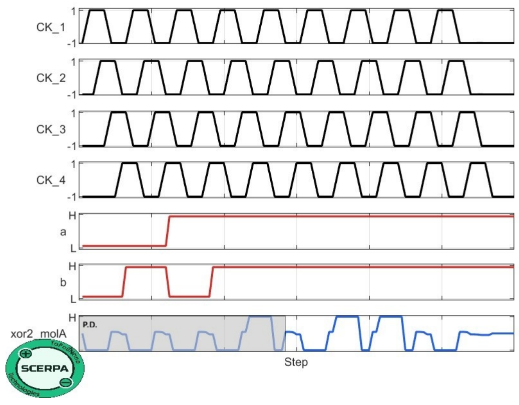


Figure 4.29: XOR2 waveform characterization

4.4.4 MUX21

The 2:1 multiplexer (MUX21) implements a data selector function that forwards one of two input data signals to the output based on a selection signal. The function is: OUT $= \overline{SEL} \cdot IN_0 + SEL \cdot IN_1$, where SEL determines which input propagates to the output. The MUX21 circuit requires one inverter, one OR gate, two AND gates, four BUS, one T_DXUP, two L_DXUP and two L_DXDW. Figure 4.30 shows the complete layout of the MUX21 multiplexer.

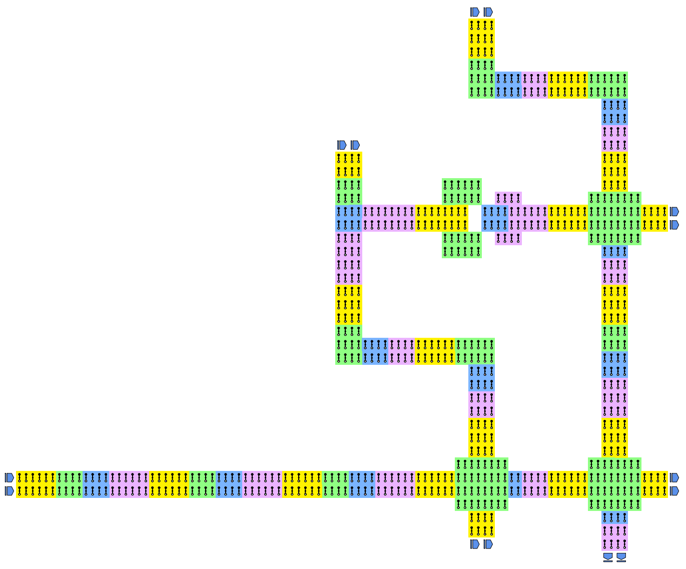


Figure 4.30: MUX21 layout

Table 4.5 presents the complete truth table for the MUX21 multiplexer, organized to show how the SEL signal controls which input appears at the output.

IN2 (SEL)	IN0	IN1	Output
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Table 4.5: MUX21 truth table

Figure 4.31 shows the waveform characterization that **validates the truth table** across all eight input combinations. The simulation demonstrates the correct multiplexing behavior:

- **SEL='0' combinations (rows 1-4)**: Output follows IN0, independent of IN1 value
- **SEL='1' combinations (rows 5-8)**: Output follows IN1, independent of IN0 value

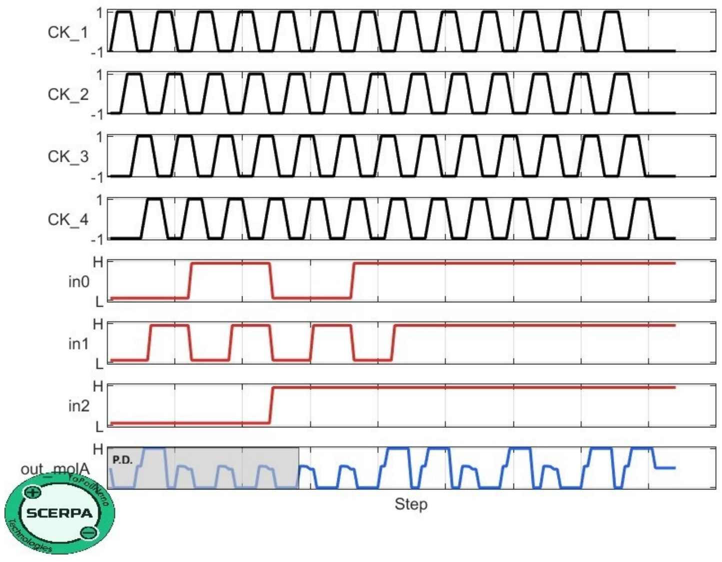


Figure 4.31: MUX21 waveform characterization

4.4.5 HA (Half-Adder)

The half-adder performs binary addition of two single-bit inputs, producing a sum output and a carry output. It implements two functions simultaneously: SUM $= A \oplus B$

and CARRY $= A \cdot B$. The circuit is composed of two AND gates, one OR gate, one inverter, one T_DXUP, one T_DXDW, two L_DXUP, two L_DXDW, and four BUS. Both outputs are computed in parallel.

Figure 4.32 shows the complete layout of the half-adder.

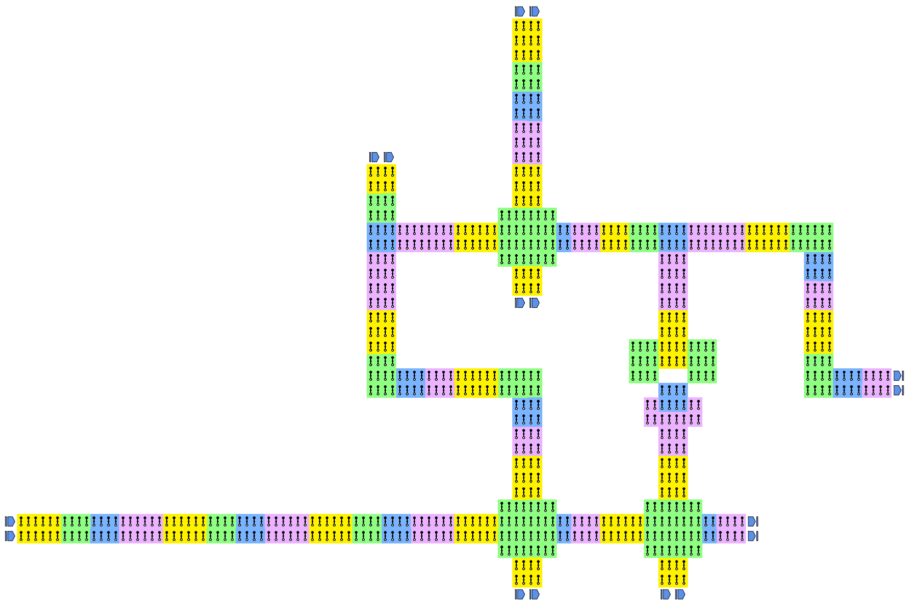


Figure 4.32: Half adder layout

Table 4.6 presents the complete truth table for the half-adder, showing both SUM and CARRY outputs for all input combinations.

Input A	**Input B**	**SUM**	**CARRY**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 4.6: Half-Adder truth table

Figure 4.33 shows the waveform characterization that **validates the truth table** across all four input combinations. The simulation demonstrates correct arithmetic behavior. The waveplot shows both output signals correctly implementing binary addition, with the SUM following the XOR pattern ('0','1','1','0') and the CARRY showing the AND pattern ('0','0','0','1').

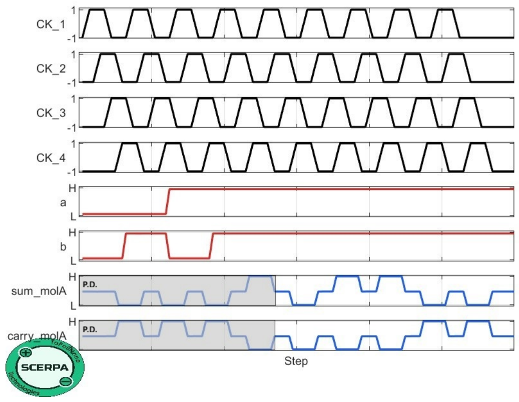


Figure 4.33: Half-Adder waveform characterization

4.4.6 1-bit Comparator

The 1-bit comparator compares two single-bit inputs A and B, producing three outputs indicating the relationship: a<b (A less than B), a=b (A equals B), and a>b (A greater than B). The circuit is composed of three inverters, two AND gates, one OR gate, two T_DXDW, one T_DXUP, three L_DXDW, two L_DXUP and ten BUS.

Figure 4.34 shows the complete layout of the 1-bit comparator.

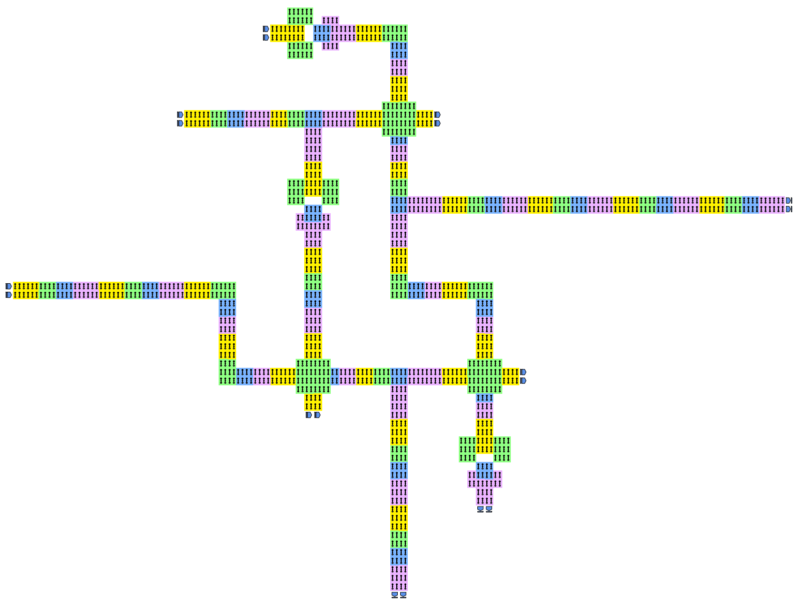


Figure 4.34: Comparator layout

Table 4.7 presents the complete truth table for the 1-bit comparator, showing all three output signals.

Input A	Input B	a<b	a=b	a>b
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

Table 4.7: 1-bit Comparator truth table

Figure 4.35 shows the waveform characterization that **validates the truth table**. The simulation demonstrates correct comparison behavior:

- **Input (00) → a=b = '1'**: Both inputs equal to 0; equality output high
- **Input (01) → a<b = '1'**: A less than B; less-than output high
- **Input (10) → a>b = '1'**: A greater than B; greater-than output high
- **Input (11) → a=b ='1'**: Both inputs equal to 1; equality output high

Note that exactly one output is high for each input combination.

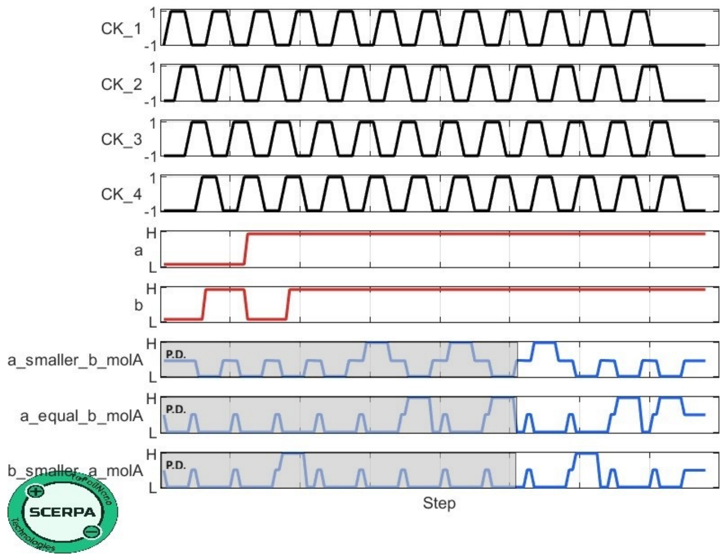


Figure 4.35: 1-bit Comparator waveform characterization

4.4.7 C17

The C17 circuit is a benchmark from the ISCAS'85 suite, representing a small combinational circuit used for testing synthesis and verification tools. The circuit is composed of one inverter, four AND gates, two OR gates, two T_DXUP, four L_DXUP, two L_DXDW and seventeen BUS.

Figure 4.36 shows the automatically generated layout of the C17 benchmark circuit.

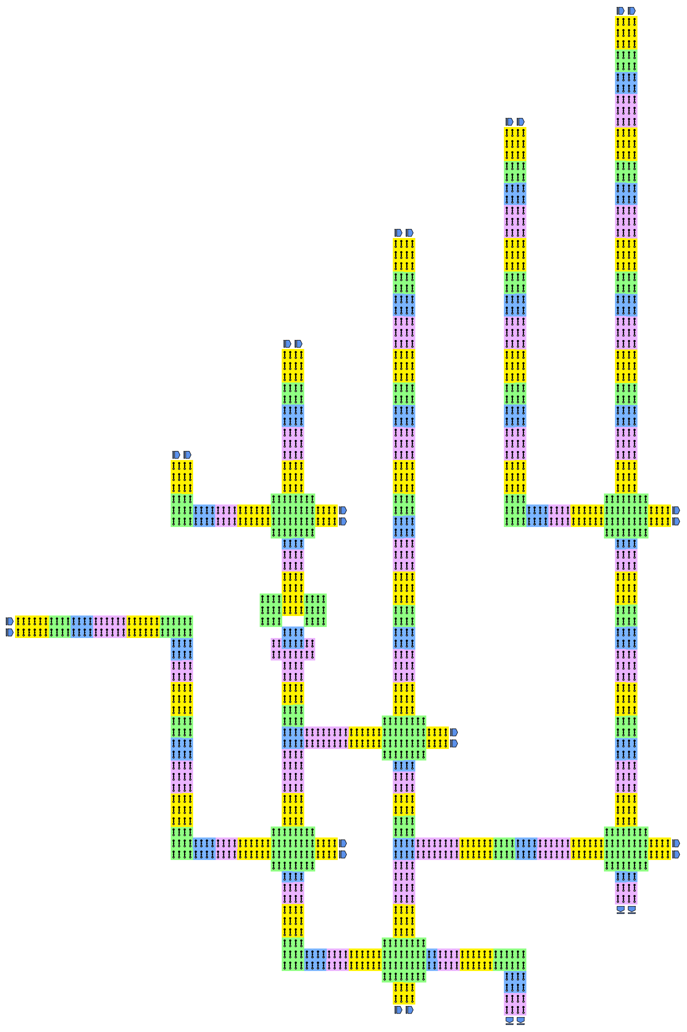


Figure 4.36: C17 layout

Figure 4.37 shows the waveform characterization validating the correct functionality of the C17 circuit.

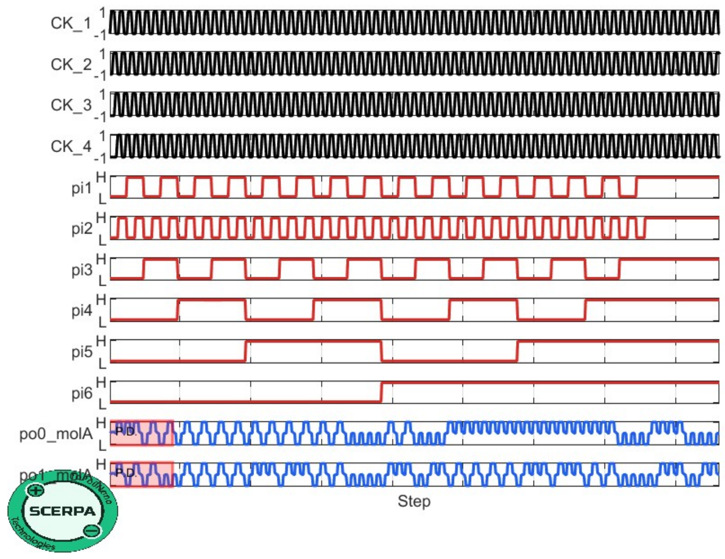


Figure 4.37: C17 waveform characterization

Chapter 5

Conclusion

This thesis represents a significant contribution toward the practical realization of Molecular Field-Coupled Nanocomputing, addressing the critical challenge of the absence of design automation tools for molecular circuits.
The work has led to the development and validation of the first standard cell library for molFCN physically characterized through SCERPA simulations, bridging the gap between molecular-level physical simulation and automatic circuit design. The SIM(7) library, developed in collaboration with the Technical University of Munich, comprises seven fundamental devices (bus, L-wire, inverter, fan-out, majority voter, AND and OR gates) implemented on a standardized 10×10 cell grid. Simulations demonstrated 100% functional correctness for all cells and their geometric rotations, validating not only individual devices but also complex benchmark circuits such as NAND, XOR, multiplexer, half-adder, comparator, and the C17 circuit from the ISCAS85 suite. These results are particularly significant because they demonstrate the possibility of fully automatic synthesis, from Verilog specification to physical molecular implementation, validated with SCERPA.
The developed and validated standard cell library, the systematic methodology for its creation, and the integration with the Fiction framework represent concrete contributions that enable new research and development possibilities for the entire scientific community. This transforms molFCN from a purely theoretical technology to a concrete design platform, enabling fair comparisons with other FCN technologies (QCA, iNML, SiDB) already supported by Fiction and freeing researchers from device-by-device manual design.
However, it is important to recognize the methodological limitations. The use of an ideal molecule, while offering advantages in reproducibility, introduces uncertainties regarding the transferability to real molecules such as bis-ferrocene or diallyl-butane, which exhibit complex behaviors dependent on the applied clock field. The simulations assume ideal conditions with atomically flat substrates, absence of defects, and controlled temperatures, while experimental reality inevitably presents surface roughness, deposition variations, and contaminations.

Despite these limitations, the developed SCERPA-Fiction framework provides the community with tools for molecule-circuit co-design, guiding future development toward functional prototypes through an approach that allows systematic exploration of the design space before committing to costly fabrication experiments.

5.1 Future perspective

Despite the progress in enabling design automation for molFCN, fundamental technological challenges remain for the transition from simulation to the realization of functional prototypes. The most critical challenge remains the absence of an actually fabricated prototype, which requires resolving complex issues from chemical synthesis to nanofabrication. Fabrication requires controlled molecular deposition on atomically flat substrates with sub-nanometric precision, where techniques such as template-stripping and Self-Assembled Monolayers (SAM) show potential but present challenges in scalability, orientation control, and uniform coverage. The nanolithographic patterning of clock structures, which must generate electric fields of 1-2 V/nm with sharp spatial transitions, represents another critical issue where conventional techniques (electron beam lithography, focused ion beam) are slow and expensive, while hybrid top-down/bottom-up approaches require further developments for precise alignment. The molFCN-CMOS interfacing constitutes a fundamental challenge since molFCN encodes information in molecular charge position, a radically different paradigm from CMOS currents and voltages. Single-Electron Transistors (SET), although sensitive to single electrons, typically operate only at cryogenic temperatures, while alternatives such as carbon nanotube or graphene-based sensors could offer complementary pathways. Defect tolerance remains critical, requiring quantitative understanding of how variations in deposition, substrate defects, or clock structure fluctuations affect functionality.
In the medium-to-long term, the objective is to bring molFCN closer to functional prototypes through coordinated progress across the entire technology stack. The developed methodology allows inverting the approach: defining molecular requirements from circuits and collaborating with chemists to synthesize molecules that satisfy them. The criteria include maximization of electrostatic coupling (dependent on geometry and polarizability of active dots), low native dipole moment to minimize unwanted interactions, high polarizability under external fields, chemical stability and oxidation resistance, and functional groups for stable anchoring (thiols for gold, silanes for silicon). The design of complex benchmark circuits such as complete ALUs, molecular memories exploiting intrinsic bistability[19], and specialized architectures for neural networks or neuromorphic processors would enable quantitative evaluations on realistic workloads and identification of advantageous application niches. Integration with industrial CAD flows through plugins for commercial tools (Cadence, Synopsys, Mentor Graphics) and format standardization (Liberty, LEF)[41] would facilitate adoption by the community, while specific Design for Manufacturing methodologies would ensure fabricability with acceptable yields.
The long-term roadmap includes partially overlapping phases: iterative molecular synthesis with spectroscopic and electrochemical characterization to validate requirements; development of scalable nanofabrication techniques[13] combining lithography for templates

and guided self-assembly; SET interfacing systems at room temperature or alternatives based on new materials; advanced non-destructive characterization techniques (ultrafast STM microscopy, time-resolved X-ray spectroscopy, near-field optical microscopy) with nanometric spatial resolution and picosecond temporal resolution; continuous evolution of the MoSQuiTo framework incorporating fabrication variation models, realistic clock field generation, and experimental feedback. Applications could emerge where molFCN advantages are most pronounced: ultra-low power computing for IoT and wearables benefiting from the absence of leakage current, molecular neuromorphic architectures exploiting ultra-high density and computation-in-memory for AI edge, and quantum-classical hybrid computing where room-temperature molFCN circuits interface with cryogenic qubits, providing control and readout for scalable quantum systems.

Bibliography

[1] International Technology Roadmap for Semiconductors (ITRS), "Beyond CMOS," 2015. [Online]. Available: http://www.itrs2.net/

[2] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, 1993.

[3] C. S. Lent, "Bypassing the transistor paradigm," *Science*, vol. 288, no. 5471, pp. 1597–1599, Jun. 2000.

[4] J. Timler and C. S. Lent, "Power Gain and Dissipation in Quantum-Dot Cellular Automata," *Journal of Applied Physics*, vol. 91, no. 2, pp. 823–831, 2002.

[5] N. G. Anderson and S. Bhanja, *Field-Coupled Nanocomputing.* Springer Berlin, Heidelberg, 2014.

[6] F. S. Torres, M. Walter, R. Wille, D. Große, and R. Drechsler, "Synchronization of clocked field-coupled circuits," in *2018 IEEE 18th International Conference on Nanotechnology (IEEE-NANO)*, 2018, pp. 1–5.

[7] M. Vacca, M. Graziano, L. Di Crescenzo, A. Chiolerio, A. Lamberti, D. Balma, G. Canavese, F. Celegato, E. Enrico, P. Tiberto, L. Boarino, and M. Zamboni, "Magnetoelastic clock system for nanomagnet logic," *IEEE Transactions on Nanotechnology*, vol. 13, no. 5, pp. 963–973, 2014.

[8] M. Graziano, M. Vacca, A. Chiolerio, and M. Zamboni, "An ncl-hdl snake-clock-based magnetic qca architecture," *IEEE Transactions on Nanotechnology*, vol. 10, no. 5, pp. 1141–1149, 2011.

[9] Y. Wang and M. Lieberman, "Thermodynamic Behavior of Molecular-Scale Quantum-Dot Cellular Automata (QCA) Wires and Logic Devices," *IEEE Transactions on Nanotechnology*, vol. 3, no. 3, pp. 368–376, 2004.

[10] F. Riente, G. Turvani, M. Vacca, M. R. Roch, M. Zamboni, and M. Graziano, "ToPoliNano: A CAD Tool for Nano Magnetic Logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 7, pp. 1061–1074, 2017.

[11] U. Garlando, M. Walter, R. Wille, F. Riente, F. S. Torres, and R. Drechsler, "ToPoliNano and Fiction: Design Tools for Field-coupled Nanocomputing," in *2020 23rd Euromicro Conference on Digital System Design (DSD)*, 2020, pp. 408–415.

[12] C. S. Lent, B. Isaksen, and M. Lieberman, "Molecular Quantum-Dot Cellular Automata," *Journal of the American Chemical Society*, vol. 125, no. 1, pp. 1056–1063, 2003.

[13] F. Ravera, G. Beretta, Y. Ardesi, M. Krzywiecki, M. Graziano, and G. Piccinini, "A roadmap for molecular field-coupled nanocomputing actualization," in *2023 IEEE*

Nanotechnology Materials and Devices Conference (NMDC). IEEE, Oct. 2023.

[14] Y. Ardesi, U. Garlando, F. Riente, G. Beretta, G. Piccinini, and M. Graziano, "Taming Molecular Field-Coupling for Nanocomputing Design," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 19, no. 1, Article no. 1, Dec. 2022.

[15] A. Pulimeno, M. Graziano, A. Antidormi, R. Wang, A. Zahir, and G. Piccinini, "Understanding a Bisferrocene Molecular QCA Wire," in *Field-Coupled Nanocomputing*, Springer Berlin, Heidelberg, 2014, pp. 381–398.

[16] R. Listo, F. Ravera, G. Beretta, Y. Ardesi, G. Piccinini, and M. Graziano, "Unveiling Charge Dynamics in Molecular Field-coupled Nanocomputing," in *IEEE-NANO*. IEEE, 2024, pp. 424–429.

[17] K. Hennessy and C. S. Lent, "Clocking of Molecular Quantum-dot Cellular Automata," *Journal of Vacuum Science & Technology B*, vol. 19, no. 5, pp. 1752–1755, 2001.

[18] Y. Ardesi, A. Pulimeno, M. Graziano, F. Riente, and G. Piccinini, "Effectiveness of Molecules for Quantum Cellular Automata as Computing Devices," *Journal of Low Power Electronics and Applications*, vol. 8, no. 3, 2018. [Online]. Available: https://www.mdpi.com/2079-9268/8/3/24

[19] Y. Ardesi, G. Beretta, M. Vacca, G. Piccinini, and M. Graziano, "Impact of Molecular Electrostatics on Field-Coupled Nanocomputing and Quantum-Dot Cellular Automata Circuits," *Electronics*, vol. 11, no. 2, p. 276, Jan. 2022.

[20] Y. Ardesi, A. Gaeta, G. Beretta, G. Piccinini, and M. Graziano, "Ab initio molecular dynamics simulations of field-coupled nanocomputing molecules," *Journal of Integrated Circuits and Systems*, vol. 16, no. 1, pp. 1–8, 2021.

[21] Y. Ardesi, R. Wang, G. Turvani, G. Piccinini, and M. Graziano, "SCERPA: A Self-Consistent Algorithm for the Evaluation of the Information Propagation in Molecular Field-Coupled Nanocomputing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2749–2760, 2019.

[22] Y. Ardesi, G. Turvani, M. Graziano, and G. Piccinini, "Scerpa simulation of clocked molecular field-coupling nanocomputing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 3, pp. 558–567, 2021.

[23] Y. Ardesi, F. Mo, M. Vacca, G. Piccinini, and M. Graziano, "Guesstimation of Molecular Ensemble Electrostatics Properties through SCERPA-DFT Calculation: Molecular Field-coupled Nanocomputing as a Case Study," *Advanced Theory and Simulations*, no. e00812, Jun. 2025.

[24] G. Beretta, Y. Ardesi, G. Piccinini, and M. Graziano, "vlsinanocomputing/SCERPA: SCERPA v4.0.1," 2022.

[25] G. Beretta, Y. Ardesi, M. Graziano, and G. Piccinini, "vlsinanocomputing/BBchar: v1.0.1," 2023.

[26] G. Beretta, "Modeling and Simulation Strategies for Advancing Molecular FCN: Pioneering Design Rules Toward Implementation from Molecules to Devices," Ph.D. dissertation, Politecnico di Torino, Turin, Italy, 2024. [Online]. Available: https://iris.polito.it/handle/11583/2991352

[27] U. Garlando, F. Riente, D. Vergallo, M. Graziano, and M. Zamboni, "ToPoliNano & MagCAD: A Complete Framework for Design and Simulation of Digital Circuits

Based on Emerging Technologies," in *2018 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2018, pp. 153–156.

[28] F. Riente, U. Garlando, G. Turvani, M. Vacca, M. R. Roch, and M. Graziano, "MagCAD: Tool for the Design of 3-D Magnetic Circuits," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 3, pp. 65–73, 2017.

[29] S. S. H. Ng, R. A. Wolkow, K. Walus, J. Retallick, H. N. Chiu, R. Lupoiu, L. Livadaru, T. Huff, M. Rashidi, W. Vine, and T. Dienel, "SiQAD: A Design and Simulation Tool for Atomic Silicon Quantum Dot Circuits," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 137–146, 2020.

[30] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Fiction: An Open Source Framework for the Design of Field-coupled Nanocomputing Circuits," arXiv preprint arXiv:1905.02477, 2019.

[31] M. Walter, J. Drewniok, S. Hofmann, B. Hien, and R. Wille, "The Munich Nanotech Toolkit (MNT)," in *2024 IEEE International Conference on Nanotechnology (IEEE NANO)*, 2024, pp. 454–459.

[32] A. Mishchenko, S. Chatterjee, and R. Brayton, "DAG-aware AIG Rewriting a Fresh Look at Combinational Logic Synthesis," in *Proceedings of the 43rd Annual Design Automation Conference (DAC)*, 2006.

[33] A. Mishchenko, R. Brayton, J.-H. R. Jiang, and S. Jang, "Scalable Don't-Care-Based Logic Optimization and Resynthesis," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 4, no. 4, pp. 1–23, 2011.

[34] A. T. Calvino, H. Riener, S. Rai, A. Kumar, and G. De Micheli, "A Versatile Mapping Approach for Technology Mapping and Graph Optimization," in *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2022, pp. 410–416.

[35] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "An exact method for design exploration of quantum-dot cellular automata," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 503–508.

[36] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Scalable design for field-coupled nanocomputing circuits," in *2019 24th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2019, pp. 1–6.

[37] S. Hofmann, M. Walter, and R. Wille, "A* is born: Efficient and scalable physical design for field-coupled nanocomputing," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*, 2024, pp. 80–85.

[38] B. Hien, M. Walter, S. Hofmann, and R. Wille, "A Fully Planar Approach to Field-coupled Nanocomputing: Scalable Placement and Routing Without Wire Crossings," 2025. [Online]. Available: https://arxiv.org/abs/2504.09012

[39] M. Walter, R. Wille, D. Große, F. Sill Torres, and R. Drechsler, "Placement & Routing for Tile-based Field-coupled Nanocomputing Circuits is NP-complete," *ACM Journal on Emerging Technologies in Computing Systems*, 2019.

[40] S. Hofmann, M. Walter, and R. Wille, "Scalable Physical Design for Silicon Dangling Bond Logic: How a 45° Turn Prevents the Reinvention of the Wheel," in *2023 IEEE 23rd International Conference on Nanotechnology (IEEE-NANO)*, 2023, pp. 872–877.

[41] D. A. Reis, C. A. T. Campos, T. R. B. S. Soares, O. P. V. Neto, and F. S. Torres, "A Methodology for Standard Cell Design for QCA," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 2114–2117.

[42] M. Walter, S. S. H. Ng, K. Walus, and R. Wille, "Hexagons are the Bestagons: Design Automation for Silicon Dangling Bond Logic," in *Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 739–744.

[43] UC Berkeley, "berkeley-abc/abc: ABC - A System for Sequential Synthesis and Verification." [Online]. Available: https://github.com/berkeley-abc/abc

[44] M. Soeken, H. Riener, W. Haaswijk, and G. De Micheli, "The EPFL Logic Synthesis Libraries," 2018. arXiv:1805.05121. [Online]. Available: https://github.com/lsils/mockturtle

[45] M. Walter, W. Haaswijk, R. Wille, F. Sill Torres, and R. Drechsler, "One-pass Synthesis for Field-coupled Nanocomputing Technologies," in *2021 26th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 574–580.

[46] M. Walter and R. Wille, "Efficient Multi-Path Signal Routing for Field-coupled Nanotechnologies," in *Proceedings of the 17th ACM International Symposium on Nanoscale Architectures (NANOARCH)*, 2023, Article no. 7.

[47] K. Walus, T. J. Dysart, G. A. Jullien, and R. A. Budiman, "QCADesigner: A Rapid Design and Simulation Tool for Quantum-Dot Cellular Automata," *IEEE Transactions on Nanotechnology*, vol. 3, no. 1, pp. 26–31, 2004.

[48] V. Arima, M. Iurlo, L. Zoli, S. Kumar, M. Piacenza, F. Della Sala, F. Matino, G. Maruccio, R. Rinaldi, F. Paolucci, M. Marcaccio, P. G. Cozzi, and A. P. Bramanti, "Toward Quantum-Dot Cellular Automata Units: Thiolated-Carbazole Linked Bisferrocenes," *Nanoscale*, vol. 4, no. 3, pp. 813–823, 2012.

[49] B. Hien, D. Quinci, Y. Ardesi, G. Beretta, F. Ravera, M. Walter, and R. Wille, "vlsi-nanocomputing/The-OpenSource-MolPDK: v1.0," Zenodo, 2024. [Online]. Available: https://doi.org/10.5281/zenodo.17311212

[50] B. Hien, D. Quinci, Y. Ardesi, G. Beretta, F. Ravera, M. Walter, and R. Wille, "Bridging the Gap Between Molecular FCN and Design Automation with SIM(7)-MolPDK: A Physically Simulated Standard-Cell Library," in *2025 IEEE Latin American Conference on Nanotechnology (IEEE LANANO)*, 2025.

Appendix A Tiles layout

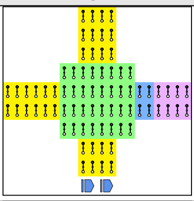
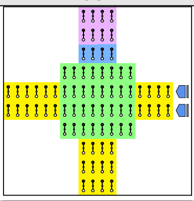
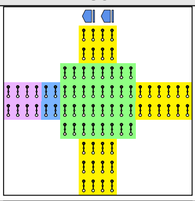
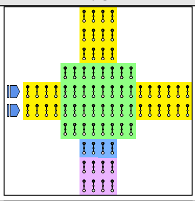
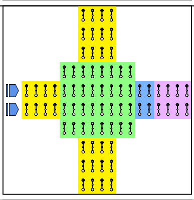
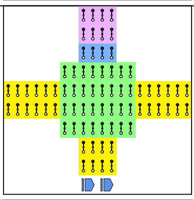
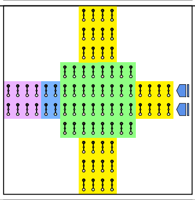
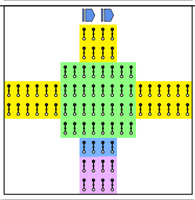
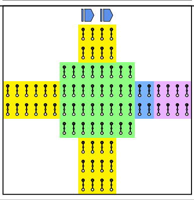
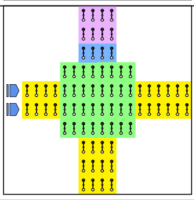
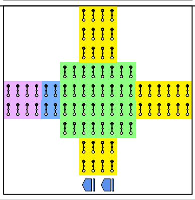
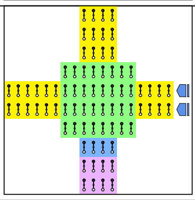
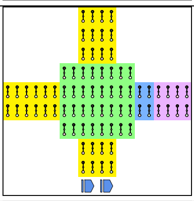
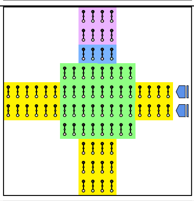
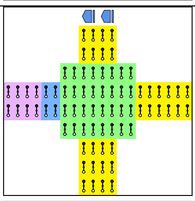
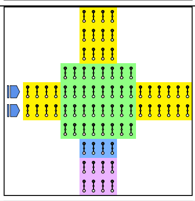
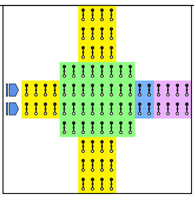
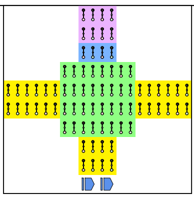
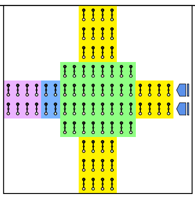
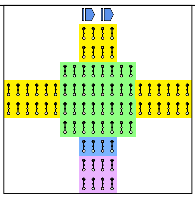
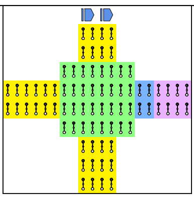
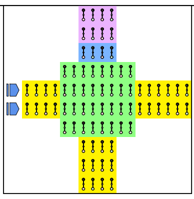
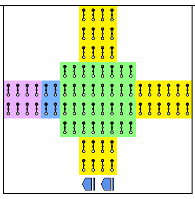
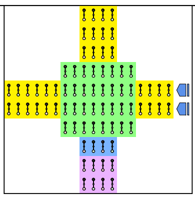
	Tile	0°	90°	180°	270°
AND	AND_DW				
	AND_LH				
	AND_UP				
OR	OR_DW				
	OR_LH				
	OR_UP				

Table A.1: Tiles - all orientations (Part I)

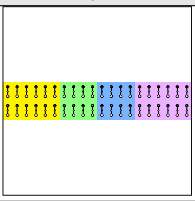
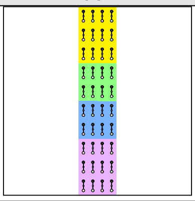
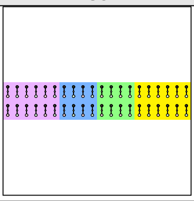
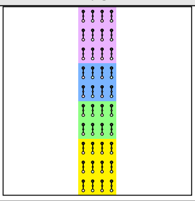
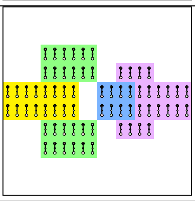
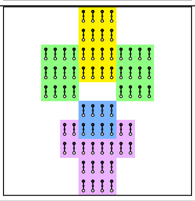
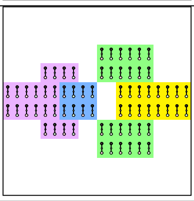
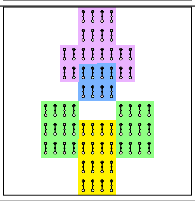
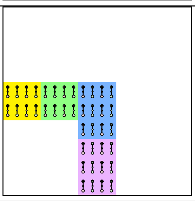
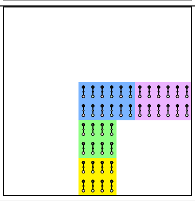
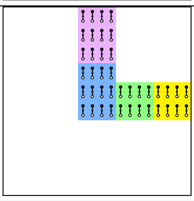
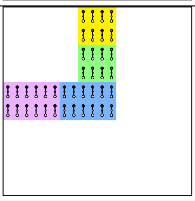
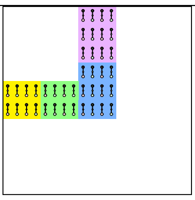
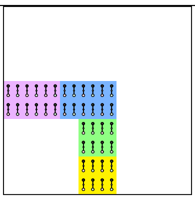
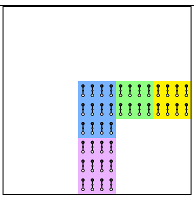
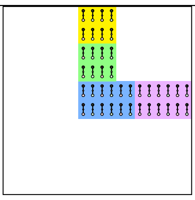
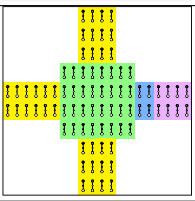
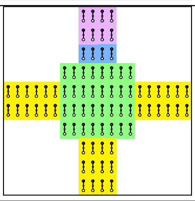
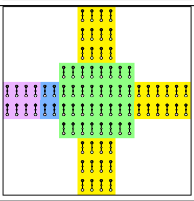
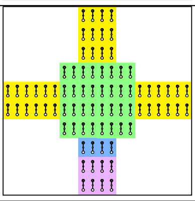
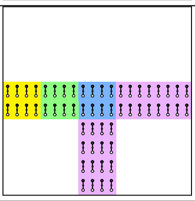
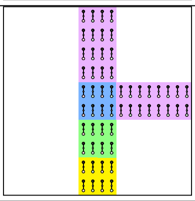
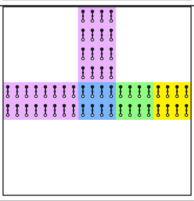
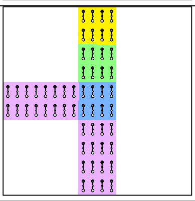
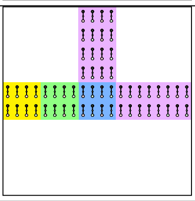
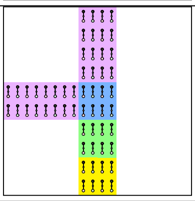
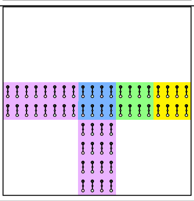
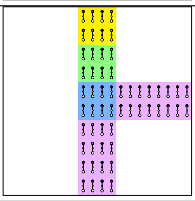
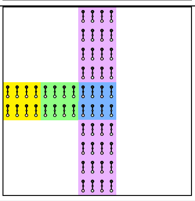
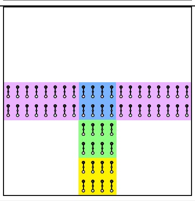
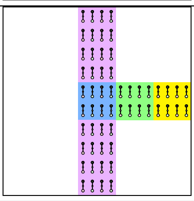
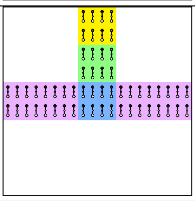
Tile	0°	90°	180°	270°
BUS				
INVERTER				
LWIRE_DXDW				
LWIRE_DXUP				
Majority Voter				
T_DXDW				
T_DXUP				
T_UPDW				

Table A.2: Tiles - all orientations (Part II)

Appendix B Tiles Waveplots

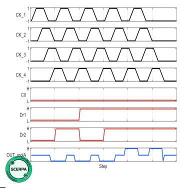
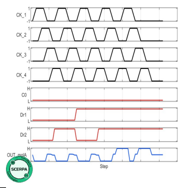
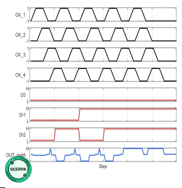
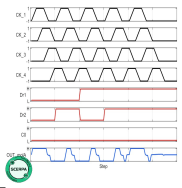
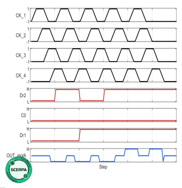
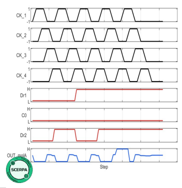
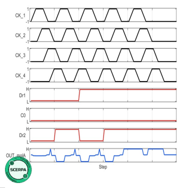
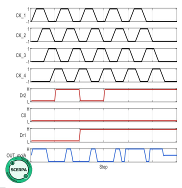
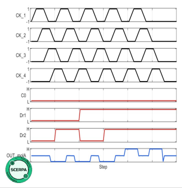
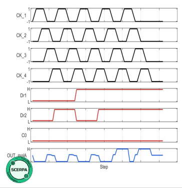
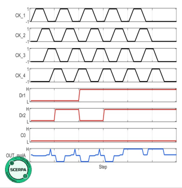
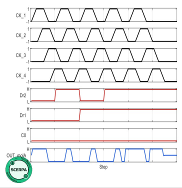
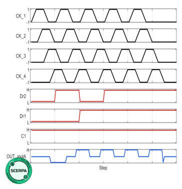
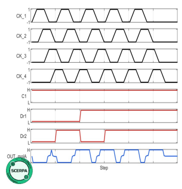
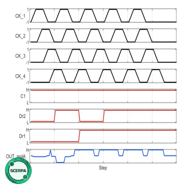
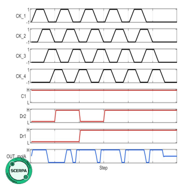
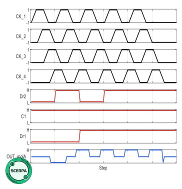
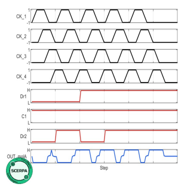
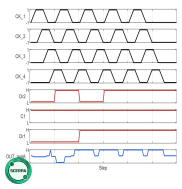
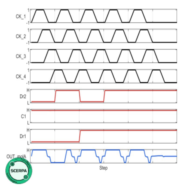
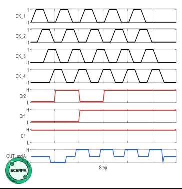
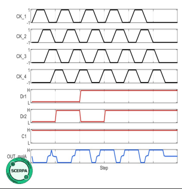
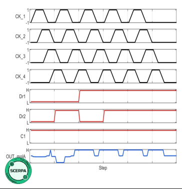
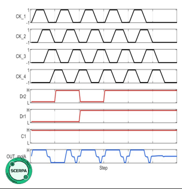
	Tile	0°	90°	180°	270°
AND	AND__DW				
	AND__LH				
	AND__UP				
OR	OR__DW				
	OR__LH				
	OR__UP				

Table B.1: Waveplot results (Part I)

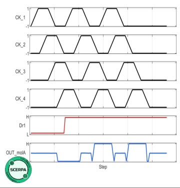
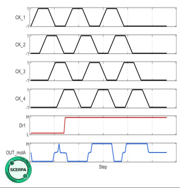
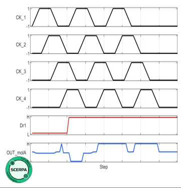
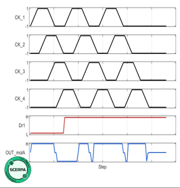
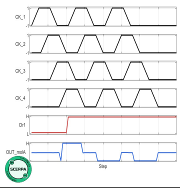
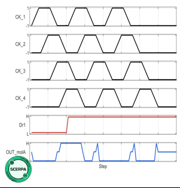
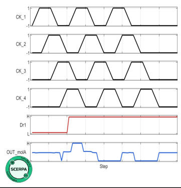
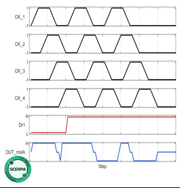
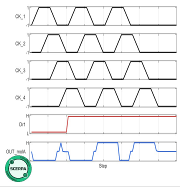
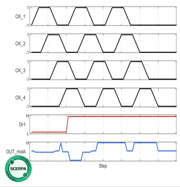
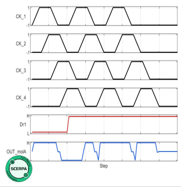
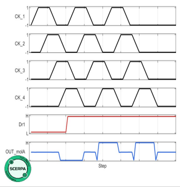
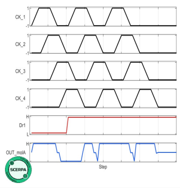
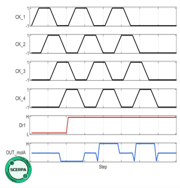
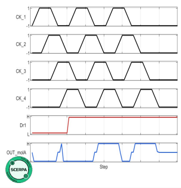
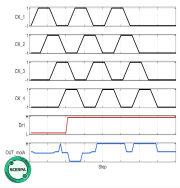
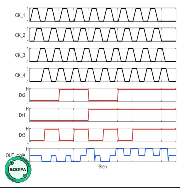
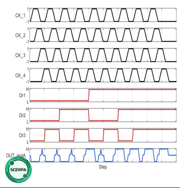
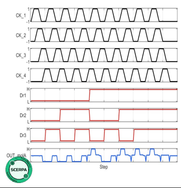
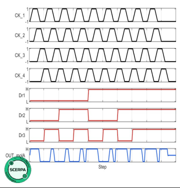
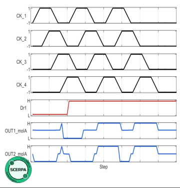
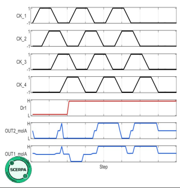
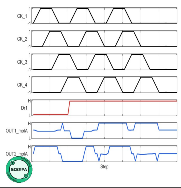
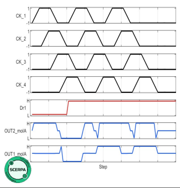
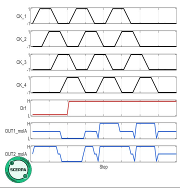
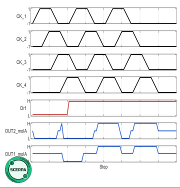
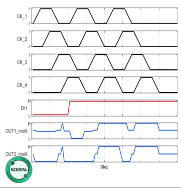
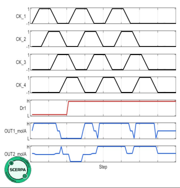
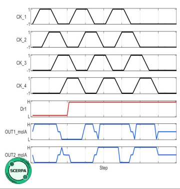
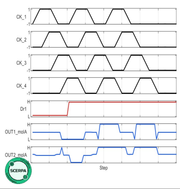
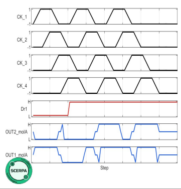
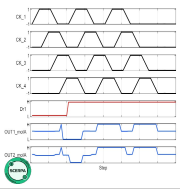
Tile	0°	90°	180°	270°
BUS				
INVERTER				
LWIRE_DXDW				
LWIRE_DXUP				
Majority Voter				
T_DXDW				
T_DXUP				
T_UPDW				

Table B.2: Waveplot results (Part II)

Appendix C   MATLAB script for SCERPA simulation

```
clear variables
close all

%% Clock parameters definitions
clock_low = -1;
clock_high = +1;
clock_step = 7;
clock_phases = 4;

%% Clock phases generation
pSwitch  = linspace(clock_low, clock_high, clock_step);
pHold    = linspace(clock_high, clock_high, clock_step);
pRelease = linspace(clock_high, clock_low, clock_step);
pReset   = linspace(clock_low, clock_low, clock_step);
pCycle   = [pSwitch pHold pRelease pReset];

%% Logic drivers definition
D0 = num2cell(+4.5*ones(1, clock_step*4));  % Logic '1'
D1 = num2cell(-4.5*ones(1, clock_step*4));  % Logic '0'
R0 = D1;  % Receiver '0'
R1 = D0;  % Receiver '1'
ND = num2cell(zeros(1, clock_step*4));      % No Drive

%% Layout (MagCAD)
file = '<device_name>.qll';  % Device layout file
settings.out_path = fullfile(pwd, '<device_name>');

%% Circuit configuration
circuit.dist_z = 10;                     % Molecules z-distance [nm]
circuit.magcadImporter = 1;              % Import from MagCAD
circuit.doubleMolDriverMode = 1;         % Double molecule driver
circuit.magcadMolOverwrite = 1;          % Overwrite molecule
   parameters
circuit.molecule = '62';                 % Ideal molecule
circuit.qllFile = fullfile(pwd, file);   % Layout file path

%% Driver configuration (multi-input device example)
circuit.Values_Dr = {
    'Dr1'   D0{:} D0{:} D1{:} D1{:} ND{:} 'end'
    'Dr2'   D0{:} D1{:} D0{:} D1{:} ND{:} 'end'
    % Add additional drivers if needed
};

%% Multi-zone clock configuration (pipeline)
circuit.stack_phase(1,:) = [pCycle pCycle pCycle ... pReset pReset
   ];
circuit.stack_phase(2,:) = [pReset pCycle pCycle ... pReset pReset
   ];
circuit.stack_phase(3,:) = [pReset pReset pCycle ... pCycle pReset
   ];
circuit.stack_phase(4,:) = [pReset pReset pReset ... pCycle pCycle
   ];

%% SCERPA settings
settings.doubleMolDriverMode = 1;
settings.damping = 0.6;
settings.activeRegionThreshold = 0.005;
settings.verbosity = 1;
settings.conv_threshold_HP = 0.005;
settings.enableRefining = 0;
settings.enableActiveRegion = 0;
settings.dumpDriver = 1;
settings.dumpOutput = 1;
settings.dumpClock = 1;
settings.dumpVout = 1;
settings.dumpComputationTime = 1;

%% Plot settings
plotSettings.plot_3dfig = 1;
plotSettings.plot_logic = 1;
plotSettings.plot_potential = 1;
plotSettings.plotSpan = clock_step;
plotSettings.fig_saver = 0;
plotSettings.plotList = 0;
plotSettings.plot_waveform = 1;
plotSettings.plot_waveform_index = 1;

% Copy output path if specified
if isfield(settings, 'out_path')
    plotSettings.out_path = settings.out_path;
end

%% SCERPA execution
this_path = pwd;
scerpa_path = fullfile('<path_to_scerpa>');
cd(scerpa_path)
SCERPA('generateLaunch', circuit, settings);
SCERPA('plotSteps', plotSettings);
cd(this_path)
```

Appendix D MATLAB script for BBChar simulation

```
clear variables
close all

%% Path definitions
myDataPath = '~';
BBcharPath = fullfile(myDataPath, 'BBchar');
BBcharCodePath = fullfile(BBcharPath, 'Code');
scerpaPath = fullfile(myDataPath, 'scerpa');
libraryPath = fullfile(BBcharPath, 'Lib');

outputPath = fullfile(BBcharPath, 'Layouts', '<device_name>');
file = '<device_name>.qll';  % Device layout file

%% Clock signal parameters
clock_low = -1;
clock_high = +1;
clock_step = 7;

% Clock phases generation
pSwitch = linspace(clock_low, clock_high, clock_step);
pHold = linspace(clock_high, clock_high, clock_step);
pRelease = linspace(clock_high, clock_low, clock_step);
driverPara.pReset = linspace(clock_low, clock_low, clock_step);
driverPara.pCycle = [pSwitch pHold pRelease driverPara.pReset];

%% Driver parameters
driverPara.doubleMolDriver = 1;
driverPara.Ninputs = <N>;  % Number of physical inputs in the
    layout
driverPara.driverNames = [{'in1'} {'in2'} ... {'inN'}];

% Input combinations matrix for exhaustive characterization
% Each row = one input, each column = one logic combination
driverPara.driverModes = [
    {'0'} {'0'} ... {'1'} {'1'};  % Input 1: all combinations
    {'0'} {'1'} ... {'0'} {'1'};  % Input 2: all combinations
    ...
    {'0'} {'0'} ... {'1'} {'1'}   % Input N: all combinations
];
```

```
% Available driver modes:
%   '1'          -> fixed value logic '1'
%   '0'          -> fixed value logic '0'
%   'sweep'      -> sweep from logic '0' to logic '1'
%   'not_sweep' -> sweep from logic '1' to logic '0'

driverPara.sweepType = 'lin';       % 'lin' or 'log'
driverPara.NsweepSteps = 1;
driverPara.cycleLength = length(driverPara.pCycle);
driverPara.clockStep = clock_step;
driverPara.NclockRegions = 4;       % Number of clock zones in the
    layout
driverPara.phasesRepetition = N;    % Clock zone repetitions
driverPara.maxVoltage = 1;          % Maximum driver voltage [V]

%% Termination settings
terminationSettings.enableTermination = 1;  % Add termination
terminationSettings.customLength = 0;       % Custom length (0=auto
    )
terminationSettings.busLayout = 1;          % 1=bus, 0=single line

%% SCERPA settings
circuit.qllFile = fullfile(pwd, file);
circuit.magcadImporter = 1;
circuit.doubleMolDriverMode = driverPara.doubleMolDriver;
circuit.outIsPin = 0;  % 0=last cell, 1=pin as output

settings.out_path = outputPath;
settings.damping = 0.6;
settings.verbosity = 0;  % 0=silent for batch characterization
settings.dumpDriver = 1;
settings.dumpOutput = 1;
settings.dumpClock = 1;
settings.dumpVout = 1;
settings.enableRefining = 0;

%% Display settings (disabled for batch)
plotSettings.plot_waveform = 1;
plotSettings.plot_waveform_index = 1;
plotSettings.plot_3dfig = 0;
plotSettings.plot_logic = 1;
plotSettings.plot_potential = 1;
plotSettings.plotSpan = clock_step;
plotSettings.fig_saver = 0;
plotSettings.out_path = settings.out_path;

%% Characterization settings
charSettings.LibPath = libraryPath;
charSettings.LibDeviceName = '<device_name>';
charSettings.out_path = outputPath;
```

```
%% BBchar execution
simulate = 1;          % 1=simulate with SCERPA
characterize = 1;      % 1=characterize and generate LUT

cd(BBcharCodePath)

% Automatic driver and clock construction from parameters
circuit.Values_Dr = buildDriver(driverPara);
circuit.stack_phase = buildClock(driverPara);

% Add termination if enabled
if terminationSettings.enableTermination
    [circuit, terminationCircuit] = add_termination(...
        circuit, terminationSettings, driverPara.pCycle, ...
        length(driverPara.pReset));
    circuit.qllFile = terminationCircuit.filepath;
end

cd(thisPath)

% Launch simulation and characterization
if simulate
    cd(scerpaPath)
    SCERPA('plotSteps', plotSettings);
    cd(thisPath)

elseif characterize
    cd(BBcharCodePath)
    tic
    characterization(charSettings, terminationSettings, ...
                     terminationCircuit, driverPara, circuit.
                        Values_Dr);
    charTime = toc;
    fprintf('Characterization␣completed␣in␣%.2f␣seconds\n',
        charTime);
    cd(thisPath)
end
```