Architecture Design for Analog Oscillatory Neural Networks

Corentin Delacour

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITE DE MONTPELLIER

En Systèmes Automatiques et Microélectroniques

École doctorale : Information, Structures, Systèmes

Unité de recherche UMR5506 (LIRMM)

Architecture Design for Analog Oscillatory Neural Networks

Présentée par Corentin Delacour

Le 19 Décembre 2023

Sous la direction de Aida Todri-Sanial et Nadine Azemard-Crestani

Devant le jury composé de

Jean-Michel Portal, Professeur, Aix-Marseille Université	Président
Nikhil Shukla, Assistant Professor, University of Virginia	Rapporteur
Ian O'Connor, Professeur, Ecole Centrale de Lyon	Rapporteur
Damien Querlioz, Chargé de recherche, Université Paris-Sud, CNRS	Examinateur
Fernando Corinto, Professeur, Politecnico di Torino University	Examinateur
Kerem Çamsarı, Assistant Professor, University of Santa Barbara	Examinateur
Aida Todri-Sanial, Professeur, Eindhoven Technical University	Directrice de thèse
Nadine Azemard-Crestani, Chargée de recherche, LIRMM, CNRS	Co-directrice de thèse
Thierry Gil, Ingénieur de recherche, LIRMM, CNRS	Invité

UNIVERSITÉ DE MONTPELLIER

Je dédie cette thèse à mes parents pour leur soutien permanent et leurs encouragements,
ainsi qu'à Fullmoona pour ses bonnes ondes.

ACKNOWLEDGEMENTS

To all my dear colleagues and friends who accompanied me during this journey, I give you my warmest thanks. Special thanks to my thesis supervisors Aida Todri-Sanial and Nadine Azemard for guiding me while giving me so much freedom to explore new ideas. I particularly thank Aida for putting scientific curiosity at the forefront and exploring uncharted paths. Through her inspirational teaching, I have realized there is nothing more important than remaining curious and creative while having fun during research. Above all, I thank her for the fantastic training and for the amazing opportunities to present this work; I feel grown up both professionally and at a personal level. I am also very grateful to Nadine for all the professional advice and for helping me during the thesis writing. It was so uplifting to have her feedback during this busy period. On the engineering side, I cannot thank Thierry Gil enough for his amazing support and for all the inspiring discussions we have had for more than three years. Thierry is the captain that everyone would love to have on board and was always here to bring the ship to its destination. Most of the thesis achievements would not have been possible without his FPGA expertise. I am also very grateful to the researchers Nikhil Shukla, Ian O'Connor, Damien Querlioz, Fernando Corinto, Kerem Çamsarı, and Jean-Michel Portal for honoring me as members of my thesis jury and for the very interesting exchanges we had during the defense.

A huge thanks to my colleagues and friends from LIRMM for the prolific scientific exchanges and the great moments we have spent together. In particular, I was fortunate to walk alongside Madeleine Abernot during this adventure. Madeleine has always been there to bounce ideas and save me from getting lost in the strange digital world. I also would like to give her a big shout-out for organizing so many events that greatly contributed to building our amazing corridor team. Also, this thesis's proof of concepts would not have been possible without the immense support from Laurent Deknyff. I deeply thank Laurent for sharing and applying his great PCB design expertise to make concepts a reality. I also thank Jeremie Salles for his support during the IC design, especially for the regular meetings and for his help just before the tape out. I express all my gratitude to Fathi Ben Ali who supported me when I was in a difficult situation. In a few hours, Fathi shared the invaluable tips that made the IC design a success.

I am also thankful to my colleagues from the NeurONN project for the fruitful collaboration during the past three years. I particularly thank Juan Núñez, Manuel Jiménez, María José Avedillo, and Bernabe Linares-Barranco from CSIC, IMSE in Sevilla for the regular interactions and interesting discussions about ONN. I thank them as well for their warm welcome during the final consortium meeting and for making it memorable. Many thanks to Elisabetta Corti, Olivier Maher, and Siegfried Karg from IBM in Zurich for sharing their expertise about oscillating vanadium dioxide devices and for the great collaboration. I also thank Ahmed Nejim from Silvaco for providing VO_2 compact models and for his great sense of humor.

Among our unique team at LIRMM, also known as the gourmet squad, I deeply thank Stefania Carapezzi for the fruitful discussions on the whiteboard and for explaining so well the complicated physics of devices. Above all, I thank Stefania for the joy and laughs she brought to the team, and for pushing colleagues to visit sushi restaurants, a passion that we both share. Many thanks to Gabriele Boschetto for his advice and for bringing such a good atmosphere into the lab and during our trips. Special thanks to Siyuan Niu for the great moments we have spent here in Montpellier and also in the bay. I also thank her for trying to demystify quantum computing for me, and I am looking forward to the next discussions. I thank Eirini Karachristou for the sunshine she has brought every day and all the organizing tips. Big thanks to Camille Couralet for being the coolest office neighbor and for always motivating the team to go out. Also, many thanks to Cécile Romane for bringing joy at every lunch and for all the advice.

Je remercie également Geneviève Carrière d'avoir fait vivre le couloir au quotidien et pour ses boosts énergétiques qui remontent le moral. Un grand merci aussi à Virginie Fèche pour sa bienveillance et son dévouement envers les doctorants. Sans oublier Ana Tacuri et Faiza Laachir que je remercie particulièrement pour leur assistance administrative et pour avoir organisé chaque mission. Merci à l'équipe STI-RX de m'avoir dépanné de nombreuses machines et en particulier Olivier Floucat qui a sauvé mon PC une semaine avant le rendu de thèse. Je remercie aussi mes ami(e)s du département microélectronique pour nos excursions salvatrices au R.A. suivies de l'indispensable pause-café. En particulier, merci Paul, Thomas, Pierre, Ismael, et Julien pour tous les bons moments passés en soirée où à la plage à vos côtés. Je remercie aussi Sarah de m'avoir invité en Nouvelle Zélande à l'occasion de sa future reconversion professionelle. Un immense merci à Elena pour son énergie positive qu'elle a si bien sû transmettre, et de m'avoir motivé à la course quand il le fallait.

Enfin et surtout, je remercie infiniment la famille et en particulier mes parents qui m'ont toujours soutenu et encouragé pendant les études. J'espère que vous verrez en cette thèse la continuation de mes loisirs créatifs avec juste un peu plus de maths. Merci mes frères pour tous ces bons moments qui rechargent les batteries, en particulier Joe pour les inoubliables sorties en montagne, et Antoine qui m'a donné goût à l'électronique analogique pendant qu'on bricolait des systèmes hi-fi. Merci à mes grands-parents pour leur soutien lorsque j'étudiais en région parisienne et pour les bons plats qui remontent le moral. Je remercie aussi Kerim et mes amis de palier pour les incroyables voyages de ces dernières années qui m'ont permis de déconnecter. Merci à Rémi pour toutes les sessions de glisse épiques, en particulier à Maguelone. Enfin, un grand merci à tous mes ami(e)s qui me suivent de près ou de loin depuis l'enfance. Un gros big up à Séda, Skaner, Mock, GZU, PLF, Gagui, Jo et Nilsou.

ABSTRACT

DIGITALIZATION of society creates important quantities of data that have been increasing at an exponential rate during the past few years. Despite the tremendous technological progress, digital computers have trouble meeting the demand, especially for challenging tasks involving artificial intelligence or optimization problems. The fundamental reason comes from the architecture of digital computers which separates the processor and memory and slows down computations due to undesired data transfers, the so-called *von Neumann bottleneck*. To avoid unnecessary data movement, various computing paradigms have been proposed that merge processor and memory such as neuromorphic architectures that take inspiration from the brain and physically implement artificial neural networks. Furthermore, rethinking digital operations and using analog physical laws to compute has the potential to accelerate some tasks at a low energy cost. This dissertation aims to explore an energy-efficient physical computing approach based on analog oscillatory neural networks (ONN). In particular, this dissertation unveils (1) the performances of ONN based on vanadium dioxide oscillating neurons with resistive synapses, (2) a novel mixed-signal and scalable ONN architecture that computes in the analog domain and propagates the information digitally, and (3) how ONNs can tackle combinatorial optimization problems whose complexity scale exponentially with the problem size. The dissertation concludes with discussions of some promising future research directions.

RÉSUMÉ DE LA THÈSE

La transformation de nos sociétés par le digital génère des quantités importantes de données dont la croissance a atteint une vitesse exponentielle depuis les dernières années. En dépit du progrès technique en matière de calcul, les ordinateurs digitaux actuels suivent difficilement cette tendance et sont dépassés par l'ampleur de certains problèmes, notamment liés aux algorithmes d'intelligence artificielle et aux problèmes d'optimisation de grande échelle. La limitation principale est liée à l'architecture même des calculateurs digitaux, à savoir la séparation du processeur et de la mémoire qui induit un ralentissement par le transfert des données, aussi appelée le *goulot d'étranglement de von Neumann*. Afin de contourner cette limitation, d'autres méthodes de calcul furent proposées distribuant le processeur et la mémoire telles que les architectures neuromorphiques basées sur l'implémentation de réseaux de neurones artificiels inspirés du cerveau. En outre, repenser la manière digitale de calculer comme par exemple utiliser les lois physiques et analogiques a le potentiel de réduire l'impact énergétique de certains calculs tout en les accélérant. Cette thèse a pour objectif principal d'explorer une approche physique du calcul fondée sur des réseaux de neurones oscillants (ONN) analogiques à faible coût énergétique. En particulier, ce travail se concentre sur (1) les performances d'une architecture ONN basée sur des neurones oscillants à partir de dioxyde de vanadium et couplés par des résistances, (2) une nouvelle architecture d'ONN à signaux mixtes calculant dans le domaine analogique, et propageant l'information de manière digitale afin de faciliter la conception à grande échelle, et (3) comment les ONNs peuvent résoudre des problèmes d'optimisation combinatoire dont la complexité croît de manière exponentielle avec la taille du problème. Pour conclure, de potentielles applications et futurs axes de recherche sont discutés.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Acronyms

AI	Artificial Intelligence.
ANN	Artificial Neural Network.
ASIC	Application Specific Integrated Circuit.
CB	Cross Bar.
CIM	Computing-In-Memory.
CMOS	Complementary Metal-Oxide Semiconductor.
COP	Combinatorial Optimization Problem.
CPU	Central Processing Unit.
DRAM	Dynamic Random-Access Memory.
DTC	Digital to Time Converter.
FN	False Negative.
FP	False Positive.
FPGA	Field Programmable Gate Array.
GDS	Graphic Design System.
GW	Goemans Williamson.
GWN	Gaussian White Noise.
HNN	Hopfield Neural Network.
HPC	High Performance Computing.
HRS	High Resistance State.
IC	Integrated Circuit.
ICT	Information and Communication Technologies.
IM	Ising Machine.
IMT	Insulator to Metallic Transition.
IO	Input-Output.
KKT	Karush Kuhn Tucker.
LagONN	Lagrangian Oscillatory Neural Network.
LEGION	Locally Excitatory Globally Inhibitory Network.
LRS	Low Resistance State.
MAC	Multiply and Accumulate.
Max-3-SAT	Maximum 3 Satisfiability.
MC	Monte Carlo.
MOM	Metal Oxide Metal.
MTJ	Magnetic Tunnel Junction.
NDR	Negative Differential Resistance.
NP	Nondeterministic Polynomial-time.
ODE	Ordinary Differential Equation.
OIM	Oscillatory Ising Machine.
ONN	Oscillatory Neural Network.

OPA	Operational Amplifier.
OTA	Operational Transconductance Amplifier.
p-bit	Probabilistic bit.
PCB	Printed Circuit Board.
PCM	Phase-Change Memory.
PDK	Process Design Kit.
PIM	Processing-In-Memory.
PLL	Phase-Locked Loop.
PPV	Phase Perturbation Vector.
PSD	Power Spectral Density.
RO	Ring Oscillator.
RRAM	Resistive Random-Access Memory.
SDE	Stochastic Differential Equation.
SDP	SemiDefinite Programming.
SHIL	Sub Harmonic Injection Locking.
SI	Strong Inversion.
SKONN	Saturated Kuramoto Oscillatory Neural Network.
SNN	Spiking Neural Network.
SOP	Synaptic Operation.
SR	Synaptic Range.
SRAM	Static Random-Access Memory.
STNO	Spin-Torque Nano Oscillator.
STT	Spin Transfer Torque.
TCAD	Technology Computer Aided Design.
TMO	Transition Metal Oxide.
TP	True Positive.
TSP	Traveling Salesman Problem.
VO_2	Vanadium Dioxide.
ZIF	Zero Insertion Force.

CHAPTER 1

INTRODUCTION

1.1 The Need for Rethinking Computation

TODAY'S societies are facing a huge demand for extended computing capability. Since the first industrial revolution enabled by steam-powered machines, technology has been shaping societies. The rate of change, however, has not been linear throughout history and has accelerated in the past decades. Not long ago, sending e-mails using the Internet was a revolution of its own whereas now, users can exchange and access *in real-time* any kind of information such as news, multimedia content, etc. In 2018, the production and use of information and communication technologies (ICT), represented more than 2% of global carbon emissions, which was similar to the aviation industry [1, 2]. While ICT has the potential to reduce carbon emission across other sectors [2], storing and processing data does have a cost. In 2018, it represented around 1% of the world's electricity just for data centers [1]. In 2020, it is estimated that around 4-6% of global electricity was consumed by the use of ICT [3]. Moreover, processing data using artificial intelligence (AI) becomes exponentially difficult. Since 2012, the number of operations to train AI algorithms has doubled every 3.4 months [4] with very large models like GPT-3 requiring more than 10^{23} training operations [5]. To put this number in perspective, using a single best-in-class chip computing 10^{14} operations per second [6] would require computing for more than 30 years. In this context, it becomes crucial to minimize the energy footprint of computation.

Unfortunately, machines generally have a complexity gap between generating and processing data. For instance, it is quite "simple" for a device such as a smartphone to take a photograph, whereas identifying objects or people in the photograph is a much "harder" task. Here "simple" and "hard" can be quantified as *processing time*, and hence energy. Time, energy, and space are the fundamental variables quantifying a computing device's efficiency. Looking at the evolution of electronics, the space variable has been optimized at an exponential rate thanks to semiconductor technology. Moore was indeed right when predicting that the number of transistors per unit area (elementary brick in today's computation) would double every two years [7]. Despite the mind-blowing progress achieved in chip design and fabrication (the 3-nm process node is now commercially available [8]), many tasks still remain costly for machines.

As an example, consider a self-driving car that has to spot people and objects while adapting and planning its navigation in real-time. Currently, the combination of these tasks is so hard

that the car's computer consumes about 850 W just for computation [9]. Compared to the 20 W that our brain dissipates on average, there is a clear performance gap between the best-in-class electronic systems and living intelligence. One could argue that the computational cost could be further reduced in the near future with more advanced technologies and chips. However, the challenge seems even bigger when thinking about how multitasking our brain is, compared to the very specialized chips such as the ones embedded in a self-driving car.

At first glance, we can wonder if this performance gap is due to a difference in the number of computing elements. Neuroscientists estimate the total number of neurons in the brain as 85 billion consuming 20 W of power [10]. In comparison, NVIDIA's recent H100 Tensor Core chip fabricated with a 4-nm process has about 80 billion transistors and dissipates up to 700 W [11]. Although neurons are different from transistors, the numbers of computing units in our brain and today's largest chips are comparable. It is then reasonable to believe that for some tasks, the brain's higher performance not only comes from the number of neurons but mainly from 1) its *organization* and 2) *the way it solves the problem*.

This second assumption further questions the definition of computation itself. *To compute* is defined as "to calculate an answer or amount by using a machine" [12]. In agreement with this definition, we generally think of computing as manipulating numbers, like we would do with pencil and paper, to solve a problem. This is similar to what digital computers do with bits '0' or '1'. However, at first sight, the brain does not seem to operate like a digital computer (mental calculating is actually hard for most of us). Hence the question: Do we always need to *calculate*, i.e. manipulate numbers, to solve a problem?

1.2 Computing with Neural Networks

In the brain, information is processed by neurons that are connected by synapses (Fig.1.1a). The field of *connectomics*, which focuses on studying neural interconnections, tells a lot about how brains differ from conventional digital computers [10]. The connectivity map of our brain consists of a highly complex network of 85 billion neurons, where on average each neuron is connected to about 1000 other neurons. In contrast, transistors in digital computers are generally locally confined in small circuits that only communicate with a few other neighbors.

The information encoding also differs. Neurons communicate with each other by sending electrical pulses or *spikes* propagated along the *axon* that are converted to chemical signals when reaching a synapse [13]. The information is then integrated into the *soma* of the receiving neuron, which produces a new spike when its membrane potential is above a threshold. Hence, compared to digital computing, the neurons also have *analog* variables, i.e. not only binary ones, which evolve according to differential equations [14]. Moreover, time plays a role in the brain operation as a burst of input spikes is more likely to produce an output spike compared to the same number of spikes stretched in time (leaky neuron integration).

Since the last century, scientists have been inspired by the brain principle to design new ways of computing. One of the first brain-inspired models of computation was proposed by McCulloch and Pitts in 1943 [15]. Motivated by the "all-or-none" type of information propagation (spike), their goal at the time was to model nervous activity with propositional logic. This led to the first representation of an *artificial neuron* that produces a logical output (true or false)

depending on its variables, the types of synaptic connections (excitatory or inhibitory), and the neuron threshold. Under certain conditions detailed in [15], the authors showed that a network of neurons can implement logical propositions and vice-versa.

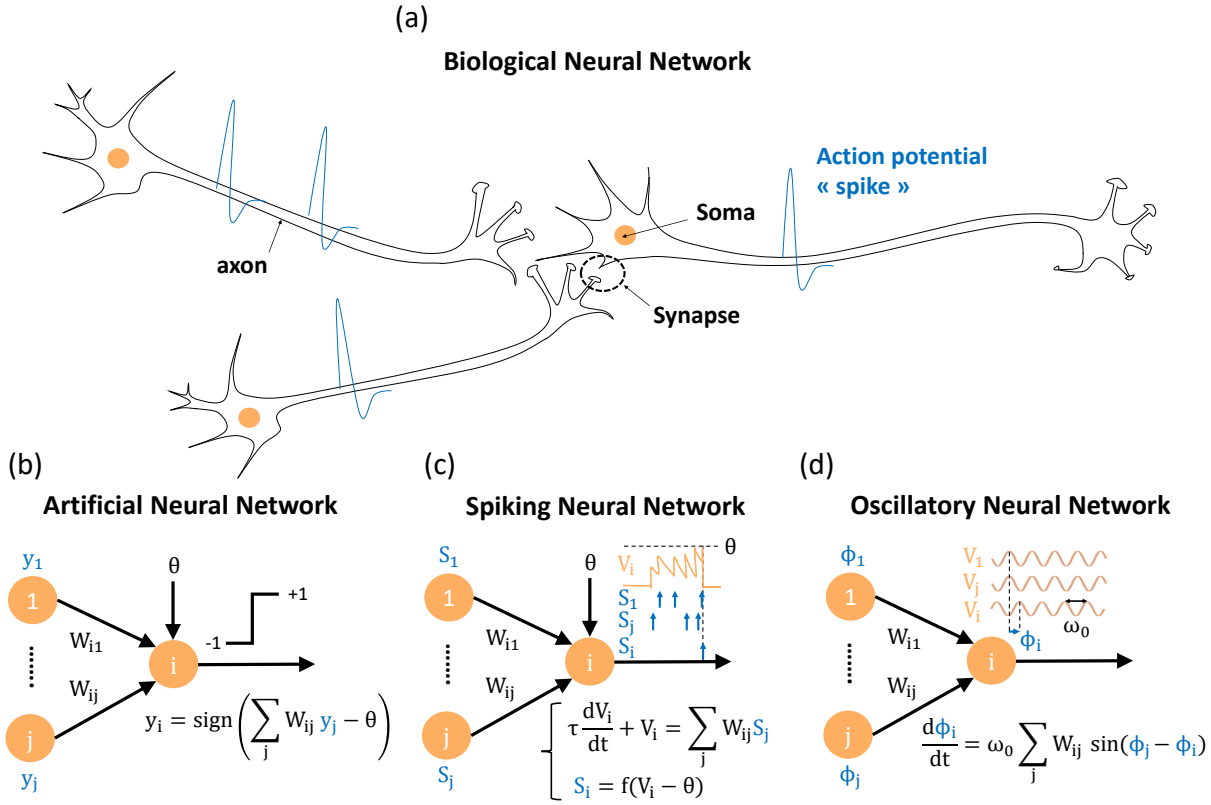


Figure 1.1: *a) Illustration of a biological neural network. Information is transmitted through action potentials that propagate along axons [13]. Note that the transmission at a synapse involves chemical exchanges. The information is finally integrated into the receiving soma which increases its membrane potential. When enough charge is accumulated (threshold), the neuron emits a spike. b) Model of artificial neural network (ANN) using perceptrons. It is time-agnostic and only models the neuron threshold and synaptic signs (inhibitory or excitatory behavior). c) Model of spiking neural network (SNN) using integrate-and-fire neurons. It describes the integration and emission of spikes. d) Model of oscillatory neural network (ONN). Neurons transmit oscillating signals that hold the information in their respective phase.*

In 1957, the first hardware artificial neuron called *perceptron* was proposed by Rosenblatt [16] and is conceptually shown in Fig.1.1b. Similar to a biological neuron, it consists of a processing unit that outputs '1', i.e. a spike if the input weighted sum exceeds a threshold. Otherwise, the output is '-1'. The input-output relation, here defined as the sign function, is called the *activation function* of the neuron. More than 60 years ago, Rosenblatt was already concerned by the limitations of digital computing to solve *associative* tasks, e.g. identifying an object from various angles, luminosity, etc., as the only solution at the time was to store all possible configurations. In his groundbreaking work [16], he remarks that having a limited memory for solving those tasks "seems to be incompatible with the nature of conventional computer systems", and thus proposes a network of perceptrons that can classify large sets of input data, without explicitly storing each element. With this model, Rosenblatt introduced the notion of network *capacity* which quantifies how many patterns, e.g. sensory data, the network can *learn* while accurately assigning inputs to the closest learned items. He also extensively

contributed to multi-layer perceptrons and for instance showed that a 3-layer network is a "universal system, for which a solution exists for every possible classification" [17]. More than 25 years later, Hornik further showed that adding a third layer of neurons enables a *universal approximation*, i.e. a 3-layer network can approximate any function with a given accuracy by adding a sufficient number of hidden neurons and having the right parameters [18].

These pioneer works are the foundations of today's *deep learning* techniques based on many layers of perceptrons (with various activation functions) connected in cascade [19] and generally trained using the back-propagation algorithm [20]. Modern neural networks can now solve a multitude of problems with high accuracy such as the classification [21] or generation of multimedia content [5], solving optimization tasks [22], discovering new drugs [23] or new algorithms [24], etc. Although artificial neural networks were initially brain-inspired, they have been mainly deployed on conventional digital computers which do not exploit the distributed organization of neural networks.

1.3 From von Neumann's to Neuromorphic Architectures

Most digital computers are based on the von Neumann architecture that consists of 1) an arithmetic unit 2) a control unit, 3) a memory, and 4) input/output units [25], as illustrated in Fig.1.2a. Like in Turing's abstract model of computation [26], the system operates in a sequential manner and is meant to execute an *algorithm*, i.e. a list of instructions for solving a problem. It started the era of digital computers that we know today. Despite the amazing versatility of the architecture (it can execute any program given enough memory), the execution speed is limited by data transfers between the memory unit and the processing unit, the so-called *von Neumann bottleneck*. Moreover, moving data also has an energy cost as in practice the bus wires have non-zero electrical impedance. When deploying neural networks on a von Neumann architecture, most of the energy dissipation actually originates from the data movement corresponding to the synaptic weights. For instance, the energy cost to fetch data from dynamic random-access memory (DRAM) can be as large as 200x the energy to compute in a 65 nm process [27].

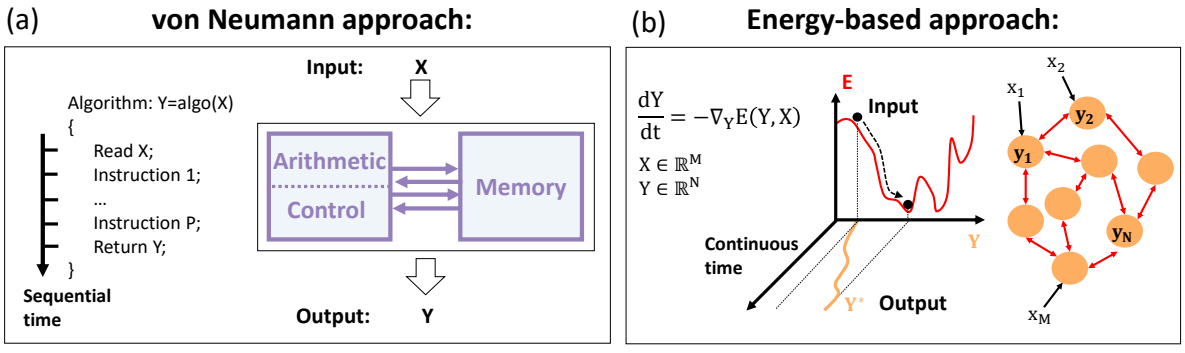


Figure 1.2: . *a) Example of algorithm and von Neumann architecture to execute it. Data related to 1) the problem and 2) intermediate results travel back and forth from the memory to the processor. b) Illustration of an energy-based computing approach. The problem is programmed by tuning the coupling elements (memory) between several processing units that consist of oscillators in this work. The input corresponds to the initial state $Y(t=0)$ set by X, or X itself. The results are encoded in the final state value Y^* reached at a minimum of of a scalar energy that is minimized in continuous time.*

As this energy loss is due to a physical separation between processor and memory, bringing the two together promises remarkable energy savings. It is the principle of *computing-in-memory* (CIM) that proposes to compute directly with memory elements without storing intermediate results [28]. CIM architectures have been used as accelerators for various applications using deep neural networks [29] as in principle, matrix-vector multiplication can be computed in a single step in the analog domain using Kirchoff's laws [30]. In essence, the idea is to physically encode the matrix elements in physical devices laid out in a cross-bar configuration, where the input vector consists of voltages applied to the rows, and the output vector is formed by column currents or voltages. The matrix memory elements can be implemented using many various devices having their pros and cons in terms of access time, endurance, power consumption, and precision [28]. Some of the most used and commercially available are charge-based circuits such as static random-access memory (SRAM), DRAM, flash memory, or resistive-based memories like resistive random-access memory (RRAM), phase change memory (PCM), spin transfer torque devices (STT), etc. [28, 29]. Note that the concept of bringing the memory as close to the processing unit is also applicable to digital architectures that are called processing-in-memory (PIM) architectures.

As highlighted by Mead in the 1980s [31], mitigating the data movement is the first but not the final step before reaching brain-like performances, and rethinking elementary computation is also necessary to further decrease the energy cost of computation. Mead stated that the combination of a CIM architecture, and "using the physics of the device to do the operation" (rather than using many transistors for a single operation), could bring up to six orders of magnitude energy improvement. Mead proposes to harness analog computation primitives that are different from logical operations and rather based on signal amplification (exponential activations), time-domain integration, and addition, similar to neural networks in the brain.

Such an approach is called *neuromorphic* and englobes various domains from material, devices, circuits, architecture, and algorithms, up to the applications [32]. The end goal of neuromorphic computing is to build an efficient physical implementation of an artificial neural network, where synapses and neurons are spatially distributed as in the brain. Such an ambitious and fascinating approach led to the development of *spiking neural networks* (SNN), sometimes called "the third generation of neural networks" [33] (the first started with the binary perceptron [15, 17] and the second with neurons having continuous activations). Essentially, SNNs mimic the spiking behavior of biological neural networks to some extent with various degrees of fidelity [33]. The key difference with previous approaches is that SNNs have the additional *time* variable as spiking neurons integrate their inputs in the time domain (Fig.1.1c). From this perspective, the static outputs of conventional ANNs can be thought of as the *mean firing rate* of an SNN. In fact, it has been shown that *rate-based* SNNs, i.e. which encode information in the spike frequency, do not bring a significant performance improvement compared to feedforward ANNs [34, 35]. Instead, SNNs have the potential to take advantage of richer information encoding in the time domain such as time-to-first spike (TTFS) that encodes information in the first spike arrival time and is at least 4x more efficient than rate coding for image classification [36].

Various large-scale SNN systems have been designed with different levels of biological plausibility, depending on the applications [32]. There are fully digital systems dedicated to brain simulation such as the SpiNNaker digital architecture [37] that can model up to 10^9 neurons with 10^{12} synapses using "medium-complexity" biological models and distributed on 10^6 cores. Also for neuroscience emulation, systems like BrainScaleS [38], Neurogrid [39], or

ROLLS [40] are closer to biological SNN as they involve complex neural emulators based on analog circuits. However, compared to digital approaches, adapting and handling different time scales can be challenging when using these SNNs for diverse applications [32]. Thus, instead of manipulating analog signals, some SNN chips use simplified digital neural responses to facilitate scaling and programmability with dedicated software. Two famous examples are True North [41], and Loihi [42] chips developed by IBM and Intel which have 4096x256 and 128x1024 cores x neurons in a single chip, respectively. Although these approaches can be thought of as neural simulation rather than emulation, they enable remarkable performances when processing *event-based* sensory data, i.e. the data is transmitted at the time there is a change in its value, such as event-based cameras or tactile sensors used in robotics [35, 43]. Moreover, owing to their rich temporal dynamics, SNNs excel at solving problems having time dependencies such as *recurrent* networks, i.e. networks that have synaptic feedback loops, or combinatorial optimization problems using the spike dynamics and stochasticity to explore energy landscapes [35, 44].

1.4 Using Physics to Solve Hard Problems

Time is a powerful ally for neural networks as we just saw that SNNs can be more efficient than von Neumann's computers for solving tasks involving vision, perception, or even optimization. At the heart of SNNs, there are ordinary differential equations (ODE) with respect to time that determine their dynamics (note that stochasticity can further be included [44]) rather than arithmetic circuits. Thus, neuromorphic computing is a remarkable example highlighting that computing in *continuous-time* can be more efficient than sequential digital calculation such as defined by Turing [26]. We can then wonder: Are there other *physical dynamical systems* that can be more efficient than digital computers? Note that this does not question the use of digital computation in general, as it is an extremely powerful concept that can in principle solve any problem as long the solving procedure can be described by an algorithm, also known as the Church-Turing thesis [26].

The first outstanding analog computer that comes to mind is the *quantum computer* [45]. It is based on quantum mechanics, in particular the law of *state superposition*, i.e. the quantum system is simultaneously in a superposition of states where, in the context of computation, each state corresponds to a computational result [46]. Intuitively, it is as if the quantum computer would consider "all possible paths simultaneously" [46]. The reason is that compared to a 1-dimensional bit in digital computers, a *qubit* can be described by a 2-dimensional unit complex vector $\Psi = a|0\rangle + b|1\rangle$ where $|0\rangle$ and $|1\rangle$ are basis vectors. Adding a qubit to the system doubles the space dimension so that an N-qubit state becomes a linear combination of 2^N basis vectors. The "trick" of quantum computing is to exploit this high dimensionality to accelerate computations. Similar to the Turing machine, an abstract computation model for quantum computers was proposed by Deutsch [45] whose state is defined by a basis vector. The quantum Turing machine is currently the only model of computation showing an *exponential speedup* when solving some challenging tasks for von Neumann computers such as integer factorization using Shor's algorithm [46]. Unfortunately, quantum computers currently come with an important circuit overhead such as cryogenic cooling for superconducting qubits, control, and error mitigation circuits which overall dissipate several kW of power [47]. Hence, the current focus for quantum computers is high-performance computing (HPC) rather than solving problems in

embedded systems like robotics.

What makes quantum computers unique is their exponentially large space with the number of qubits. However, they are not the only physical systems harnessing some kind of parallelism to compute. In fact, there are many physical-based approaches that can solve problems based on variables that *naturally* evolve (due to physical laws) and interact simultaneously [48]. The underlying principle of non-quantum physical-based computers is to encode the solutions to a problem as *attractors* of a dynamical system. In other words, the physical system is designed in such a way that its state naturally *flows*, i.e. evolves toward a particular region of space that encodes the solution [49]. This natural motion can also be thought of as parallel processing as generally, all components of the state vector evolve simultaneously in continuous-time. In the case of SNNs, the state vector would be composed of the neuron variables such as membrane potentials. The flow would then be determined by the synaptic connections between the neurons, the neuron input-output function, and how neurons are initialized. Compared to quantum computers, however, adding a neuron to the network does not double the space size but instead adds a single extra dimension. Some models of computation and their corresponding hardware are illustrated in Fig.1.3.

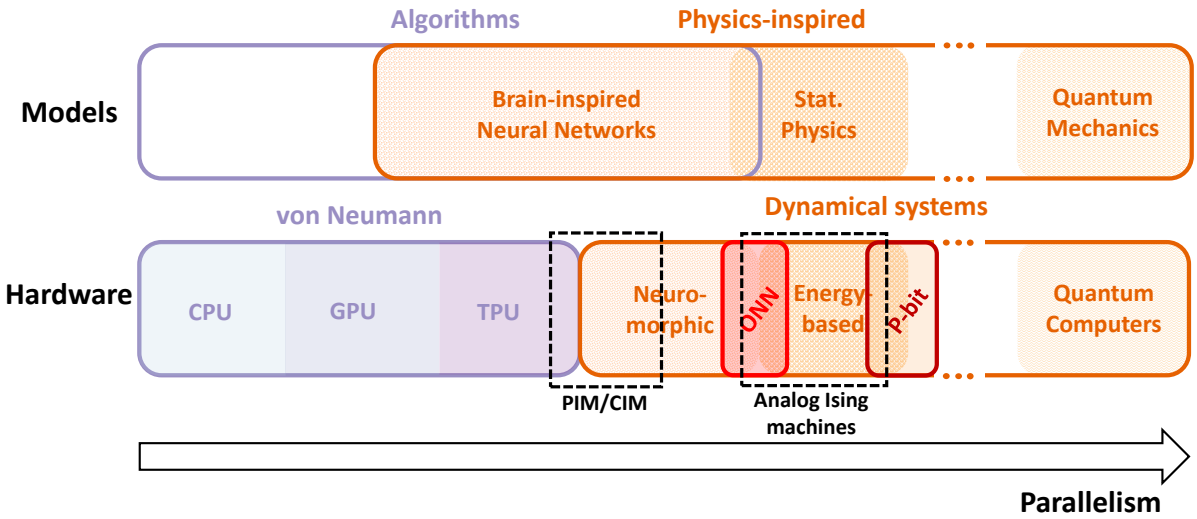


Figure 1.3: . *Main models used for computing and their corresponding hardware. Most of today's computers execute algorithms that are sequential by definition. Note that the classification is not exhaustive and algorithms can be executed on many more custom architectures such as ASICs. Similarly, the set of possible dynamical systems used for computing is very large. Most of these approaches can be described by ordinary differential equations that define the computation process. ONN has roots in neuromorphic computing but can also be described as an energy-based physical system.*

Although not necessary, an interesting particular case occurs when the dynamics of the physical system can be interpreted as the minimization of some *energy landscape*, i.e. a scalar quantity that is a function of the state values, and which can be thought of as a high-dimensional surface in the state space. Having this idea in mind, the state motion can then be interpreted as a "ball" that naturally "rolls" following the landscape curvatures (Fig.1.2b). Imagine now that the problem to solve is an optimization task having various solutions whose quality is quantified by a cost function depending on the problem variables. By mapping the cost function to the energy landscape, and the variables to some components of the state vector, one can then obtain

a solution when the "ball" stops at low energy values. Currently, there are multiple existing systems called *Ising machines* (IM) governed by different physical laws which follow a common goal: minimizing the *Ising energy* [50]. This concept was named after the physicist Ernst Ising [51] who studied ferromagnetic materials in the 1920s. It describes how interacting particles with 1-dimensional magnetic moments (spins $S_i = \pm 1$) tend to reach a state that minimizes the Ising Hamiltonian H defined as:

$$H = -\sum_{i,j} J_{ij} S_i S_j - \sum_i h_i S_i \tag{1.1}$$

where J_{ij} models the interaction between particles i and j, and h_i is the *external field*, i.e. a force that favors a spin value for particle i. This model is of particular interest as the problem of finding the spin values that minimize H for a 3-dimensional lattice is *as hard* as any nondeterministic polynomial-time (NP)-complete problem [52, 53], for which there is no known algorithm running in polynomial time [54]. For instance, consider the traveling salesman problem (TSP) which given a list of cities on a map, consists in finding the shortest tour that visits every city and returns to the starting city. A brute force algorithm would test all possible $(N-1)!/2$ tours, a number that grows exponentially with the number of cities N. However, it is clearly not realizable for a relatively large number of cities like $N = 100$, as it would involve way more operations than the total number of atoms in the observable universe (estimated around 10^{80}). Despite the existence of better exact algorithms like *branch and bound* or *backtracking* which avoid testing every single tour, the worst-case number of operations still scales exponentially [55].

In the 1980s, Tank and Hopfield [56] proposed to solve TSP with a physical artificial neural network (made of electrical components) whose energy landscape corresponds to the Ising energy when neurons take binary values. This groundbreaking work triggered the study of various physical systems for solving the Ising problem. Remarkable examples are coherent Ising machines that use laser pulses to emulate up to 10^5 Ising spins [57, 58], quantum annealers based on the adiabatic theorem to find the ground states of H [59], or probabilistic-bit (p-bit) computers [60, 61] that implement *Boltzmann machines* [62] where bi-stable stochastic p-bits devices randomly switch such that the probability distribution of the global spin state follows a Boltzmann distribution $\propto \exp(\eta H)$. Interestingly, η is the "inverse temperature" and can be used for *simulated annealing* [63], i.e. to emulate the temperature decrease that reduces the particle agitation until "freezing" to a fixed state of low energy. However, one important restriction for reaching the desired Boltzmann distribution is to *sequentially* update the p-bit values [62, 64], as Boltzmann machine inferences are based on Gibbs sampling, a particular instance of Markov chain Monte Carlo algorithm [65]. Nevertheless, as stated by authors in [60] and [66], an analog p-bit computer could be efficiently implemented at a very large scale as noisy devices such as magnetic tunnel junctions (MTJ) would "naturally lead to an asynchronous updating of p-bits in the absence of a global clock signal".

Another promising low-power approach is to use classical **coupled oscillators** that can also be integrated into embedded systems at room temperature and do not require bulky equipment [67]. At first glance, oscillating devices seem similar to p-bits as by definition they are unstable and oscillate between two maximum values. However, in general, the dynamics of coupled oscillators are deterministic and can be interpreted as continuous trajectories in their state space rather than a stochastic search [68]. Interestingly, when oscillators have a uniform frequency and couplings are symmetric, i.e. $J_{ij} = J_{ji}$, the phase motion minimizes an analog relaxation E of the Ising energy H (such that $H \subseteq E$) [69]. Hence, coupled oscillators are similar to

continuous Hopfield neural networks (HNN) [70] and can solve various problems by "shaping" the energy landscape such that the minima correspond to solutions. In the context of AI, this process would be called *training*, while for solving optimization tasks the energy landscape would capture the problem *constraints*. Despite the limited use of continuous HNN in modern AI models, its computational power has been intensively studied [70–73] and proved to be at least as powerful as space-bounded Turing machines and can simulate "any converging discrete-time computation" [73]. Nowadays, the oscillatory implementation and in particular the recent development of *oscillatory Ising machines* (OIM) [69] sheds new light on a particularly efficient approach for solving hard optimization problems [74].

1.5 Oscillatory Neural Networks

1.5.1 Brief ONN Introduction

Due to the distributed architecture of coupled oscillators, and because oscillating signals can also be measured in nervous systems [75], coupled oscillators are sometimes called *oscillatory neural networks* (ONN) [68]. ONNs are known for their:

1. **Computational interest**: One can exploit the ONN synchronization mechanism to compute by encoding the information in frequency or phase between oscillators. In the latter case and with symmetric coupling ($J_{ij} = J_{ji}$), their phase dynamics minimize an energy function that is equal to the Ising energy when phases take binary values [68].
2. **Potential large-scale fabrication**: ONNs can be built from classical electronic oscillators at room temperature [67].
3. **Possible efficient operation**: Oscillators running at high frequency could solve problems at very low energy cost. For instance, a current CMOS-ONN running at 1 GHz dissipates only around 10 fJ per oscillation [76].

Since Huygens who first observed synchronization between coupled pendulum [77], researchers like von Neumann [78] and Goto [79] have studied the computational properties of coupled oscillators and first engineered them to compute Boolean functions in the 1950s. Labeled *parametron*, Goto's oscillatory circuit was used as a majority gate to solve Boolean logic, and machines with up to 9600 parametrons were used in the early 1960s. Note that this approach has been recently reconsidered using recent oscillator designs [80, 81]. However, at that time, oscillatory-based logic could not compete with more scalable transistor-based logic and disappeared during the following decade. A few years later, the pioneering work from Hopfield [70, 82] has brought neural networks together with concepts of statistical physics describing spin-glass systems with the Ising model [51]. This triggered the rebirth of oscillatory-based computing paradigms that have been shown to minimize Ising-like energies [68, 83] and follow remarkable dynamics derived by Kuramoto a few years earlier [84]. Nowadays, the design of coupled oscillator systems is driven by the development of novel oscillator circuits that promise higher scalability compared to classical LC-based oscillators. Researchers are investigating

transition-metal-oxide devices (TMO) [85, 86], spintronic devices [87], microelectromechanical systems (MEMS) [88], optical systems [58], or CMOS accelerators [74, 89, 90] as potentially scalable solutions [91] for applications like digital logic [80, 81], classification [92–94], computing convolutions [85, 95], or solving associative tasks [68, 89, 96–98] and combinatorial optimization problems (COP) [69, 86, 99, 100]. Fig.1.4 shows a timeline of oscillatory-based computing systems.

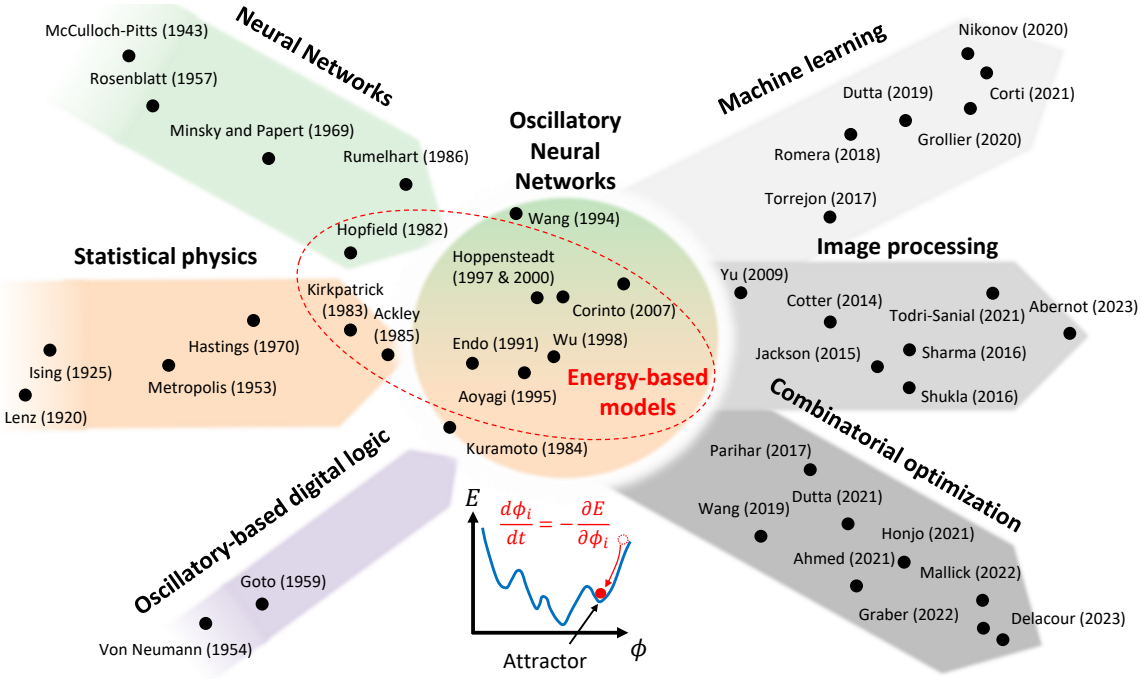


Figure 1.4: *Timeline of the ONN computing paradigm highlighting the inspirations and current trends.*

1.5.2 Computing with ONN

ONN computing is based on the synchronization of coupled oscillators, a phenomenon that naturally appears in biology, physics, or social interactions [101] and was first conceptualized by Winfree in the 1960s [102]. To illustrate oscillator synchronization, consider a crowd of people applauding during an event. The reader probably experienced a situation where the initial state is quite chaotic but after a sufficient time, most of the participants end up applauding in synchrony. From a coupled oscillator network perspective, one can model each individual as an oscillator adjusting its frequency, i.e. increasing or reducing the clapping rate to get synchronized with the neighboring crowd [103]. The Kuramoto model predicts such phenomenon [84] which expresses the time derivative of oscillators' phases, i.e. instantaneous frequencies as:

$$\frac{d\phi_i}{dt} = \omega_i + \sum_j K_{ij} \sin(\phi_j - \phi_i) \tag{1.2}$$

where ω_i is the oscillator free-running frequency. The sinusoidal interaction terms are responsible for the frequency adjustment and model the adaptation of individual i to other clapping agents j in our example. Despite its simple expression, the Kuramoto model produces very complex dynamics depending on the connectivity (K_{ij}) and frequency distributions [104]. The model describes a large variety of rhythmic behaviors [105], including ONN dynamics.

The non-linear dynamics of coupled oscillators such as (eq.1.2) can be harnessed in many different ways to perform intelligent tasks. However, most ONN developments fall into two classes depending on the type of input/output encoding:

1. **Frequency-based ONN**: inputs are oscillator frequencies and outputs are the synchronization levels between oscillators.
2. **Phase-based ONN**: oscillators have the same frequency and input/output are encoded in phase between oscillators.

In a frequency-based ONN (Fig.1.5a), frequency-dependent input signals are injected into the ONN that reacts to the input perturbations. The computation outcome consists of groups of oscillators that lock in frequency, i.e. are synchronized. This computation scheme has been used for image processing [95, 96, 106, 107], associative memory tasks [97, 108], or spoken vowel classification [94, 109].

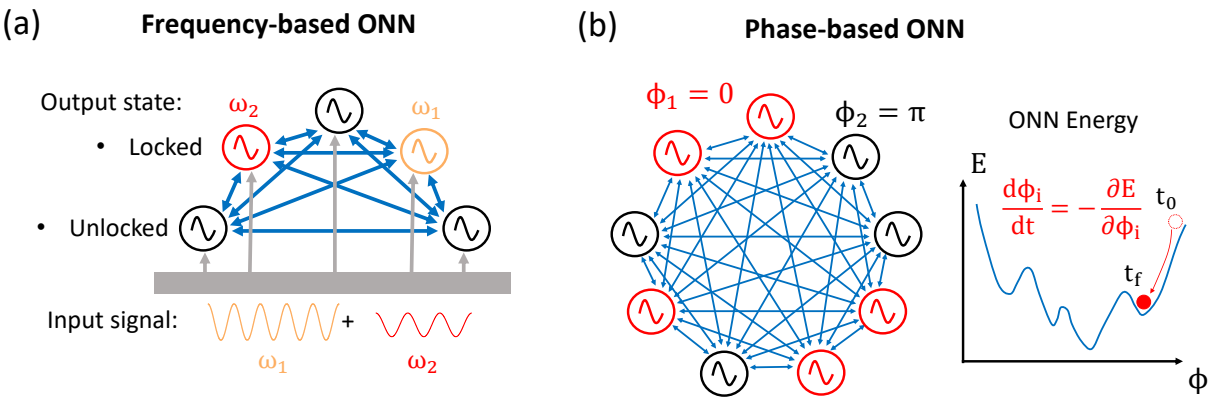


Figure 1.5: *a) Illustration of a frequency-based ONN. The input consists of frequency-dependent signals fed into the ONN. The latter can have oscillators locking to a common frequency and providing the computational result. b) Schematic of a phase-based ONN. The oscillators have a uniform frequency and evolve in the phase domain, minimizing a kind of Hopfield energy. Both modes are generally based on synchronization, i.e. oscillators are locked in phase and/or in frequency.*

In the special case where oscillator frequencies are identical and coupling elements are symmetric ($K_{ij} = K_{ji}$), Hoppensteadt and Izhikevich [68] have shown that the ONN is a gradient system that minimizes an energy function through time as:

$$\begin{cases} d\phi_i/dt = -\partial E/\partial \phi_i \\ E = -\frac{1}{2}\sum_i \sum_j K_{ij} \cos(\phi_i - \phi_j) \end{cases} \tag{1.3}$$

In this regime, the inputs and outputs are encoded in phase between oscillators whose dynamics depend on the ONN energy and phase initialization (Fig.1.5b). It turns out that the ONN energy is similar to Hopfield's energy [82] and to the Ising Hamiltonian [69], enabling a broad range of applications from machine learning and image processing to combinatorial optimization.

1.5.3 ONN Design Challenges

Despite the well-established ONN formalism and the various promising implementations, we identify three main *practical* implementation challenges for phase-based ONNs:

1. **Mapping challenge:** Given a physical ONN implementation, to which extent can we use conceptual models (such as Kuramoto's (eq.1.2)) to model the ONN computation? Moreover, how to map conceptual synaptic weights to actual hardware components?
2. **Scaling challenge:** How to ensure robust dynamics and scale analog ONNs? Today's problems involve thousands, even millions of variables. What kind of architecture and control strategy could enable such large-scale implementation in *continuous-time*?
3. **Operational challenge:** How to escape undesired energy minima? There is not yet rigorous formalism unifying *a priori* two different concepts: the continuous ONN dynamics vs. simulated annealing. Are there other deterministic approaches to escape local minima?

1.6 Thesis Contributions

The current thesis focuses on energy-efficient **electrical ONN architecture design** with the objective of exploring ONN computing capabilities and taking aim at the three previous challenges (Fig.1.6). Overall, the research work can be grouped into three main chapters, each one proposing various ONN architectures and techniques aiming to cope with these problems.

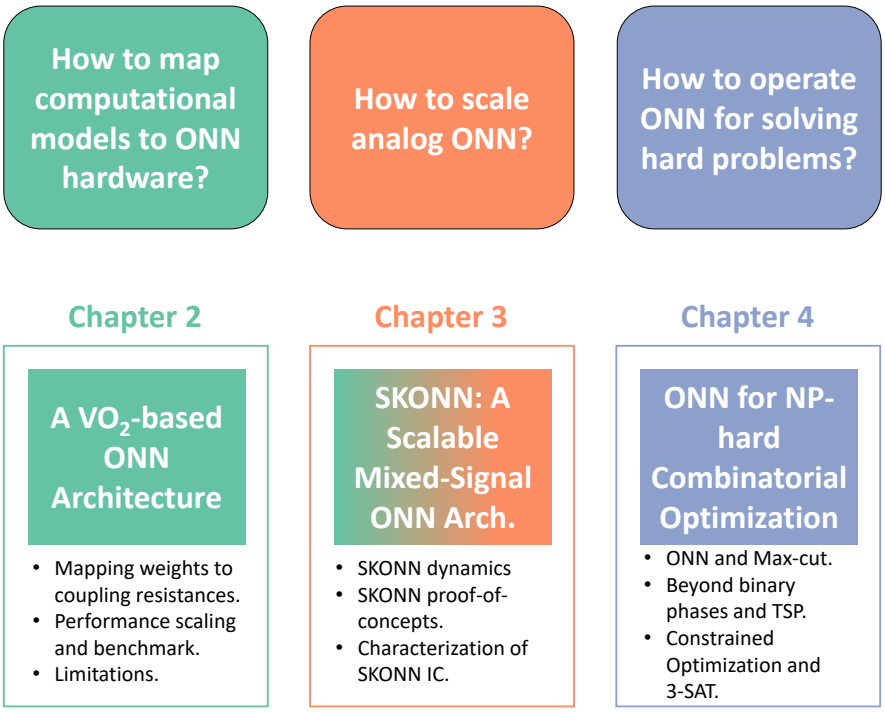


Figure 1.6: *Main thesis contributions.*

1.6.1 A VO_2-based ONN Architecture with Resistive Coupling

In Chapter 2, we explore a simple yet promising **VO_2-based ONN architecture** composed of resistively coupled relaxation oscillators made of vanadium dioxide devices (VO_2) which are transition metal oxide devices (TMO) [85, 86, 110]. Acting as hysteresis switches, VO_2 devices trigger interest since a single device in series with a transistor can produce oscillations at room temperature [110]. Despite its compactness, a VO_2-ONN has complicated dynamics and there is no clear mapping between existing ONN formalism and the hardware. In particular, mapping weights to coupling resistance values is currently performed by trials and errors, thus preventing the design and learning of large-scale VO_2-ONNs. Consequently, it is unknown how the VO_2-ONN memory capacity scales with the number of oscillators. Further, the ONN performance scaling such as computation time and energy consumption remain undetermined.

To enable a more systematic design, we combine the HNN formalism with the specific VO_2 dynamics to derive a *mapping function* which assigns conceptual weights to coupling resistance values. Similar to HNN [82], we find in simulations that a fully connected VO_2-ONN trained with the Hebbian learning rule has a similar memory capacity that scales linearly with the number of oscillators N, i.e. $C \propto 0.15N$. Then, we evaluate the VO_2-ONN performance scaling at the device and architecture level. We find that VO_2 devices in crossbar configuration undergo a quadratic energy decrease when scaling down the device dimensions. At the architecture level, the high ONN parallelism allows computing in quasi-constant time and enables a linear energy scaling with increasing coupling resistance values. Finally, we benchmark the unconventional approach with state-of-the-art neural accelerators and neuromorphic chips at the neuron level. Our study indicates that the intrinsic operation of VO_2-ONN can be competitive for oscillating frequencies above 100 MHz.

Despite these promising simulation results, we identify two main limitations when using a resistively coupled VO_2-ONN. First, the coupling resistances producing the best accuracy only differ by 10% at maximum, which would challenge practical implementation, especially with novel memristive devices. Second, the binary-phase locking property is only possible when the oscillator is biased at the edge of the oscillation region, which is non-robust as VO_2 device variations could prevent oscillations.

1.6.2 SKONN: A Scalable Mixed-Signal Approach

In Chapter 3, we propose a novel mixed-signal ONN architecture labeled *Saturated Kuramoto ONN* (SKONN) that mitigates some of the previous concerns. There are three main motivations behind the proposed architecture. In response to the mapping and scaling challenges, SKONN has:

1. **Controllable analog dynamics** while computing in continuous time.
2. **Digital propagation** to ease scaling and interfacing with digital circuits.
3. **A simple synaptic design** such as a linear weight mapping with a large synaptic range.

SKONN consists of relaxation oscillators that feed their digital waveform via **synaptic capacitors** to the analog input of the receiving oscillators. Such architectural choice lead to the following interesting features:

- **At the hardware level**: By construction, the information propagates in one direction thus enabling *feedforward* ONNs that we demonstrate with a XOR example. Moreover, in the context of Ising machines, feedforward synapses can be used to implement the external field term h_i (eq.1.1). Also, oscillators are AC-coupled and are then more robust to any variation of the oscillator operating point. Moreover, synapses can easily be programmed using banks of switched capacitors.

- **At the computational level**: SKONN electrical dynamics can be reduced to phase dynamics with high fidelity. The obtained phase dynamics are very similar to the Kuramoto model (eq.1.2), except for the sinusoidal term that becomes *saturated*, i.e. a square function. Interestingly, SKONN's energy landscape in its phase state is closely linked to the best-known approximation algorithm for solving the NP-hard Max-cut problem [111]. Finally, SKONN's phases tend to naturally reach binary phase fixed points, which makes it particularly suited for solving Ising-like problems.

To evaluate SKONN's performance in practice, we designed a first proof-of-concept with 9 oscillators on a printed circuit board (PCB) to solve instances of the Max-cut optimization problem. The promising results further led to the design of a **16-neuron SKONN integrated circuit** using a 65 nm CMOS technology with fully connected capability, i.e. 256 synapses with 5 bits of resolution. The chip characterization confirmed its functional operation but also highlighted some issues related to *undesired* couplings among oscillators.

1.6.3 ONN for NP-hard Combinatorial Optimization

Chapter 4 of the thesis further explores ONNs for solving hard combinatorial optimization problems (COPs). First, we evaluate SKONN performances at the chip level and with large-scale simulations when solving the Max-cut problem. Simulations up to 7000 oscillators using the G-SET benchmark indicate that SKONN in *free regime*, i.e. without external control, is as accurate as the Goemans Williamson [111] and CirCut [112] algorithms for Max-cut while suggesting SKONN computes in logarithmic time only.

Second, we investigate how N coupled sinusoidal oscillators can approximate the N-city traveling salesman problem (TSP), rather than using the Ising approach which requires N^2 neurons as originally proposed by Tank and Hopfield in their seminal paper [56]. Inspired by a recent conceptual work [113] which proposes to interpret the phase permutation on the unit circle as the TSP tour, we propose a simple implementation using negatively coupled sinusoidal oscillators. Phase-based simulations indicate that the simple ONN can find TSP tours shorter than 1.5x the optimal solution for $N < 45$. However, as the ONN is not constrained to have uniform spacing between phases, for some instances phases form clusters which would challenge the phase read-out in a real implementation.

Finally, we propose a novel ONN architecture for constrained COPs designed to **escape local energy minima** by going *uphill* in the energy landscape. Our approach is to apply the *Lagrange multipliers* formalism from mathematical optimization to ONNs by combining both gradient descent and ascent mechanisms to *satisfy* constraints. Similar to Lagrangian neural networks [114], the proposed architecture labeled *LagONN* implements competitive dynamics between classical and Lagrangian oscillators, resulting in a search for a saddle point in the aug-

mented phase space. At this point, the energy is minimized along the classical phase directions and maximized in the Lagrangian phase directions to meet the constraints. Harnessing high-order interactions between oscillators, a LagONN architecture is specifically designed to solve the NP-hard Maximum-3-Satisfiability problem (Max-3-SAT). LagONN simulations using the SATlib benchmark with up to 200 variables and 860 clauses suggest that the approach can find near-optimal solutions in polynomial time as $O(N^4)$ where it remains at most a single unsatisfied clause in 75% of trials. For $N = 200$, this corresponds to 99.88% of the 860 clauses being satisfied with a median computation time of around 100k oscillation cycles. As Max-3-SAT is NP-hard, finding the optimal solution probably takes exponential time. However, compared to other physical-based approaches that trade time with exponential power, our approach could be implemented using a bounded circuit power due to **1)** variables encoded in phases and **2)** synapses having fixed amplitudes. Overall, a LagONN physical machine could be used for applications tolerating near-optimal solutions while requiring fast inference such as in real-time systems.

CHAPTER 2

A VO_2-BASED ONN ARCHITECTURE

THIS chapter explores an ONN architecture where neurons are based on vanadium dioxide devices (VO_2) that act as switches with hysteresis and synapses are resistive devices such as resistors, transistors, or memristors. There are two main motivations behind this architecture choice:

1. The oscillator circuit is very compact with only three components: a VO_2 device, a load, and a capacitor.
2. The synaptic matrix can be programmed using memristors laid out in a crossbar array.

Despite the simplicity of the architecture, little is known about **1)** the ONN programmability, **2)** its performances such as computation time, energy consumption, and **3)** its accuracy for a given application. In this chapter, we focus on these three points by combining analytical, numerical, and hardware experiments for the pattern recognition application as with Hopfield neural networks (HNNs).

2.1 Introduction

2.1.1 Existing ONN Architectures

ONN for Image Processing

Although oscillatory-based computing has been studied since the 1950s [78, 79], the first ONN hardware realizations (other than for Boolean logic) appeared only 40 years later. In 1994, Wang and Terman [115] proposed the locally excitatory, globally inhibitory network (LEGION) for image segmentation using an array of oscillators having excitatory couplings with nearest neighbors and a global inhibitor. The idea is to map each pixel of an image to an oscillator where two neighboring oscillators are coupled if the corresponding pixels have a similar intensity. This way, oscillators belonging to the same region remain synchronized together, what the authors call "oscillatory correlation", whereas two different regions are not synchronized due

to global inhibition. The first LEGION implementation was proposed in 1999 [116] using CMOS analog circuits and further developed in 2006 [117] with a 32x32 oscillator chip. More recently, coupled oscillators using beyond-CMOS devices have been studied for various image-processing tasks such as edge detection [96, 118] or stereo vision [106]. All these approaches have the potential to be implemented at a large scale as oscillators are locally coupled.

ONN as an Associative Memory

In 1982, Hopfield [119] proposed a recurrent neural network (HNN) to implement *associative memories* (AM), i.e. memories where stored items (with no known storage location) are recalled by inputting partial memory items. For instance, an HNN programmed as an AM can be used to remove noise from input images that are associated with stored images ("stored" in the synaptic array). Intrigued by the oscillatory behavior of some neural circuits in the brain, or simply to exploit the high parallelism of coupled oscillators, Noest [120] and Endo [99] proposed to program coupled oscillators for associative memory applications. The dynamics of coupled oscillators were further formalized by Hoppensteadt and Izhikevich [121], in particular how to derive phase-based models that abstract the remaining oscillator variables such as the voltage orbit under the weak coupling assumption. This work is the foundation for phase-based ONN computing as the authors derived a convergence theorem for ONN which ensures global stability for symmetric coupling by providing a Lyapunov function for the system (also called energy function). Overall, ONN for AM behaves similarly to HNN, i.e. stored items correspond to some minima of the energy function. Compared to HNN, however, ONNs have the potential to learn patterns encoded with continuous values on the unit circle by extending synaptic values to the complex domain (adding synaptic delays) [122].

The first ONN hardware for AM was proposed in [68] using phase-locked loop circuits (PLL) whose phase evolution satisfies the ONN convergence theorem with various oscillating waveform shapes. PLL-based ONNs were further studied by Jackson and Shi [123, 124] but the impact of undesired transmission delays finally led the authors to choose a PLL-free architecture demonstrated with a fully connected 100-neuron chip in 28 nm CMOS technology and operated in the digital domain [89]. Inspired by Jackson's work, digital architectures have been recently developed by Abernot on FPGAs [125] (120 fully connected neurons) which facilitate the ONN programmation and controlling the dynamics. However, there is currently some resource overhead that prevents large FPGA scaling due to digital arithmetic circuits based on multiply and accumulate operations (MAC), compared to analog CIM architectures that harness analog operations.

Another fascinating approach proposed by Hoppensteadt and Izhikevich in 1999 [126] is to replace the pairwise static connections J_{ij} with *dynamic connections*, i.e. synaptic couplings are embedded in a time-varying signal to avoid implementing N^2 synapses. Having oscillators with different frequencies, the idea is to connect each oscillator to a central unit (via N connections) which in turn sends a signal $a(t)$ that is the sum of N^2 harmonic signals weighted by all possible synaptic weight J_{ij}. The role of each harmonic signal is to "connect" two oscillators such that the average phase difference follows Kuramoto's dynamics (eq.1.2). Thus, the N^2 synaptic overhead is replaced by the challenge of generating $a(t)$. This concept was further studied and formalized by Itoh, Chua [127], Corinto and colleagues [128] with the ONN *star* topology. Notably, they have shown that in addition to reaching static phase fixed points cor-

responding to memorized items, the ONN can also successively find several patterns or create new spurious patterns, what the authors call *dynamic memories*. Interestingly, the dynamic creation of spurious patterns would be analogous to the emergence of new ideas in our brain, or "flash of inspiration" that could help us learn new concepts [127]. Recently, this architecture has been further explored in the context of Ising machines [129]. Although the authors discuss the potential implementation of a modular architecture where each module is driven by its own modulation signal $a(t)$, it is not clear yet how to generate such modulation.

ONN for Classification

ONN synchronization is also interesting for accelerating classification tasks in modern neural networks. Authors like Corti [85] and Nikonov [95] both proposed to use ONNs to accelerate convolution operations that are at the heart of modern convolutional neural networks (CNN) for image classification [21]. Moreover, computing in the frequency domain is not restricted to energy-based computation and broadens the spectrum of applications. For instance, in [109] and [94], the authors propose to use frequency-based spin-torque and phase transition nano-oscillators to solve spoken vowel classification tasks in real time, respectively. The principle is to reduce input vowels to two frequencies f_A and f_B called "formants" and inject them into a 4-node ONN that assigns each input sample to a vowel by reading the synchronization state of the oscillators with inputs f_A and f_B. As demonstrated in these studies [94, 109], operating the ONN in the frequency domain enables a relatively simple supervised learning procedure which consists of 1) applying the training frequencies, 2) measuring the resulting ONN synchronization frequencies and computing the error (or loss), and 3) tuning the individual oscillator frequencies to match the training labels. This last step is possible because the gradient of the loss with respect to some oscillator parameter is known (such as the oscillator bias current). This contrasts with phase-based ONNs for which the frequency is set and learning consists in finding the ONN synaptic weights. Given the final phase state and a phase target, it is not straightforward to find the weight update, although there are promising local learning rules such as equilibrium propagation [130, 131] and its holomorphic variant [132].

When learning the weights is too difficult, another approach is to use coupled oscillators in a *reservoir* topology [133], also called *echo-state networks* [134]. The principle of reservoir computing (RC) is to harness the non-linear input-output function of a dynamical system whose internal configuration is unknown (black box) to classify or generate data. The key advantage of RC is that the reservoir weights are randomly set, thus greatly simplifying the training phase. Overall, the reservoir's role is to project the input data to a higher dimensional space before it can be classified by an output linear function. In RC, the reservoir has a *fading memory*, i.e. a short-term memory, so it can process time-dependent signals that are compatible with ONNs. For instance, using the non-linear amplitude of a single magnetic tunnel junction (MTJ), authors in [93] have shown that the oscillating device can implement a reservoir computer to classify spoken digits with high accuracy by scaling the input signal to the oscillator time constants (short-term memory). Similar spintronic-based reservoirs have also demonstrated the potential combination of amplitude, frequency, and phase non-linearity for classification [135, 136].

ONN for Combinatorial Optimization

One of the most promising applications for ONN is to solve NP-hard combinatorial optimization problems (COP). Like for HNNs, the motivation arises from the intimate link between the ONN energy E (eq.1.3) and the Ising Hamiltonian H (eq.1.1). For most ONNs, E forms an analog relaxation of H where $E(\phi^*) \propto H(S)$ when phases take binary values as $\phi^* = \pi(1+S)/2$. We take the Kuramoto dynamics (eq.1.2) as an example and rewritten here for $K_{ij} = K_{ji}$:

$$\begin{cases} E = -\frac{1}{2}\sum_i \sum_j K_{ij}\cos(\phi_i - \phi_j) \\ d\phi_i/dt = -\partial E/\partial \phi_i = \sum_j K_{ij}\sin(\phi_j - \phi_i) \end{cases} \tag{2.1}$$

Observing that $\cos(\phi_i^* - \phi_j^*) = S_i S_j$, it is clear that $E(\phi^*) \propto H(S)$ and ϕ^* is a fixed point of the dynamics, i.e. $d\phi^*/dt = 0$. Unfortunately, ϕ^* is not necessarily a *stable* fixed point and is often a saddle point in practice as pointed out by Bashar et al. [137]. In other words, there is no guarantee that the ONN reaches ϕ^*. To overcome this situation and reach Ising-like energy minima, Wang et al. [69] proposed to use sub-harmonic injection locking (SHIL) which consists in injecting a harmonic signal to every oscillator at twice the oscillating frequency so that the oscillator dynamics become:

$$\begin{cases} E = -\frac{1}{2}\sum_i \sum_j K_{ij}\cos(\phi_i - \phi_j) - \frac{K_S}{2}\sum_i \cos 2\phi_i \\ d\phi_i/dt = \sum_j K_{ij}\sin(\phi_j - \phi_i) - K_S \sin 2\phi_i \end{cases} \tag{2.2}$$

with K_S the strength of the SHIL signal. Its effect can be deduced from the new expression for E: SHIL adds "wells" of depth $K_S/2$ at every multiple of π in all phase directions. When K_S is large enough, it is likely to transform ϕ^* into a stable fixed point meaning that in practice, the oscillators tend to phase-lock to binary phase states ϕ^*. In their groundbreaking work, Wang et al. [69] named such ONN configuration *Oscillatory Ising Machine* (OIM) and implemented proof-of-concepts on PCB with up to 240 spins. This work triggered great enthusiasm in the field as the results show that a physical OIM can find approximate solutions for the NP-hard Max-cut problem much faster than software approaches with similar or higher accuracy.

Following a similar principle, many various OIM chips have been showcased using CMOS technology or novel devices. Remarkable CMOS chips with 560 [74] and 1968 [76] ring oscillators (RO) were designed using a digital flow achieving both low area and high speed (1968 ROs in 2.1 mm^2 @ 1 GHz in [76]). At this scale, it is generally not feasible to implement the full synaptic array and oscillators are coupled to nearest neighbors with 6 [74] or 8 connections [76]. Impressive fully analog OIMs were also reported by Bashar, Mallick, and colleagues with a first OIM chip with 30 fully connected relaxation oscillators [138], followed by an extended architecture containing 600 oscillators [139]. In the latter, each tile of 16 oscillators can be coupled to 6 neighboring tiles, thus each oscillator has programmable connections with 111 neighbors which enables relatively dense networks. Graber and Hofmann further proposed a 400-oscillator analog OIM chip [90, 140] with 8 nearest neighbor connections per oscillator and a 6-bit high synaptic resolution using digital-to-analog converters. As pointed out by the authors in another work [141], mapping arbitrary problems to the OIM is not directly possible due to the limited connectivity between oscillators. The mapping procedure called *graph embedding* generally requires more physical spins, i.e. oscillators, than the initial number of nodes. Finding the optimal graph embedding can be a hard problem by itself which could undermine the final OIM computational speed-up. A very interesting approach to facilitate graph

embedding was proposed by the same authors in [140]. The idea is to use "switch blocks" between tiles of 12 oscillators and a global routing bus that allows any oscillator connection in the chip (each tile has 4 connections to the global routing bus). Such architecture is an interesting trade-off between fully connected capability and graph embedding difficulty.

Despite the impressive performances of CMOS OIM chips, there is also a great interest in studying beyond-CMOS OIM systems as oscillatory circuits with fewer transistors can be implemented with novel devices such as spintronic devices [87], transition metal oxide devices [86, 142], memristors [143], RRAM [144], or ferroelectric transistors [145]. By reducing the number of components, the common goal of these approaches is to reduce the overall ONN area and energy footprints to allow a potential large-scale integration.

2.1.2 Compact Oscillatory Neurons

Sustained oscillations can appear in many dynamical systems involving mechanical, optical, electrical, biological, or chemical processes [84]. Despite the great variety of oscillators, they are generally described mathematically as dynamical systems $dx/dt = F(x) \in \mathbb{R}^m$ having a limit cycle attractor [128, 146], i.e. an orbit $\gamma \subset \mathbb{R}^m$ with period T such that $\gamma(t) = \gamma(t+T)$. For instance, van der Pol derived famous oscillatory dynamics corresponding to an electrical harmonic oscillator with non-linear damping [147]; and Hodgkin–Huxley's neuron is known to produce oscillations with a sufficient input current [14]. When computing with ONNs, the system dynamics are often reduced to frequency and phase dynamics by considering weak coupling between oscillators, and the oscillation amplitude is not considered [68, 146]. However, intrinsic parameters such as waveform shape and response to perturbations determine the collective ONN performances and can be tuned for specific applications [69, 148], just like activation functions in neural networks.

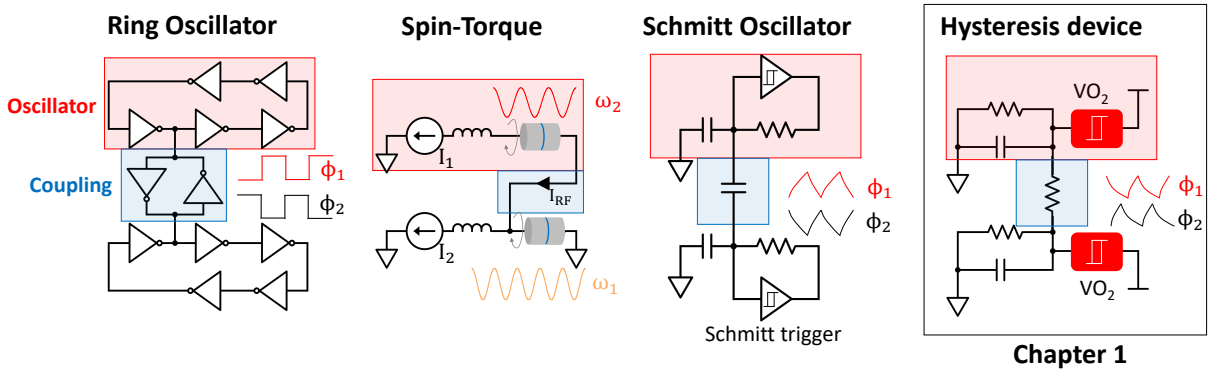


Figure 2.1: *2-coupled oscillators implemented with various technologies. Ring oscillators can be coupled by back-to-back inverters [74] or using transmission gates [76]. Relaxation oscillators consist of a hysteresis device that charges/discharges a load capacitor, producing analog oscillations. The hysteresis component can be implemented by a Schmitt trigger [138] or beyond-CMOS devices that have a negative differential resistance region in their I-V characteristic. A partial list of potential devices includes VO_2 [85, 86, 149, 150], TaO_x and TiO_x oxide [142], NbO_x memristors [143], $PrMnO_3$ RRAM [144] or ferroelectric transistors [145].*

Fig.2.1 shows various electrical oscillator implementations used in ONNs. One of the simplest, yet promising oscillators for ONN is the ring oscillator (RO) which consists of a closed

loop of CMOS inverters in series. ROs benefit from advanced CMOS technologies and can be integrated at a very large scale with low energy consumption and high speed (1968 oscillators demonstrated @1 GHz [76]) although it is still unknown whether ROs provide any computational advantages over other types of oscillators. Spin-torque nano oscillators (STNO) are promising for ONNs as they operate at radio frequencies from 100 MHz to the GHz range and could be integrated at a very large scale with CMOS electronics [87]. An STNO consists of a magnetic tunnel junction biased by a DC current which induces a magnetization precession and consequently voltage oscillations across the device [151]. STNOs are very versatile as they react to various magnetic or electrical perturbations, enabling various types of couplings and ranges [152]. Additionally, STNOs also possess memory in the oscillation amplitude which can be harnessed for processing time series in real-time [93].

Another interesting CMOS oscillator is the Schmitt-based relaxation oscillator where a Schmitt trigger alternately charges and discharges a load capacitor through a resistor. This oscillator is generally bulkier than RO but exhibits interesting properties when carefully tuned, such as phase binarization which is particularly useful for solving COPs [153]. Moreover, beyond-CMOS devices holding hysteresis behavior such as transition metal oxide devices (TMO) [85, 86, 142, 154], volatile memristors [143], RRAM [144] and ferroelectric transistors [145] are intensively studied to replace Schmitt triggers and implement more compact relaxation oscillators. A promising TMO is vanadium dioxide (VO_2) which operates at room temperature and can transition from a semiconductor (insulating) state to a metallic state by Joule effect [150, 155]. When the VO_2 device is biased in its negative differential resistance region, the circuit is unstable and produces sustained oscillations.

2.1.3 Motivations and Contributions

How to program the coupling elements between VO_2 oscillators is not formally defined and still remains empirical [110], although it is known that ONNs behave like HNNs [68, 119]. In the simplest HNN, every neuron is connected to all the others, and synaptic weights are computed with the Hebbian rule ([68]). Given N fully-coupled VO_2-oscillators, it is yet unknown how to transform the coefficients obtained analytically via the Hebbian learning rule to coupling resistor values among oscillators. Further, how can one interpret the coefficient signs W_{ij} such as positive or negative values? Finding a mapping from weights to resistances is a crucial step to allow large-scale ONN design exploration. A greedy approach would be to tune the coupling elements corresponding to the most negative and most positive weights and linearly interpolate all the other weight values. However, this would be impractical. It would require repeated simulations and re-tuning coupling resistances when changing any oscillator parameter; hence, it is not suitable for large-scale ONN design.

Moreover, it is believed that scaled VO_2 devices would provide fast and energy-efficient oscillations and has been validated experimentally with up to eight coupled oscillators [86]. But, little is known about ONN energy when scaling up the network size, whereas energy and power are among the most important specifications for edge devices. Likewise, it is still unknown how the computation time evolves for a large ONN when used as an associative memory. Prior experimental works using VO_2 oscillators have reported ONN performances for less than ten oscillators, but information on **1)** VO_2 device scaling and **2)** ONN architecture scaling are yet to be explored. For example, for image processing applications, Shukla et al. reported the power

consumption for six-coupled VO_2-oscillators [96, 107] but do not mention the energy and delay for larger networks. However, at the device level, a power projection motivates the scaling down of the VO_2 channel length in planar geometry. For spoken vowel detection, Dutta et al. [94] propose to use four coupled planar VO_2 oscillators that consume 6 μW each, but scalability and computation time are not discussed. Corti et al. [85] describe how four and nine coupled VO_2 oscillators can be used as input filters in convolutional neural networks and make a projection of the ONN energy-delay for scaled VO_2 devices in crossbar geometry. However, the estimation remains empirical as VO_2 device physics and coupling elements parameters are not considered.

The chapter is divided into two parts. First, we formally link HNN to VO_2-ONN and propose a framework to map Hebbian coefficient values to ONN coupling resistances. Second, we investigate the VO_2-ONN scaling and performances at device, circuit, and architecture levels. Overall, the main contributions of this chapter are:

1. **At the architecture level:**
   - To propose a *mapping function* between Hebbian coefficients and ONN coupling resistances.
   - To determine the ONN linear energy scaling and quasi-constant computation time with the number of oscillators.
   - To benchmark the VO_2-based ONN energy and delay with respect to state-of-the-art neural accelerators and neuromorphic chips. It appears that ONN can be a competitive computing paradigm for high oscillating frequencies.
   - Finally, we showcase a VO_2-based ONN benchmark for image edge detection and compare it with the state-of-the-art CMOS ASICs.
2. **At the device level:**
   - How to minimize the oscillating energy for crossbar VO_2 devices.

The findings of this chapter have led to two peer-reviewed journal articles:

- C. Delacour and A. Todri-Sanial, "Mapping Hebbian Learning Rules to Coupling Resistances for Oscillatory Neural Networks ", in *Frontiers in Neuroscience*, vol.15, 2021.
- C. Delacour, S. Carapezzi, M. Abernot, and A. Todri-Sanial, "Energy-Performance Assessment of Oscillatory Neural Networks Based on VO_2 Devices for Future Edge AI Computing," in *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

2.2 VO_2 Device

Vanadium dioxide (VO_2) is a material whose resistivity can vary abruptly with temperature [156]. The timescale of the insulator-metal transition is ultrafast, spanning from ns to fs [157]. A sudden resistive switching is also observed in VO_2 channels of two-terminal devices, where self-heating due to Joule effect is driving the change. The channel is in high resistance state (HRS) or low resistance state (LRS) according to its temperature being below or above temperature thresholds for state transitions. The volatility of resistance states of VO_2 devices is key to designing VO_2 relaxation oscillators. Overall, the temperature thresholds are not unique and depend on temperature history, i.e. if the VO_2 channel is in the heating/cooling stage during the device operation. Specifically, VO_2 needs to cool down to transition back to the insulating

state and thus presents a hysteresis behavior (Fig.2.2a). Since the temperature of the channel is triggered by self-heating, such threshold temperatures are achieved at given threshold voltages: V_H, for the transition from HRS to LRS, V_L for the transition from LRS to HRS. Hence, VO_2's $I-V$ curve also has a hysteresis (Fig.2.2c).

The state-of-the-art approach for simulating VO_2 is to use the compact model from Mafezzoni et al. [158] which reproduces the VO_2 hysteresis along with continuous and abrupt transitions between the two states. In this model, the VO_2 hysteresis behavior is conceptually emulated by an amplifier with positive feedback that charges or discharges an RC circuit when the VO_2 is in a metallic or insulating state, respectively. The voltage V_c across the capacitor commands the VO_2 conductance G_{VO_2} as:

$$G_{VO_2} = \frac{1-V_c}{R_{ins}} + \frac{V_c}{R_{met}} \tag{2.3}$$

where R_{ins} and R_{met} are the VO_2 resistances in insulating and metallic state. The dynamics of the VO_2 conductance is given by the RC circuit where τ_0 is its time constant modeling the transition time of the VO_2:

$$\tau_0 \frac{dV_c}{dt} + V_c = 1 - V_0 \tag{2.4}$$

where V_0 is the output of the positive feedback amplifier (gain α) and is expressed as:

$$V_0 = \frac{1}{2}\left[1 + \tanh\left(2\alpha\big((V_H - V_L)V_0 + V_L - V\big)\right)\right] \tag{2.5}$$

Note that this model does not include VO_2 thermal variables that impact the oscillator frequency and energy consumption.

2.3 VO_2 Oscillator

The VO_2 hysteresis behavior has been widely used in literature to design relaxation oscillators [86, 158, 159]. It consists of biasing the VO_2 device with a load resistance R_S in series and connecting a capacitor C_P in parallel with the output node V_{out} to adjust the oscillation frequency (Fig.2.2b). To produce oscillations, the load line I_L set by V_{DD} and R_S must cross the VO_2 I-V curve in its negative differential resistance region (NDR) to obtain an unstable fixed point (Fig.2.2c). The VO_2 device state hence alternates between the metallic and insulating state. When the VO_2 device is in the insulating state, the parallel capacitance C_P at the output node discharges through R_S until the VO_2 voltage reaches V_H and transitions to the metallic state. Then, C_P charges through the VO_2 device until its voltage reaches V_L and a new cycle begins (Fig.2.2d). The oscillator dynamics can be described by Kirchoff's law as follows:

$$C_P \frac{dV_{out}}{dt} = \big(V_{DD} - V_{out}\big) G_{VO_2} - \frac{V_{out}}{R_S} \tag{2.6}$$

Note that despite the first order differential equation, oscillations can occur as equations (eq.2.3), (eq.2.4) and (eq.2.5) describe the hysteresis behavior of G_{VO_2}. Fig.2.2d shows an example of oscillating dynamics where at t=0: $V_{out} = 0\,V$, $G_{VO_2} = 1/R_{ins}$, and $V_{DD} = 2.5\,V$. In this chapter, we use the same VO_2 and circuit parameters listed in Table 2.1 if not stated otherwise. The VO_2 parameters were shared by our IBM colleagues after characterization.

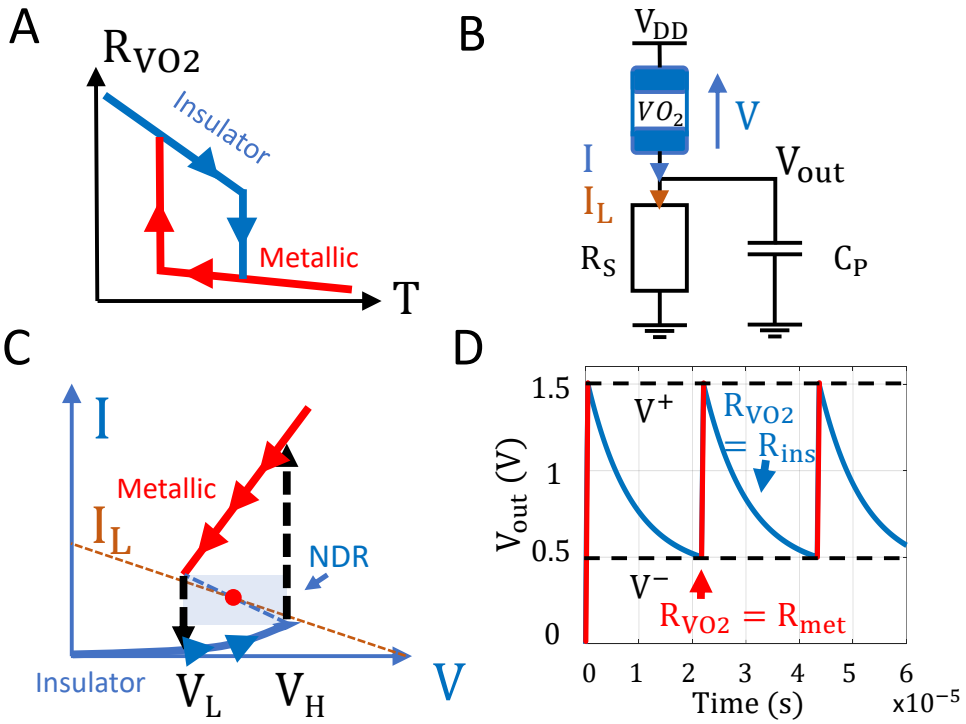


Figure 2.2: *(A) VO_2 resistance versus temperature. R_{VO_2} axis is in logarithmic scale. (B) VO_2 oscillator circuit (C) VO_2 I-V curve showing the device hysteresis behavior. When its voltage reaches a threshold V_H, it transitions from insulating to the metallic state. To transition from a metallic to an insulating state, V must be lower than the threshold V_L. (D) If the device is biased in its Negative Differential Resistance region (NDR), its state alternates between the metallic and insulating state, producing electrical oscillations.*

Parameter	V_{DD}	R_S	C_P	R_{ins}	R_{met}	V_L	V_H	α	τ_0	$V^+ = V_{DD} - V_L$	$V^- = V_{DD} - V_H$
Value	2.5 V	20 kΩ	500 pF	100.2 kΩ	0.99 kΩ	1 V	1.99 V	200	10 ns	1.5 V	0.501 V

Table 2.1: *VO_2 and circuit parameters used in Chapter 2.*

2.4 Two-coupled VO_2-Oscillators

2.4.1 Phase Initialization

To set an initial phase state with respect to a reference oscillator (the first one in this manuscript), the oscillator starting times can be delayed via V_{DD}. Assuming the oscillators have the same period T_{osc}, any input delay Δt_{init} is equivalent to an initial phase relation as:

$$\Delta\phi_{init} = 2\pi\frac{\Delta t_{init}}{T_{osc}} \tag{2.7}$$

However, if the two oscillators are always connected, they might have different oscillation periods during initialization and their initial phase relation cannot be defined. For example, as shown in Fig.2.3a, the second oscillator starts $\Delta t_{init} = 0.5\ T_{osc}$ after the first one to set an initial phase relation $\Delta\phi_{init} = \pi$. For $t < \Delta t_{init}$, the second oscillator is off, and its output resistance is $R_S//R_{ins} \approx R_S$. Therefore, during this time, the equivalent load resistance of the first oscillator is $R_S//(R_C + R_S)$, which induces a shorter period of oscillation $T'_{osc} < T_{osc}$ and hence no control on the initial phase. To tackle this lack of control, one can introduce switches between each

oscillator and coupling elements as in Fig.2.3b. Then, the oscillators can evolve freely with a known oscillation period T_{osc} before coupling them at t_{on}, instant where their initial conditions are known using (eq.2.6). The ONN initialization is improved at the cost of one additional switch per oscillator such as a transfer gate.

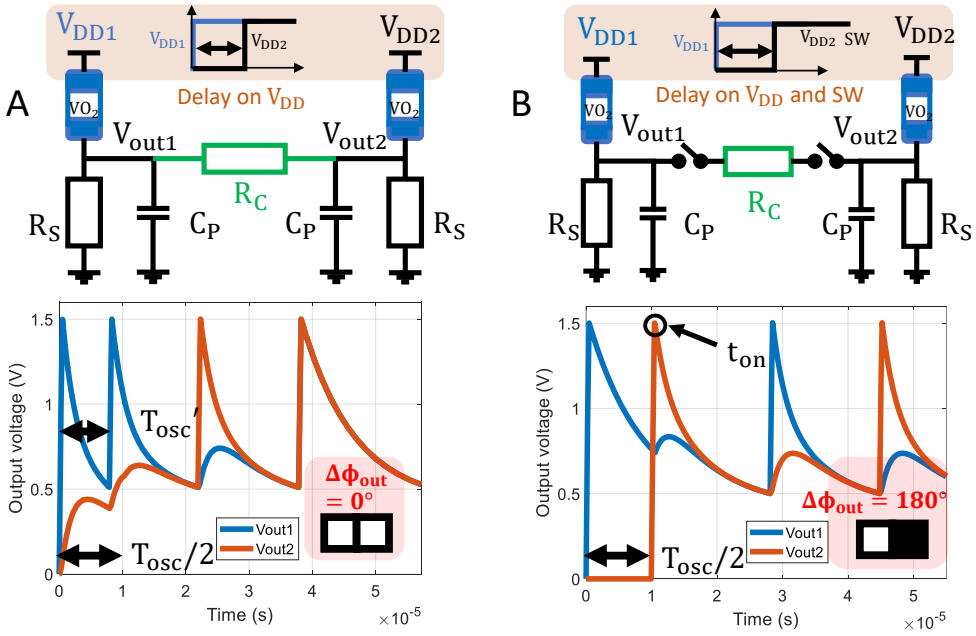


Figure 2.3: *(A) Two oscillators are coupled by a resistance R_C=10 kΩ without coupling switches in between. V_{DD2} is turned on $0.5T_{osc}$ after V_{DD1} to set an initial phase of 180°. However, for $t < 0.5T_{osc}$ the first oscillator period is decreased due to the shunt R_C at its output node, and we cannot control the initial phase difference. The two oscillators are in phase after convergence. (B) Same circuit with coupling switches to isolate the oscillators during initialization. In this case, the $0.5T_{osc}$ input delay sets a desired 180° initial phase state. Here, the switches are closed at $t_{on} = 0.5T_{osc} + t_c$ such that $V_{out2}(t_{on}) = V^+$. The two oscillators converge to an 180° phase state relation.*

2.4.2 Memory of Two-coupled Oscillators

Analyzing the dynamics of two coupled oscillators gives information about the phase fixed points which depend on **1)** the coupling strength R_C and **2)** the initial phase state. The dynamics of two coupled oscillators can be expressed using Kirchhoff's laws:

$$\begin{cases} C_P \frac{dV_{out1}}{dt} = \left(V_{DD1} - V_{out1}\right) G^1_{VO_2} - V_{out1}/R_S + I_{c1} \\ C_P \frac{dV_{out2}}{dt} = \left(V_{DD2} - V_{out2}\right) G^2_{VO_2} - V_{out2}/R_S + I_{c2} \end{cases} \tag{2.8}$$

where currents are $I_{c1} = -I_{c2}$ represent the coupling element's current flow. Fig.2.4 shows a simulation where V_{DD2} is turned on $0.1T_{osc}$ after V_{DD1} which initializes a light-gray pixel for the input image of oscillator 2. For a small coupling resistance R_C=10 kΩ, the ONN converges to a stable state with both oscillators in phase (0°,0°). Whereas, for R_C=100 kΩ, it converges to an out-of-phase state (0°,180°). More interestingly, varying both parameters R_C and Δt_{init} highlights the ONN associative memory behavior. Fig.2.5A shows the simulation results. As already

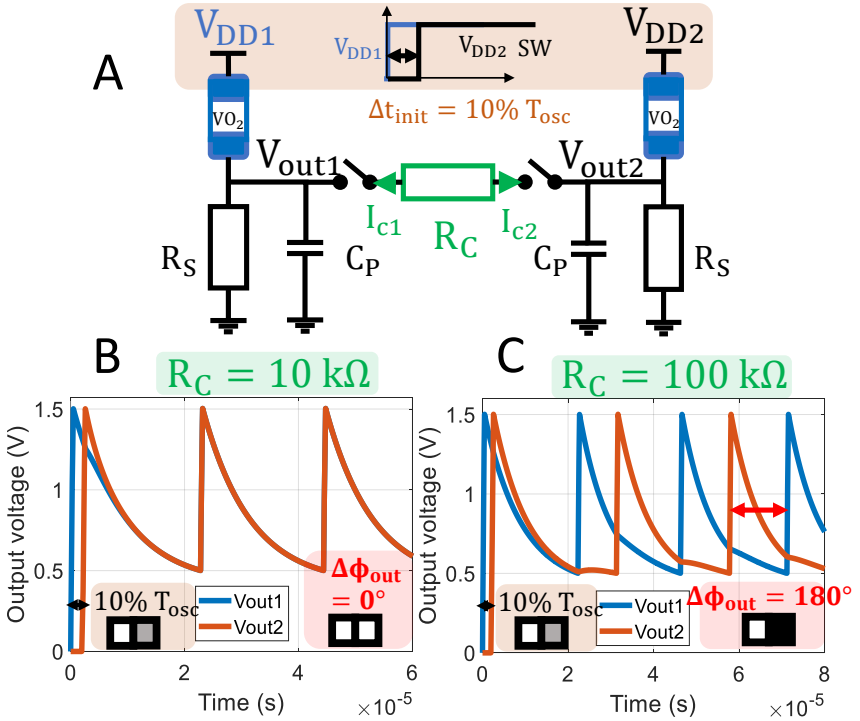


Figure 2.4: *(A) Two identical VO_2 oscillators coupled by a resistance R_C and switches. V_{DD2} starting time and coupling time t_{on} are delayed by $0.1T_{osc}$ with respect to the first oscillator, representing a light-gray second-pixel ONN input. (B) Output voltages for $R_C = 10k\Omega$: the oscillators converge to an in-phase state (0°,0°) and the corresponding output pattern corresponds to two white pixels. (C) Output voltages for $R_C = 100k\Omega$: the oscillators are out-of-phase (0°,180°) and the output pattern corresponds to white and black pixels.*

observed by [159], a large coupling resistance $R_C > 40$ kΩ induces oscillators in out-of-phase relation (0°,180°) for any input delay, whereas a small coupling resistance $R_C < 10$kΩ aligns oscillators in-phase (0°,0°) for any input delay. In contrast, we examine the region between these two ranges, highlighted in the center of Fig.2.5A. It appears that for 10 kΩ $\leq R_C \leq$ 40 kΩ both states co-exist and the oscillators store two patterns (0°,0°) and (0°,180°) that can be retrieved by adjusting the input delay. The line *transition function* between the in-phase and out-of-phase regions represents our analytical function for the ONN memory with respect to coupling resistance and initialization. It is defined as:

$$\zeta : R_C \longrightarrow \Delta t_{transit} \tag{2.9}$$

with $\Delta t_{transit}$ the initial delay such that:

$$\Delta t_{transit} = \zeta(R_C) \mid \begin{cases} \Delta t_{init} < \Delta t_{transit} \Rightarrow \Delta\phi_{out} = 0° \\ \Delta t_{init} \geq \Delta t_{transit} \Rightarrow \Delta\phi_{out} = 180° \end{cases} \tag{2.10}$$

To study $\zeta(R_C)$ in experiments, we emulated two coupled VO_2 oscillators with off-the-shelf components on a printed circuit board (PCB). The relaxation oscillator circuit consists of an inverting Schmitt trigger [160] operational amplifier (OPA) which implements the VO_2 hysteresis behavior (Fig.2.5b). The OPA saturates to $+V_{sat}$ and $-V_{sat}$ while the 1.8 nF output capacitor charges and discharges, respectively. Fig.2.5c shows the voltage across the output capacitor for a single oscillator. Similarly to a VO_2 oscillator, the OPA transitions to another state when the voltage across the output capacitor reaches V^+ or V^-. The 5.6 kΩ resistor implements the

metallic VO_2 resistance, whereas the 100 kΩ resistor corresponds to the load R_S. For a fixed oscillating period of T_{osc}=200 μs, we varied R_C and measured $\Delta t_{transit}$ values that define the experimental transition function $\zeta(R_C)$ (Fig.2.5d). There is a good match between experimental $\zeta(R_C)$ data points and the analytical transition function derived in the next section. Such formulation $\zeta(R_C)$ is of interest as it represents a closed-form representation of ONN memory instead of repeating numerical simulations for different oscillator parameters.

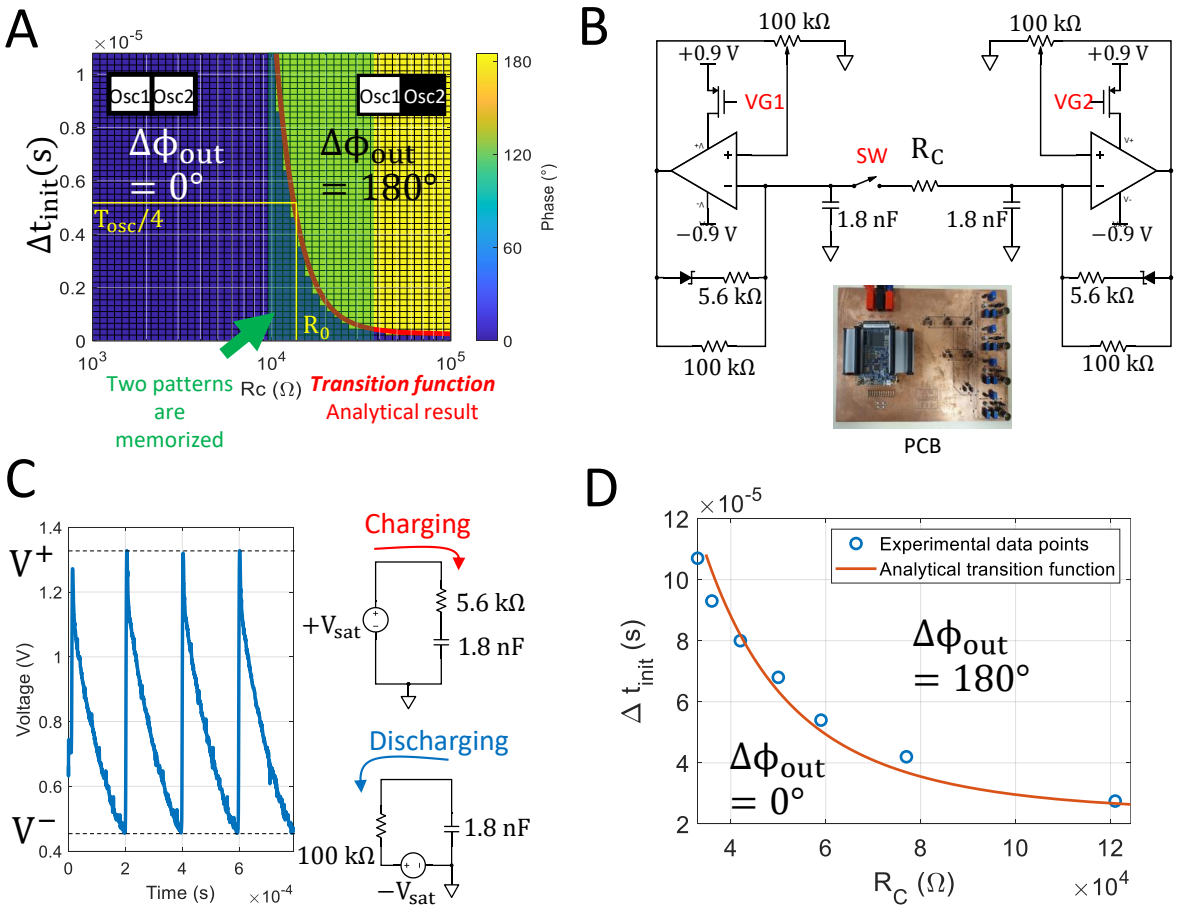


Figure 2.5: *(A) Plot showing the phase relation between two oscillators for every set of parameters $(R_C, \Delta t_{init})$. As expected, small coupling resistances tend to pull the oscillator phase together, whereas large coupling resistances push the phase away. The middle green region shows the coupling resistance range 10kΩ $< R_C <$ 40kΩ in which two patterns (0°,0°) and (0°,180°) are memorized and can be retrieved by adjusting the input delay. The red curve is our analytical model describing the transition between the two phase states in the plan $(R_C, \Delta t_{init})$, and plays a major role in the ONN ability to memorize patterns. (B) Experimental set-up of two coupled relaxation oscillators based on MCP6001 OPAs. We delay VG2 with respect to VG1 by Δt_{init} to set the initial phase, and close SW after initialization. (C) Experimental oscillating waveform and equivalent circuits during charge and discharge of the output capacitor. (D) Experimental phase transition curve and analytical model in plain line.*

2.5 Mapping Hebbian Weights to Coupling Resistances

2.5.1 Phase Transition Function

The phase transition function has already been observed [161] but to the best of our knowledge, no closed-form expression has ever been reported. To derive the transition function, we analytically solve the node voltage equations for two coupled oscillators during initialization (see Appendix A). The oscillator output voltages can be expressed as:

$$\Delta V = V_{out2} - V_{out1} = \left(V^0_{out2} - V^0_{out1}\right) \exp\left(-\frac{t}{\tau'(R_C)}\right) \tag{2.11}$$

where V^0_{out1} and V^0_{out2} are the initial voltages when coupling the oscillators. $\tau'(R_C)$ is a function of R_C and defined in the Appendix (eq.A.18). Eq.2.11 describes how both oscillator output voltages are attracted via the coupling resistance R_C. If the coupling is strong enough (small R_C), the oscillators are rapidly pulled together with a speed determined by τ'. Then, if $\Delta V < \varepsilon$ (ε defined in the Appendix (eq.A.22)) before reaching the VO_2 threshold V^-, both oscillators transition to low resistive states, and the exponential term in (eq.A.19) keeps the two voltages locked. This concept is illustrated in Fig.2.6A when both oscillators are in-phase. However, if V_{out1} reaches V^- before V_{out2} such that $\Delta V > \varepsilon$ as in Fig.2.6B, the first oscillator transitions to a LRS while the other oscillator is still in HRS. The two oscillators are then in opposite states, and this leads to an out-of-phase state.

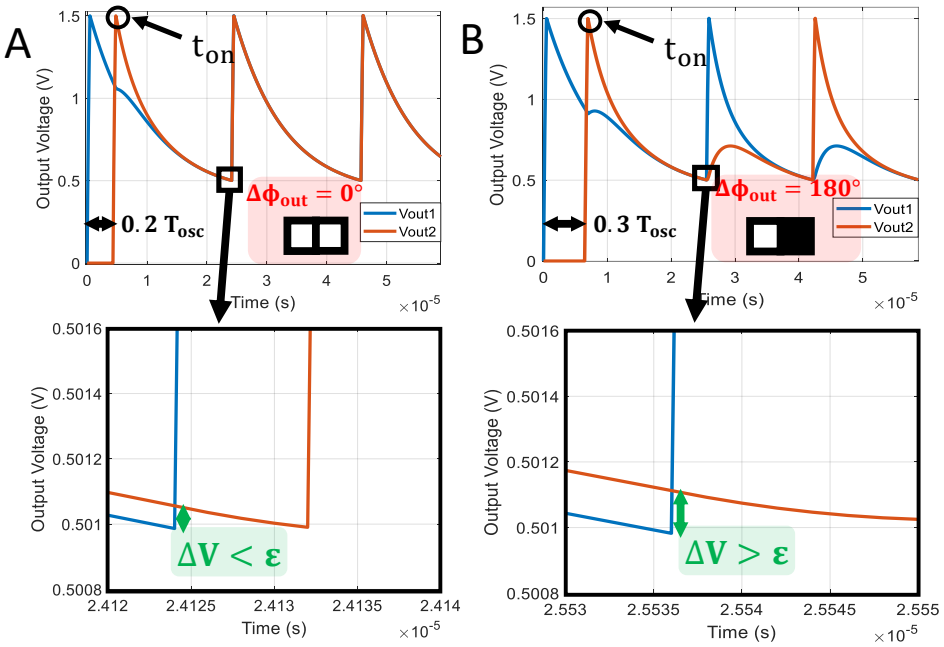


Figure 2.6: *Two identical oscillators coupled by $R_C = 12\,k\Omega$. (A) The second oscillator is turned on at 0.2 T_{osc} after the first one. By zooming on the waveform when the first oscillator reaches V^-, there is a voltage difference $\Delta V < \varepsilon$ and the oscillators converge to an in-phase state. (B) The second oscillator is turned on 0.3 T_{osc} after the first one. In this case, the voltage difference $\Delta V > \varepsilon$ and the oscillators converge to an out-of-phase state. We express ΔV to derive the phase transition function ζ.*

Thus, the transition function ζ (eq.2.9) can be derived when both $\Delta V = \varepsilon$ and $V_{out1} = V^-$ (see Appendix A for details). By combining Appendix equations (eq.A.17), (eq.A.24) and (eq.A.25) the coupling resistance can be expressed as:

$$R_C = 2\frac{R_S R_{ins}}{R_S + R_{ins}} \frac{\log\left(\frac{V^- - V_{std}^{ins}}{V_{out1}^0/2 + V_{out2}^0/2 - V_{std}^{ins}}\right)}{\log\left(\frac{\varepsilon(R_C)}{V_{out2}^0 - V_{out1}^0}\right) - \log\left(\frac{V^- - V_{std}^{ins}}{V_{out1}^0/2 + V_{out2}^0/2 - V_{std}^{ins}}\right)} \tag{2.12}$$

where V_{std}^{ins} is defined in (eq.A.5). Finally, introducing (eq.A.9) into (eq.2.12) produces a relation between R_C and $\Delta t_{transit}$. Note that as ε is a function of R_C, (eq.2.12) cannot be solved analytically. Instead, one can numerically solve (eq.2.12) using Newton-Raphson's algorithm for $\Delta t_{transit}$ values. Finally, the R_C values that describe the inverse of the transition function correspond to:

$$\zeta^{-1} : \Delta t_{transit} \longrightarrow R_C \tag{2.13}$$

The transition function ζ is plotted as the curve line (red line) in Fig.2.5A. There is an excellent fit between our analytical model, simulations (transition region between dark and light green in $(R_C, \Delta t_{init})$ plan) and the transition curve obtained experimentally with off-the-shelf relaxation oscillators (Fig.2.5D). As by definition, both output phase states can equally occur for $W = 0$, its corresponding coupling resistor R_0 can be expressed as follows:

$$R_0 = \zeta^{-1}(\Delta t_{init} = T_{osc}/4) \tag{2.14}$$

Finally, the final phase relation can be expressed based on the transition function as:

$$\Delta\phi_{out} = \pi\left(\text{sign}\Big(R_C - \zeta^{-1}(\Delta t_{init})\Big) + 1\right)/2 \tag{2.15}$$

Analogous to ANNs, (eq.2.15) can be thought of as oscillator's activation function because it provides the oscillator's output phase based on its input phase (set by Δt_{init} (eq.2.7)) and the weight implemented by R_C.

2.5.2 Linking ONNs to Hopfield Neural Networks

Here, we exploit the HNN formalism to build an analogous representation in ONN. For equivalence, ONN oscillators are treated similarly to HNN neurons. Such as, we consider a neuron i with two possible states S_i that can be thought of as equivalent to ONN oscillators with 0° or 180° phase relations as:

$$S_i = \begin{cases} +1 \\ -1 \end{cases} \iff \Delta\phi_i = \begin{cases} 0^\circ \\ 180^\circ \end{cases} \tag{2.16}$$

In HNN, each neuron output state is dynamically determined by a sigmoid activation function $g(x) = \big(\tanh(\beta x) + 1\big)/2$ with $\beta > 0$ [13] and shown in Fig.2.7.

For a neuron i, g gives the probability to reach one of the two states at $t + \Delta t$ for a given input weighted sum $h_i(t)$ as:

$$P\Big(S_i(t + \Delta t) = +1 \mid h_i(t)\Big) = g\big(h_i(t)\big) \tag{2.17}$$

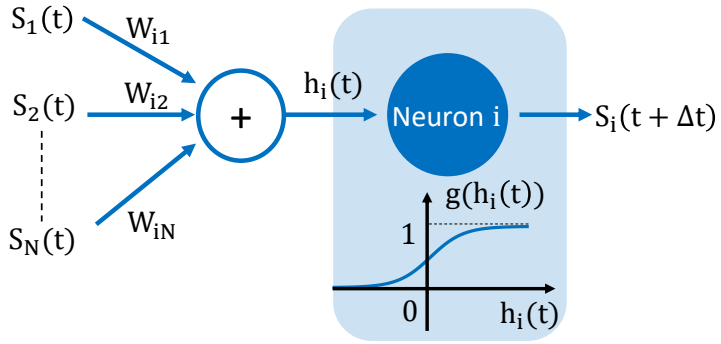


Figure 2.7: *Model of artificial neuron used to construct our mapping function μ_N. The neuron's output state $S_i(t)$ is either +1 or -1 and is dynamically updated at each time-step Δt according to the sigmoid activation function g. Here, g gives the probability to have one of the two states at $t+\Delta t$ for a given input weighted sum $h_i(t)$.*

with:

$$h_i(t) = \sum_{j=1}^{N} W_{ij} S_j(t) \tag{2.18}$$

In ONNs, (eq.2.17) would represent the probability of oscillator i to be in phase with the reference at time-step Δt. For the two-oscillator case, the weighted input sum of the second oscillator is given by:

$$h_2(t) = W_{21}\, S_1(t) = W_{21} \tag{2.19}$$

Because the first oscillator is the reference. Thus, the probability of the second oscillator being in phase with the reference can be derived by (eq.2.17) and (eq.2.19), as:

$$P\Big(S_2(t+\Delta t) = +1 \mid h_2(t)\Big) = P_{in-phase} = g(W_{21}) \tag{2.20}$$

2.5.3 Mapping Function

Applying the above definitions from the HNN formalism, a mapping function can be derived as:

$$\mu_N : W_{ij} \longrightarrow R_{ij} \tag{2.21}$$

where normalized weights are $-1 \leq W_{ij} \leq 1$ and N is the ONN size. Before scaling to N oscillators, we first propose a mapping function μ_2 for two coupled oscillators. The unifying step between HNN and ONN is the recasting of the phase transition curve, ζ as the probability $P_{in-phase}$ for a given coupling resistance R_C. In ONNs, the input delay Δt_{init} can be considered as a uniform random variable taking values between 0 and $T_{osc}/2$ and the transition function ζ would give the probability $P_{in-phase}$ (for example for $R_C > 10\ \text{k}\Omega$):

$$P_{inphase} = \zeta(R_C)\,\frac{2}{T_{osc}} \tag{2.22}$$

By equalizing (eq.2.20) and (eq.2.22), the coupling resistance is expressed as:

$$\mu_2(W_{21}) = R_C = \zeta^{-1}\left(\frac{T_{osc}}{2} g(W_{21})\right) \tag{2.23}$$

This mapping function is represented in Fig.2.8 for three different values of the sigmoid parameter, β. We see that β sets the range of R_C and could be adapted for different ONN sizes. Interestingly, it appears that the derivative $|dW_{21}/\dot{R_C}|$ is quite large for a positive weight, whereas it is much smaller for a negative one. Actually, Fig.2.8B shows that the function ζ^{-1} is a logarithmic function; thus, any small variation in ΔR_C around 10 kΩ is likely to change the final phase state outcome. Whereas for large R_C, the two oscillators are almost always out-of-phase. This asymmetry in ζ^{-1} comes from the oscillator waveform type, as ζ^{-1} is derived from (eq.2.12), which is specific for relaxation oscillator waveform type. Hence, we expect some change for other types of waveforms, such as linear sawtooth, but the formulation of mapping (eq.2.23) is general enough to be applied to any relaxation oscillators.

For large-scale ONN with N oscillators, we scale μ_2 (eq.2.23) by a factor $N-1$ to ensure the conservation of the current flow in coupling resistances, i.e. an average constant fan-out current at the oscillator output. We finally obtain:

$$\mu_N(W_{ij}) = R_{ij} = (N-1)\,\zeta^{-1}\left(\frac{T_{osc}}{2}g(W_{ij})\right) \tag{2.24}$$

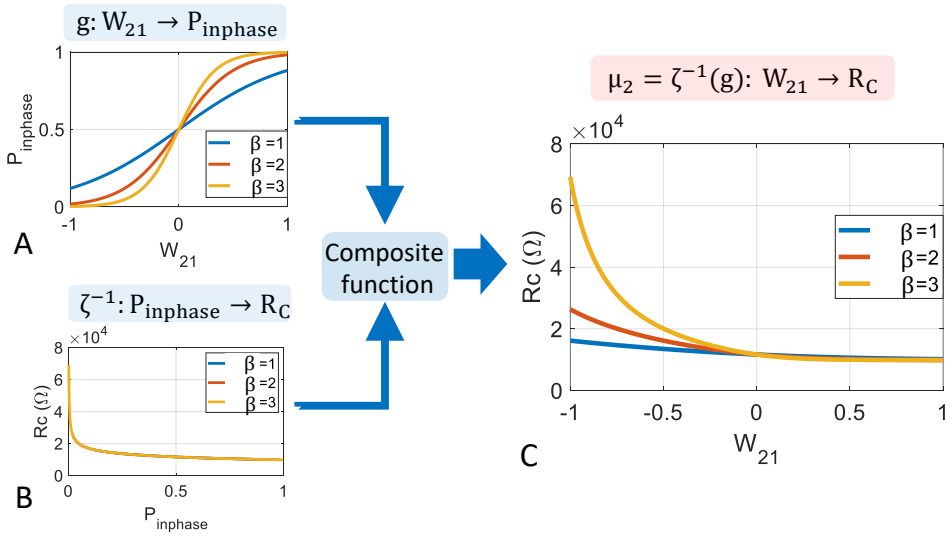


Figure 2.8: *Mapping function for two coupled oscillators. (A) The sigmoid activation function presents the probability $P_{in-phase}$ for the two oscillators to be in phase. (B) The inverse of the transition function ζ^{-1} determines the coupling resistance R_C for a given probability $P_{in-phase}$. (C) The mapping function μ_2 is obtained by the composite function $\zeta^{-1}(g)$.*

2.6 ONN Design for Pattern Recognition

Here, we show the effectiveness and limitations of the proposed mapping function (eq.2.24) in simulations up to 100 coupled oscillators for pattern recognition as in HNN.

2.6.1 ONN Training and Mapping

We propose a design flow as shown in Fig.2.9A for pattern recognition with ONNs using the proposed mapping function. We first compute the weights associated with the M stored binary patterns $\{\xi^1, \xi^2, ...\xi^M\}$ using the Hebbian Rule [68], as:

$$W_{ij} = \frac{1}{N}\sum_{k=1}^{M} \xi_i^k \xi_j^k \tag{2.25}$$

where $\xi_i^k = 1$ if the pixel i is white, or $\xi_i^k = -1$ if the pixel i is black. We store $M = 6$ images representing digits '0','1' to '5' as shown in Fig.2.9B.

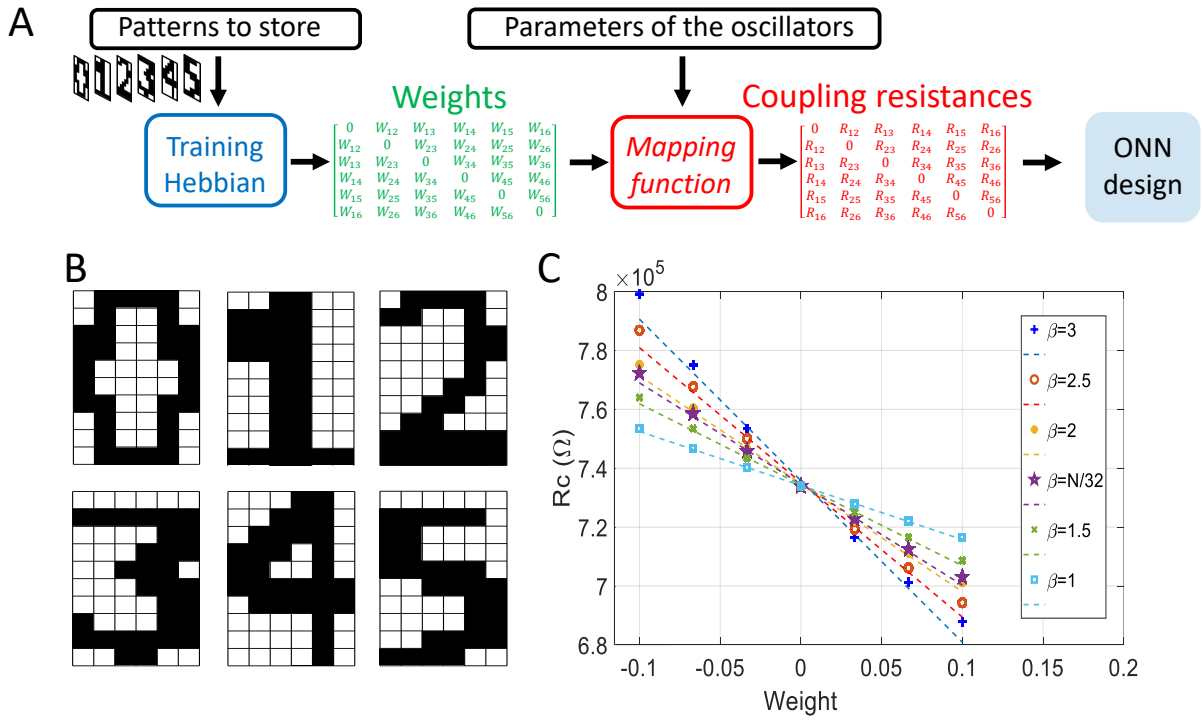


Figure 2.9: *(A) Illustration of the ONN design flow for the associative memory application. Patterns to store can be represented as black-and-white images from which we compute weights with the Hebbian rule during the training process. Then, we use the mapping function μ_N to get the coupling resistances, allowing a systematic ONN design. (B) Stored patterns. (C) Coupling resistances as a function of Hebbian weights, computed with the mapping function μ_N for different values of parameter β.*

Next, we use the mapping function to compute the coupling resistances associated with the Hebbian coefficients. The mapping is represented in Fig.2.9C for different values of parameter β which sets the slope of $\mu_N(W_{ij})$. Increasing β induces a larger coupling resistance range. Because the Hebbian rule normalizes the weights by the network's size N (eq.2.25), we scale β with N to keep a relative range of R_{ij} approximately constant when increasing the ONN size. As shown in the next simulations, we find that the best accuracy is obtained for $\beta = N/32$.

2.6.2 ONN Inference

We have developed an ONN circuit simulation platform in Matlab that includes the VO_2 device parameters (compact model [158]) and coupling parameters to allow transient simulation of different size ONNs. Each oscillator is a pixel of an image where a black pixel is initialized with a delay $\Delta t_{init} = T_{osc}/2$ to set an initial out-of-phase relation (eq.2.7) whereas, for a white pixel, no delay is introduced. A noisy gray pixel corresponds to an input delay between 0 and $T_{osc}/2$. After a few oscillations, the ONN settles, retrieves the noiseless pattern and the phase relations are measured. An example of ONN voltage dynamics is presented in Fig.2.10A, showing the initialization and the settling time before the ONN stabilizes. Fig.2.10B and C show two examples of input images where 15 pixels have been randomly altered by a uniform distribution taking values between -1 and +1. When the number of noisy input pixels is too large such as in Fig.2.10D (20 noisy pixels), the ONN converges toward a wrong spurious state that is different from the stored patterns.

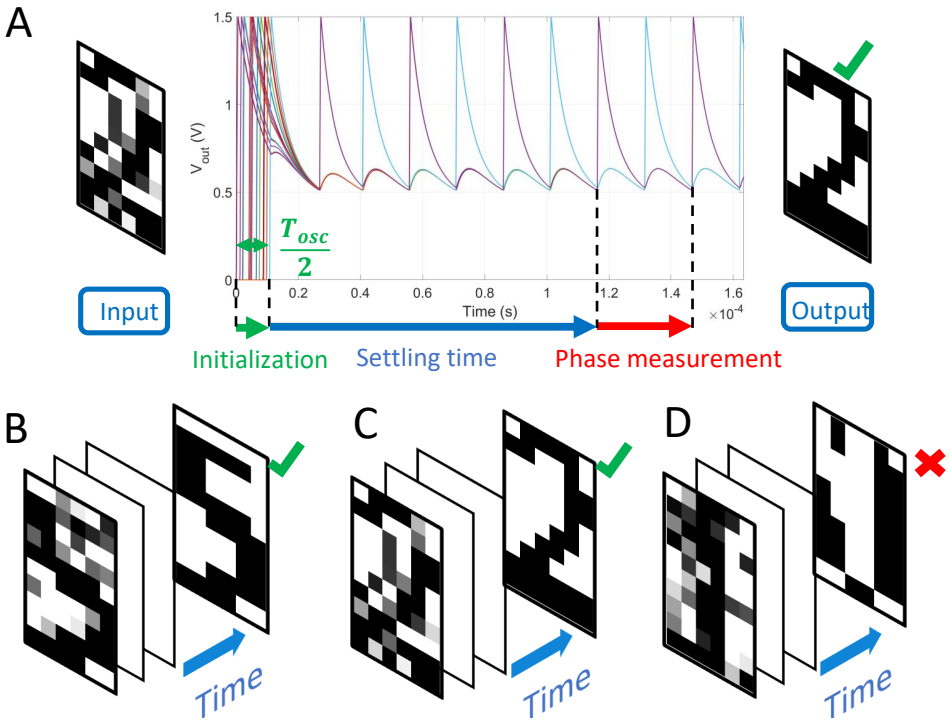


Figure 2.10: *(A) Noisy input image '2' with 15 random altered pixels and voltage waveforms of 60 oscillators. ONN is initialized during $T_{osc}/2$ with the noisy input image. After few oscillation cycles, ONN settles, and phases are measured. ONN retrieves the correct output image that corresponds to the stored pattern '2'. (B) The input image '5' has been altered at 15 random pixel locations by a uniform distribution. ONN retrieves the corresponding stored pattern. (C) Similarly, 15 random pixels of the input image '2' are altered, and ONN retrieves the correct corresponding pattern. (D) In this example, 20 noisy input pixels are introduced to digit '1', and ONN converges toward a spurious state.*

2.6.3 ONN Memory Capacity

We report on the ONN memory capacity when trained using the Hebbian learning rule and our proposed mapping (eq.2.24). For different ONN sizes from N=8 up to N=100, several training

set sizes M are considered. To avoid biased results due to specific training sets, we generate random training sets composed of M random black-and-white patterns that have the same proportion of black-and-white pixels to avoid any effect emerging from unbalanced patterns. To generate test sets, we add a random uniform noise between -1 and +1 to the training patterns (noisy pixels are gray), as in the example of Fig.2.10A. The number of noisy pixels varies from 0 up to 50%.

During inference, the accuracy is '1' if and only if the ONN completely retrieves the pattern. Otherwise, it is '0'. Fig.2.11A shows the ONN recognition accuracy for test images having 10% of noise. As expected, larger networks can store more patterns. An ONN with N=100 stores M=16 patterns with more than 50% of recognition accuracy. Whereas 16 oscillators are limited to M=6 patterns for a similar accuracy. This trend is in accordance with Hopfield's results [82]. Based on Fig.2.11A, we extract the ONN memory capacity when the recognition accuracy reaches 50%. Results are shown in Fig.2.11B. It appears that the ONN memory capacity grows linearly with a fitted slope of 0.146, in accordance with the scaling factor of 0.15 derived by Hopfield [82]. The trained ONN can hence implement an HNN as there is a good match between the results and the original observations from Hopfield [82].

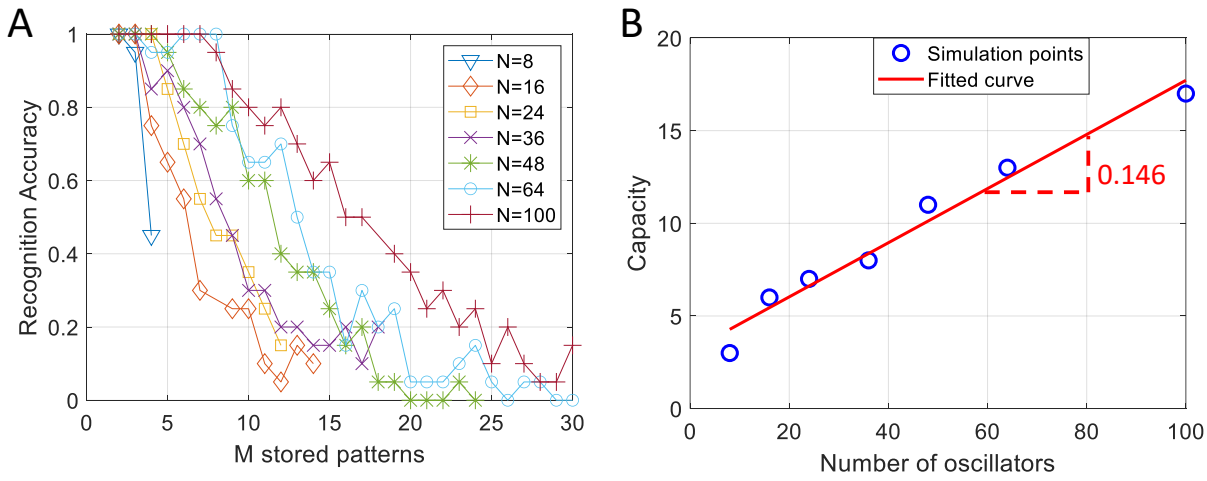


Figure 2.11: *(A) ONN recognition accuracy with respect to the number of stored patterns M. Test images have 10% of noise. Each data point is the recognition average computed over 20 different trials. (B) ONN capacity extracted for a 50 % recognition accuracy. ONN capacity scales linearly. The slope is close to the 0.15 value theoretically obtained with HNN using the Hebbian rule [82]*

2.7 ONN Energy Scaling

Here, we study the ONN energy scaling using a bottom-up approach, i.e., starting from device and circuit level up to the architecture level. It is shown analytically and by circuit simulations that the ONN energy scales linearly with the number of oscillators.

2.7.1 Single Oscillator Energy Footprint

From circuit equations and Fig.2.2A, the instantaneous power consumption of a single oscillator can be expressed as:

$$P(t) = V_{DD}(\frac{V_{out}}{R_S} + C_P\frac{dV_{out}}{dt}) \tag{2.26}$$

As V_{out} is a T_{osc}-periodic signal, the oscillator energy loss for one oscillation is given by:

$$E_{osc} = \frac{V_{DD}}{R_S}\int_0^{T_{osc}} V_{out}\, dt \tag{2.27}$$

Then, we introduce the output mean voltage $\overline{V_{out}} = 1/T_{osc}\int_0^{T_{osc}} V_{out}\, dt$ to reformulate the last expression as:

$$E_{osc} = \frac{V_{DD}}{R_S}\overline{V_{out}}T_{osc} \tag{2.28}$$

As $T_{osc} \propto R_S C_P$ (see Appendix A), the expression is similar to the dynamic energy loss due to the charge and discharge of load capacitors in digital circuits, i.e. $E_{dyn} = C_P V_{DD}^2$. However, in our case, the oscillator energy loss (eq.2.28) is modulated by the DC output voltage operating point $\overline{V_{out}}$. Note that closed-form expressions for $\overline{V_{out}}$ and T_{osc} are established in the Appendix A but are not listed here for clarity. However, the mean power consumption of a single oscillator can be expressed as:

$$P_{osc} = \frac{V_{DD}}{R_S}\overline{V_{out}} \tag{2.29}$$

Hence, the two key knobs to obtain low-energy ONNs are low operating voltages and low parasitics. The next section presents the oscillator energy when coupled to $N-1$ other oscillators.

2.7.2 ONN Synaptic Operations

We first define the intrinsic synaptic operation between coupled oscillators. For oscillator i, its synaptic input weighted sum $h_i(t)$ can be conceptually expressed as:

$$h_i(t) = \sum_{j=1}^{N-1} W_{ij}\,\Delta\phi_j(t) \tag{2.30}$$

where W_{ij} are the synaptic weights and $\Delta\phi_j(t)$ are the phases of other oscillators (Fig.2.12a). Then, the role of the oscillating neuron is to produce an output phase by applying a non-linear activation function a to its input:

$$\Delta\phi_i = a(h_i) \tag{2.31}$$

We define a synaptic operation (SOP) in ONN as the evaluation of the quantity $W_{ij}\Delta\phi_j$. Note that up to now, the study is hardware-agnostic, and a SOP could be implemented in various manners such as with digital circuits [125] or using the analog Ohm's law. Using these definitions, the neuron energy is expressed as the sum of two contributions:

$$E_{neuron} = E_{input} + E_{activation} \tag{2.32}$$

E_{input} is the loss related to the evaluation of the input weighted sum, whereas $E_{activation}$ is the energy needed to produce an output, i.e., to determine the phase difference. Again, (eq.2.32)

is general enough so it can capture any type of implementation and computing (sequential or parallel). In the interesting case where neurons process information in parallel, the neuron energy becomes:

$$E_{neuron} = \left((N-1)\,E_{SOP} + E_{osc}\right) N_{cycles} \tag{2.33}$$

where N_{cycles} is the number of oscillating cycles before settling to a stable output phase state, and E_{osc} is the energy of a single oscillation. One interesting aspect of analog ONN is that sometimes SOP can be energy-free. For instance, when two coupled oscillators are in-phase the synaptic current is null and $E_{SOP} = 0$ (see Fig.2.12c). The worst-case SOP energy occurs when two oscillators are out-of-phase: the maximum amplitude across the synaptic resistor reaches $\Delta V_{max} = V_H - V_L$ and induces Joule's loss (see Fig.2.12b). As the SOP analytical expression depends on the oscillating waveform, we evaluate here the worst-case for simplicity and consider that a DC voltage ΔV_{max} is applied to every coupling resistor R_{C_N} during the entire oscillating period:

$$E_{SOP} = \frac{\Delta V_{max}^2}{R_{C_N}} T_{osc} \tag{2.34}$$

To assess how the ONN energy scales with N, the ONN computation time, N_{cycles}, must be first evaluated. The next section presents circuit simulations of various ONN sizes dedicated to pattern recognition to estimate N_{cycles}.

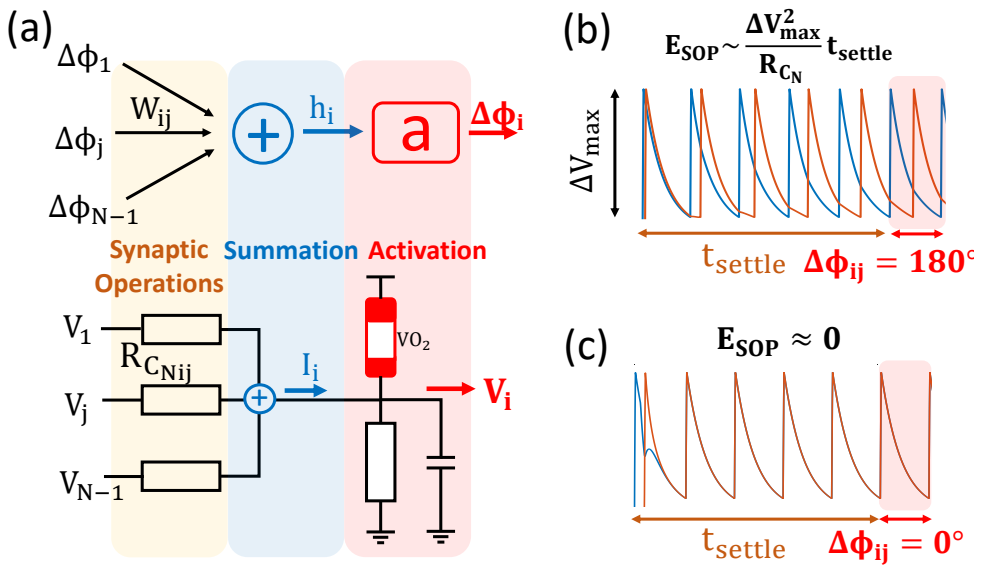


Figure 2.12: *(a) ONN neuron model (top) and analog implementation using coupling resistors and a VO_2-oscillator. In this work, synaptic operations occur via current flow through coupling resistors and the input summation naturally happens in current mode. (b) When two coupled oscillators are out of phase, a synaptic current flows through the coupling resistor, and energy is lost by the Joule effect. The case where the two oscillators are out-of-phase during t_{settle} corresponds to the maximum SOP energy loss. (c) The other extreme case occurs when two coupled oscillators are in phase during t_{settle}; the current flow is null and $E_{SOP} = 0$.*

2.7.3 ONN Settling Time and Energy Scaling

We define the *ONN settling time* as the time t_{settle} required for ONN signals to be periodically stable:

$$t_{settle} = N_{cycles} T_{osc} \tag{2.35}$$

For $t \geq t_{settle}$, ONN phases can be measured as they are stable. For example in Fig.2.10A, the ONN stabilizes to a stable pattern after $N_{cycles} \approx 5$ cycles. Simulations for different ONN sizes allow the estimation of the ONN settling time by varying **1)** the number M of stored patterns and **2)** the number of noisy pixels in test images from 10% to 50%. Fig.2.13 shows the simulation results. Interestingly, the ONN settling time is approximately constant and is smaller than 5 cycles in most cases. Hence, the ONN parallel computation allows computing in constant time even for large networks. This result corroborates what has been observed with oscillator-based Ising machines [162], i.e., coupled oscillators converge to a solution (not necessarily the optimal one) in quasi-constant time. In fact, we will see in Chapter 3 that the computation time seems to scale logarithmically.

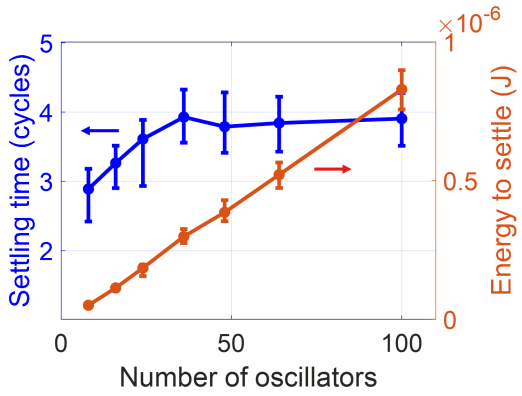


Figure 2.13: *ONN settling time and energy for different values of N. Settling time remains approximately constant when scaling up N. Oscillators truly act as parallel processing units. Energy to settle scales linearly with N. Medians, first and third quartiles of simulation results are represented.*

Moreover, the ONN energy scales linearly (see Fig.2.13) when ONN satisfies the two following properties:

1. ***Parallelism***: the computation time t_{settle} remains quasi-constant.
2. ***Downscaling of synaptic energy***: the coupling resistors R_{C_N} are scaled as:

$$R_{C_N} = (N-1)R_{C_2} \tag{2.36}$$

where R_{C_2} is the coupling resistance between two coupled oscillators (eq.2.23). The synaptic loss E_{SOP} becomes:

$$E_{SOP} = \frac{\Delta V_{max}^2}{(N-1)R_{C_2}} T_{osc} \tag{2.37}$$

Therefore, even though the number of synapses grows quadratically, the ONN energy grows only linearly with the number of oscillators. This can be verified using our previous definitions

(eq.2.28, eq.2.33, eq.2.37):

$$\begin{aligned} E_{ONN}^{analog} &= N\,E_{neuron} \\ &= N\big((N-1)\,E_{SOP}+E_{osc}\big)\,N_{cycles} \\ &= N(\frac{\Delta V_{max}^2}{R_{C_2}}+\frac{V_{DD}}{R_S}\overline{V_{out}})N_{cycles}T_{osc} \end{aligned} \tag{2.38}$$

Note that we have not yet considered any peripheral circuits that could change the ONN energy scaling law when implemented in real hardware. For instance, although the energy of the analog ONN computing core grows linearly (eq.2.38), there would still be a quadratic number of synapses that would need to be programmed. But in terms of computing, the analog ONN is promising when compared to digital architectures. In the latter case, the energy of a synaptic operation (such as multiply and accumulate (MAC)) generally remains constant and can hardly be scaled down. By considering a fully digital ONN computing with MACs rather than analog currents, its total energy would grow quadratically as:

$$E_{ONN}^{digital} = N\big((N-1)\,E_{MAC}+E_{osc}\big)\,N_{cycles} \tag{2.39}$$

In the previous simulations of Fig.2.13, we considered ONNs with a large supply voltage $V_{DD} = 2.5$ V leading to an important energy consumption of 2 nJ/oscillator/cycle. Whereas for instance Jackson et al. in [89] have designed an ONN consuming 1.21 pJ/oscillator using a hybrid design (analog synapses and digital neurons) in 28 nm CMOS technology. The next section presents how to scale VO_2 devices to achieve competitive performances with respect to state-of-the-art solutions.

2.7.4 Oscillator Energy Minimization using Scaled VO_2 Devices

Here, we study how to minimize the energy for a VO_2-based oscillator using the formulation (eq.2.28) and assisted by TCAD simulations. The TCAD modeling and simulation flow is described in recent work [150, 163]. VO_2 devices in crossbar (CB) geometry [85] is considered as a potentially scalable geometry to lower the oscillator energy consumption (Fig.2.14a). By reducing the VO_2 CB size, the overall VO_2 thermal dissipation decreases and the VO_2 device can transition to a metallic state with less power [150]. The applied voltage can then be reduced for given insulator and metallic states that are set by the material properties and contact area (Fig.2.14b).

As our model predicts that the oscillator energy scales quadratically with voltages (eq.2.28), it is of interest to scale down the VO_2 CB dimensions. Fig.2.15 shows results of TCAD simulations for various CB (500 nm, 1 μm, 1.5 μm, 2 μm, 3 μm, and 4 μm) and biasing parameters. It can be seen from Fig.2.15a that the VO_2 threshold voltages V_H and V_L are approximately proportional to CB and allow a linear V_{DD} scaling. With reduced CB, the oscillating voltage amplitude can be decreased (Fig.2.15b) for low power operation (Fig.2.15c). As the material, contact area, and load capacitor remain the same for all CB sizes, the oscillating period does not vary significantly and the minimum energy is obtained for CB=500 nm (Fig.2.15d).

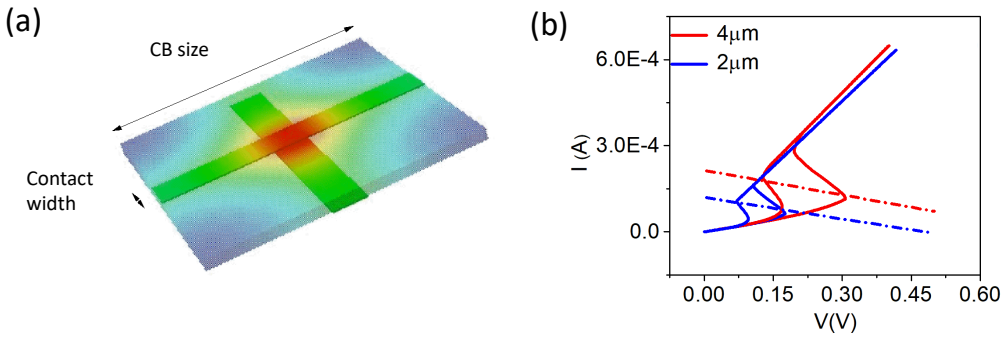


Figure 2.14: *(a) Structure of the VO_2 crossbar (CB) device. The top and bottom electrodes are in cross-like configuration. They have the same contact width of 250 nm. The VO_2 layer of 80 nm thickness is sandwiched between them. The color map overlapped with the geometrical structure accounts for the temperature distribution across the device at the highest simulated voltage. (b) Device $I-V$ obtained through electrothermal TCAD simulation of CB=4 μm (red solid line) and CB=2 μm (blue solid line) devices. The simulations have been performed in voltage-controlled mode, by applying the voltage to a circuit composed of the VO_2 device connected in series to an external resistor of R_S = 1 kΩ. The dashed-dotted lines represent the associated load lines.*

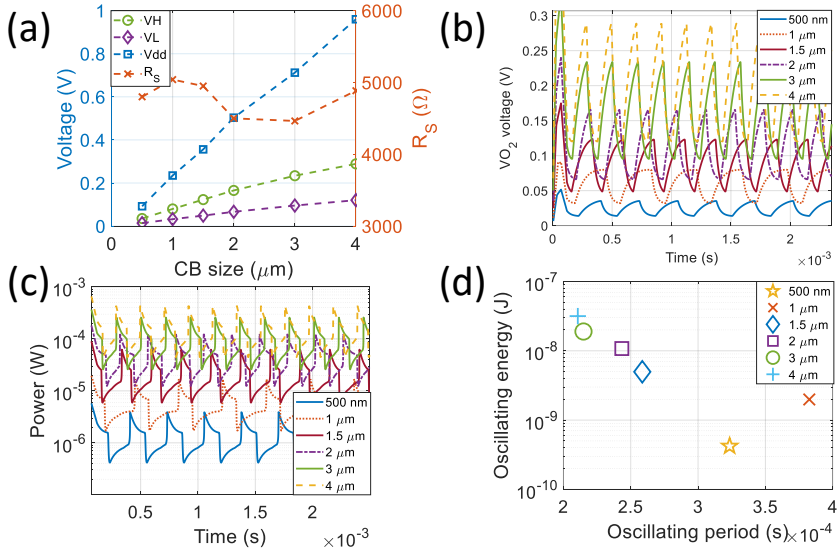


Figure 2.15: *TCAD simulation results for the same crossbar (CB) geometry and C_P=5 nF. (a) Oscillator parameters with respect to the VO_2 CB. By scaling down the VO_2 CB, the thermal dissipation decreases and the device needs less power to transition from one state to the other. Therefore, the VO_2 thresholds V_H and V_L decrease with CB. V_{DD} is scaled down approximately linearly with V_H and V_L. The load resistor R_S is adapted in each case to place the load line in the NDR region and obtain oscillation. (b) Transient voltage across VO_2 devices for different CB. (c) Instantaneous power for different CB sizes. Scaling down CB leads to low oscillation amplitude and low power. (d) Oscillator energy vs period for various CB.*

Fig.2.16 shows the comparison between our analytical model (eq.2.28) and mean power and energy computed with TCAD for different CB sizes. There is a good match for the mean power but some deviation when evaluating the energy. We believe this is mainly due to non-linearities induced by thermal effects, which result in a larger oscillating period and thus a higher energy consumption [163]. This aspect is not captured by our analytical formalism as it only considers electrical variables (Fig.2.16b). Nevertheless, the scaling trend of our model is in agreement with TCAD simulations and we use it for benchmarking ONN with state-of-the-art chips.

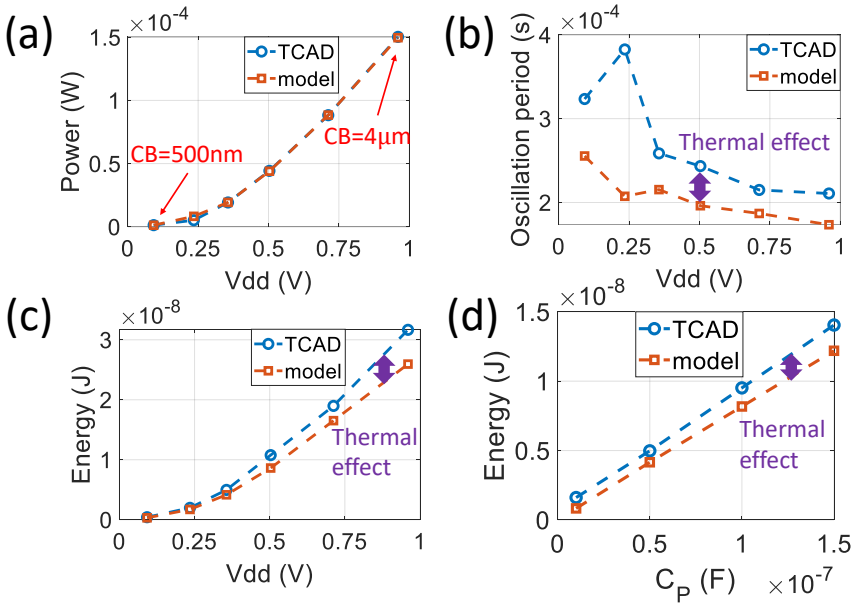


Figure 2.16: *Comparison between TCAD and analytical model for (a) Oscillator mean power (b) Oscillating period and (c) Oscillator energy with respect to V_{DD}. Our analytical model does not include thermal effects which slow down oscillations and increase the energy. Nevertheless, our model (eq.2.28) captures well the quadratic V_{DD} scaling law. (d) Both TCAD and our model predict a linear energy scaling law with respect to the oscillator load capacitance. CB=1.5µm is considered here.*

2.8 ONN Benchmarking

2.8.1 Neuron Energy-Delay Benchmark

Benchmarking ONN with other architectures is not trivial as ONN is a phase-based neuromorphic system and does not perform conventional MAC operations. However, the concept of synaptic operation (SOP) is shared among all sorts of neural inference chips and can serve as common ground for benchmarking. In ANNs, an SOP is defined by the multiplication between the input and the synaptic weight. Then, it can be naturally implemented in digital hardware by a MAC operator and in this case, there is the equivalence 1 SOP≈1 MAC [91]. In SNNs, an SOP is generally defined as the event where an action potential (or spike) propagates through a synapse [36]. Similarly, we previously defined an SOP in ONN as the oscillation propagation through a synapse during one oscillation period. Beyond SOP metrics, information about the time and energy dissipated by a single neuron to produce an output is challenging to find in the literature. Generally, the accessible metrics common to all sorts of chips are the following:

- Chip synaptic throughput T_{ch} in SOP/s.
- Chip total power P_{ch}.
- Number of neurons N and number of synapses per neuron N_S.

To find the average processing time of a neuron, we propose a simplistic approach that is nevertheless in agreement with existing benchmarks recently proposed by Nikonov and Young [91]. We first make distinctions between the SNN, ANN, and ONN paradigms. Fig.2.17 illustrates SNN, ANN, and ONN chips having the same synaptic throughput $T_{ch} = 30$ SOP/s, $N = 2$ and $N_S = 3$.

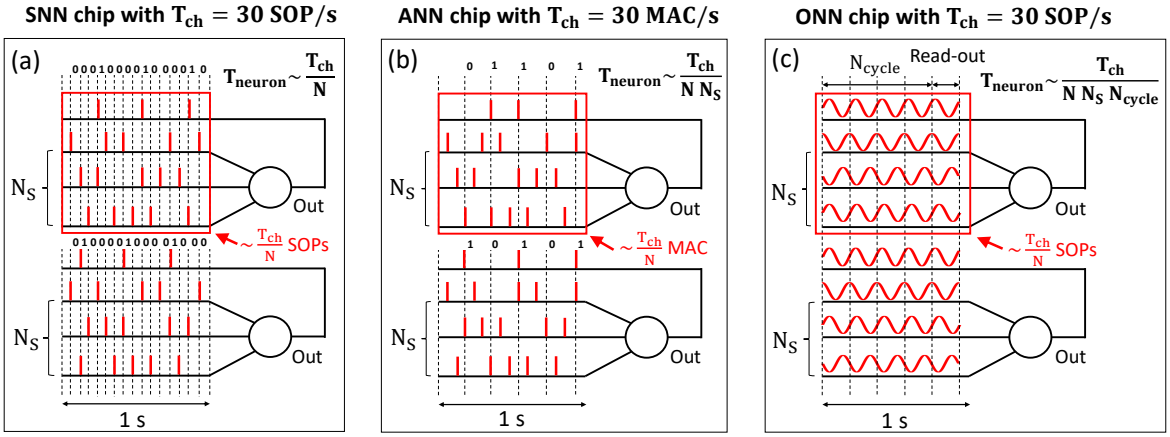


Figure 2.17: *Illustration of synaptic and neuronal throughput for (a) an SNN chip, (b) an ANN chip, and (c) an ONN chip. The chips have $N = 2$ neurons with $N_S = 3$ synapses each. The synaptic throughput for all chips is set to $T_{ch} = 30$ SOP/s. In SNNs, the neuron membrane potential is updated for each input spike and can potentially produce an output spike when receiving a single weighted spike. In ANNs, each neuron needs to receive all the input MAC before producing an output. In ONNs, neurons' outputs are measured after N_{cycle} oscillations.*

Note that this is purely illustrative, as SNN and ANN chips have T_{ch} in the range of $[10^6 - 10^{10}]$ SOP/s and $[10^{10} - 10^{14}]$ SOP/s, respectively. Moreover, the schematic assumes a uniform distribution of SOPs, whereas in practice SOPs could occur differently depending on the architecture and information encoding [36]. Given T_{ch}, N, and N_S, we can only infer the average number of SOPs per neuron within a time window.

In SNNs, each synaptic spike (SOP) updates the neuron membrane potential which can induce an output spike when the potential is above a threshold. In other words, the neural output is sensitive to each SOP and produces an output, '0' or '1', according to its internal state (Fig.2.17a). We define the neuron throughput in SNN as the neural output rate $T_{neuron}^{SNN} = T_{ch}/N$, which is different from the output firing rate (rate of '1's). Consequently, we define the average neuron delay in neuromorphic chips as:

$$delay_{neurom} = \frac{1}{T_{neuron}^{SNN}} = \frac{N}{T_{ch}} \tag{2.40}$$

In ANNs, the neurons need to evaluate all the input SOPs (MACs) before updating the output. Hence, the neuron throughput is $T_{neuron}^{ANN} = T_{ch}/(NN_S)$. Note that this is also valid when the MAC operations are calculated in parallel, which is the main objective of neural accelerators. Thus, we define the average neuron delay in ANN accelerator chips as:

$$delay_{acc} = \frac{1}{T_{neuron}^{ANN}} = \frac{NN_S}{T_{ch}} \tag{2.41}$$

Knowing the power consumed in the chip P_{ch}, we express the neuron energy loss to produce an output for SNN chips as:

$$E_{neurom} = \frac{P_{ch}}{N} delay_{neurom} = \frac{P_{ch}}{T_{ch}} \tag{2.42}$$

The last expression is equal to the energy lost while processing a single SOP and is in agreement with our previous definition of SNN neural output. Similarly, we express the neuron energy loss

to produce an output for ANN accelerators as:

$$E_{acc} = \frac{P_{ch}}{N} delay_{acc} = N_S \frac{P_{ch}}{T_{ch}} \tag{2.43}$$

In ONNs, the computation is based on the synchronization of coupled oscillators which require an average of N_{cycle} oscillation cycles before producing a steady output. Thus, the oscillator delay is simply expressed as:

$$delay_{ONN} = N_{cycle} T_{osc} \tag{2.44}$$

and the oscillator energy becomes:

$$E_{ONN} = \frac{P_{ch}}{N} delay_{ONN} = \frac{P_{ch}}{N} N_{cycle} T_{osc} \tag{2.45}$$

In general, ONNs need less than 10 oscillation cycles to settle and we set $N_{cycle} = 5$ in this benchmark, as observed in section 2.7.3.

The VO_2-ONN architecture is benchmarked using VO_2 devices with CB=500nm, ΔV_{max}=21 mV, and various load capacitors. TCAD simulations were initially carried out to fit experimental oscillations where circuits employ nano-farad load capacitors [110]. However, in literature, faster VO_2 oscillations up to 9 MHz have been reported [164] and we believe crossbar VO_2 could reach a similar speed with lower load capacitors. Thus, we project the oscillator energy and delay for lower capacitances down to 500 fF using our analytical model (eq.2.28) and (eq.2.35).

The power dissipated by peripheral circuits, i.e., the phase initialization and measurement circuits, are included to obtain a more precise energy assessment. In the worst case, one digital-to-time converter (DTC) per oscillator is needed to set the oscillator input phase. As an example, we consider a 9-bit DTC consuming 31 μW at 40 MHz in 28 nm CMOS technology [165] suitable for low-power edge applications. For the phase measurement, we take the example of the circuit described in [166] that consumes 20.5 μW in 28 nm CMOS technology. Overall, the peripheral circuits clocked at 30 MHz consume P_{periph}=60 μW per oscillator, which gives 2 pJ per cycle. As a first-order estimation, P_{periph} is considered proportional to the neuron oscillating frequency which gives a constant peripheral energy loss $E_{periph} = P_{periph} T_{osc} N_{cycles}$. The number of cycles is $N_{cycles} \approx 5$ and E_{periph}=10 pJ. Note that our estimation remains optimistic as we use a bottom-up type of energy-delay assessment, whereas state-of-the-art data correspond to real chip measurements.

Fig.2.18 shows the neuron energy delay for various SNN neuromorphic chips (blue circled dots), digital neural accelerators (red squared points) considered in previous work [91] and VO_2 oscillators with different load capacitances. The benchmark data is listed in Tables 2.2, 2.3 and 2.4. When the oscillator load capacitance increases, the oscillator slows down and its energy to produce a stable output phase increases. Similarly, neuromorphic SNN chips lie on the right-hand side of the plot as they generally produce spikes at lower frequencies than digital neural accelerators [91]. From the neuron energy point of view, it appears that VO_2-based ONNs can compete with state-of-the-art SNN neuromorphic chips for a similar neuron delay.

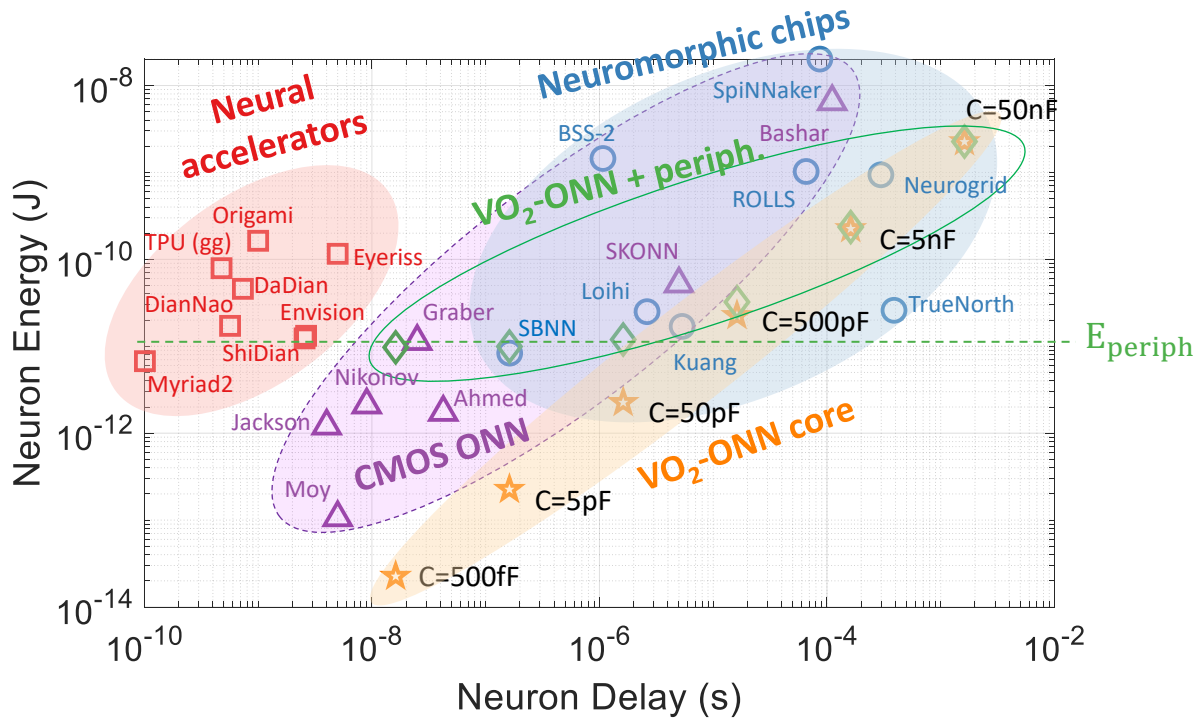


Figure 2.18: *Neuron energy and delay to produce an output for various chips. Red squared markers are digital neural accelerators optimized for efficient matrix-vector product (MAC operations). Blue circled points are neuromorphic chips that implement SNNs. Green diamond markers are VO_2 oscillators with CB=500 nm and various load capacitances including peripheral circuits. Orange star markers are ONN neurons standalone without any peripheral circuits. Purple triangular points are ONNs designed with CMOS technology.*

	BSS-2 ASIC [38]	SpiNNaker [167]	Neurogrid [39]	ROLLS [168]	Loihi [169]	True North [41]	SBNN [165]	Kuang et al. [170]
Application	Neuroscience simulation				Neuromorphic computing			
#core	1	768	15	1	128	4096	64	64
N/core	512	150	256x256	256	1024	256	64	1024
N_S	256	750	7980	512	1000 (1bit)	256	256	1024 (1 bit)
T_{ch} (MSOP/s)	477	1330	3300*	3.9*	50 000 (@1V)	2700	25 200 (@0.9V)	12 290 (@1.2V)
P_{ch} (W)	0.69	26.8	3.1	0.004 (@1.8V)	1.25*	0.07	0.21*	0.208 (@1.2V)
Firing rate (Hz)	3600*	15	0.42	30	380*	20	24 000*	183*
Time step of dynamics	NA	-	NA	NA	-	1 ms	-	-
$\mathbf{delay_{neurom} = N/T_{ch}}$ (μs)	**1***	**90***	**300***	**64***	**2.6***	**400***	**0.16***	**5.3***
$\mathbf{E_{neurom} = P_{ch}/T_{ch}}$ (pJ)	**1400***	**20 000***	**940***	**1025***	**25***	**26***	**8.3***	**17***
E_{SOP} (pJ)	-	7000	-	3.7	-	-	-	4.6 (@1.2V)

Table 2.2: *High-level features of Neuromorphic chips implementing SNNs. *Derived values.*

	DianNao [171]	ShiDianNao [172]	DaDianNao [173]	Origami [174]	Eyeriss [175]	Envision [176]	TPU v4 Google [177]	Myriad2 [178]
Application	CNN/DNN	CNN	CNN/DNN	CNN	CNN	CNN	DNN/CNN	Vision processing
#core	1	1	16	1	1	1	2x4	12
N/core	16	64	16	4	12	16	128	2 (16-bit)
N_S	16	8	16	49	14	16	128	8 (16-bit)
T_{ch} (GOP/s)	452	194	5580	196 (@1.2V)	33.6 (@1V)	100	275 000	950 (16-bit)
P_{ch} (W)	0.485	0.32	15.97	0.654 (@1.2V)	0.278	0.077	170	0.8
Clock frequency (GHz)	0.98	1	0.606	0.5	0.2	0.2	1.05	0.6
$\mathbf{delay_{acc} = NNs/T_{ch}}$ (ns)	**0.57***	**2.6***	**0.73***	**1***	**5***	**2.6***	**0.48***	**0.2***
$\mathbf{E_{acc} = N_S P_{ch}/T_{ch}}$ (pJ)	**17***	**13***	**46***	**163***	**116***	**12***	**79***	**6.7***

Table 2.3: *High-level features of digital accelerators. *Derived values.*

	Jackson et al. [89]	Nikonov et al. [95]	Ahmed et al. [74]	Moy et al. [76]	SKONN [179]	Bashar et al. [138]	Graber et al. [90]
Application	Image Processing		Combinatorial Optimization				
#core	1	1	1	1	1	1	1
N/core	100	26	560	1968	16	30	400
N_S	100	1	6	4	16	30	4
f_{osc} (MHz)	1000	8000	118	1000	1	0.045	200
P_{ch} (mW)	303	6.76	23	42	0.167 (core)	1.76	182
$\mathbf{delay_{ONN} = N_{cycle}/f_{osc}}$ (ns)	**4***	**9**	**42***	**5***	**5 000***	**111 000***	**25***
$\mathbf{E_{ONN} = N_{cycle}T_{osc}P_{ch}/N}$ (pJ)	**1.21***	**2.1***	**1.74***	**0.11***	**52***	**6500***	**11.3***

Table 2.4: *High-level features of ONN chips. *Derived values.*

With real chip measurements which would include all peripheral energies, the ONN region would probably shift up and lie in the SNN neuromorphic region in the worst case. The VO_2 oscillator could compete with neural accelerators at energy level but would be orders of magnitude slower with load capacitances larger than 500 fF. For instance, a neuron from DianNao [171] accelerator produces an output after 570 ps whereas it would take 16 ns to phase lock for a scaled VO_2 oscillator with C_P=500 fF. Interestingly, peripheral circuits set the minimum achievable neuron energy for load capacitances smaller than 50 pF (green diamond points), whereas the energy of the ONN neuron standalone can reach sub-picojoule ranges (orange star points). From our first-order estimation, we conclude that the energy and delay of a VO_2-ONN can be very competitive under the two following conditions:

1. The oscillating frequency is in the GHz range, i.e., the load capacitance C_P <50 fF and assuming that the VO_2 thermal time constant remains negligible [150].
2. A careful design of peripheral circuits is required to fully take advantage of the ONN phase computing paradigm.

As an alternative to VO_2 oscillators, CMOS ONNs (purple triangular points) are currently very competitive as they use scaled transistors from a mature CMOS technology. For instance, the first phase-based ONN chip ever reported for pattern recognition is the digital ONN designed by Jackson et al. [89] with 100 neurons and 10,000 synapses using a 28 nm CMOS technology. Their results are promising as they measured a 1.21 pJ neuron energy and 4 ns delay. For fast convolution inference, Nikonov et al. recently reported on an ONN chip fabricated in a 22 nm FinFet CMOS process that computes in less than 10 ns and consumes 2 pJ/oscillator [95]. In the field of oscillator-based Ising machines (OIM) [162], Moy et al. [76] revealed an OIM composed of 1968 ring oscillators in 65 nm CMOS technology that only consume 110 fJ/oscillator for N_{cycles}=5 and f_{osc}=1 GHz. These recent examples further highlight the ONN potential to perform various tasks at high speed and low energy.

Finally, it is important to stress that benchmarking different architectures at the neuron level only gives a limited vision of the chips' potential as they are ultimately used to solve practical problems. For example, Nikonov's ONN and the neural accelerator DianNao [171] have almost the same energy and delay when used to compute convolutions [95]. Next, we choose to benchmark a VO_2-ONN in the case of image edge detection which is a widely used task in image processing.

2.8.2 Edge Detection Benchmark with ONN

Here, we aim to benchmark VO_2-ONNs with other works on a specific image edge detection application. Similar to edge detection algorithms that employ 3x3 or 5x5 convolution kernels [180], the input image is scanned by a 3x3 ONN to extract edges. Analogous phase-based edge detection algorithms have already been proposed in literature [118, 181] but we rather focus on the analog hardware implementation to assess how a VO_2-ONN benchmarks with the state-of-the-art edge detection hardware.

We consider a fully-coupled ONN composed of 10 oscillators and 45 coupling resistors where 9 oscillators scan the input image with a padding of 1, and the 10th oscillator makes the final decision (Fig.2.19a). Using the Hebbian learning rule [68], the ONN is trained to detect edges in the vertical, horizontal, and diagonal directions and the Hebbian coefficients are mapped to coupling resistors using the mapping function defined in [98] (Fig.2.19b). To detect the background, we bias the VO_2 oscillators such that the 0° phase state is more likely to occur (R_S=6 kΩ instead of 20 kΩ), further explained in section 2.9.3. As shown in Fig.2.19c and Fig.2.20b, the oscillators converge in-phase when initialized with similar input phases and the ONN detects the background. Fig.2.20a shows an example where the ONN detects a vertical edge. Note that the 10th output oscillator is always initialized with an input phase of 90° to not favor any particular output state. As already highlighted in section 2.7.3, the ONN makes the decision after a few oscillation cycles only (between 3 and 5).

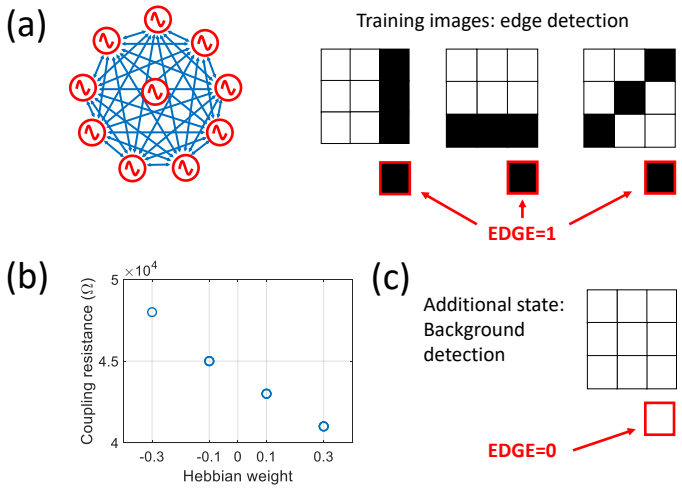


Figure 2.19: *a) 10 fully-connected oscillators trained to detect vertical, horizontal, and diagonal edges in images. b) Mapping of Hebbian coefficients to coupling resistors. c) ONN state that detects the image background.*

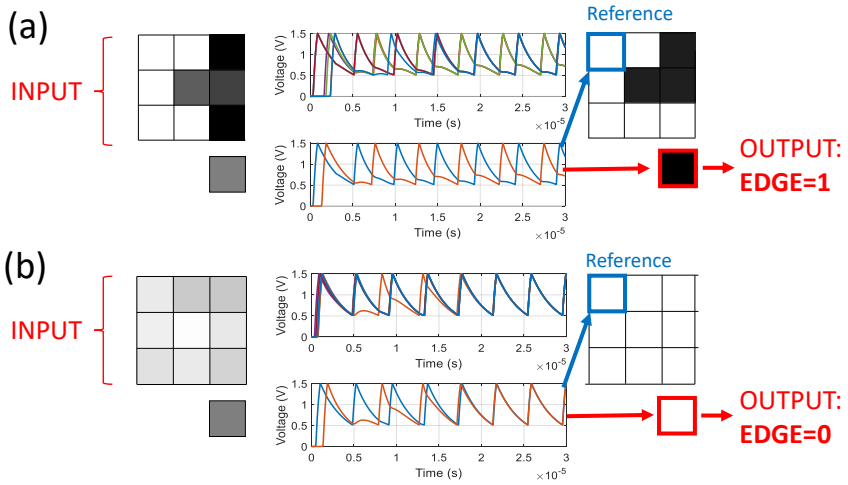


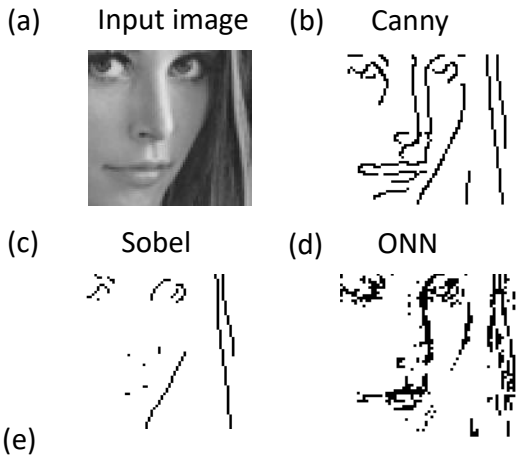
Figure 2.20: *From left to right: 3x3 portion of an input image, oscillators' waveforms, output ONN state in the case of a) vertical edge and b) uniform background.*

We compare our ONN image detection with the state-of-the-art Sobel and Canny edge detection methods [180, 182] that are evaluated in Matlab using built-in functions. To quantitatively benchmark edge detection algorithms, most methods compare the output image with a ground truth solution [183]. We consider Canny's result as the ground truth solution as it is a widely used edge detection algorithm known for its good performances [184, 185]. Given the set of positive pixels (edges) P_c produced by the Canny algorithm, we compute four sets that quantify how well the produced set of edges P matches the ground truth solution. True Positives are pixels that match the ground truth positive pixels: $TP = P_c \cap P$; False Positives are positive pixels that do not match the ground truth: $FP = \overline{P_c} \cap P$; False Negatives quantify the missing edges: $FN = P_c \cap \overline{P}$ and True Negatives are correctly undetected edges: $TN = \overline{P_c} \cap \overline{P}$.

With the 8-bit 64x64 gray-scale example in Fig.2.21a, Canny produces $|P_c|$=469 positive pixels that are considered as the ground truth set (Fig.2.21b). The ONN outputs more edges than Sobel (578 vs 184) and some regions such as Lena's nose and mouth are more visible. The ONN has more TP pixels (179 vs 137) but most of the ONN edge pixels are not aligned with Canny's edges as the ONN produces 399 FP pixels (69%); whereas Sobel only has 47 FP pixels (25%). This misalignment is also reflected by FN pixels that correspond to missed edges as $|FN|/|P_c|$=62% of Canny's edges are missed by the ONN edge detector. With respect to the Jaccard similarity defined as $J = |TP|/\big(|TP| + |FP| + |FN|\big)$ [186, 187], Sobel is more similar to the ground truth than ONN as $J(C,S)$=0.265 and $J(C,ONN)$=0.206.

Using a similar approach, we benchmark VO_2-ONNs on nine different 512x512 black and white images [188] (Fig.2.22a). By considering Canny's edge set as the ground truth, it appears that Sobel produces more TP pixels and fewer FN pixels than the ONN edge detector (Fig.2.22b). Hence, there is a better match between Sobel's output and the ground truth compared to the ONN output. For instance, Fig.2.23b illustrates the good match between Canny and Sobel outputs as $|TP| > |FN| > |FP|$ (black and magenta pixels are dominant); whereas the ONN output in (Fig.2.23c) is dominated by green FP pixels, suggesting again that the ONN edges are at different locations compared to Canny's. Finally, the larger mismatch between ONN and Canny also appears in the Jaccard similarity measure for the nine images as $J(C,S) > J(C,ONN)$. Overall, the ONN produces at least as many edges as Canny (except for Image 9: see Fig.2.22b) but are at different locations. In contrast, the Sobel algorithm detects

fewer edges that are better matched to Canny's.



	|P|	|TP|	|FP|	|FN|	|TN|	J
Sobel	184	137	47	332	3580	0.265
ONN	578	179	399	290	3228	0.206

Figure 2.21: *a) 64x64 8-bits gray scale image [188]. b), c) and d) are the output images using Canny, Sobel, and ONN edge detection methods, respectively. e) Comparison between Sobel and ONN edge detections by considering the Canny edge set P_c as ground truth with $|P_c|$=469. $|P|$ is the number of positive pixels that correspond to edges. $|TP|$, $|FP|$, $|FN|$, $|TN|$ are the number of True Positive, False Positive, False Negative and True Negative pixels compared to Canny, respectively. J is the Jaccard similarity with Canny's edge set.*

Table 2.5 shows the performances of edge detection ASICs implemented in 65 nm [189] and 45 nm [190] CMOS technologies. Both accelerators are optimized to run the Canny algorithm and are suitable for edge applications thanks to their low power consumption. A VO_2-ONN with a crossbar size of 500 nm is considered to achieve low power operations along with various load capacitances to set the oscillating frequency. A single ONN running at 31 MHz would process a 512x512 image in 42 ms and would be 100x slower than Soares's ASIC [190]. By reducing the capacitance load to 500 fF and parallelizing at least 10 ONNs, ONN could compete with state-of-the-art to achieve 0.42 ms/image.

	Hardware	**Frequency**	**Mean Power**	**Image size**	**Time /image**	**Energy/pixel**
Lee 2018 [189]	ASIC (65 nm)	500 MHz	5.48 mW	1280x720	2.2 ms	13.2 pJ
Soares 2020 [190]	ASIC (45 nm)	350 MHz	6.7 mW	512x512	0.42 ms	10.7 pJ
ONN1	10 VO_2-oscillators C=5 pF	31 MHz	13 μW + 330 μW (periph.)	512x512	42 ms	2.1 pJ + 53 pJ (periph.)
ONN2	10 VO_2-oscillators C=500 fF	310 MHz	13 μW + 3.3 mW (periph.)	512x512	4.2 ms	0.21 pJ + 53 pJ (periph.)

Table 2.5: *Edge Detection Benchmark*

Again, the peripheral circuits' energy could become dominant for scaled VO_2-oscillators and would be 253x larger than the oscillator energy in this first-order estimation. This points out that a VO_2-ONN requires specific and optimized peripheral circuits to fully take advantage of the ONN paradigm. We also believe there is room for improvement in terms of power management as ONN only needs initialization and phase measurement circuits during the first and last oscillating cycle, respectively.

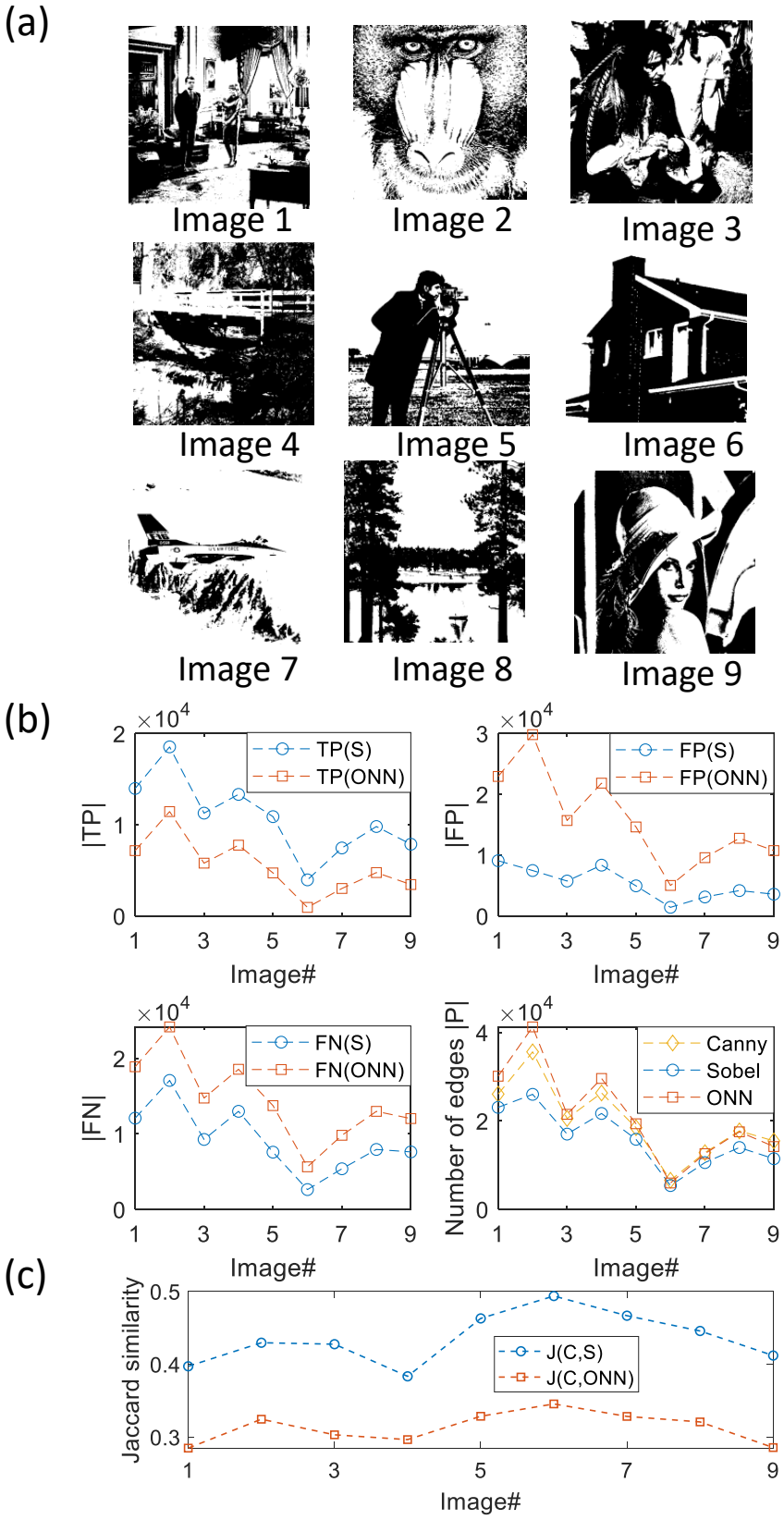


Figure 2.22: *a) 512x512 black and white images. b) Numbers of True Positive, False Positive, and False Negative pixels obtained by considering Canny as the ground truth. The bottom right plot shows the number of detected edges for each edge detector. c) Jaccard similarities between (Canny, Sobel) and (Canny, ONN).*

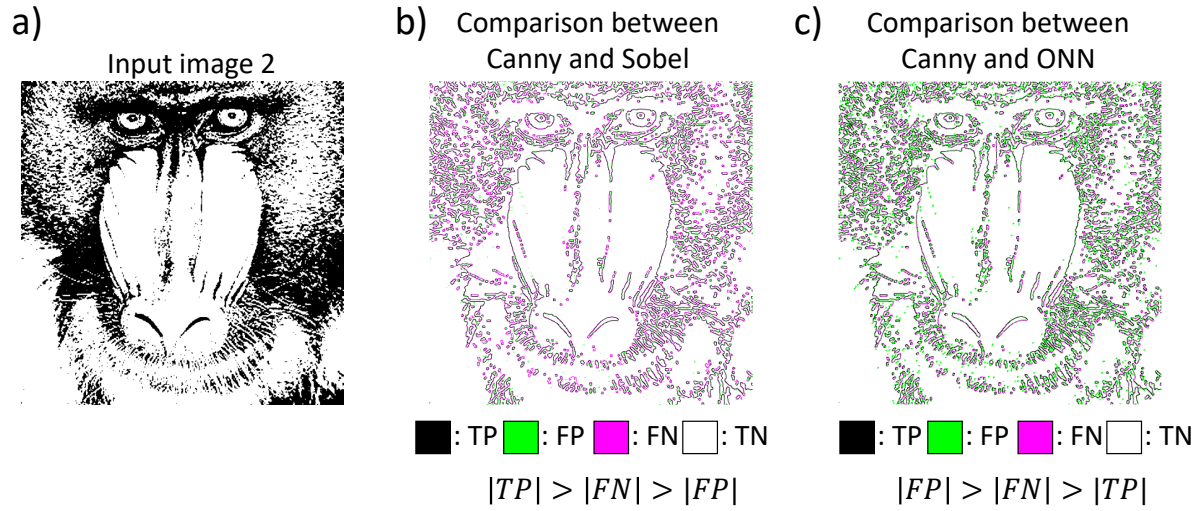


Figure 2.23: *a) Monkey binary 512x512 image. b) and c) Comparison between Sobel and ONN outputs, respectively, with Canny considered as the ground truth. Black, green, magenta and white pixels represent True Positive, False Positive, False Negative, and True Negative pixels, respectively.*

2.9 Architecture Limitations

While VO_2-based ONNs are promising for low-energy applications, the proposed architecture has three main limitations that are further described below. Overall, we find that the pattern recognition accuracy is greatly affected by the:

1. Synaptic resistance range.
2. VO_2 device variations.
3. Oscillator waveform shape.

2.9.1 Challenge of Synaptic Implementation

Here, we highlight the sensitivity of the proposed VO_2-ONN architecture with respect to the synaptic resistance range for pattern recognition applications. Our strategy is to evaluate the recognition accuracy of a 60-neuron ONN (Fig.2.10) trained with different coupling resistance ranges set by the mapping function parameter β. The test set consists of 20 different subsets S_k, $k \in \{1,2,..,20\}$ in which 60 different test patterns have k randomly located fuzzy input pixels. We notice from Fig.2.24C that the ONN achieves the best accuracy for an optimum value $\beta = N/32 = 1.875$. In this case, the ONN recognizes more than 80% of test images with up to 20% of noise. As seen in Fig.2.9C, the ONN accuracy is sensitive to the slope parameter β that sets the coupling resistance range. For instance, while increasing $\beta = 2$ to $\beta = 2.5$ only slightly changes the coupling resistance range, the accuracy decreases drastically in the latter case. This could be an issue in real circuits where the coupling resistance precision is limited by the technology process.

Thus, we assess the impact of R_C's relative variations and R_C's mean value on the ONN accuracy. R_C^{min} is the minimum resistance common to all coupling resistances, and ΔR is the additional series resistance to distinguish between weights (Fig.2.24A). Using the Hebbian rule,

weights are located near '0' coefficient as in Fig.2.9C and our mapping function can be fitted linearly (dashed lines). Therefore, every coupling resistances can be approximated by $R_C \approx R_C^{min} + n\Delta R^{min}$ with $n \in \{0,1,2,..,M\}$. Using this linear approximation, we check that the ONN accuracy is similar to the nominal mapping function with $\beta = N/32$, as shown in Fig.2.24C with the magenta dashed line.

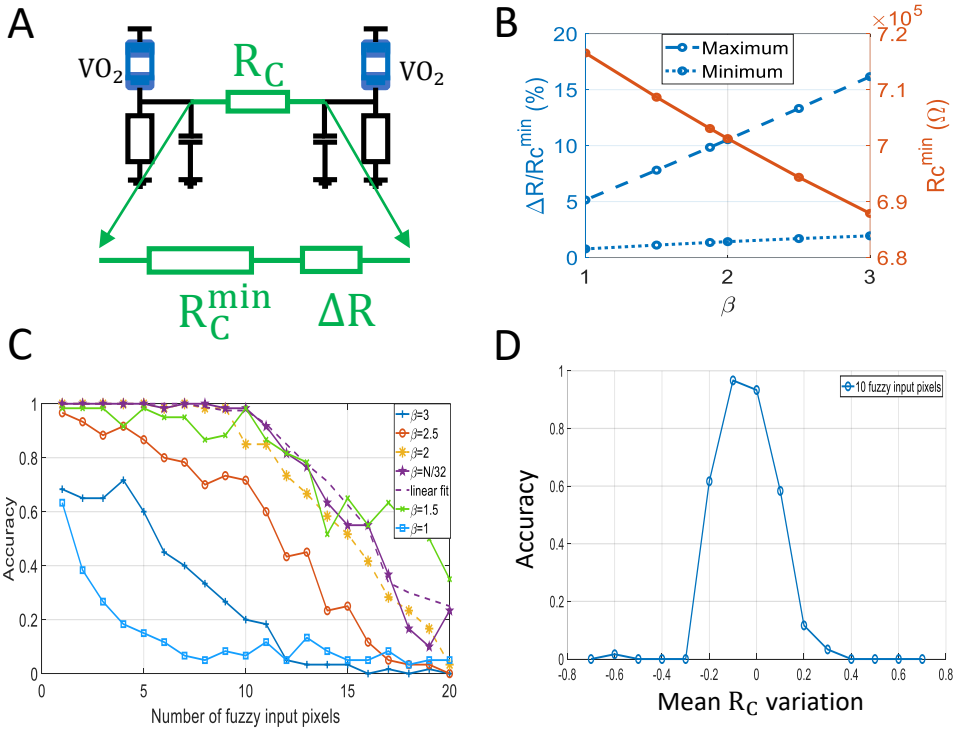


Figure 2.24: *(A) Two oscillators coupled by R_C, which can be decomposed in two series resistances: $R_C = R_C^{min} + \Delta R$. (B) Evolution of ΔR, R_C^{min} with respect to β. $\Delta R^{max} \approx 10\%\ R_C^{min}$ gives the best accuracy results. Note that $\Delta R^{min} = 1.7\%\ R_C^{min}$. (C) ONN recognition accuracy for different values of β. The dashed line is obtained for a linear fit of the mapping μ_N, i.e., with coupling resistances that are linearly spaced. (D) Impact of R_C's mean variation on recognition accuracy.*

It appears that the ONN accuracy is quite sensitive to the coupling resistances. We obtain $\Delta R^{max} \approx 15\%\ R_C^{min}$ for $\beta = 3$, and $\Delta R^{max} \approx 5\%\ R_C^{min}$ for $\beta = 1$ (Fig.2.24B). For these two cases as shown in Fig.2.24C, the ONN shows a poor accuracy. It is rather for $\beta = N/32 = 1.875$ with $\Delta R^{max} \approx 10\%\ R_C^{min}$ that the ONN accuracy is higher than 90%. To achieve the best ONN accuracy, a very good resistor matching is required, with a precision of $\Delta R^{min} = 1.7\%\ R_C^{min}$ between two consecutive weights. To study the influence of the R_C's mean value only, we apply the same variation to all coupling resistances for $\beta = N/32$ and for 10 fuzzy input pixels (Fig.2.24C). We notice that the mean value of coupling resistances can vary from -10% up to +5% from its nominal value to achieve a similar accuracy.

2.9.2 Impact of VO_2-oscillator variations

Fabricating reliable VO_2 devices is challenging [85] and ONN experiments with VO_2 are currently limited to few devices because of device variability [85, 107]. Here, we study the impact of VO_2 variability on the ability to phase-lock and on the synaptic range. The transition function $\zeta(R_C)$ (eq.2.9) defines the boundary between the binary phase regions and allows direct

identification of the neutral coupling resistor R_0 corresponding to the weight $W = 0$ (eq.2.14). Thus, we use ζ and R_0 as metrics to assess the impact of VO_2 parameters' variations. We apply relative variations on VO_2 parameters one at a time from -20% up to +20%, as shown in Fig.2.25A. Note that we vary V_H from -4% up to +4% only, as for larger positive variations oscillations do not occur (load line crosses the insulating branch and forms a fixed point). We solve (eq.2.12) to find ζ with the varied VO_2 parameters by checking that ζ also matches the phases boundary obtained via transient simulations, as in Fig.2.5A.

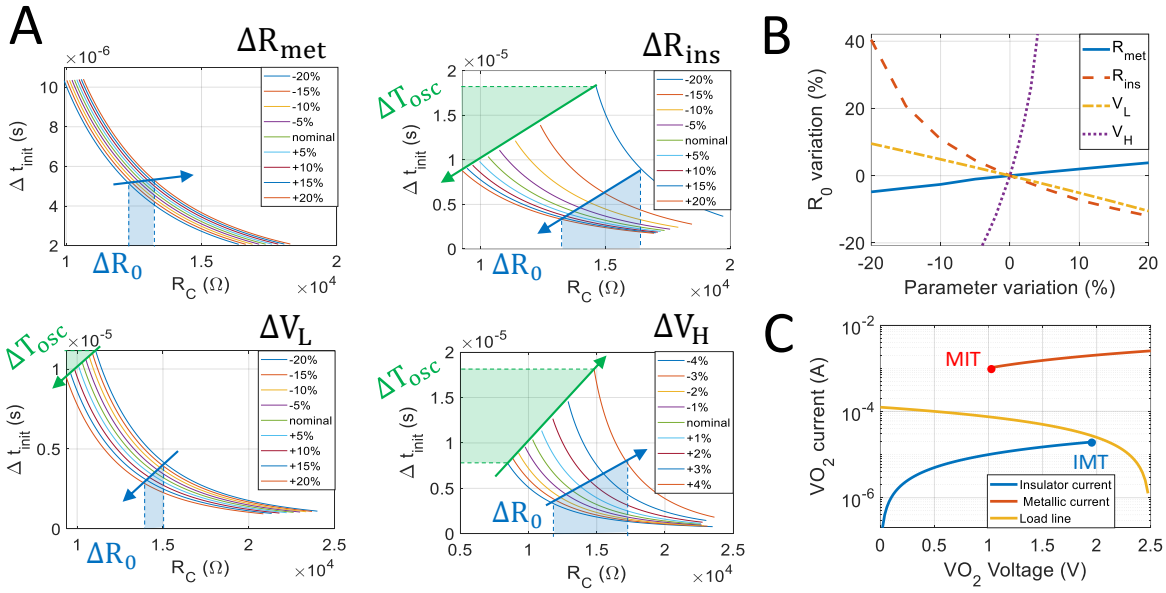


Figure 2.25: *(A) Phase transition curves $\zeta(R_C)$ when varying VO_2 parameters R_{met}, R_{ins}, V_L from -20% to +20%, and V_H from -4% to +4% for circuit parameters listed in Table* **??**. *Left hand side of the transition curve corresponds to inputs $(\Delta t_{init}, R_C)$ inducing $\Delta\phi_{out} = 0°$ whereas right hand side corresponds to $\Delta\phi_{out} = 180°$ phase region. V_H and R_{ins} variations correspond to IMT point variations and have the most detrimental impact on the transition function variations. The oscillating period almost doubles due to IMT variations. (B) Variations of the neutral synaptic resistance R_0 with respect to VO_2 parameters' variations. R_0 is very sensitive to V_H as -4% and +4% V_H variations induce -20% and +40% R_0 variations, respectively. (C) The ONN sensitivity to IMT point is mainly due to the load line $I_L = (V_{DD} - V)/R_S$ placed close to IMT point. Any IMT variation greatly impacts the oscillators' dynamics defined by $C_P\, dV/dt = I_L - I$.*

Fig.2.25A shows the set of phase transition curves obtained when varying R_{met}, R_{ins}, V_L, and V_H. Note that our current formalism assumes matched oscillators and hence, variations are applied to both coupled oscillators. For all curves, the maximum Δt_{init} value corresponds to an input delay of $T_{osc}/2$ and shows the oscillation period variation (highlighted in green in Fig.2.25A). Finally, we extract R_0 for each configuration (Fig.2.25B). We observe that variations on the IMT point (defined by R_{ins} and V_H) induce the largest T_{osc} and R_0 variations. With our biasing scheme set by R_S and V_{DD} (Table 2.1), the most sensitive VO_2 parameter is V_H as +4% and -4% V_H variations induces +40% and -20% R_0 variations, respectively. As the dynamic of the voltage V across the VO_2 device is given by $C_P\, dV/dt = I_L - I$, we believe this sensitivity is mainly due to the load line that passes very close to the IMT point on the VO_2 $I-V$ characteristic (Fig.2.25C). In this case near IMT, $I_L(V_H) - I(V_H)$ is small and the voltage "slows down" and is very sensitive to any IMT variation. When applying -4% up to +4% V_H variations, the oscillating period T_{osc} almost doubles (same remark with -20% and +20% R_{ins}

variations). Ideally, we would then place the load line at equal distances between MIT and IMT points ($I(V_L) - I_L(V_L) \approx I_L(V_H) - I(V_H)$) to homogenize the impact of VO_2 variations. However, we show in the next subsection that such biasing would prevent binary phase locking and that resistively coupled oscillators need a very asymmetric waveform to phase-lock to 180°.

2.9.3 Impact of the Waveform Shape

Oscillators' circuit parameters listed in Table 2.1 influence the oscillating frequency, amplitude, and waveform shape. The oscillating waveform shape has a major influence on ONN phase-locking capability and has been studied for PLL-based ONNs by [68]. Here, we study the impact of the oscillating waveform shape on the capability for pairs of oscillators to lock to the 180° phase state. We characterize the oscillating waveform shape with the ratio τ_d/τ_c, where τ_d and τ_c are the discharging and charging time constant, respectively (defined in Appendix A (eq.A.7), (eq.A.6)). Our transition function ζ links ONN phase-locking properties to the metric τ_d/τ_c, as ζ only depends on oscillators' internal parameters.

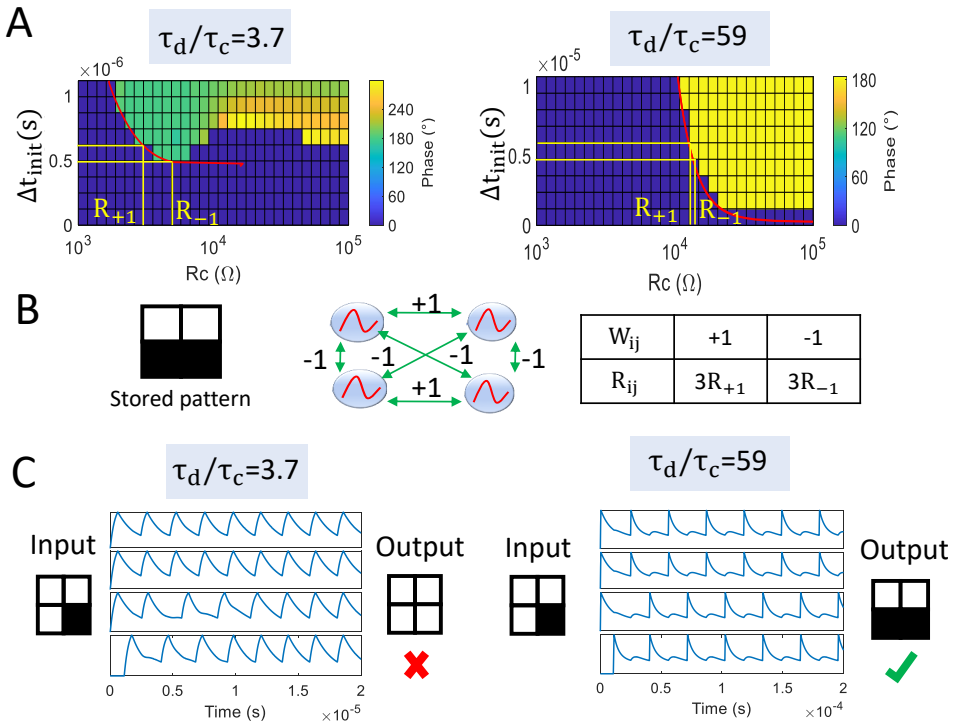


Figure 2.26: *(A) Phase plots showing $\Delta\phi_{out}$ with respect to Δt_{init} and coupling resistance R_C between two oscillators. For $\tau_d/\tau_c = 3.7$, 180° phase state is not reachable for low Δt_{init} values. In contrast for $\tau_d/\tau_c = 59$, 180° phase-locking can occur for any Δt_{init} value for large R_C. The red line is our analytical model $\zeta(R_C)$ and captures well the boundary between phase regions. (B) 4 coupled oscillators store a pattern composed of 2 white and 2 black pixels. Positive and negative weights are mapped to $3xR_{+1}$ and $3xR_{-1}$, respectively. (C) ONN inference for $\tau_d/\tau_c = 3.7$ and $\tau_d/\tau_c = 59$. The first configuration leads to a wrong in-phase relationship for all oscillators. In the latter case, ONN retrieves the correct stored pattern.*

We reproduce the previous simulation with two coupled oscillators to extract the output phase regions for different load resistances R_S that set τ_d/τ_c (Fig.2.26A). Note that we could also have varied VO_2 parameters such as R_{met}, but we instead consider the same device. For

$\tau_d/\tau_c = 3.7$ ($R_S = 3$ kΩ), we observe that the two oscillators cannot lock to $\Delta\phi_{out} = 180°$ for small Δt_{init} values. In other words, the phase state $\Delta\phi_{out} = 180°$ stored by a large R_C cannot be fully recovered. This can be an issue for some pairs of oscillators that need an out-of-phase relationship for any input delay. If $\tau_d/\tau_c = 59$ ($R_S = 20$ kΩ), the charging time is much smaller than the discharging time and the oscillating waveform becomes very asymmetrical. Interestingly, this configuration enlarges the 180° phase region and $\Delta\phi_{out} = 180°$ is reachable for any Δt_{init} value for large R_C. Our analytical model $\zeta(R_C)$ predicts the correct boundary between the two phase regions (red plain lines in Fig.2.26A).

We study a simple case where four VO_2 oscillators are coupled by resistances to store a single pattern (Fig.2.26B). Based on transition functions obtained for 2 coupled oscillators, we compute the coupling resistances R_{+1} and R_{-1} that correspond to synaptic coefficients +1 and -1, respectively. We set R_{+1} and R_{-1} around R_0 as $R_{+1} = \zeta^{-1}(T_{osc}/4 + T_{osc}/8)$ and $R_{-1} = \zeta^{-1}(T_{osc}/4 - T_{osc}/8)$, respectively. Then, we scale the coupling resistances as $3xR_{+1}$ and $3xR_{-1}$ as every oscillator is connected to 3 others (Fig.2.26B). It appears that the ONN with $\tau_d/\tau_c = 59$ retrieves the correct stored pattern whereas the ONN with $\tau_d/\tau_c = 3.7$ produces a wrong output (Fig.2.26C). In the latter case, we observe that all oscillators converge to an in-phase relationship. We believe this wrong behavior is mainly due to the small τ_d/τ_c value for which it is less likely ONN converges to $\Delta\phi_{out} = 180°$, as described by the transition function ζ.

Fig.2.27 shows results for the same experiment with τ_d/τ_c varied from 1.8 up to 59 (obtained for 2 kΩ $\leq R_S \leq$ 20 kΩ). We observe that $\tau_d/\tau_c > 20$ is required to retrieve the correct pattern. Interestingly, for $\tau_d/\tau_c \leq 20$, there are cases where the fourth oscillator locks to a phase state around 270°. 270° phase value is also obtained in the phase plot between two coupled oscillators, such as on the left-hand side of Fig.2.26A. This phenomenon would allow more than two phase values but is not captured by our current formalism. In contrast, setting $\tau_d/\tau_c = 59$ ensures binary 0° and 180° phase locking. However, this configuration lacks stability and is subject to device variations, as seen in the previous section.

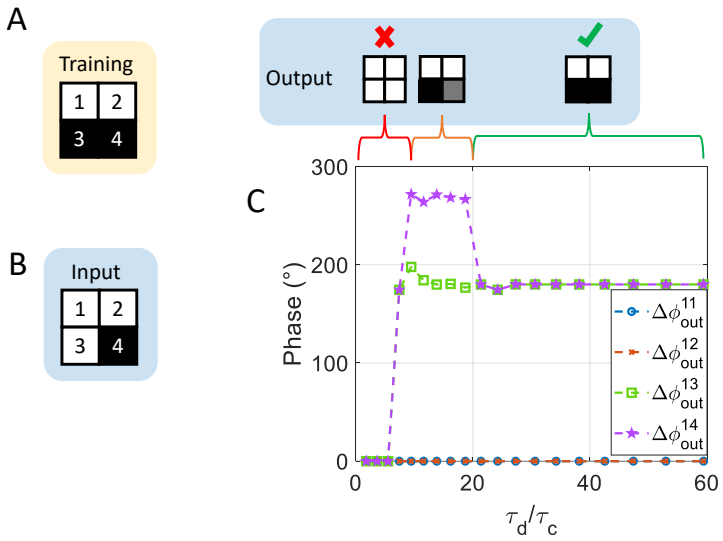
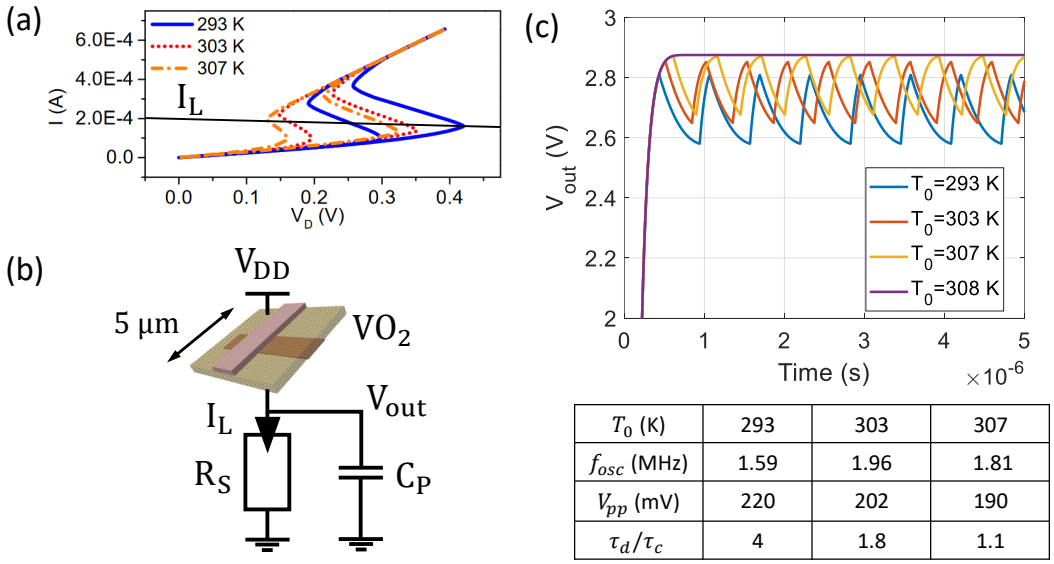


Figure 2.27: *(A) Training pattern stored by the ONN. (B) ONN input pattern. (C) Output phase with respect to τ_d/τ_c. For $\tau_d/\tau_c < 7$, all oscillators converge to a wrong in-phase relationship. For $7 \leq \tau_d/\tau_c < 20$, the fourth oscillator locks to a 270° phase state. A very asymmetrical waveform such that $\tau_d/\tau_c \geq 20$ leads to a correct binary phase-locking.*

2.9.4 Impact of the External Temperature

We saw in section 2.2 that the switching mechanism of VO_2 devices is temperature-dependent as it is driven by the temperature increase due to the Joule effect. However, such dependency is not captured by the state-of-the-art VO_2 model from Maffezzoni [158] that we have been using throughout this chapter. Instead, the VO_2 electrothermal TCAD model developed by Carapezzi et al. [150, 163] is based on the intrinsic relationship between device resistivity and temperature $\rho(T)$, as illustrated in Fig.2.2A, and has been calibrated with experimental data from IBM [150].

Fig.2.28a and b show the $I-V$ characteristic and oscillating circuit of a VO_2 device in crossbar geometry with 5-μm side and thickness of 80 nm. The contact width is 250 nm. More details can be found in [150]. TCAD quasi-static simulations show that the $I-V$ characteristic shifts to the left when increasing the external temperature. As the VO_2 device IMT point is set by a given temperature threshold, the device requires less power (i.e. current) to switch when increasing the external temperature. Of course, the amplitude of the $I-V$ shift depends on VO_2 material and device properties but for this device example, we observe a -25% V_{IMT} variation for a 14 K temperature increase. This results in modifications of the oscillation waveform as shown in Fig.2.28c) with the device biased with a series resistor $R_S = 15$ kΩ, $V_{DD} = 3$ V and $C_P = 150$ pF. The ratio τ_d/τ_c decreases with T_0 which prevents binary phase-locking, as seen earlier in section 2.9.3. Moreover, if the temperature increases too much, the load line no longer lies in the NDR region and the oscillations die, as shown in Fig.2.28b at $T_0 = 308$ K.



T_0 (K)	293	303	307
f_{osc} (MHz)	1.59	1.96	1.81
V_{pp} (mV)	220	202	190
τ_d/τ_c	4	1.8	1.1

Figure 2.28: *a) $I-V$ plot of a 5-μm VO_2 device in crossbar geometry [150] for various external temperature points T_0. b) VO_2 oscillator circuit. c) Resulting oscillations. V_{pp} is the peak-to-peak oscillation amplitude. τ_d and τ_c are the oscillation charging and discharging times.*

2.10 Discussions

Possible Design Improvement: Adding Synaptic Capacitors. In literature, authors have reported that AC coupling using a coupling capacitor favors a 180° phase shift between two coupled oscillators [149, 191]. Thus, adding a synaptic capacitor in parallel with the coupling resistor could increase the 180°-phase-locking capability while enabling a more stable oscillator biasing in the middle of the NDR region. To test our hypothesis in experiments, we have moved the biasing line closer to the center of the NDR region, resulting in phase plots having a reduced 180° phase region, similar to the phase plot in Fig.2.26A. In practice, we decreased the charge resistance using a potentiometer on our VO_2 PCB emulator (Fig.2.29A) to reduce $\tau_d/\tau_c \leq 20$ and increase the oscillator stability.

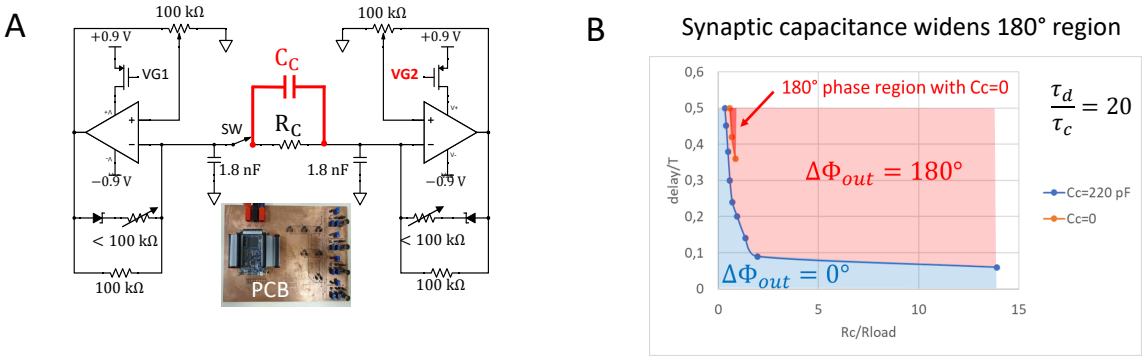


Figure 2.29: *A) Schematic of the experimental circuit on PCB to test the impact of synaptic capacitors. B) Experimental results with or without $C_C = 220$ pF.*

Fig.2.29B presents two experimental phase plots obtained by varying the synaptic resistance R_C with or without a fixed synaptic capacitance $C_C = 220$ pF = $12\%C_L$. It is clear that the synaptic capacitor widens the 180° phase region and enables binary phase locking. Adding C_C increases the exploitable R_C range by a factor of 10 at least. However, the mapping formalism that we have developed in Section 2.5 does not consider synaptic capacitors yet, and would constitute a future work.

2.11 Conclusion

Motivated by the architecture simplicity, this chapter focused on designing ONNs using VO_2 oscillators coupled by resistive devices. As fully-connected ONNs behave like HNNs, we first proposed a formalism to map Hebbian synaptic coefficients to ONN coupling resistances, thus enabling ONN design and programmation for the pattern recognition application. Then, we studied how the ONN performances scale with the network size N. It first appeared that for good accuracy, the ONN coupling resistances should linearly increase with N, thus keeping a constant synaptic current at the output of each neuron. This further induces a decrease in the power of a single synapse as $O(1/N)$, limiting the total ONN power to scale only linearly as $N^2O(1/N) = O(N)$. Furthermore, circuit simulations suggested that the ONN computation time remains quasi-constant with N, highlighting the high ONN parallelism and its advantageous energy scaling.

Moreover, we studied how to minimize the oscillator energy footprint at the VO_2 device level. TCAD simulations of VO_2 devices in crossbar configuration with fixed thickness suggested that the device threshold voltages scale as $O(CB)$, with CB the crossbar width. Hence, the oscillator power scales as $O(CB^2)$ and the VO_2 oscillating neuron is low-power when scaled down to low dimensions. When benchmarked with state-of-the-art digital neural accelerators and neuromorphic chips at the neuron level, a VO_2 oscillator with CB=500 nm becomes more energy efficient (<20 fJ/oscillation) for oscillating frequencies above 100 MHz, i.e. for capacitive loads < 500 fF.

Finally, we discussed the limitations of the architecture and found that a large-scale implementation remains challenging without architectural modifications. The two main limitations are poor oscillator stability and limited synaptic range which would both lead to an unreliable ONN operation with real devices. However, there is room for improvement such as **1)** relaxing the oscillator biasing for more robust oscillations while using synaptic capacitors and **2)** controlling the ONN dynamics with sub-harmonic injection techniques. In the next chapter, we propose a novel ONN architecture to relax these constraints and potentially facilitate ONN design at a large scale.

CHAPTER 3

SKONN: A SCALABLE MIXED-SIGNAL ONN ARCHITECTURE

DESPITE the promising results on ONN performances and scaling discussed in Chapter 2, the study of ONNs based on resistively coupled VO_2 oscillators has underlined important design and operational challenges such as:

1. A limited oscillation robustness.
2. A non-trivial synaptic mapping to the hardware with a very narrow synaptic range.
3. A non-negligible temperature dependency of the oscillating signals.

Furthermore, the previous architecture was intrinsically *recurrent* as oscillators' inputs and outputs were merged on a single node, thus preventing any kind of forward propagation of information. In this chapter, I propose a new ONN architecture to address the previous limitations with a limited increase in complexity. For this design, I wished to keep the computation in the analog domain, according to the *let physics compute* paradigm; while taking advantage of the digital world that has so much to offer in terms of programmability, modularity, and robustness to noise. Hence, the proposed ONN is a **mixed-signal architecture** with relaxed triangular oscillations to address the first challenge, and a digital propagation of information to overcome the second one. Finally, the temperature dependency is mitigated by designing the ONN with a CMOS technology.

Although the architecture development first started at transistor and circuit levels, higher levels of ONN abstraction such as phase dynamics and gradient descent naturally emerged from the composition of simple building blocks. Interestingly, we will see throughout this chapter how such high levels of ONN abstractions are linked to energy-based models and to the field of combinatorial optimization, a topic that will be the focus of the fourth chapter.

3.1 Introduction

3.1.1 Previous work

Recently, a new interest in ONN has risen thanks to the emergence of novel oscillating devices that enable the fabrication of efficient ONNs [67]. Such as spin-torque and spin Hall devices [87, 93], micro-electromechanical systems [88, 192], and transition metal oxide devices are all candidates for implementing ONNs using their oscillatory behavior and synchronization properties [85, 86, 96, 123, 150]. As seen in the previous Chapter, beyond-CMOS devices are promising as they generally allow a compact oscillator design using a single device that could be scaled down to the nanoscale. Nevertheless, CMOS-based ONNs benefit from the mature CMOS technology which enables rapid ONN development and facilitates its co-integration with conventional digital circuits [76, 89, 90, 138]. Throughout this thesis, we focus on ONNs that compute in the phase domain, i.e. with neurons that oscillate at the same frequency. However, note that it is also possible to compute with various frequencies [94, 95]. Regardless of the technology, we identify three important criteria for designing a competitive ONN that computes in phase domain. It should have:

1. Homogeneous oscillating frequencies
2. Compact and linearly programmable signed synapses
3. A scalable architecture

Even with the mature CMOS technology, achieving perfect matching between hundreds of oscillators is unfeasible for small-scale oscillators due to device-to-device variations. Hopefully, some techniques can overcome frequency mismatches such as calibration or sub-harmonic injection locking (SHIL). SHIL consists in driving the oscillators with a harmonic signal that can lock to a Fourier harmonic of the oscillating signal [193]. In case of large frequency mismatches, the injection of a strong SHIL signal ensures phase locking among the oscillators [194]. The second criterion promotes synapses that are compact, programmable with signed weights, and have a value proportional to their conceptual weight. Some architectures, such as the one studied in Chapter 2, can lead to a non-linear mapping between the conceptual weights and their hardware implementation, or even be unknown due to the high complexity of the dynamics. Finally, the ONN architecture must be scalable to compete with conventional computing and solve large-scale problems involving thousands or millions of synapses. For this reason, we believe that the ONN architecture should be modular, i.e. to support the interconnection of smaller sub-ONNs to build a larger system and avoid the implementation of a fully-connected network.

Table 3.1 presents the state-of-the-art ONN architectures and their features. We only consider ONN computing in the phase domain and based on electrical oscillators. For solving combinatorial optimization problems (COPs), a general approach is to map the input graph to the ONN where vertices are oscillators, and edges are synapses. Some architectures such as [86, 139] are dedicated to finding the maximum cut of a graph with weights of the same sign, as the synapses only implement negative weights. The main drawback is that both coupling capacitors and resistors are required to program negative and positive weights, respectively. Other architectures using differential LC oscillators enable signed weights using resistors only [162, 195] but are not scalable on chip due to the bulky LC tanks and resistors. Digital ONNs are promising as they are scalable and modular, as demonstrated by Moy et al. with their 1968

ring oscillators chip [76]. A recent promising fully-analog architecture for solving COPs has also been proposed by Graber et al. [90] that consists of 400 oscillators coupled with nearest neighbors.

	Goto [79]	Jackson et al. [89]	Wang et al. [162].	Chou et al. [195].	Mallick et al. [139].	Dutta et al. [86].	Moy et al. [76].	Graber et al. [90].	This work
Size	9600	100	240	4	600	8	1968	400	16
Oscillator	Analog LC	Digital	Analog LC	Analog LC	Analog relaxation	Analog relaxation (PTNO)	Ring Oscillator	Analog differential	Analog relaxation
SHIL or Calibration	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Coupling	Transformers	Resistors	Resistors	Resistors	Capacitors	Capacitors Resistors	Transmission gates	Current sources with DACs	Capacitors
Signed weights	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes
Weight precision	1 bit	5 bits	8 bits	5 bits	1 bit	-	5 levels	6 bits	5 bits
Modular	Yes	-	Yes	-	Yes	-	Yes	Yes	Yes
Feedforward	Yes	Yes	-	Yes	No	No	No	Yes	Yes
Initial phase control	Yes	Yes	-	-	-	-	-	No	Yes
Application	Digital logic	Pattern recognition	COP	COP	COP	COP	COP	COP	COP Image processing

Table 3.1: *State-of-the-art ONN Architectures*

3.1.2 Contributions

In this work, we introduce a new mixed-signal ONN architecture, named *Saturated Kuramoto oscillatory neural network* (SKONN) that leverages both analog and digital domains to satisfy the three ONN design criteria. SKONN takes inspiration from the state-of-the-art analog ONN architectures for which the dynamics evolve *naturally* in continuous time and can easily be described by phase models like Kuramoto's or Izhikevich's [146], thus facilitating the exploration of potential applications. SKONN's main novelty consists of setting the computation and propagation in the analog and digital domains, respectively. Fig.3.1 illustrates SKONN with 4 fully-coupled neurons.

Digital propagation has several advantages such as greater noise immunity, a higher fan-out, and smoother interfacing with other digital circuits. Moreover, the separation between the computation and propagation induces a natural implementation of feed-forward synapses that have never been implemented in literature, to the best of our knowledge. The development of the proposed ONN architecture led to the following contributions:

- SKONN circuit and architecture description.
- Derivation of SKONN's phase dynamics and energy landscape.
- Design of a first 3x3 SKONN proof-of-concept on PCB.
- Integrated circuit (IC) design of a 4x4 SKONN demonstrator in 65 nm CMOS technology.

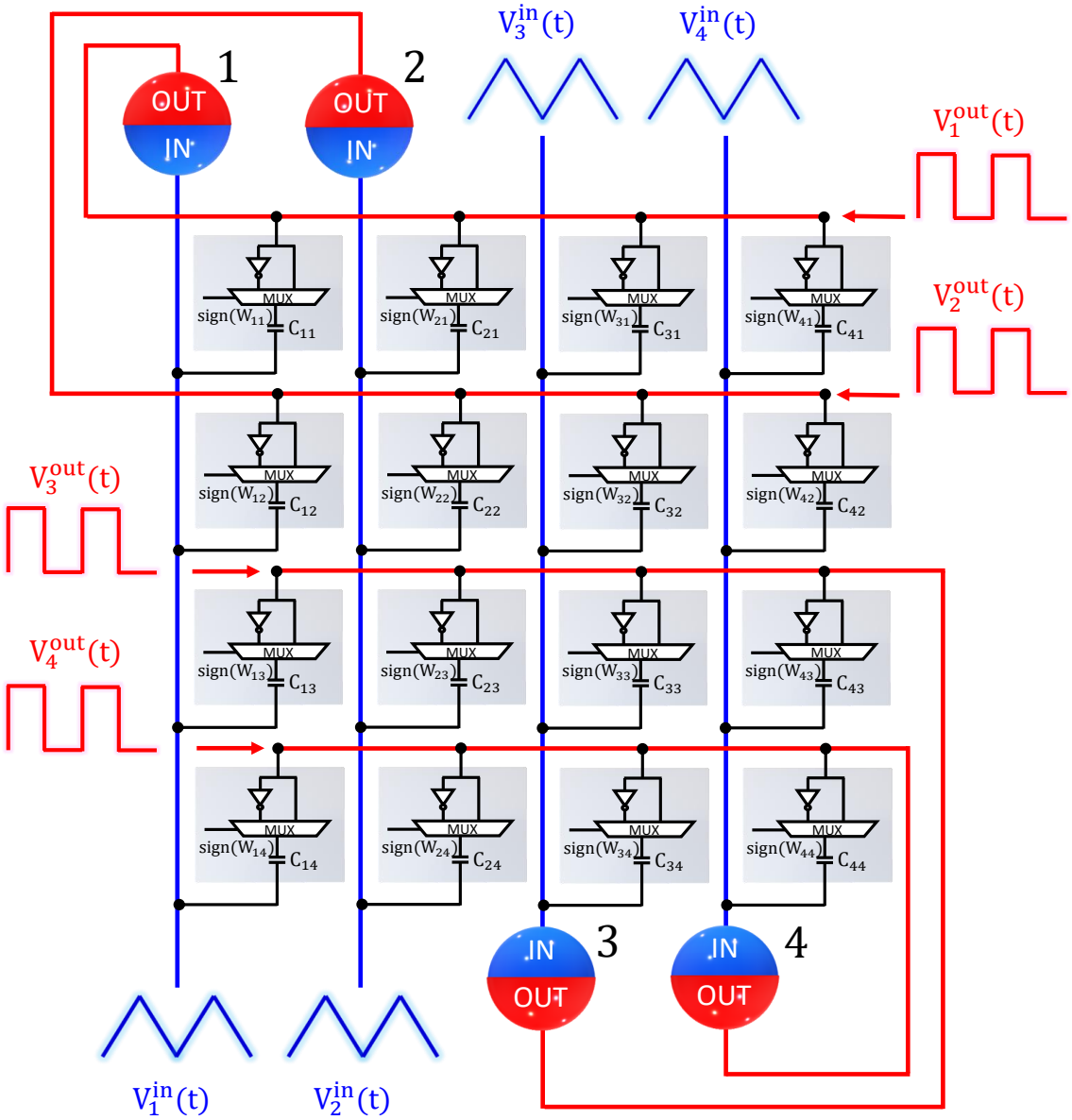


Figure 3.1: *Illustration of a SKONN architecture composed of 4 oscillators and 16 synapses. Neuronal input and output lines are laid out vertically and horizontally, respectively. For neuron i, the input $V_i^{in}(t)$ and output $V_i^{out}(t)$ are synchronized such that $V_i^{out}(t)$ commands the generation of $V_i^{in}(t)$. A synapse S_{ij} consists of a capacitor C_{ij} that converts the digital signal $V_j^{out}(t)$ in current spikes sent to the input node i. The multiplexer sets the weight sign by selecting $V_j^{out}(t)$ or $\overline{V_j^{out}(t)} = V_j^{out}(t - T/2)$. The triangular analog input oscillation $V_i^{in}(t)$ receives the synaptic current spikes, i.e. the charges Q_{ij}, that shift the phase ϕ_i.*

Part of this chapter's findings led to the following peer-reviewed journal article:

C. Delacour, S. Carapezzi, G. Boschetto, M. Abernot, T. Gil, N. Azemard and A. Todri-Sanial, "A Mixed-Signal Oscillatory Neural Network for Scalable Analog Computations in Phase Domain", in *Neuromorphic Computing and Engineering*, vol.3, 2023.

3.2 SKONN Architecture Description

3.2.1 A Mixed-Signal Oscillating Neuron

A SKONN neuron consists of a relaxation oscillator producing analog and digital oscillations with period T at its input and output nodes, respectively. Fig.3.2a shows the block diagram of the oscillating neuron. It consists of a hysteresis circuit that commands a shaper block to charge and discharge a capacitor C_L with constant current I_{bias}. The voltage across the capacitor V_i^{in} is fed back to the hysteresis comparator that switches between V_{DD} and 0 when V_i^{in} reaches the thresholds V_H and V_L, thus producing oscillations. V_i^{out} holds the phase state in the digital domain, whereas V_i^{in} is the analog evolution of the oscillation. Note that the input impedance of the oscillator is purely capacitive in the ideal case so that any charge sent to the input node causes an instantaneous phase shift. The analog waveform V_i^{in} supports the computation and is separated from the digital propagation V_i^{out}, enabling a feed-forward propagation of the phase information.

Fig.3.2c shows an example of feed-forward propagation between two oscillators. The computation occurs in the analog domain at the input node of neuron i that gathers the output signals from neuron j. The oscillator output signal is a square digital-like signal that carries the oscillator state and evolves until the phase dynamics settle to a fixed point. Choosing a triangular waveform at the analog input leads to simple yet rich phase dynamics that are similar to the Kuramoto model, which is known to have interesting computational properties [146]. Moreover, it skips the use of bulky LC tanks needed for producing sinusoidal oscillations. The neuron voltage dynamics are expressed in Appendix B.1 for completeness, although this work focuses on phase dynamics that are more suitable to study phase-based ONNs. The comparison between SKONN circuit dynamics and phase dynamics is presented in Appendix B.4.

3.2.2 Synaptic Design and Weight Sign

A SKONN synapse S_{ij} consists of a capacitor C_{ij} that transmits current pulses, i.e. charges Q_{ij}, from the output of oscillator j to the input of oscillator i. C_{ij} can easily be programmed using a capacitor bank and a register, as shown in Fig.3.2b. Instead of propagating the sensitive analog signal, SKONN only transmits the oscillator phase information in a robust manner. The digital output voltage V_j^{out} is applied to C_{ij} that creates current spikes holding the phase information ϕ_j. The synaptic spike train can be expressed as follows:

$$I_{ij} = C_{ij}\left(\frac{dV_j^{out}}{dt} - \frac{dV_i^{in}}{dt}\right) \tag{3.1}$$

The synaptic capacitor can be thought of as a digital-to-analog phase converter. The synaptic weight consists of the capacitance value C_{ij} that linearly modulates the charge sent to the oscillating input node i as $Q_{ij} = C_{ij}V_{DD}$, thus inducing phase shifts in the oscillation i as shown in Fig.3.2c. To implement a negative weight, the complementary of V_j^{out} defined as $\overline{V_j^{out}(t)} = V_j^{out}(t - T/2)$ is selected using a multiplexer and applied to C_{ij}. Compared to resistors, synaptic capacitors have several advantages for upscaling the ONN:

1. ONN computation models are generally based on the weak coupling assumption [69, 146] and necessitate weak synaptic signals. This means the ONN needs either large coupling resistors or small capacitors, the latter being much more scalable in a chip.

2. For a limited neuron output strength, the only way of increasing the synaptic fan-out is to reduce the synaptic current, which again would lead to bulky resistors or smaller capacitors in the case of SKONN.

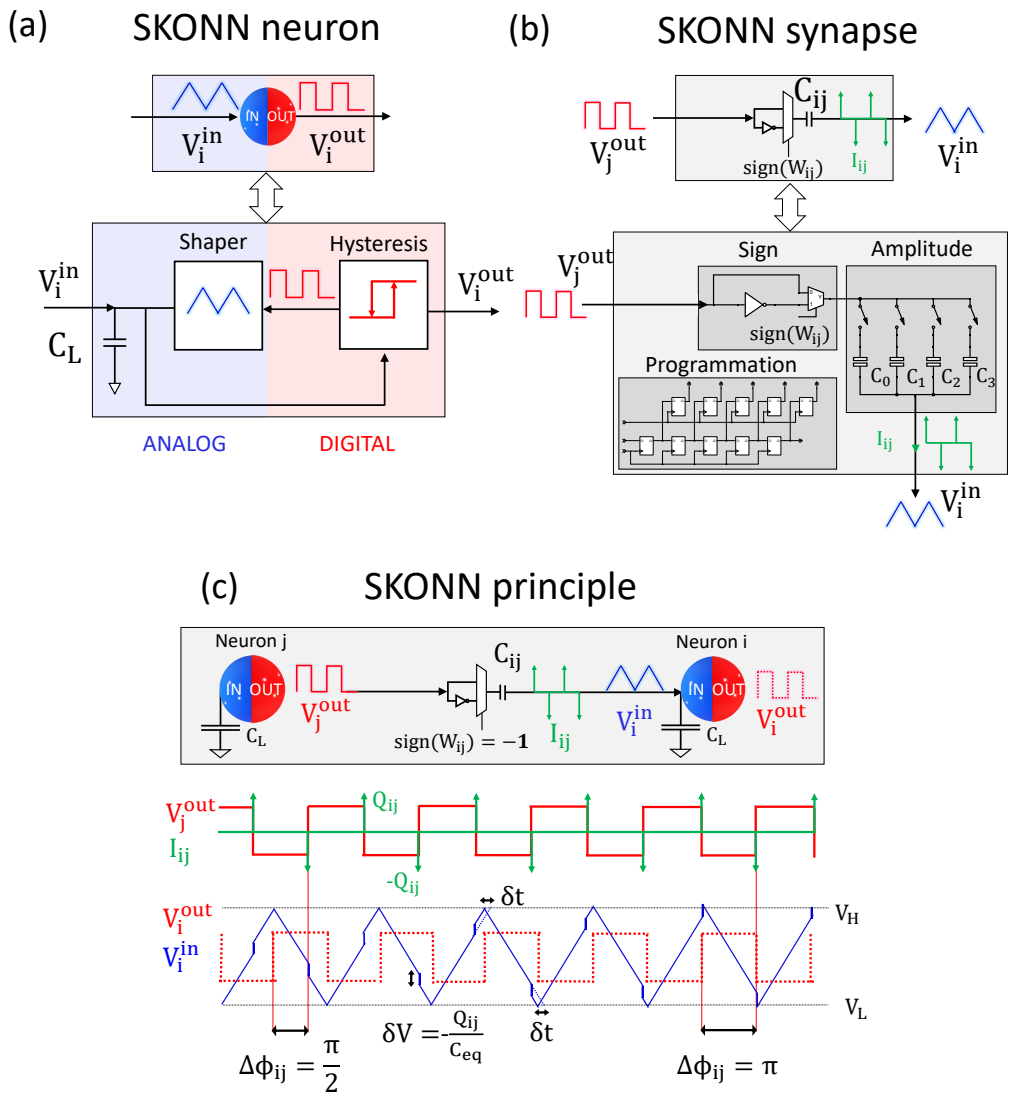


Figure 3.2: *a) A SKONN neuron is a relaxation oscillator composed of two blocks: the hysteresis circuit that holds the neuron state and drives the shaper stage. The latter produces an analog triangular waveform at the input whereas the hysteresis block outputs a digital waveform. b) A SKONN synapse consists of a capacitor bank setting the weight amplitude. A multiplexer selects V_j^{out} or $\overline{V_j^{out}}$ to set the weight sign. c) SKONN principle of computation illustrated with a negative weight. The multiplexer selects $\overline{V_j^{out}}$ that is fed into C_{ij}, creating current spikes I_{ij} aligned with the rising and falling edges of $\overline{V_j^{out}}$. The injected charges $\pm Q_{ij}$ to V_i^{in} induce voltage jumps $\pm\delta V$ that cause time shifts δt. After a few cycles, the two oscillators lock to $\Delta\phi_{ij} = \pi$.*

3.3 SKONN Phase Dynamics

3.3.1 Two Coupled Oscillators

SKONN computing mechanism is illustrated in Fig.3.2c with the case of a neuron j feeding its phase to another neuron i in a feed-forward manner with a negative weight.

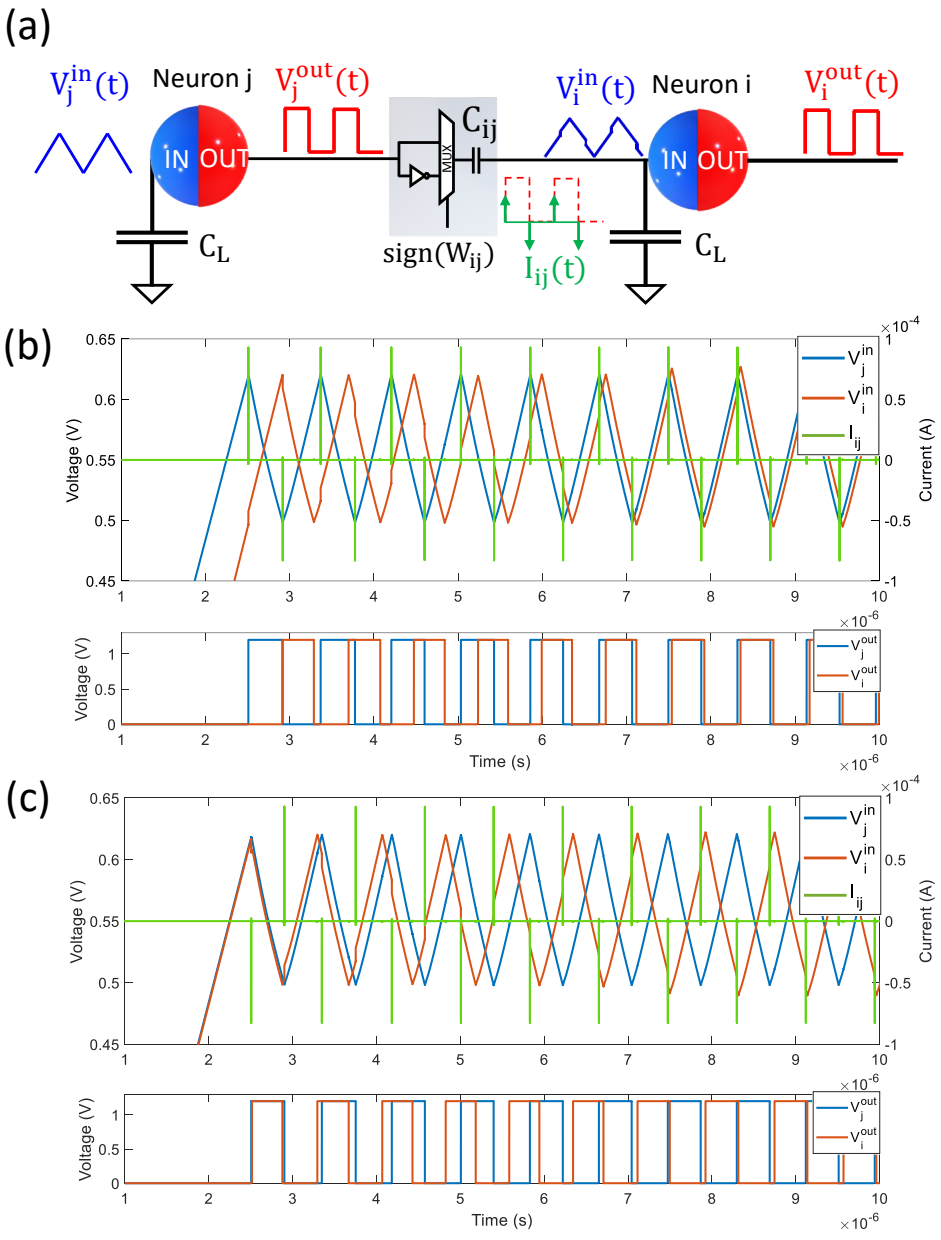


Figure 3.3: *(a) Illustration of a neuron that drives a second neuron in a feed-forward manner. The multiplexer sets the weight sign and the capacitor C_{ij} converts the digital signal $V_j^{out}(t)$ into current spikes $I_{ij}(t)$ that induce phase shifts of the input $V_i^{in}(t)$. b) Transistor-level transient simulation with a positive weight $W_{ij} > 0$. The second phase catches up with the first one after a few cycles such that $\Delta\phi_{ij} \approx 0°$. (c) Simulated dynamics with a negative weight $W_{ij} < 0$. The second phase is pushed such that $\Delta\phi_{ij} \approx 180°$ after a few cycles.*

This means that $\overline{V_j^{out}}$ is selected by the multiplexer and applied to C_{ij}, thus creating current spikes $+Q_{ij}\delta(t)$ and $-Q_{ij}\delta(t-T/2)$ that are aligned with the rising and falling edges of $\overline{V_j^{out}}$, respectively. Each injected charge $\pm Q_{ij}$ induces a voltage jump $\delta V = \pm Q_{ij}/C_{eq}$ at the input node, with $C_{eq} = C_L + C_{ij}$. As V_i^{in} is a triangular waveform, δV provokes a time shift $\delta t = \pm C_{eq}\delta V / I_{bias}$, where $\pm$ indicates here the sign of V_i^{in}'s slope. Knowing the period of the triangular oscillation $T = 2C_{eq}\Delta V / I_{bias}$ where ΔV is V_i^{in}'s peak-to-peak amplitude, we can then express the phase shift related to a single current spike:

$$\begin{aligned}\delta\phi &= 2\pi\frac{\delta t}{T}\\ &= \pi\frac{\pm Q_{ij}}{C_{eq}\Delta V}\\ &= \pi\frac{\pm C_{ij}}{C_{eq}}\frac{V_{DD}}{\Delta V}\\ &\approx \pi\frac{\pm C_{ij}}{C_L}\frac{V_{DD}}{\Delta V} \text{ if } C_L >> C_{ij}\end{aligned} \tag{3.2}$$

SKONN's unique feature consists of this simple relation (eq.3.2) between coupling capacitors and phase shift, thus enabling well-controlled phase dynamics and a precise weight mapping to the coupling capacitor C_{ij}. As we will see later, the quantity $\beta_0 = |\delta\phi/Q_{ij}|$ provides the neuron phase sensitivity with respect to the charge perturbation. It is linked to the phase perturbation vector (PPV) of the oscillator which is key for deriving SKONN's phase dynamics [194]. SKONN's PPV is defined and derived in Appendix B.2.

Two coupled oscillators converge either in or out of phase, depending on the synaptic sign. To show this property, we use SKONN's phase dynamics that are derived in Appendix B.3 using the PPV formalism [194]. Under the weak coupling assumption ($C_{ij} << C_L$), the phase dynamics of oscillator i can be expressed as follows:

$$\frac{d}{dt}\phi_i = 2\beta_0\frac{Q_{ij}}{T}\,\text{square}(\phi_i - \phi_j) \tag{3.3}$$

with the 2π-periodic function:

$$\text{square}(\theta) = \begin{cases}-1, & \text{if } 0 < \theta < \pi\\ +1, & \text{if } \pi < \theta < 2\pi\end{cases} \tag{3.4}$$

The phase fixed points can be derived from (eq.3.3) and are expressed in the next proposition.

Proposition 3.1. *If the injected charge $Q_{ij} \neq 0$ then the two SKONN oscillators admit a unique stable fixed-point $\Delta\phi^* = (\phi_i - \phi_j)^*$ such that*

$$\Delta\phi^* = \begin{cases}0, & \text{if } Q_{ij} > 0\\ \pi, & \text{if } Q_{ij} < 0\end{cases} \tag{3.5}$$

The proof is shown in Appendix B.3. In other words, propagating a spike train defined as (eq.3.1) induces an in-phase or out-of-phase locking, depending on the polarity of Q_{ij}. Each current spike produces a local phase shift to the analog input oscillation, resulting in an average phase shift $\Delta\phi = \pm 2\beta_0 Q_{ij}$ after each cycle (eq.3.3). Fig.3.3b shows a transistor-level simulation

of the positive weight case. I_{ij} perturbs V_i^{in} until the oscillators converge in phase. Similarly, Fig.3.3c shows the same configuration with a negative weight and the oscillators are out-of-phase.

Note that the phases measured from the rising edges of V_i^{out} and V_j^{out} are slightly shifted from the theoretical fixed points (eq.3.5). This is mainly due to the limited bandwidth of the hysteresis block which does not switch instantaneously when reaching its thresholds. This non-ideality can be compensated and is further discussed in Appendix B.6. Interestingly, this phase shift disappears with symmetric synapses as both oscillators are equally delayed (see Fig.3.6c).

3.3.2 N Coupled Oscillators

The phase dynamics of N sinusoidal coupled oscillators are often expressed using the Kuramoto model [68, 69, 146]:

$$\frac{d}{dt}\phi_i = -\omega_0 \sum_{j=1}^{N} K_{ij} \sin\left(\phi_i - \phi_j\right) \tag{3.6}$$

where ω_0 is the frequency in rad/s and K_{ij} the coupling coefficients. Similarly, we derive SKONN's phase dynamics for N oscillators as follows:

$$\frac{d}{dt}\phi_i = \omega_0 \frac{V_{DD}}{\Delta V} \sum_{j=1}^{N} \frac{C_{ij}}{C_L} \text{square}\left(\phi_i - \phi_j\right) \tag{3.7}$$

where we replaced β_0 and Q_{ij} from (eq.3.3) by their expressions $\beta_0 = \pi/(\Delta V C_L)$ and $Q_{ij} = C_{ij} V_{DD}$. The derivation is detailed in Appendix B.3. V_{DD} is the digital voltage swing, ΔV is the peak-to-peak triangular voltage amplitude at the input, C_{ij} is the synaptic capacitance value, and C_L is the neuron input capacitance.

SKONN's phase dynamics are very similar to the Kuramoto model (eq.3.6) except for its sinusoidal function replaced by a *saturated* square function in this work. It induces a binarization behavior that is useful for solving some optimization problems as shown next. Note that similar dynamics have already been explored in simulation by Wang et. al. in their work about oscillatory Ising machines (OIM) [69]. The authors studied the case where the sinusoidal term $\sin(\Delta\phi)$ from Kuramoto (eq.3.6) is replaced by $\tanh(\alpha \sin\Delta\phi)$ with $\alpha = 10$. As SKONN's square interaction can be thought as $\text{square}(\Delta\phi) \approx -\tanh(\alpha \sin\Delta\phi)$ for $\alpha >> 1$, we expect SKONN to have good performances when solving NP-hard combinatorial optimization problems.

3.4 SKONN Stability and Energy Landscape

SKONN stability can be proved by applying the convergence theorem for ONNs derived by Hoppensteadt and Izhikevich [121]. With an odd coupling function (eq.3.4) and symmetric coupling $Q_{ij} = Q_{ji}$, the theorem ensures that the phase differences converge to a stable fixed point. The proof consists of finding a Lyapunov function for the dynamics (eq.3.7) that is bounded below and minimized through time. A candidate for the SKONN Lyapunov function

is:

$$E = \frac{\beta_0}{T} \sum_{i,j}^{N} Q_{ij} \,\text{triangle}\big(\phi_i - \phi_j\big) \tag{3.8}$$

with:

$$\text{triangle}(\theta) = \begin{cases} \theta - \pi/2, & \text{if } 0 \leq \theta \leq \pi \\ 3\pi/2 - \theta, & \text{if } \pi \leq \theta \leq 2\pi \end{cases} \tag{3.9}$$

Under the assumption that $Q_{ij} = Q_{ji}$, one can check that:

$$\begin{aligned} \frac{\partial E}{\partial \phi_k} &= \frac{\beta_0}{T} \Big(- \sum_{j=1}^{N} Q_{kj} \,\text{square}\big(\phi_k - \phi_j\big) \\ &\qquad + \sum_{i=1}^{N} Q_{ik} \,\text{square}\big(\phi_i - \phi_k\big) \Big) \\ &= - \frac{d\phi_k}{dt} \end{aligned} \tag{3.10}$$

Thus, SKONN minimizes E over time:

$$\begin{aligned} \frac{dE}{dt} &= \sum_{k=1}^{N} \frac{\partial E}{\partial \phi_k} \frac{d\phi_k}{dt} \\ &= - \sum_{k=1}^{N} \Big(\frac{d\phi_k}{dt} \Big)^2 \leq 0 \end{aligned} \tag{3.11}$$

By considering $\phi_1 = 0$ as the reference oscillator, SKONN's and Kuramoto's energy landscapes are represented for the three-oscillator case with negative weights in Fig.3.4. In this simulation, the phases are initialized with $\phi(t=0) = (0,5,2)^\circ$, and the dynamics (eq.3.6) and (eq.3.7) are numerically solved using Matlab. Although Kuramoto's and SKONN's dynamics seem similar, i.e. the phases are "pushing" each other, it appears from Fig.3.4b that SKONN's trajectory converges faster to a different phase fixed point $\phi_S(t^S_{end}) = (0,90,180)^\circ$, compared to $\phi_K(t^K_{end}) = (0,120,240)^\circ$ for Kuramoto. Visualizing the trajectories on the energy landscapes (Fig.3.4) confirms the stability of both ONNs, although reaching different local minima. Note how SKONN's energy landscape is sharper than Kuramoto's. Surprisingly, in this example, Kuramoto's minima are translated as flat regions or *plateaus* in SKONN's energy landscape. The shape of the energy landscape will be further discussed in Chapter 4 when solving optimization problems.

For more than 3 oscillators, the only way to plot the energy landscapes in 3D is to take "snapshots" along some phase variables. Fig.3.5a shows an ONN example with four oscillators and recurrent negative weights. Here, the phase fixed points are the same as $\phi_S(t^S_{end}) \approx \phi_K(t^K_{end}) = (0,180,360,180)^\circ$, but we observe different convergence times $t^K_{end} >> t^S_{end}$ as Kuramoto's trajectory seems to **1)** be slowed down by a saddle point and **2)** is slower to reach the local minimum (especially ϕ_2 and ϕ_4, see Fig.3.5b). Fig.3.5c shows a snapshot of the energy landscapes for $\phi_4 = 110^\circ$ when the Kuramoto ONN slows down. For this example of initialization, SKONN's trajectory follows a much steeper route than Kuramoto's. We will see in Chapter 4 that SKONN is indeed statistically faster than Kuramoto to reach energy minima.

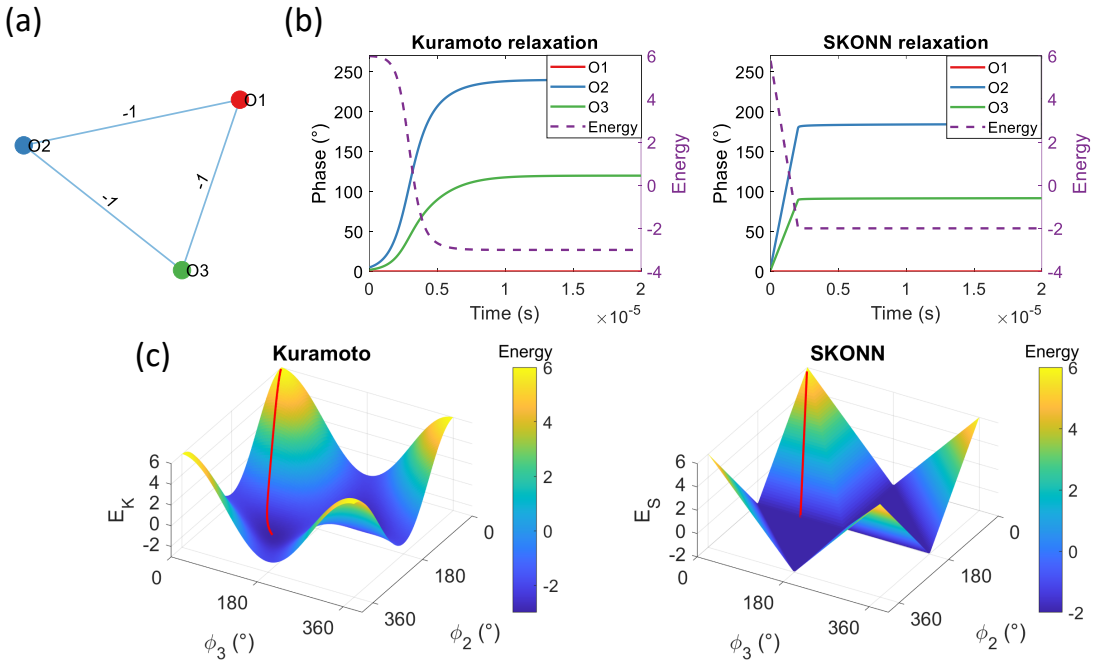


Figure 3.4: *a) Three oscillators are recurrently coupled by a negative weight. b) Example of Kuramoto and SKONN dynamics for an initial phase state $\phi(t=0) = (0,5,2)°$. c) Corresponding energy landscapes minimized through time. The red plain lines show both trajectories.*

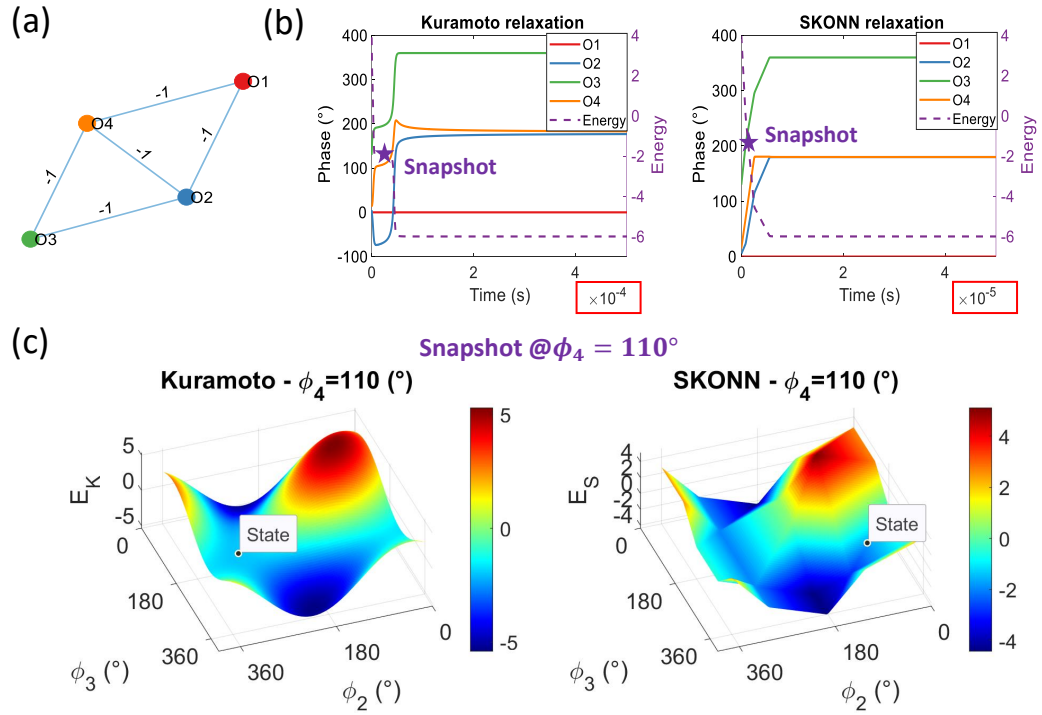


Figure 3.5: *a) 4-neuron ONN with recurrent synapses and negative weights. b) Example of dynamics with $\phi(t=0) = (0,5,130,15)°$. The star marker indicates when a snapshot of the energy landscape is captured at $\phi_4 = 110°$. Notice that SKONN's settling time is 10x faster than Kuramoto's. c) Energy landscapes at the snapshot $\phi_4 = 110°$. Kuramoto and SKONN trajectories take different paths.*

3.5 9-Neuron SKONN Proof-of-Concept

3.5.1 SKONN Design with Discrete Components

We designed a 3x3 SKONN on PCB with fully connected capability and 81 synapses (Fig.3.6a) as a proof of concept for the SKONN architecture. Due to area constraints, we only implemented negative weights that we program by placing discrete capacitors C_{ij} manually. Fig.3.6b shows the oscillating neuron based on a Schmitt trigger (U1) with feedback resistor R_3 that charges/discharges a load capacitor C_L, producing a triangular-like waveform with 720 mVpp amplitude. The neuron output is a square-like waveform oscillating between $V_{DD} = +0.9$V and $V_{SS} = -0.9$V. Using an FPGA, we set the initial phase state by delaying the oscillator's starting time (via transistor Q1). The FPGA measures the neurons' output voltages and allows phase post-processing with a maximum precision of $\varepsilon = 360° \, f_0/f_{FPGA}$. In our experiments we set $f_0 = 4$kHz, f_{FPGA} =50 MHz and $\varepsilon \approx 0.03°$.

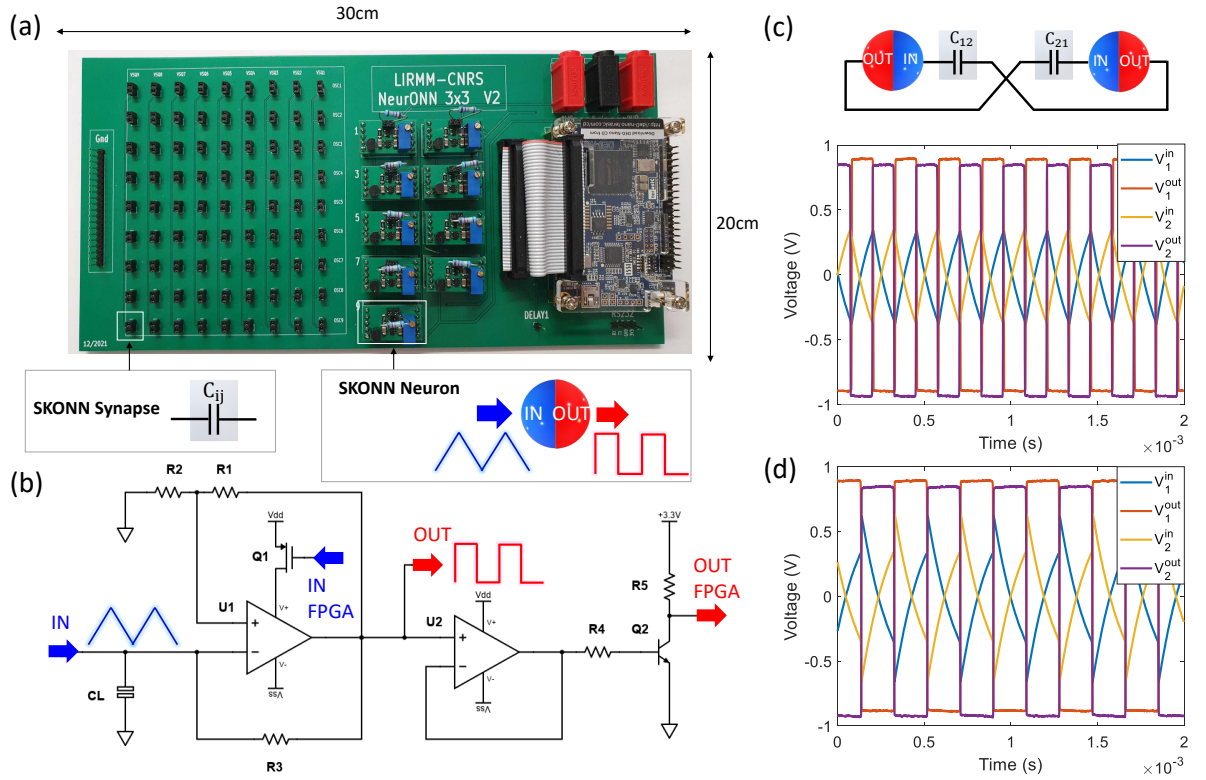


Figure 3.6: *(a) SKONN on PCB with 9 oscillators and 81 synapses. We set the weight amplitude with the synaptic capacitance value C_{ij}. The FPGA initializes the phases by delaying the oscillators' starting time by switching Q1 and measures the digital oscillations buffered by the output stage (U2 and Q2). (b) The oscillator consists of an OPA Schmitt trigger with feedback via the resistor R_3 to produce self-oscillations. $R_1//R_2$ sets the analog oscillation amplitude. (c) Two coupled oscillators with $C_{12}/C_L = C_{21}/C_L = 1\%$. (d) Two coupled oscillators with $C_{12}/C_L = C_{21}/C_L = 10\%$.*

Fig.3.6c shows an experiment of two oscillators weakly coupled by $C_{12} = C_{21} = 1\%C_L$ whereas Fig.3.6d is a strong coupling with $C_{12} = C_{21} = 10\%C_L$. In both experiments, the oscillators are out-of-phase but the strong coupling case leads to a frequency reduction of -34% as the voltage jumps $\Delta V = (V_{DD} - V_{SS})C_{12}/C_L$ produced by each current spike are too large with

respect to the analog amplitude. This phenomenon can induce frequency mismatches between groups of strongly coupled oscillators and groups of weakly coupled oscillators. Frequency mismatches still need to be investigated and here we empirically choose $C_{ij} < 5\%C_L$ to guarantee phase locking among oscillators.

3.5.2 Weighted Max-Cut Experiments

The test case consists of the Max-cut problem with 2-bit positive weights. Given a graph, the Max-cut objective is to find two complementary graph subsets A and B such that the sum of weighted edges between A and B is maximum. It has been shown in the literature that ONNs can solve the Max-cut problem by mapping the graph vertices to oscillators and assigning the weighted edges to negative weights in ONNs [69, 138, 139]. Chapter 4 provides more rigorous definitions and here we rather focus on SKONN's experimental results compared to Kuramoto simulations. We generate random instances of Erdos-Rényi graphs $G(N,p)$ [196] with N=9 nodes and p is the probability to have an edge between a pair of vertices such that the total number of edges $m = pN(N-1)/2$. For each graph edge, the weight is randomly selected from the list [0 10 22 47]/47 that corresponds to discrete capacitors used experimentally.

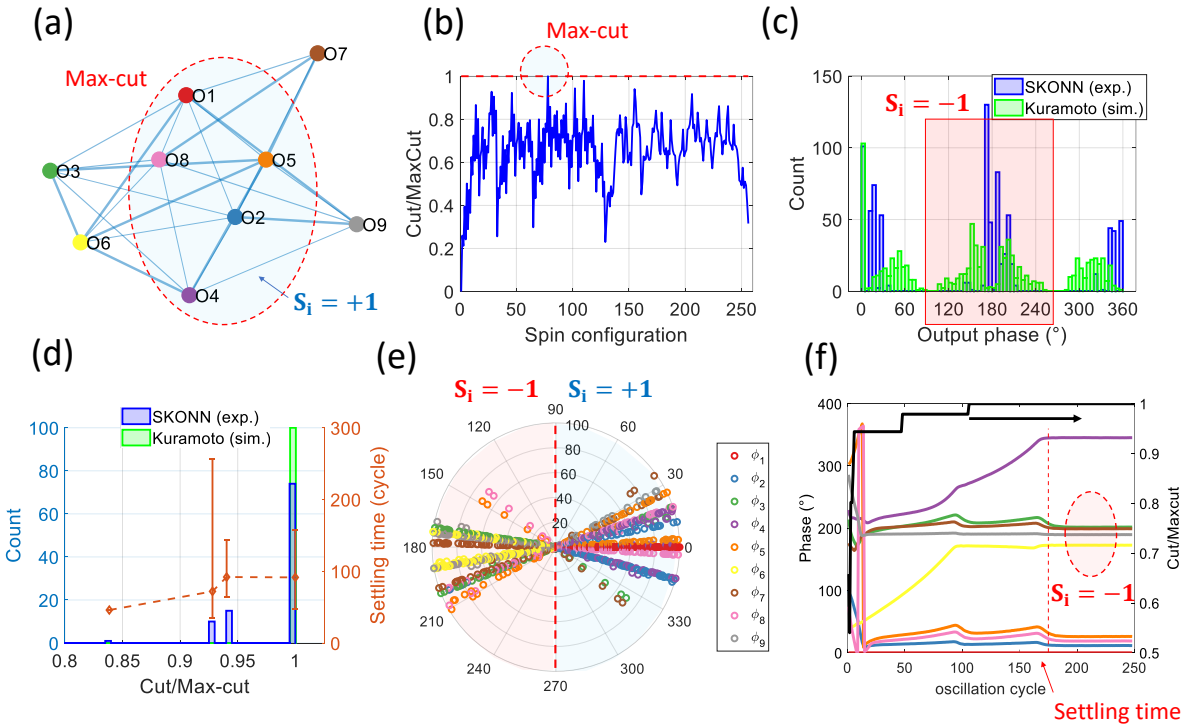


Figure 3.7: *(a) Random instance of $G(9,0.75)$ with 2-bit weighted edges. (b) Cut value vs spin configuration. (c) SKONN phases measured after 1000 oscillation cycles and compared with Kuramoto simulations. (d) Histogram of Max-cut solutions for 100 trials and measured settling time (e) SKONN experimental phase distribution. The polar amplitude represents the trial number (100 trials). We assign $-90° < \phi_i < 90° \rightarrow S_i = +1$ and $S_i = -1$ otherwise. (f) Example of phase dynamics and cut evolution.*

Fig.3.7a shows an example of a dense random graph instance with $p = 0.75$. We map the graph edges to the synaptic matrix and run 100 trials with random phase initializations. For each trial, the nine phases are sampled every oscillation period during 1000 oscillation cycles.

Fig.3.7e shows the final phases $\phi_i(t = 1000\,T)$ measured for each trial, the latter indicated as the amplitude in the polar plot. The right-hand side of the polar plot corresponds to positive spins, whereas the left-hand side corresponds to negative spins. It appears that some phases such as ϕ_2, ϕ_4 and ϕ_6 are always assigned to the same spin polarity whereas most of the phases can end up in both half-circles, depending on the phase initialization. Hence, SKONN final states depend on the initialization and several trials ensure obtaining a good solution. Fig.3.7d shows the histogram of solutions and the settling time. SKONN finds the graph Max-cut with 75% probability in less than 100 oscillation cycles on average. Fig.3.8 presents another Max-cut problem with $G(9, 0.5)$.

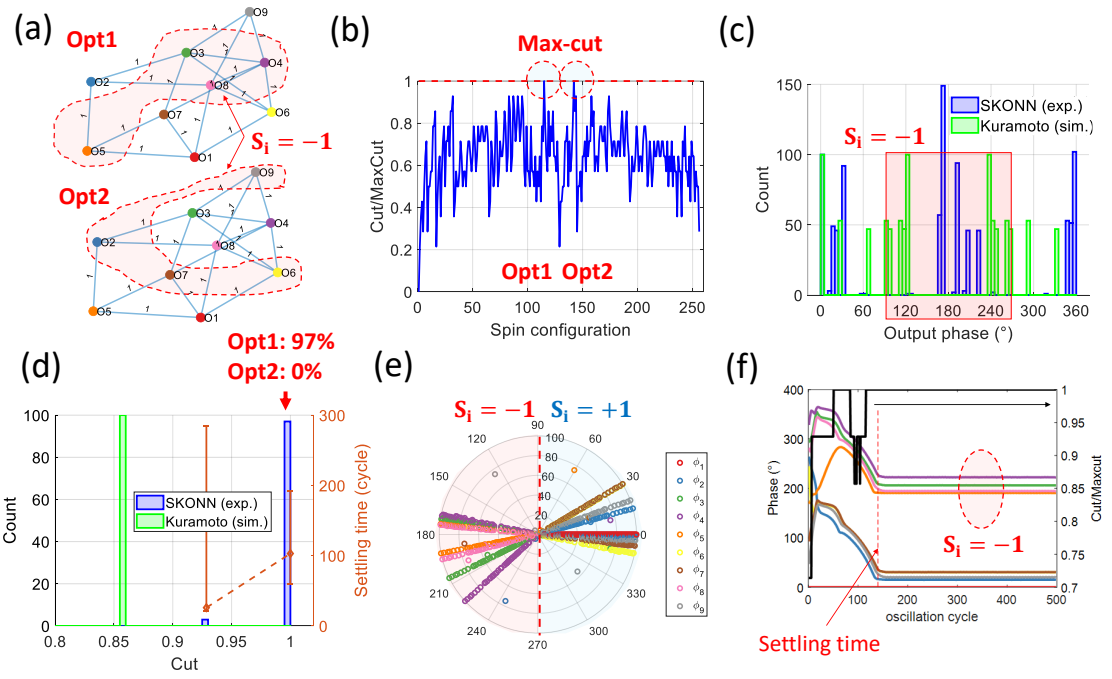
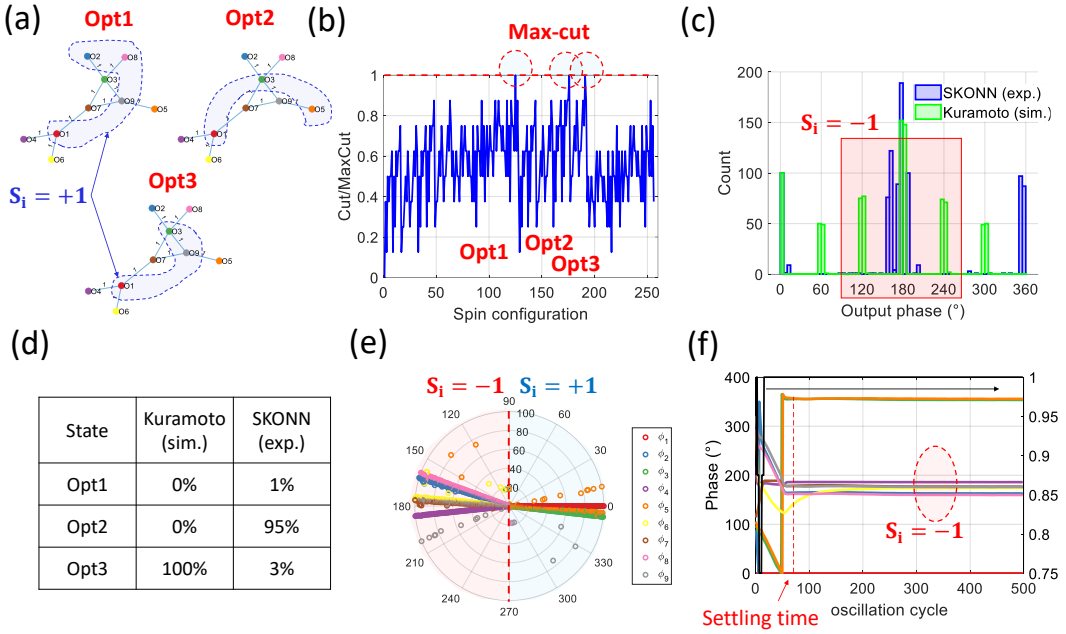


Figure 3.8: *(a) Random instance of $G(9, 0.5)$ with 1-bit weighted edges and the two optimal states. (b) Cut value vs spin configuration. (c) SKONN phases measured after 1000 oscillation cycles and compared with Kuramoto simulations. (d) Histogram of Max-cut solutions for 100 trials and measured settling time (e) SKONN experimental phase distribution. The polar amplitude represents the trial number (100 trials). We assign $-90° < \phi_i < 90° \rightarrow S_i = +1$ and $S_i = -1$ otherwise. (f) Example of phase dynamics and cut evolution.*

This instance is easier to solve as there are two optimal spin states as seen in Fig.3.8a and SKONN reaches 97% accuracy. Whereas Kuramoto simulations led to a lower accuracy due to the large phase distribution causing errors when rounding phases to spin values. Surprisingly, SKONN favors a single optimal state and never finds the second one. We observe a similar behavior even for graph instances with three optimal states as seen in Fig.3.9. This suggests there are optimal critical points that are unstable (such as saddle points) as recently highlighted by Bashar et al. [137].



State	Kuramoto (sim.)	SKONN (exp.)
Opt1	0%	1%
Opt2	0%	95%
Opt3	100%	3%

Figure 3.9: *(a) Random instance of $G(9, 0.25)$ with 1-bit weighted edges and the three optimal states. (b) Cut value vs. spin configuration. (c) SKONN phases measured after 1000 oscillation cycles and compared with Kuramoto simulations. (d) Optimal states found for 100 trials (e) SKONN experimental phase distribution. The polar amplitude represents the trial number (100 trials). We assign $-90° < \phi_i < 90° \rightarrow S_i = +1$ and $S_i = -1$ otherwise. (f) Example of phase dynamics and cut evolution.*

3.6 16-neuron SKONN CMOS Prototype

This section is dedicated to the design of a second SKONN prototype in an integrated circuit (IC) using Europractice's Miniasic program to further assess its performance and programmability. The design process started in November 2021 after discovering SKONN architecture. The chip was designed using a full custom design flow and the Virtuoso software from Cadence with TSMC's PDK @1.2V of supply voltage and 9 metal layers. The parasitic extraction was carried out with Synopsys' Calibre PEX tool. All the design steps were conducted at LIRMM, except the chip metal filling that was done by IMEC engineers before sending out the GDS file to TSMC. The chip was taped out in June 2022 and first tested in January 2023. We provide below the high-level chip specifications and the description of each block.

3.6.1 Integrated Circuit Specifications

Table 3.2 presents the specifications of the state-of-the-art ONNs designed using CMOS technology, transition metal oxide, and spintronic devices. The connectivity scheme can be all-to-all for small-sized ONNs (N$\leq$100) but is obviously reduced to nearest neighbor connections for larger ONNs such as in [76] and [90] that both chose 8 neighbors (King's graph). It appears that digital ONNs such as [89] and [76] are very energy-efficient as they produce a single oscillation with only 300 fJ and 21 fJ, respectively. In contrast, analog oscillators found in [90] and [138] consume 2.3 pJ and 1.3 nJ per oscillation, respectively. Finally, novel devices such as vanadium dioxide (VO_2 [86]) and spin-torque oscillators [109] are promising for future ONN implementation thanks to their potential dense integration and high frequency.

	Jackson et al. [89]	Moy et al. [76]	Bashar et al. [138]	Graber et al. [90]	Dutta et al. [86]	Romera et al. [109]	Our target
Technology	28nm	65nm	65nm	28nm	Simulations: 28nm CMOS + PTNO (VO_2)	Spin-torque	65nm
Neurons	100	1968	30	400	100	4	16
Connectivity	all-to-all	King's graph	all-to-all	King's graph	all-to-all	all-to-all	all-to-all
Power	303 mW	42 mW	1.76 mW	182 mW	2.56 mW	4 mW	160 μW (core)
Frequency	1 GHz	1 GHz	45 kHz	200 MHz	87 MHz	300 MHz	1 MHz
Energy/osc	300 fJ	21 fJ	1.3 nJ	2.3 pJ	300 fJ	3.3 pJ	10 pJ
Chip area	3.24 mm²	2.1 mm²	-	2.2 mm²	-	-	< 4 mm²

Table 3.2: *State-of-the-art ONNs and IC Specifications*

For this first IC proof-of-concept, we chose TSMC's 65-nm technology with the objective of integrating 16 oscillators and all-to-all connectivity (256 synapses) with 5 bits of synaptic precision as shown in Fig.3.10. Moreover, given the performances of the state-of-the-art ONNs (Table 3.2), we targeted an energy consumption of around 10 pJ per oscillation at 1 MHz only to minimize the risks. Finally, we wished to have the ability to control each one of the oscillators externally by sending its initialization and sub-harmonic injection signal, while receiving its output signal. This requirement induced a high number of Input/Output pads (IOs) estimated as follows:

- 16 inputs for initialization.
- 16 inputs for SHIL signals.
- 4 inputs for the synaptic programmation (serial).

- 8 power supply pads (digital domain).
- 8 power supply pads (analog domain).
- 4 analog pads for biasing.

Due to the high number of pads > 56, we did not have severe area constraints for the core design as the high number of IOs in the pad ring hardly fit in a square smaller than 4 mm^2. The initial design specifications are resumed in Table 3.2.

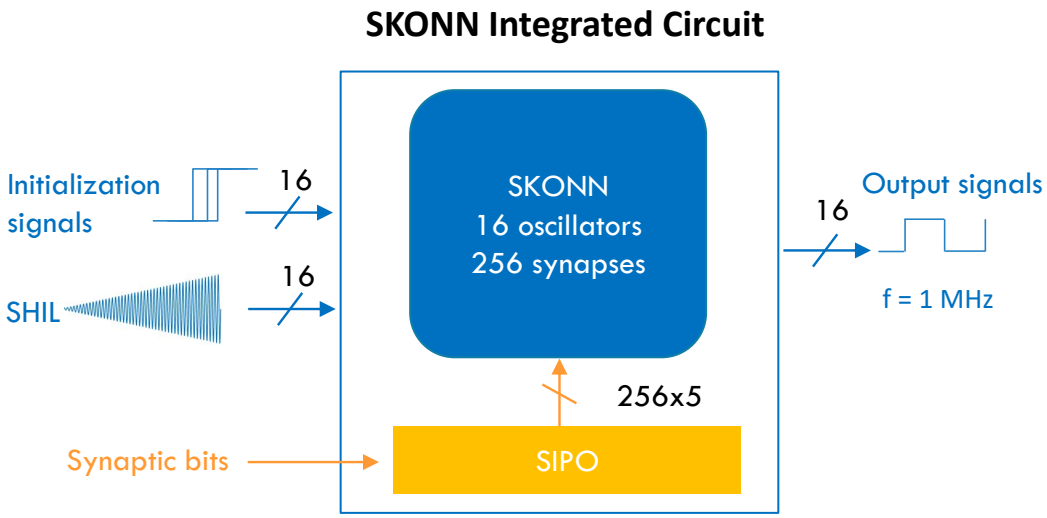


Figure 3.10: *SKONN IC High-level View*

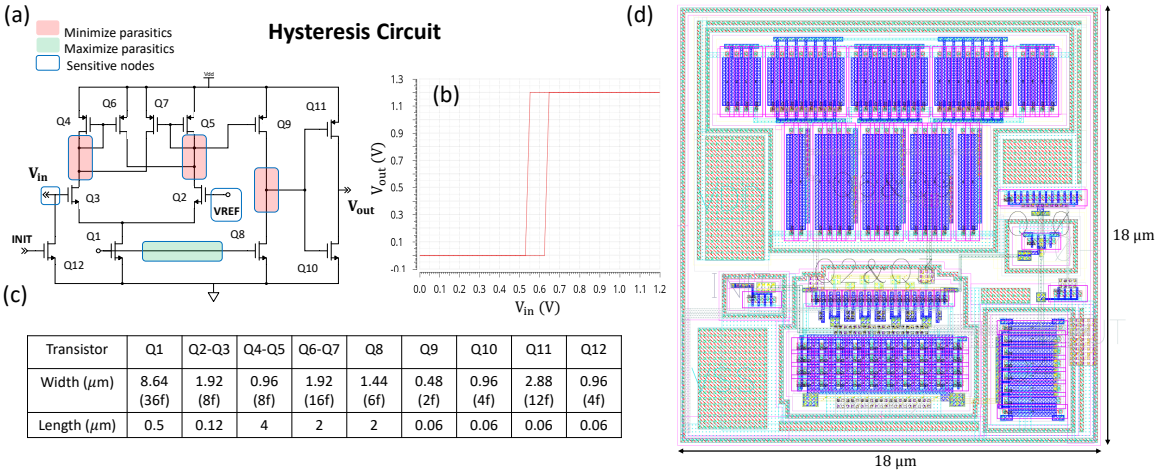
3.6.2 Oscillator Circuit Design

Hysteresis Circuit

A SKONN oscillator consists of a hysteresis and a shaper block, as shown in Fig.3.2a. There are various CMOS circuits having a hysteresis behavior such as Schmitt triggers [160] with 6 transistors only [197] that have already been used in ONNs [138]. However, in this circuit, the hysteresis thresholds are set by the supply voltage and the transistor threshold voltage which are process-dependent. Instead, we opted for a differential Schmitt trigger with regenerative current feedback [198, 199] whose hysteresis is only set by device dimensions and biasing current. The hysteresis circuit is shown in Fig.3.11a and consists of a standard operational transconductance amplifier (OTA) with positive feedback applied by transistors Q6 and Q7. Denoting $R_6 = W_6/L_6 = W_7/L_7$ and $R_4 = W_4/L_4 = W_5/L_5$, the circuit has a hysteresis effect when $k = R_6/R_4 > 1$. In our case, $k = 4$ and the hysteresis width can be calculated as [198]:

$$\Delta V_H = 2\sqrt{\frac{2nI_{REF}}{\beta(k+1)}}(\sqrt{k}-1) \approx 60\,mV \tag{3.12}$$

where $n \approx 1.1$ models the bulk effect, $I_{REF} = 4.5\,\mu$A is the bias current flowing through Q1 and $\beta = \mu_n C_{ox} W/L = 200\text{x}16\,\mu\text{A/V}^2$ for transistors Q2-3. Note that to achieve our initial target of 1 pJ per oscillation at 1 MHz Table3.2, the hysteresis block should have been biased with less than 1 μA. However, during the design process, we noticed that the hysteresis circuit did not switch fast enough for SKONN and we increased the bias current until we got satisfactory results. The relationship between hysteresis bandwidth and theoretical phase fixed points is further explained in Appendix B.6. Note that in dynamic operation, the oscillator amplitude $\Delta V \approx 100$ mV is larger than ΔV_H due to the hysteresis limited bandwidth and shaper delay. Fig.3.11a shows the circuit nodes that require minimization of parasitics for high speed, and the sensitive high impedance nodes necessitating shielding against noise (such as V_{in} and V_{REF}). These constraints were considered during the layout process (Fig.3.11d).



Transistor	Q1	Q2-Q3	Q4-Q5	Q6-Q7	Q8	Q9	Q10	Q11	Q12
Width (μm)	8.64 (36f)	1.92 (8f)	0.96 (8f)	1.92 (16f)	1.44 (6f)	0.48 (2f)	0.96 (4f)	2.88 (12f)	0.96 (4f)
Length (μm)	0.5	0.12	4	2	2	0.06	0.06	0.06	0.06

Figure 3.11: *a) Hysteresis circuit schematic. b) Hysteresis characteristic for a fixed $V_{ref} \approx 0.6$ V. c) Transistors' dimensions. d) Layout.*

Shaper Circuit

The full oscillator circuit is shown in Fig.3.12a. The hysteresis commands the shaper block which charges or discharges the load capacitor $C_L = 500$ fF via current sources $I_1 = I_2 = 200$ nA. To ensure a symmetric triangular waveform V_{in}, the current sources I_1 and I_2 copy the same current in a cascode configuration as seen in Fig.3.12b. As the analog waveform amplitude V_{in} does not exceed 100 mV, using a cascode topology was possible here without bringing Q20 and Q21 in the triode regime. Fig.3.12c shows the biasing circuits for $V_{REF} \approx 0.6$V and $I_{REF} \approx 4.5\,\mu$A. They polarize the oscillator circuit based on a bias current $I_{bias} = 200$ nA brought by the chip's main bias block.

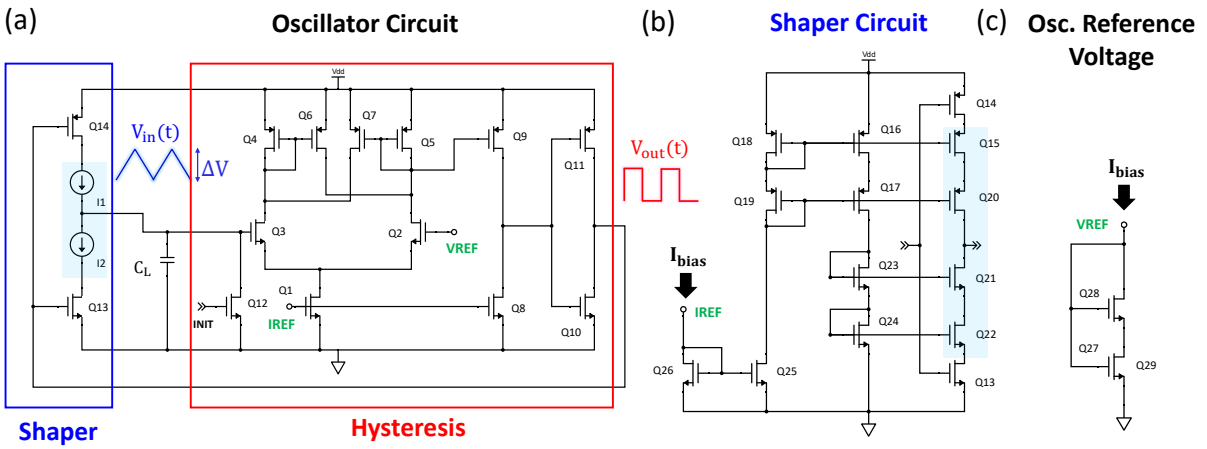


Figure 3.12: *a) Oscillator circuit. b) Full Shaper circuit. c) Bias circuit used to produce the hysteresis voltage reference V_{REF}. The current I_{bias} is distributed by the chip's global bias block.*

Optimization of the Oscillator Circuit Design

As for any phase-based ONNs, one of the main important specifications is to have homogeneous frequencies between oscillators, which is challenged by device-to-device mismatches within the chip. In our design, the oscillation frequency is expressed as:

$$f_{osc} = \frac{I_{shaper}}{2C_L\Delta V} \tag{3.13}$$

where I_{shaper} is the current sunk by current sources I_1 and I_2 (Fig.3.12a) and ΔV is the peak-to-peak analog amplitude. The relative frequency variation is then calculated as follows:

$$\frac{df_{osc}}{f_{osc}} = \frac{dI_{shaper}}{I_{shaper}} - \frac{dC_L}{C_L} - \frac{d\Delta V}{\Delta V} \tag{3.14}$$

Although there are negligible variations in the large load capacitance $C_L = 500$ fF implemented with a metal-oxide-metal capacitor (MOM), we found in Monte Carlo simulations that the main variations originate from the shaper block. Hence, the shaper block had to be carefully designed in order to minimize the current mismatches expressed as [200]:

$$\sigma_{I_D} = \sqrt{\sigma_\beta^2 + (\frac{g_m}{I_D}\sigma_{V_T})^2} \tag{3.15}$$

where I_D, g_m are the transistor current and transconductance; and σ_{V_T}, σ_β are transistors variations depending on the technology [200]. However, these two contributions can be minimized by increasing the transistor dimensions:

$$\sigma_{V_T-\beta} \propto \frac{1}{\sqrt{WL}} \tag{3.16}$$

Moreover, the equation 3.15 indicates that the quantity g_m/I_D should be minimized, i.e. the transistor should be in strong inversion (SI) regime [201] and have a low W/L for a given current I_D.

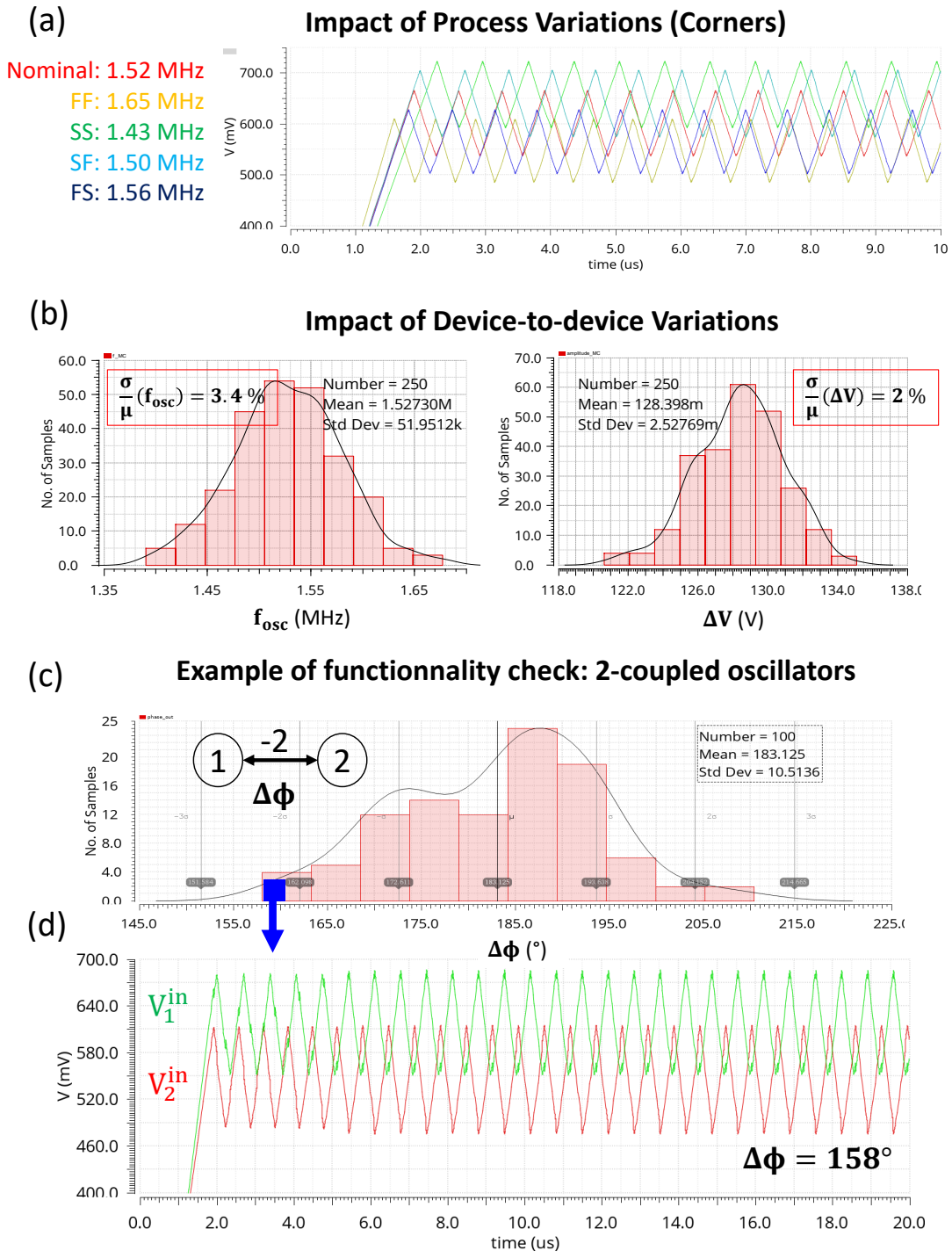


Figure 3.13: *a) Impact of process variations. SF stands for Slow NMOS, Fast PMOS. Note how the triangular waveform remains symmetric despite the NMOS/PMOS performance asymmetry. b) Monte Carlo (MC) simulation measuring the oscillator frequency and amplitude after design optimization. c) Example of functionality check carried out during the design process. The phase fixed point between two-coupled oscillators varies with device variations. d) Worst MC instance resulting in* $\Delta\phi = 158°$ *instead of* $180°$ *in theory.*

Overall, the oscillator transistor sizing was finalized after several iterative steps mainly consisting of **1)** Monte Carlo and corner transient simulations to measure the oscillator frequency variations subjected to device and process variations (Fig.3.13a and b), **2)** resizing the transistors, **3)** comparing the resulting SKONN operation with the phase-based model (eq.3.7) as

in Fig.3.13c with two-coupled oscillators. This process reduced the frequency variations from $\sigma_{f_{osc}}/\mu_{f_{osc}}$ =19% down to 3.4% (Fig.3.13b), but increased the shaper and bias transistor sizing that are listed in Table 3.3.

Transistor	Q13-14	Q15-Q19	Q20-21	Q22-24	Q25-26	Q27-28
Width (μm)	2.4 (10f)	11.52 (12f)	0.48 (1f)	11.52 (12f)	9.6 (10f)	0.24 (2f)
Length (μm)	0.06	6	0.25	6	20	7

Table 3.3: *Shaper Transistor Dimensions in the IC*

Note that Monte Carlo simulations from Fig.3.13d highlight SKONN's robustness to any shift of the DC operating point (set by V_{REF}) as the coupling between oscillators is purely AC.

Oscillator Layout and Parasitics Extraction

The final oscillator layout is shown in Fig.3.14. To minimize transistor mismatches, every group of transistors is laid out in a common-centroid topology, i.e. their fingers are interdigitated to form a symmetrical single block. Note that the MOM load capacitor C_L' = 320 fF = 500 fF-C_{par} was adjusted after extracting the parasitics C_{par} added to V_{in}'s metal line from the synaptic array. The post-layout simulation that includes parasitic capacitances shows frequency and amplitude variations of -13% and +9%, respectively. The main contributor to the +9% amplitude variation is the hysteresis block whose bandwidth is impacted by the parasitics and takes a longer time to switch, thus increasing ΔV. Moreover, assuming that the shaper current is not impacted by the parasitic capacitances, we conclude from eq.3.14 that the frequency variation is expressed as $\Delta f_{osc}/f_{osc} = -13\% \approx -\Delta C_L/C_L - 9\%$ so around $4\% C_L = 20$ fF of parasitic is added to the oscillator input node.

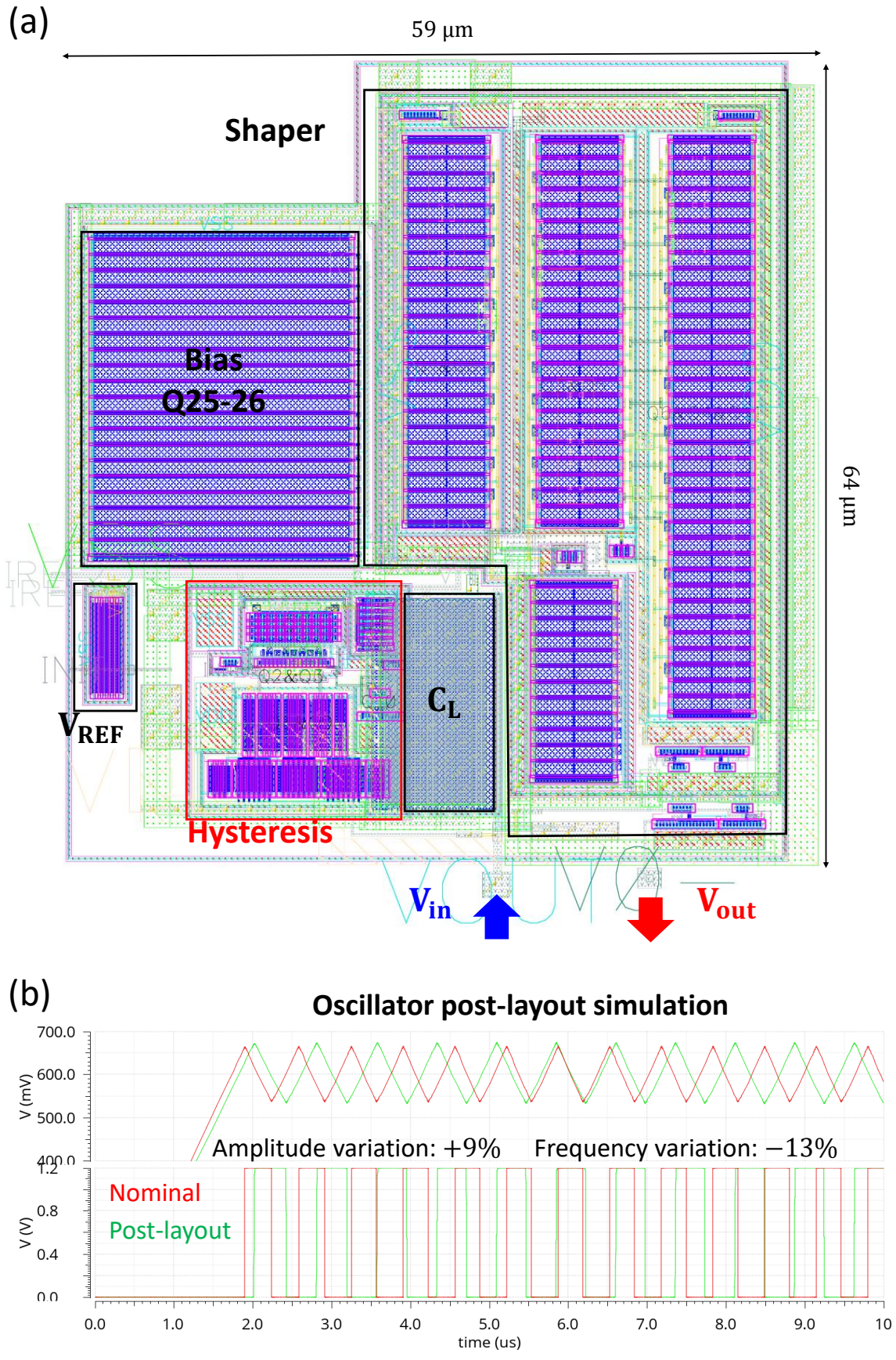


Figure 3.14: *a) SKONN Oscillator Layout. b) Post-layout simulation.*

3.6.3 Synaptic Design

A synapse consists of a programmable bank of four capacitors C_i=2.5 fF x 2^i with $i \in \{0,1,2,3\}$ connected in parallel such that a 4-bit weight W_{ij} is linearly mapped to a synaptic capacitor $C_{ij} = |W_{ij}|$x2.5 fF (Fig.3.15) with a synaptic range of [0, 37.5] fF. We chose the synaptic range as follows. From our previous PCB prototype, we concluded that having $C_{ij} < 10\%C_L = 50$ fF prevents too large frequency mismatches between groups of strongly and weakly coupled oscillators (Fig.3.6b). Moreover, the synaptic smallest capacitance was set by the device and technology. We chose to implement the capacitors with metal-oxide-metal capacitors (MOM) for enhanced linearity compared to MOS capacitors. Note that the metal-insulator-metal capacitors (MIM) are only implementable at the highest metal levels and are generally used as decoupling capacitors. In this technology, the smallest available MOM capacitor has a 2.5 fF value. It consists of an array of 6x6 100nm-width fingers per metal layer, from metal 2 up to metal 4, with a footprint of 1.7 μm x 3 μm. To shield the synaptic capacitor from the substrate, we chose an N-type MOM capacitor with an NWELL underneath.

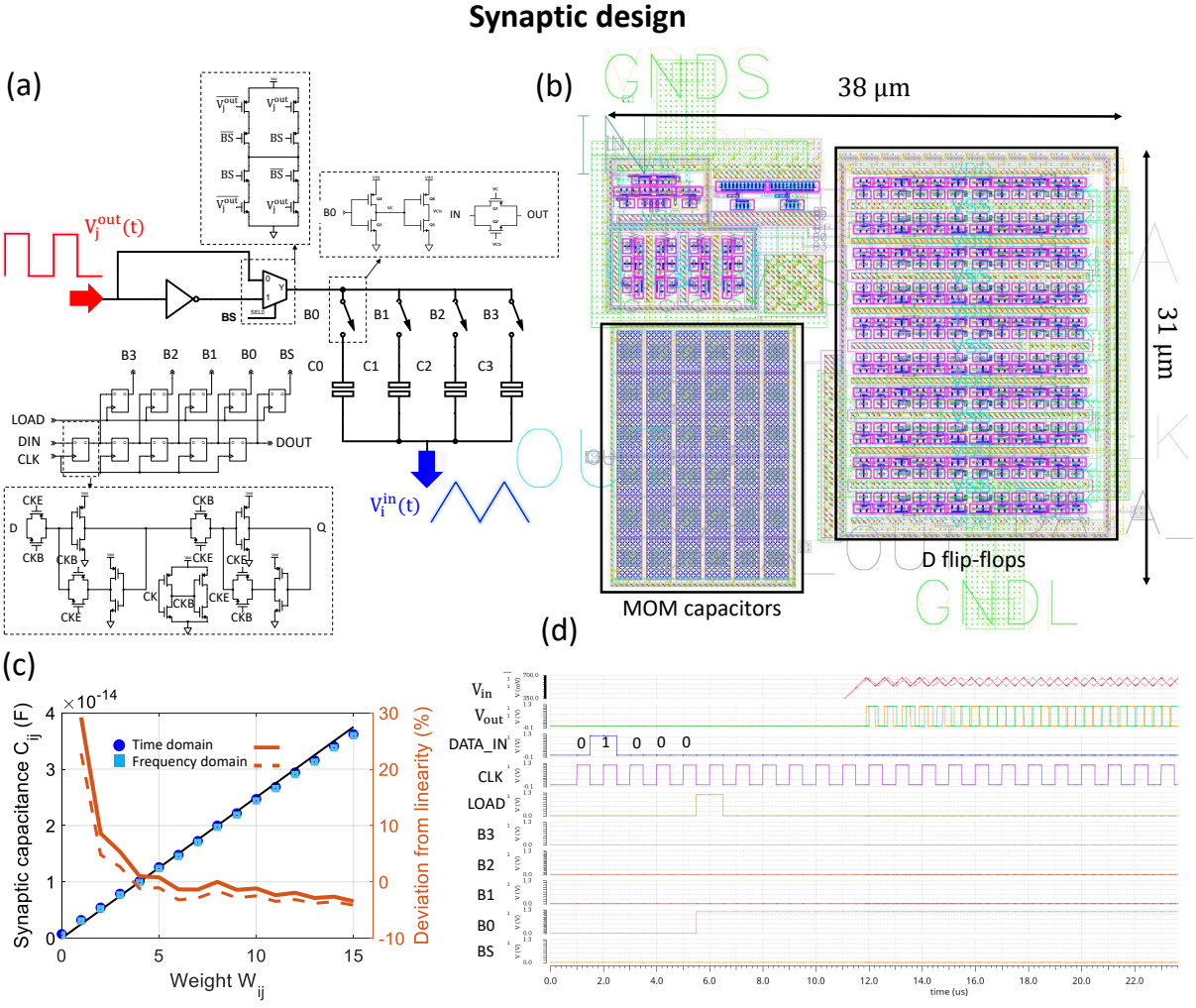


Figure 3.15: *a) Synapse circuit. b) Synapse layout. c) Post-layout synaptic capacitance vs. absolute weight value. d) Example of programmation procedure with $W_{ij} = -1$. The upper waveforms show the input and output signals for the neurons i and j.*

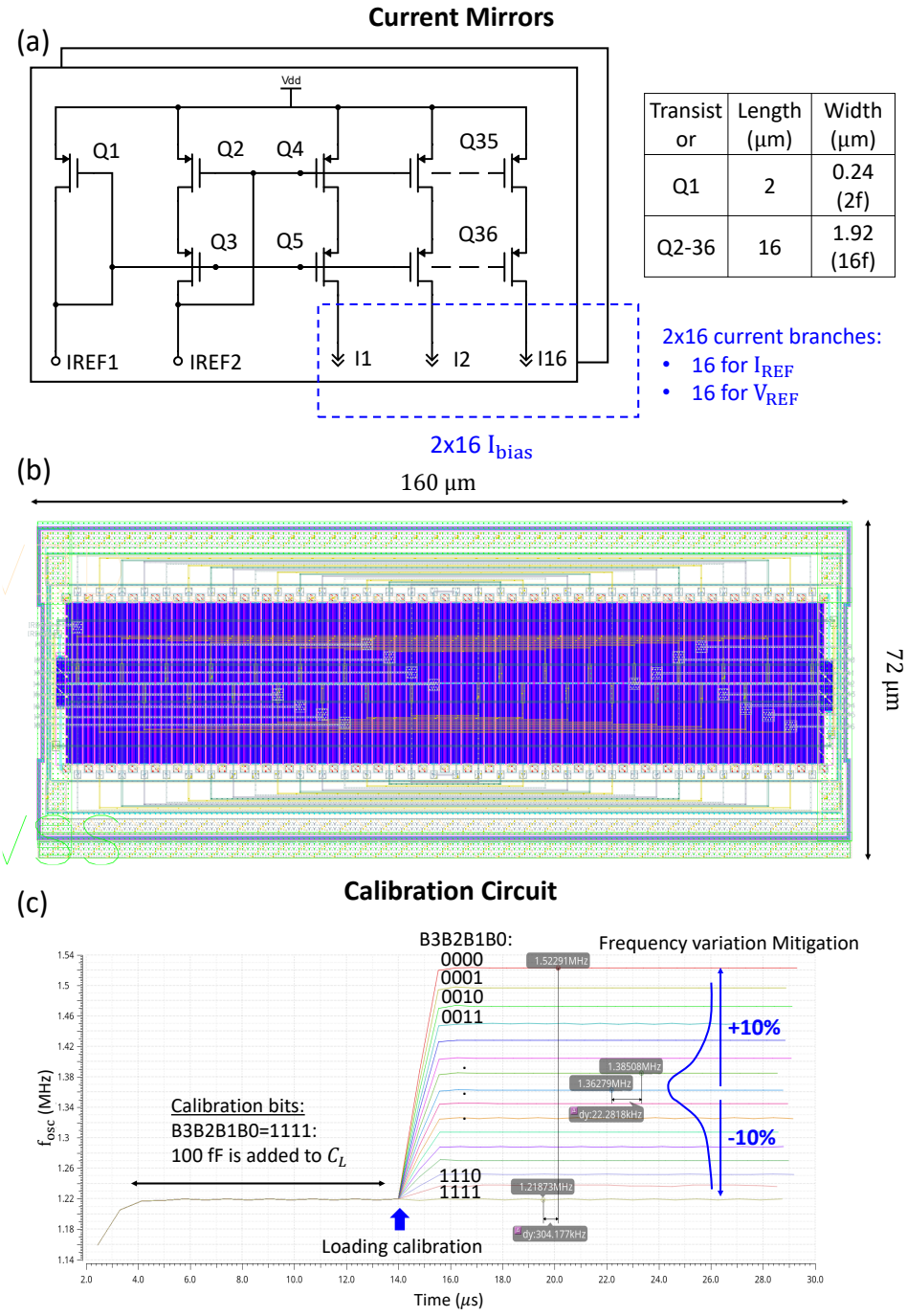
The synaptic weight sign is programmed by selecting the input signal V_j^{out} or $\overline{V_j^{out}}$ if $W_{ij} > 0$ or $W_{ij} < 0$, respectively. This is the function of the standard 2-input multiplexer shown in Fig.3.15a and commanded by the bit sign *BS*. Overall, the synaptic circuit implements a synaptic range from -15 to +15 and is programmed by a standard register consisting of five D flip-flops loading the weights serially at each clock cycle. A second register applies the voltages to the switches when receiving the 'LOAD' signal, as shown in Fig.3.15d with the programmation of $W_{ij} = -1$. Due to some challenges with our PDK, we did not have the layout views of the standard logic blocks thus we designed our own D flip-flops to program the synapse.

Fig.3.15b shows the layout view of the synaptic block. Note that the 15 MOM capacitors are encircled by dummy devices to avoid the impact of any process variation at the edge of the synaptic array. The area taken by the register is dominant here but could be reduced in a future design by using optimized standard cells provided by the PDK. Fig.3.15c shows the post-layout synaptic capacitance measured using two different methods that provide similar results. One method measures the injected charge in the time domain, while the other computes a gain in the frequency domain and they are further described in Appendix X. It appears that the deviation from linearity remains under 5% for $W_{ij} \geq 3$ but is higher for smaller synaptic capacitances due to the added parasitics. For instance, when all the switches are opened ($W_{ij} = 0$), it remains a parasitic capacitance of $C_{ij} = 0.7$ fF that would be equivalent to a weight $|W_{ij}| \approx 0.3$. Unfortunately, we realized too late that this small parasitic is sufficient to synchronize oscillators that should be disconnected (Huygens effect [77]). Using Monte Carlo simulations to emulate device-to-device variations, it appears that two-coupled oscillators can phase-lock 40% of the trials after a few tens of cycles. A possible improvement would be the addition of a 4-input OR(B0,B1,B2,B3) gate in series with the multiplexer to further shield the synaptic output from the input when $W_{ij} = 0$.

3.6.4 Additional Circuits

Biasing Circuit

The chip biasing block provides two bias currents $I_{bias} = 200$ nA for I_{REF} and V_{REF} nodes in each oscillator (Fig.3.12). Fig.3.16a shows the bias circuit which consists of two blocks of 16 current mirrors, with each block dedicated to I_{REF} and V_{REF} nodes, respectively. Similar to the shaper circuit, it is preferable to have low mismatches between each bias current so that the bias block only introduces negligible frequency mismatches among oscillators. Hence, the transistors' dimensions are critical and set such that the transistors are in strong inversion to minimize current variations (eq.3.15). Overall, we measure a relative standard deviation $\sigma/\mu = 1.6\%$ for I_{bias} during Monte Carlo (MC) simulations including device-to-device variations. Note that the previous MC simulation measuring the oscillator frequency variation (Fig.3.13b) includes the variations of the chip bias block and validates its good operation.



Transistor	Length (µm)	Width (µm)
Q1	2	0.24 (2f)
Q2-36	16	1.92 (16f)

Figure 3.16: *a) Schematic of two bias blocks. b) Layout of a single bias block (16 current branches). c) Operation of the calibration circuit.*

Calibration Circuit

The goal of the calibration circuit is to adjust the oscillation frequency of each oscillator if needed. It is similar to the synaptic circuit and consists of a 4-bit capacitor bank connected to the input line to modulate the load C_L with steps of 6.7 fF as $C_{calib} = 6.7\text{x}n$ with $n \in \{0, 1, 2, .., 14, 15\}$. Each capacitance step corresponds to a 10x10-fingers MOM capacitor laid out from M1 to M4. All the possible calibrations are shown in Fig.3.16c. The calibration circuit

can add a total capacitance of around 100 fF. Hence, for $C_L = 500$ fF, the calibration circuit can reduce the frequency down to -20% and is enough to compensate a Gaussian distribution of frequencies such as $3\sigma/\mu \leq 10\%$.

3.6.5 Floor Planning and Layout of the Prototype IC

The chip blocks are placed in a symmetric topology as shown in Fig.3.17a. Two blocks of 8 oscillators are positioned at the top and bottom of the chip to have similar output line lengths that are laid out horizontally and buffered to keep sharp digital edges. The 1-mm length oscillator input lines are laid out vertically on metal 1 layer (M1) with a 180-fF parasitic capacitance C_{par} added to the input node such that $C_L = C'_L + C_{par} = 500$ fF. Moreover, the oscillator calibration circuit is added at the end of each input line to adjust the oscillator frequency if needed. The bias block is at the center of the chip and provides 16 current paths to each part of the chip laid out on M4-5. Note that we placed M3 ground plans at the intersection between the output and bias lines to shield them from the digital signals.

All the synaptic and calibration registers are connected serially to form a 1344-bit register (256x5+16x4) whose input starts at the bottom-left of the chip. As shown in Fig.3.17a., the data goes through every line of the synaptic array and is propagated to the top-right of the chip. Hence, the chip is ready for inference after 1344 clock cycles fed through one of the DATA pads (Fig.3.17b). All the signal pads are digital except the BIAS analog pads used to polarize the chip bias block from the outside. The unannotated pads from Fig.3.17b are power pads used for both digital and analog power domains.

3.7 IC Characterization

3.7.1 IC Package and PCB

The fabricated IC die is shown in Fig.3.18a. We chose to package the die in a pin grid array (PGA) in order to use a zero insertion force (ZIF) IC support on the experimental PCB. The final package choice was mainly determined by the maximum length of the wire bound, and we finally chose the 120-pad CPG12034 package from NTK. The experimental PCB is shown in Fig.3.18a and consists of a 144-ZIF support holding the IC and communicating with an FPGA board. Two current mirror circuits bias the chip with I_{bias} ranging from 100 nA to 1 μA. The PCB was carefully designed using 4 layers to homogenize and minimize the routing line lengths. The user interface is on Matlab and communicates with the FPGA via a serial port. Then, the FPGA commands the IC depending on the user's requests. In return, the FPGA measures the chip output signals on its digital inputs and computes the phase of the oscillators with respect to a reference at each oscillation cycle. Finally, the phase data is sent back to Matlab for post-processing.

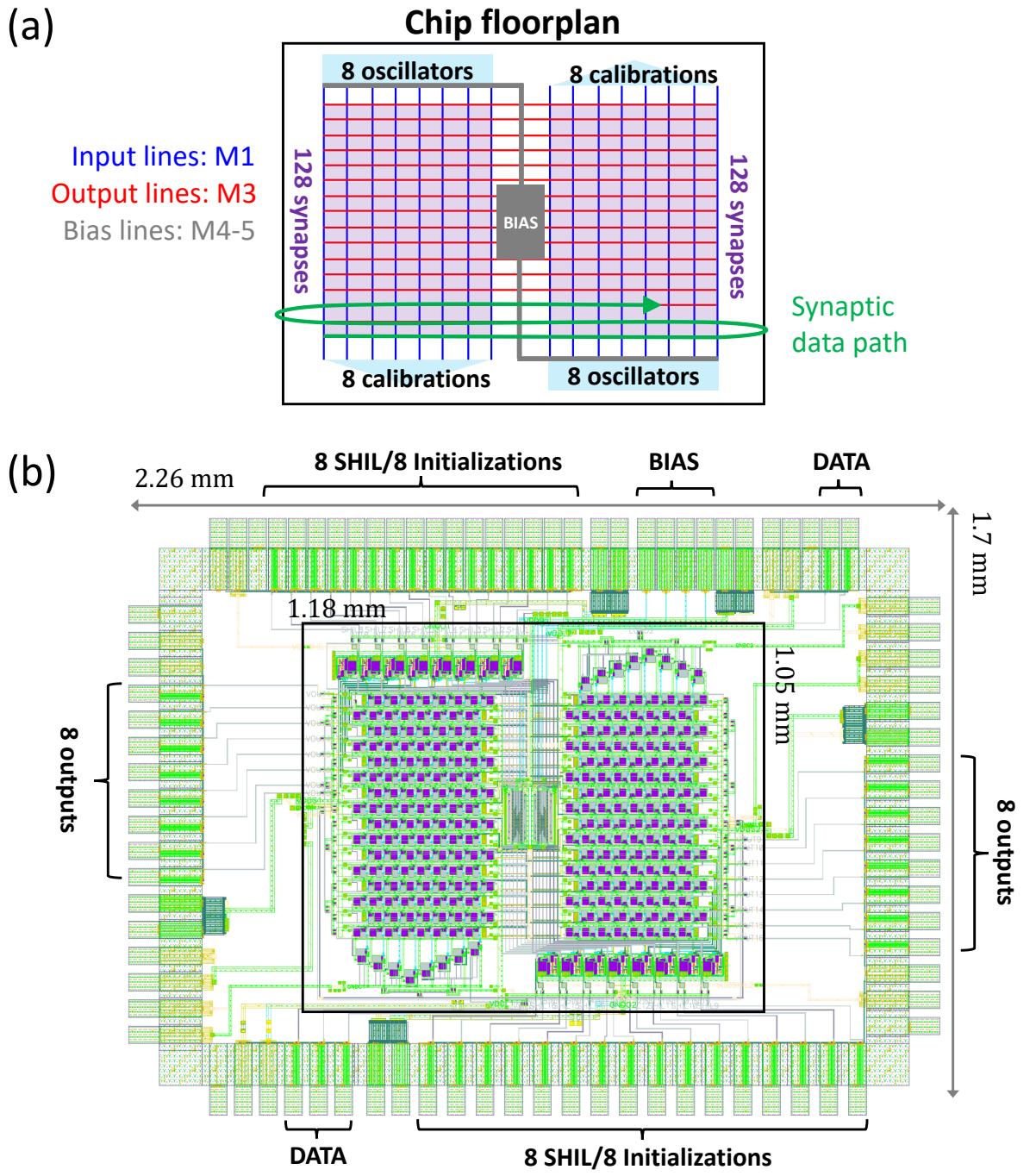


Figure 3.17: *a) Floorplan of the IC. b) Final layout of the chip.*

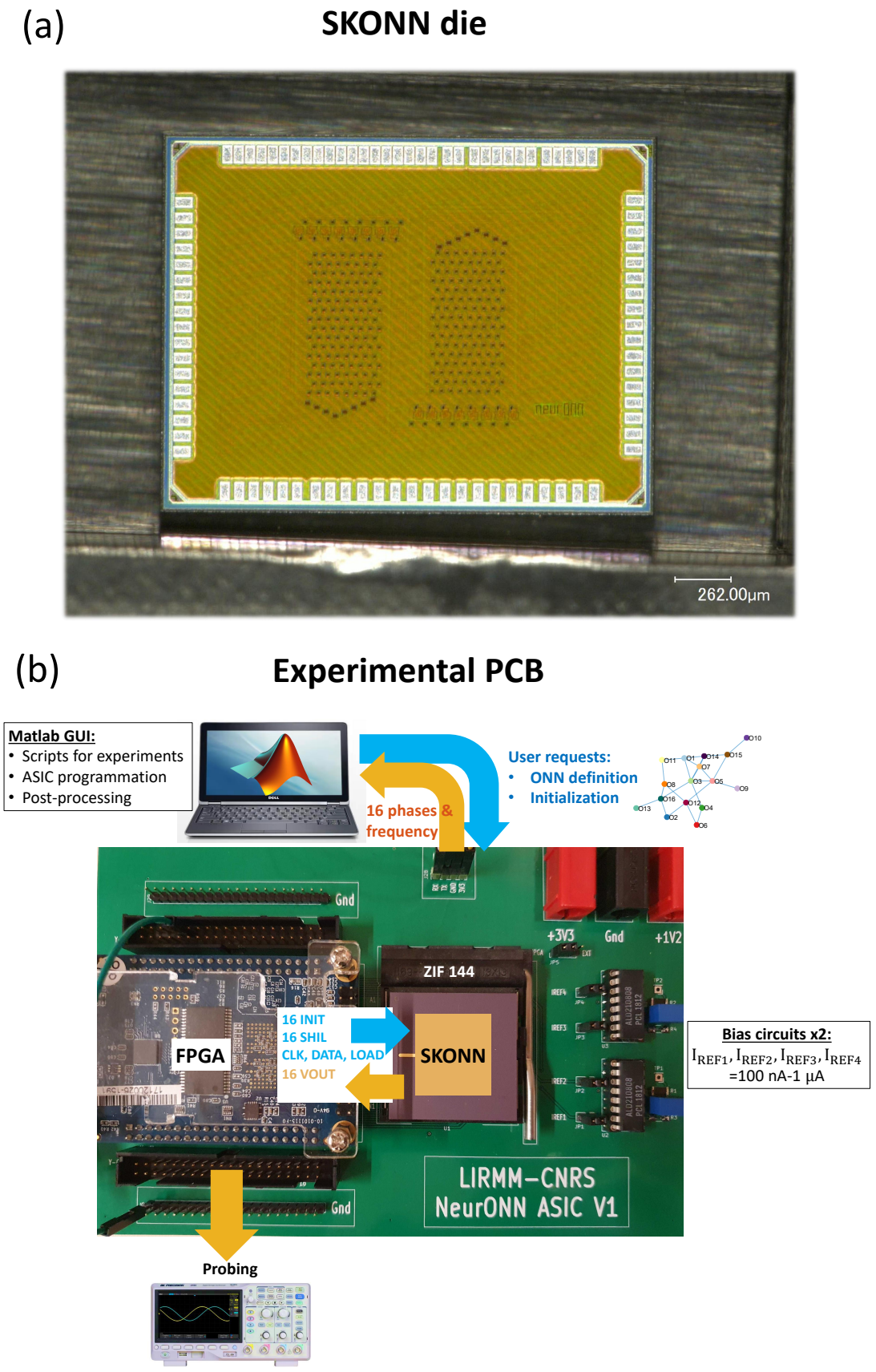


Figure 3.18: *a) Photography of SKONN die. b) Schematic of experimental PCB to test the IC.*

3.7.2 Characterization of SKONN Building Blocks

Oscillator Frequency at Steady State

The oscillator circuit was initially designed to operate at $f = 1.5$ MHz with $I_{bias} = 200$ nA. After post-layout simulations at the oscillator level, we observed a -13% frequency reduction, i.e. an expected frequency $f \approx 1.3$ MHz (Fig.3.14b). However, it is still underestimated as the simulation does not consider all the chip parasitic capacitances such as the metal fillings. In practice, we had to inject $I_{bias} = 366$ nA to get $f \approx 1.13$ MHz. Table 3.4 reports each individual oscillating frequency measured with an oscilloscope while the oscillators are uncoupled and started for a long time (minutes). The oscillating signal was analyzed using a Fast Fourier Transform (FFT) with a resolution of 150 Hz during a time window of 3.1 ms, i.e. more than 3000 oscillation periods. As the frequency mismatches are below 0.3%, it is likely that the oscillators get synchronized during the measure. This hypothesis is confirmed by measuring the phase dynamics in real time.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16
f (MHz)	1.127	1.126	1.129	1.130	1.128	1.127	1.128	1.129	1.129	1.127	1.130	1.130	1.131	1.130	1.130	1.130
$\Delta f/f$ (%)	0	-0.09	0.18	0.27	0.09	0	0.09	0.18	0.18	0	0.27	0.27	0.35	0.27	0.27	0.27

Table 3.4: *IC Frequency Measurement at Steady State*

Oscillator Synchronization (Huygens effect) and Transient Frequency

Fig.3.19a shows the oscillator phases measured during 1000 oscillations where the first oscillator is taken as a reference. It appears that the group of oscillators 1-8 are synchronized as their phases stabilize. However, the group 9-16 is not synchronized with the reference oscillator. Instead, their phase dynamics show a modulation with a frequency of around $|f_{9-16} - f_1| \approx 25$ kHz, which is the consequence of a frequency mismatch between the two groups of oscillators expressed as:

$$\cos(\phi_{9-16} - \phi_1) = \cos\left(2\pi(f_{9-16} - f_1) + \theta\right) \tag{3.17}$$

To measure the synchronization level between signals, we use the metric $\sigma(\cos\phi(t))$ which is the standard deviation of $\cos\phi_i(t)$ over time defined as:

$$\sigma(\cos\phi(t)) = \sqrt{\frac{1}{L-1}\sum_{k=1}^{L}\left(\cos\phi(kT) - \mu\right)^2} \tag{3.18}$$

where μ is the mean $\mu = \sum_{k=1}^{L}\cos\phi(kT)/L$ and T is the oscillating period of the reference oscillator. Computing $\sigma(\cos\phi_i(t))$ indicates if the phase value varies through time, i.e. if oscillator i is synchronized with the reference. A value close to 0 means the phase is stable through time, whereas a value tending to 1 indicates an unstable or rotating phase. Fig.3.19b presents the standard deviation computed over 100 trials for each oscillator. Its median value is around 0.2-0.3 for oscillators 1-8 whereas it reaches 0.6 for oscillators 9-16, highlighting the synchronization asymmetry between the two groups. However, there are trials where all the oscillators are synchronized as in the example of Fig.3.19c. Due to memory limitations in the FPGA, we

could not measure dynamics longer than 1000 cycles with the current resolution. Hence, it is possible that the unsynchronized state is only transient and that all oscillators get synchronized after several thousands of cycles. This hypothesis is confirmed by previous frequency measurements realized minutes after the oscillator starting times (Table 3.4).

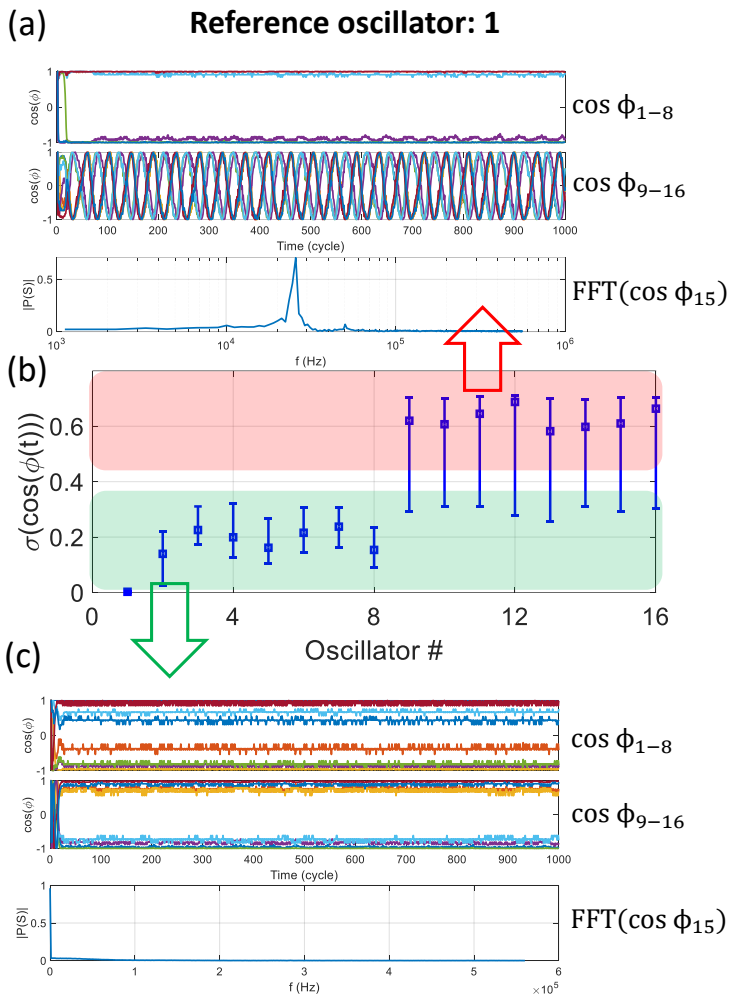


Figure 3.19: *Chip measurements when oscillators are uncoupled and with reference oscillator 1. a) Example where oscillators 9-16 are not synchronized to oscillator 1 while measuring a frequency mismatch between the two groups around 25 kHz. b) Standard deviation of spin signals measured over 1000 cycles and 100 trials. c) Trial instance where all oscillators are synchronized with the reference.*

We repeat the same experiment by taking the oscillator 16 as a reference. Fig.3.20a and c show examples of dynamics while Fig.3.20b presents the resulting standard deviations computed over 100 trials. We notice that the results are symmetric with the previous experiment, i.e. this time oscillators 1-8 are not synchronized with oscillators 9-16. Again, the phase dynamics indicate a transient frequency mismatch of around 25-30 kHz. To further assess the frequency mismatch observed during the two experiments, the FPGA is programmed to measure the oscillation frequency of the reference oscillator in real time. Over the 1000 oscillation cycles measured from experiments a) of Fig.3.19 and Fig.3.20, we finally measure a mean frequency:

$$f_{1-8} = 1.135 \text{ MHz}$$
$$f_{9-16} = 1.164 \text{ MHz}$$

which is consistent with the oscilloscope measurements at steady state (Table 3.4). Hence, oscillators 9-16 are +2.5% faster than oscillators 1-8. Note that the measured mismatch is smaller than the expected frequency standard deviation $\sigma/\mu = 3.4$ % (Fig.3.14b).

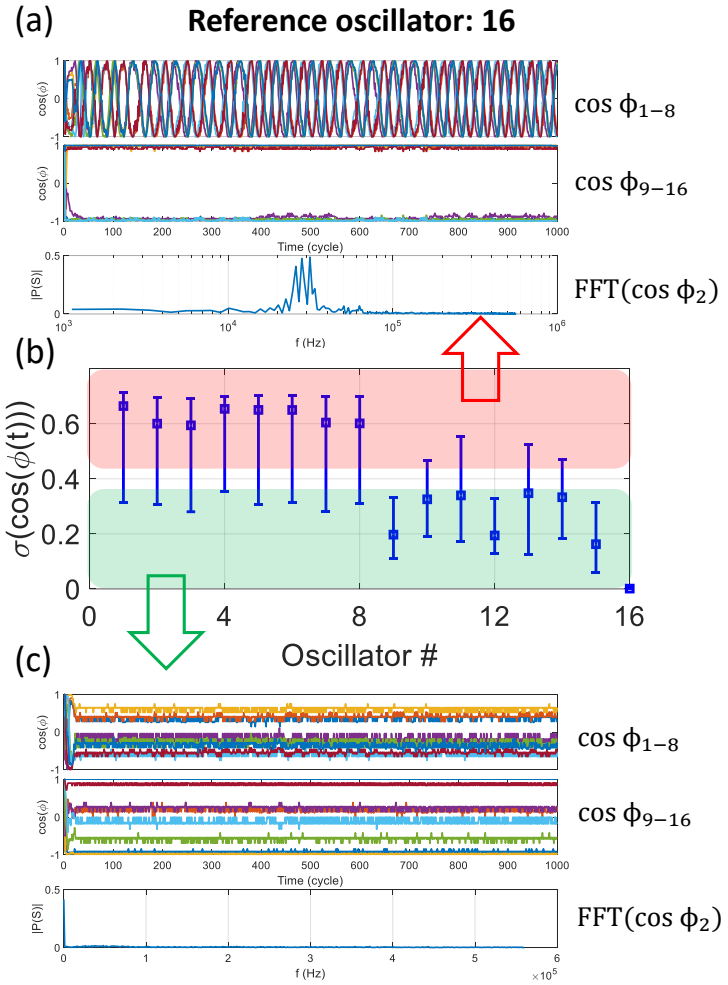


Figure 3.20: *Chip measurements when oscillators are uncoupled and with reference oscillator 16. a) Example where oscillators 1-8 are not synchronized to oscillator 16 while having a frequency mismatch around 30 kHz. b) Standard deviation of spin signals measured over 1000 cycles and 100 trials. c) Trial instance where all oscillators are synchronized with the reference.*

Overall, these experiments indicate that:

1. Oscillators 1-8 are synchronized together with frequency f_{1-8}.
2. Oscillators 9-16 are synchronized together with frequency f_{9-16}.
3. There is a frequency mismatch of 2.5% between the two groups of oscillators.
4. There are parasitic connections in the chip that connect the two groups of oscillators and are strong enough to provoke global phase locking.

The undesired coupling could originate from the substrate, the metal fillings, and the non-zero synaptic capacitances. While it is difficult to quantify these parasitic connections, we can still evaluate their impact on SKONN operation by studying two-coupled oscillators. To mitigate the frequency mismatches, we activated the calibration circuits and added more load capacitances

to oscillators 9-16 to slow them down. We obtained the best frequency matching by adding 100 fF to oscillators 9-16, and 93.3 fF to oscillators 1-8, for a final matched frequency of 970 kHz as shown in Fig.3.21.

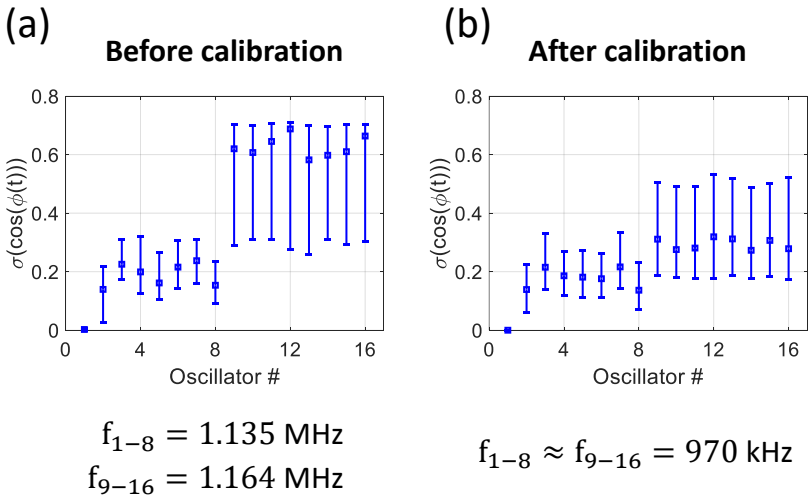


Figure 3.21: *Standard deviation of spin signals measured with 100 trials before a) and after b) calibration.*

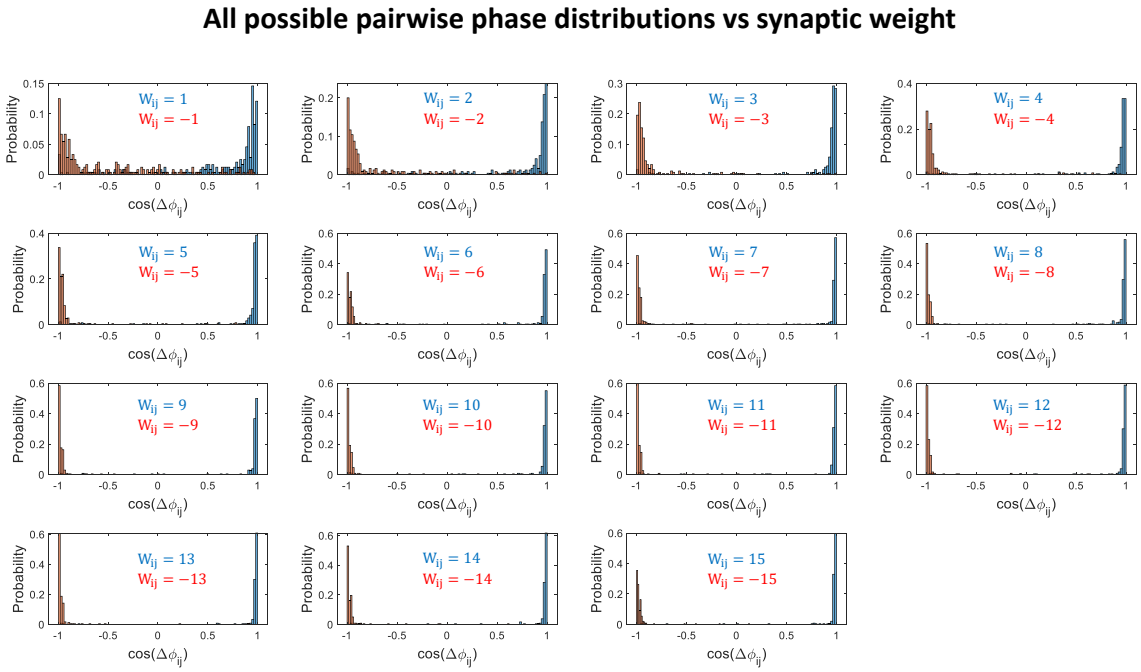


Figure 3.22: *Phase distribution vs synaptic weight. Data are mean phases (averaged for 1000 cycles and two trials) for each one of the 120 possible recurrent weights $W_{ij} = W_{ji}$.*

Synaptic Array and Undesired Coupling

We characterized each pair of synapses (S_{ij}, S_{ji}) with $i \neq j$ by varying the weight value from -15 to +15 and measuring the resulting phase. Fig.3.22 shows the distribution of all 16x15/2=120 possible pairs of phases (run twice and averaged for 1000 cycles) for each synaptic weight. It appears that weak coupling such that $|W_{ij}| < 5$ does not always induce phase fixed points of 0 or π, but also some intermediate phase values. This was expected from the previous experiment that highlighted possible weak connections within the chip. However, these parasitics are on average overcome for $|W_{ij}| \geq 5$ as the phase distributions become very sharp around 0 and π.

During this experiment, we further monitored wrong phase states, i.e. errors with respect to the theoretical phase fixed points (eq.3.5). For instance, if $W_{ij} = -10$ (resp. +10) and $\cos\Delta\phi_{ij} > 0$ (resp. ≤ 0), then it is considered as an error and qualified as *false positive* (resp. *false negative*). Fig.3.23a shows the false negative map when synapses are programmed with positive weights and represented on the synaptic array. The left-hand plot reveals that oscillators 2 and 8 are never in phase for all positive weights and trials. This means that either synapses S_{2-8} and/or S_{8-2} are dysfunctioning, or there is a strong parasitic connection between the two oscillators acting as a negative weight. If the latter is true, then the equivalent capacitance of the parasitic connection is larger than 37.5 fF which corresponds to the maximum weight amplitude $|W_{ij}| = 15$. There is the exact same situation for oscillators 9 and 15. Similarly, oscillators 3 and 7 also suffer from a false negative parasitic connection but to a lower extent as $W_{3-7} = W_{7-3} \geq 10$ compensate the undesired connection (right-hand side of Fig.3.23a). The wrong dynamics of oscillators (2,8) and (9,15) are shown in Fig.3.23b. They are also less stable compared to the wished dynamics of other oscillators such as (3,8) in Fig.3.23c). False positive weights were detected too between oscillators (7,9) and (9,10) as shown in Fig.3.24. We count two times fewer occurrences compared to false negative weights but still represent 50% of wrong dynamics for oscillators (9,10). Overall, for all the tested die samples, we found that oscillators 2,8,9, and 15 suffer from strong undesired coupling, while 3,7,10, and 14 also have non-negligeable parasitic couplings. They are resumed in Table 3.5.

Oscillator pairs	(2,8)	(9,15)	(7,9)	(9,10)	(3,7)
Negative coupling	strong	strong			weak
Positive coupling			medium	medium	

Table 3.5: *Undesired Coupling in the IC*

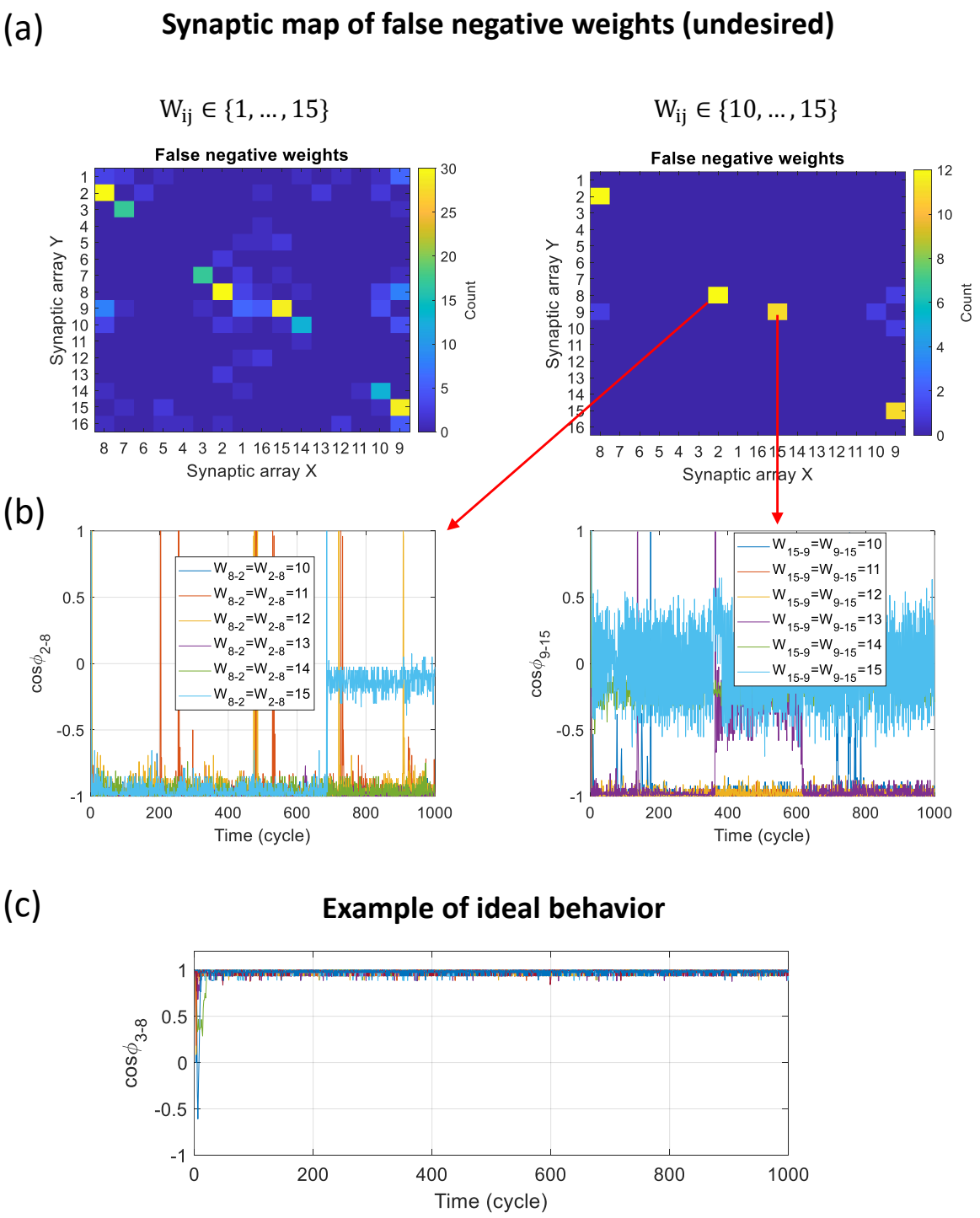


Figure 3.23: *a) Synaptic error map when weights are positive. A false negative weight between oscillators i and j indicates that an incorrect phase state* $\cos\Delta\phi_{ij} < 0$ *is measured (averaged over 1000 cycles) instead of having* $\cos\Delta\phi_{ij} \geq 0$. *b) Incorrect dynamics between oscillators (2,8) and (9,15). c) Example of desired dynamics between oscillators 3 and 8.*

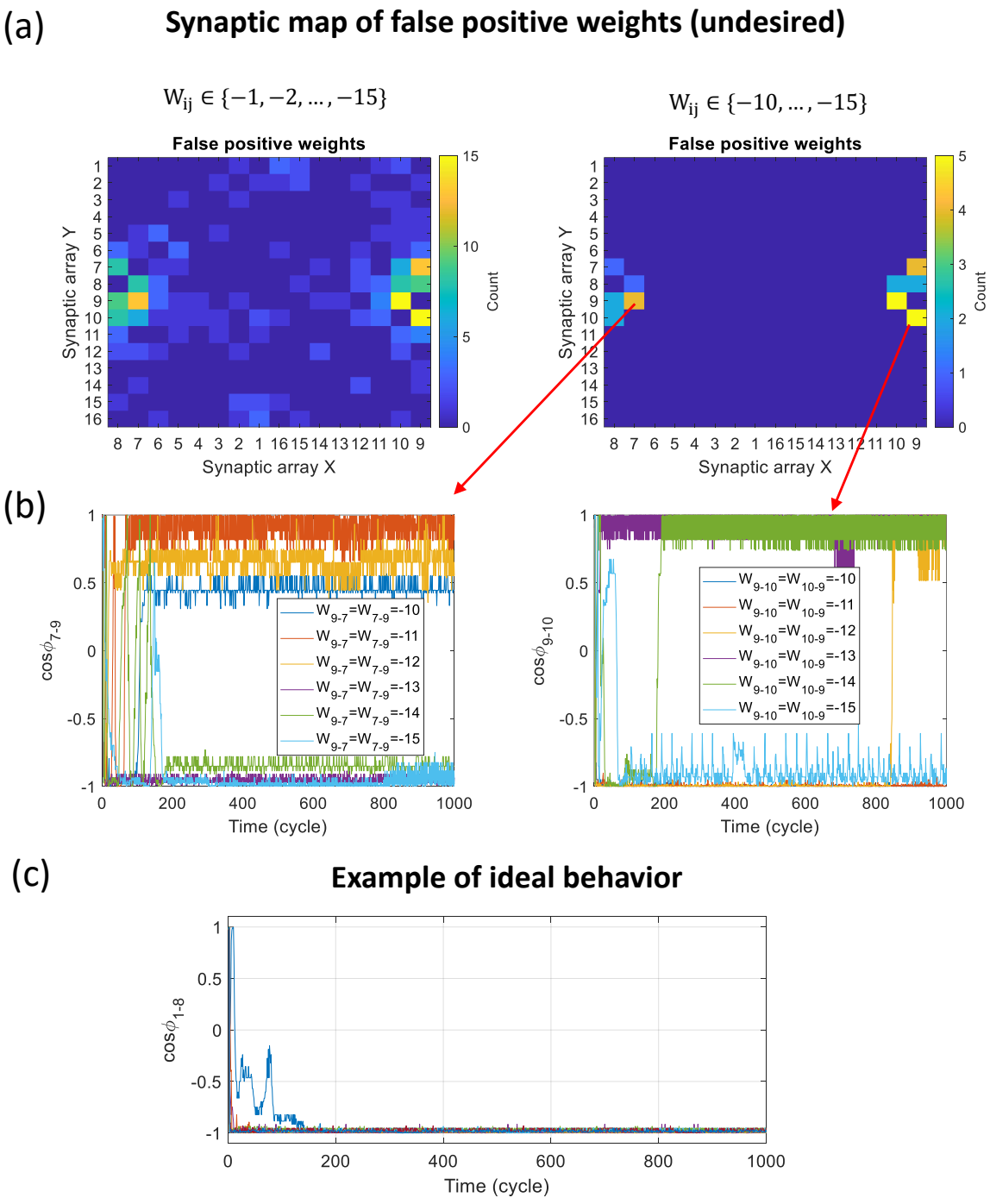


Figure 3.24: *a) Synaptic error map when weights are negative. A false positive weight between oscillators i and j means that an incorrect phase state $\cos\Delta\phi_{ij} \geq 0$ is measured (averaged over 1000 cycles) instead of having $\cos\Delta\phi_{ij} < 0$. b) Incorrect dynamics between oscillators (7,9) and (9,10). c) Example of desired dynamics between oscillators 1 and 8.*

Consider now that the undesired strong coupling is indeed capacitive to study any possible implication. We saw previously that strong capacitive coupling must provoke large voltage jumps at the oscillating input node, as seen in Fig.3.6d on the PCB with $C_{ij} = 10\%C_L$. In this example, the measured frequency reduction was -34%. For two coupled SKONN oscillators with recurrent weights, the frequency at steady state is indeed reduced as the effective voltage jump dV increases the overall analog amplitude ΔV. Hence, using eq.3.14, the resulting frequency change for the two-coupled oscillators is expressed as:

$$\frac{\Delta f_{osc}}{f_{osc}} = -\frac{dV}{\Delta V} = -\frac{C_{ij}}{C_L}\frac{V_{DD}}{\Delta V} \tag{3.19}$$

Under the current assumption, we also saw that the false negative weight between oscillators (2,8) must be larger than 37.5 fF. Considering an undesired $C_{2-8} = 40$ fF and using the post-layout values $C_L = 520$ fF and $\Delta V = 140$ mV, it leads to $\Delta f_{osc}/f_{osc} = -66\%$. Although we cannot yet draw any conclusion, this frequency reduction is in agreement with the +80% increase in bias current we had to inject to reach a frequency close to the nominal case $f_{osc} \approx 1.3$ MHz. The other way around, considering $\Delta f_{osc}/f_{osc} = -80\%$ induces a parasitic capacitance $C_{2-8} = 48.5$ fF. In such a scenario, oscillators (2,8) would "impose" their frequency to the group of oscillators (1,2,...,8) as we previously saw that they are synchronized in a free regime (Fig.3.21a). The same comment applies to oscillators (9,15) that would sort of drive the oscillators (9,10,...16) and reach a common frequency.

We are not sure about the causes of such coupling but one of our hypotheses is that the metal-filling could create parasitic connections at lower metal levels. However, it is striking how synapses S_{8-2} and S_{9-15} are symmetric in the synaptic array. The same remark applies to S_{7-9} and S_{10-9}. We asked for the metal-filling layout views from IMEC to further investigate this issue.

Neuron to Oscillator Mapping

Since there are undesired connections between some oscillators, how to assign neurons to oscillators in the chip impacts the performance and can be quantified by varying n_p, the bijective application that assigns every neuron to a unique physical oscillator. We run experiments for random networks of various densities with $W_{ij} = W_{ji} = -5$ while increasing the ONN size and assigning neurons to physical oscillators according to n_p. Fig.3.25a shows the standard deviation of spin signals (eq.3.18) when n_p is the identity function. We notice that there is a clear instability peak for $N \geq 9$. In addition to the undesired couplings (Table 3.5), groups 1-8 and 9-16 are at different locations in the chip, thus it is plausible they cause some kind of asymmetry in performances. Looking at the chip floorplan (Fig.3.25b), it is clear that the identity neuron assignment induces an asymmetric synaptic assignment with a synaptic block discontinuity when $N = 9$.

Consider now the permutation $n_p(1,..,16) = (4,13,5,12,3,14,6,11,2,15,7,10,1,16,8,9)$. Its corresponding synaptic assignment is illustrated in Fig.3.25b. With this permutation, the synaptic assignment is symmetric and the resulting dynamics are more stable for $N \geq 9$, although the case $N = 4$ is worse than with the previous identity assignment. We are not sure about what causes such discrepancy between the two permutations, but we think the main reason is that oscillators with strong undesired couplings (Table 3.5) are assigned only for $N \geq 9$.

Whereas the first permutation used oscillators 2 and 8 for $N \leq 8$. We also thought a possible explanation could be mismatches in output line lengths. For instance, there is an asymmetry in output line lengths between synapses S_{1-9} and S_{9-1} as $d_{9-1} \approx 1.5\,d_{1-9}$ (Fig.). Similarly, the second largest asymmetry is between S_{8-16} and S_{16-8} as $d_{8-16} \approx 1.5\,d_{16-8}$. With the second proposed permutation, these two mismatched lines are used only for $N \geq 12$ and could explain the peak of instability measured at $N = 12$. However, two-oscillator experiments with oscillators (1,9) and (8,16) did not confirm this hypothesis as the dynamics were comparable to other oscillator pairs.

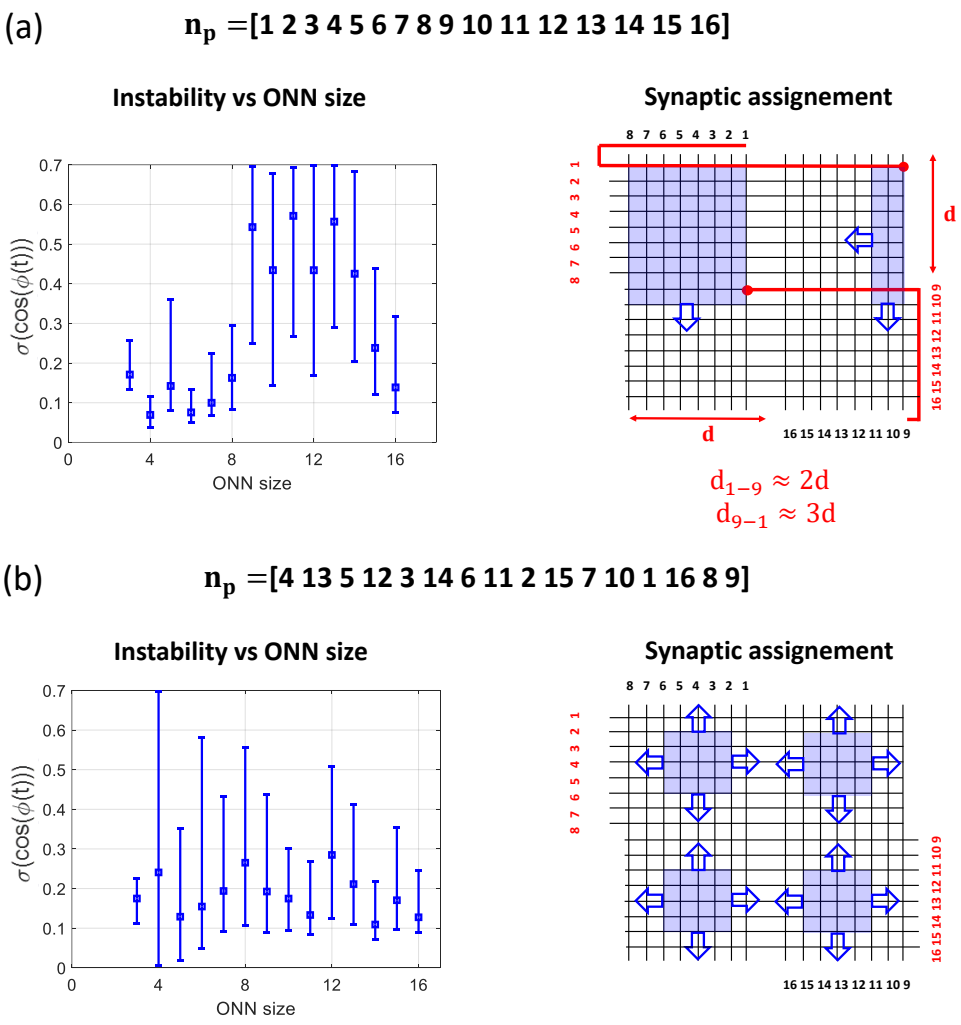


Figure 3.25: *a) Experiments with the identity oscillator assignment n_p. The left-hand plot shows the metric $\sigma(\cos\phi(t))$ computed for 10 trials per graph. Each size N has $N(N-3)/2+1$ different graphs (from sparse to fully connected) with $W_{ij} = W_{ji} = -5$. The right-hand side illustration represents the corresponding synaptic assignment when increasing the size N. b) Same experiment with a different synaptic assignment n_p that leads to the best results.*

3.7.3 Feed-Forward Network with SKONN

Propagating the information in a feed-forward manner is helpful in some applications that require driving neurons. For instance, when training an ONN with the equilibrium propagation method [130, 131], one must nudge the output oscillators toward the desired value which is challenging to obtain with recurrent synapses. Instead, teaching oscillators could drive the output oscillators using feed-forward connections without being impacted during the learning phase.

To demonstrate SKONN's feed-forward ability, we program the ASIC to solve a simple 2-input XOR operation. Inspired by the Parametron built by Goto in the 1950s [79], we use a 3-input SKONN neuron as a majority gate

$$\phi_M = (\phi_X.\phi_Y) + (\phi_X.\phi_Z) + (\phi_Y.\phi_Z) \tag{3.20}$$

where ϕ_X, ϕ_Y, $\phi_Z \in \{0°; 180°\}$ are the input binary phases thought as Boolean variables; i.e. ϕ_M is true when ϕ_M=180°. Interestingly, SKONN's odd-degree property (Proposition 4.2.1 derived in Chapter 4)) ensures that ϕ_M is binary when its inputs are also binary.

The XOR(X,Y) circuit is implemented by writing the XOR boolean expression $\phi_{XOR} = (\phi_X.\overline{\phi_Y}) + (\phi_Y.\overline{\phi_X})$. Table 3.6 summarizes the results for the 4 possible inputs W_{XZ} and W_{YZ}. By assigning the bit 0 when -90°$\leq \phi_i \leq$90° and 1 otherwise, it can be seen that the proposed network computes XOR(X,Y) in a feed-forward manner.

W_{XZ}	W_{YZ}	ϕ_X	ϕ_Y	ϕ_{XOR}	XOR(X,Y)
+1	+1	359°	56°	342°	0
+1	-1	358°	195°	161°	1
-1	+1	203°	0°	187°	1
-1	-1	199°	203°	344°	0

Table 3.6: *Solving XOR(X,Y) with SKONN IC*

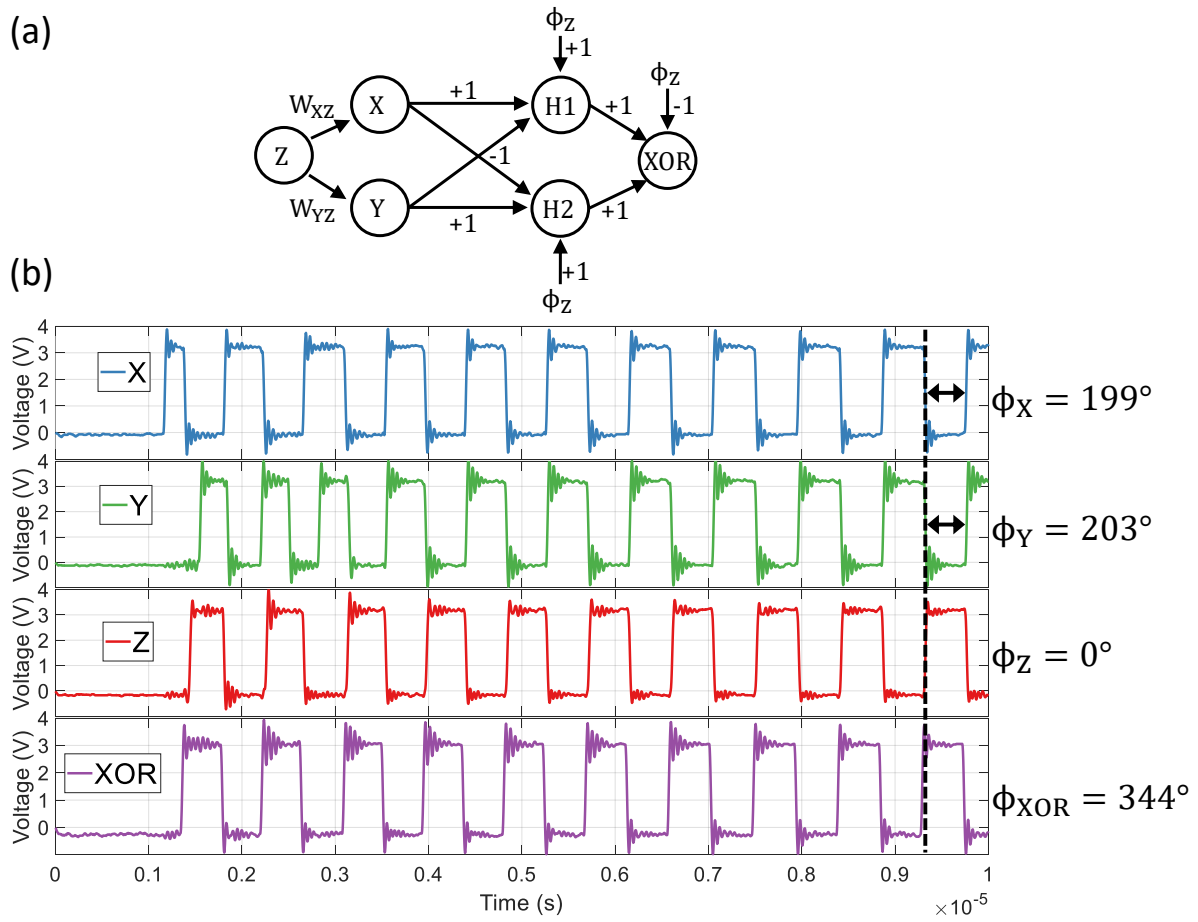


Figure 3.26: *a) XOR(X,Y) circuit using SKONN and feed-forward synapses. Neuron Z is the reference oscillator and can be thought of as the neurons' bias, similar to perceptrons. The weights W_{XZ} and W_{YZ} are the inputs and set the initial phases ϕ_X and ϕ_Y. b) ASIC experimental results when $W_{XZ} = W_{XZ} = -1$ which set $\phi_X \approx \phi_Y \approx 180°$ and corresponds to the boolean case where X=Y=1.*

3.8 Discussions

Here we resume several implementation challenges that we faced while designing this first proof-of-concept, and discuss possible improvements:

1. **Low power, low mismatch vs. circuit area**. Due to the low-power specification, we had to limit the current of each block which induced some other challenges. The main problem we faced was oscillator-to-oscillator mismatches. Most mismatch contributions come from the shaper stage, i.e. the circuit that commands the charge and discharge of the load capacitor. Ideally, for this circuit, transistors should be in strong inversion (SI) to minimize current mismatches (eq.3.14) and SI is obtained for large gate voltages. For a given transistor length, this means that the bias current has to increase. As we did not wish to increase the current consumption, the other way of reaching SI for a fixed current was to increase the transistor length. Thus, to reach our mismatch specifications and keep currents down to 200 nA, we had to use very long lengths in the shaper block up to 7 μm.

2. **Low-power vs. low-energy**. Although a low-energy system is not necessarily low-power (and vice versa), we now think that it is probably simpler to meet low-energy specifications with SKONN. By allowing an increase of current consumption in the shaper block, the power goes up but the energy loss remains quasi-constant as the oscillating frequency is proportional to the current (eq.3.13). Of course, increasing the frequency could bring other issues such as undesired synaptic delays, or limited precision for the phase measurement, but it has the potential to significantly reduce the silicon area.

3. **Circuit design**. There is room for improvement at the circuit level, especially concerning the oscillator circuit. While we chose a regenerative comparator circuit for the hysteresis block that is robust against process variations, simpler circuits could probably be used. We think that process, temperature, and supply voltage (PVT) variations are not too problematic for ONN computation as long as oscillators remain matched. Hence, simpler PVT-dependent circuits could be advantageously used for SKONN design.

4. **Oscillator isolation** We also learned from the characterization that neighbor oscillators are synchronized by default, highlighting undesired couplings (Huygens effect). While we are still investigating this issue, it is likely that oscillators exchange current through the substrate. Hence, using deep-NWELL for the PMOS transistors could enhance the oscillator isolation. Another probable cause is undesired synaptic connections due to parasitic capacitors in the chip. This is probably the main drawback of SKONN architecture.

For future work and to further assess SKONN scalability, it would be interesting to study the performance of a modular SKONN system having a locally dense, globally sparse connectivity in a single chip by assembling tiles (similar to chips in [139] or [90]) or connect several chips together. Although the first option is probably the safest, the last one would be possible by including digital input pads in series with synapses that connect oscillators, and digital output pads to propagate the phase information to other chips.

3.9 Conclusion

In this chapter, we proposed a mixed-signal ONN architecture (SKONN) that computes in the analog domain and propagates the phase information digitally. The combination of relaxation oscillators and synaptic capacitors led to simple and controllable phase dynamics, yet having interesting computational properties when the input oscillation has a triangular shape. We showed that SKONN dynamics are closely linked to the Kuramoto model, where sinusoidal interactions are replaced by a *saturated* version, i.e. a square function. Importantly, there is a one-to-one linear correspondence between conceptual synaptic weights and synaptic capacitance values. Moreover, SKONN allows feedforward ONNs that we showcased with the XOR example. After demonstrating SKONN on a 9-neuron PCB, we further designed a 16-neuron SKONN chip with all-to-all connectivity and 5 bits of synaptic resolution in a TSMC 65nm CMOS technology. In the next chapter, we further explore the SKONN model and test the chip to solve NP-hard combinatorial optimization problems.

CHAPTER 4

ONN FOR NP-HARD COMBINATORIAL OPTIMIZATION

PHASE-based ONNs naturally minimize an energy function in continuous time without any external control or algorithm. Moreover, due to their network structure, ONNs can be seen as graphs which are at the heart of combinatorial optimization problems (COPs). The nondeterministic polynomial time (NP)-hard COPs are deeply rooted in computer science as they are essential for many crucial problems in transportation, scheduling, resource allocation, engineering, etc. and no known algorithm can find an optimal solution in polynomial time. Hence, during the last five decades, scientists have designed heuristic algorithms that provide near-optimal solutions in polynomial time to daily problems. As many remarkable heuristic algorithms like simulated annealing (SA), particle swarm optimization (PSO), or genetic algorithms (GA) are inspired by natural sciences, it brings back an old question: why not solve COPs with physical objects that naturally perform some optimization in continuous-time? As analog computing suffers from a lack of programmability, variability, and noise, one can question the quest for physics-based computers. However, analog dynamical systems have the intrinsic advantage of being driven by nature, i.e. they evolve in continuous time according to physical laws. Hence, when the physical system is carefully designed to solve a particular problem instance, it does not need any algorithm and instruction fetch between a memory and a processor, which in turn can speed up computations and save energy.

This chapter aims to explore how the ONN intrinsic optimization mechanisms can also be harnessed to solve NP-hard COPs. The chapter is divided into three main parts. First, I focus on SKONN and show that it is closely linked to the best-in-class approximation algorithm for the Max-cut problem developed by Goemans and Williamson. I further evaluate SKONN's performance for solving Max-cut using the IC proof-of-concept and large-scale simulations. In the second part, I explore how Kuramoto-ONNs with sinusoidal interactions can naturally approximate the traveling salesman problem (TSP) by letting the phases relax to analog values. Finally, I explore how ONNs can escape sub-optimal solutions without injecting noise but rather *going uphill* in the energy landscape. This study led to the development of Lagrangian-ONNs (*LagONN*) that I specifically designed to solve one of the most fundamental NP-hard problems: Boolean satisfiability.

4.1 Introduction

4.1.1 Background on Combinatorial Optimization

Combinatorial Problems

Combinatorial problems consist of "finding groupings, orderings, or assignments of a discrete set of objects which satisfy certain conditions or constraints" [55]. Combinatorial problems can be stated as:

1. **decision problems**, where one needs to answer a yes-no question.
2. **search problems** that require finding a solution satisfying some conditions given by the problem instance.
3. **optimization problems** (COPs) that consist of finding a solution meeting the conditions and minimizing (or maximizing) a cost function.

For instance, a decision problem for the traveling salesman problem (TSP) would be asking the question "Is there a tour visiting every city once with distance $\leq D$"; whereas its search version would be to find or not such tour. Finally, its optimization version is to find the shortest tour. The usual method for solving combinatorial problems is to run an algorithm or program, that sequentially operates on an input string of size N that encodes the input problem with some alphabet. Formal definitions of programs using Turing machines can be found in [202] but are beyond the scope of this study.

Time Complexity and NP problems

We are interested in an important algorithmic property that is its *time complexity*. Applied to combinatorial problems, it consists of the maximum time $T(N)$ required by the program to answer the decision problem (the yes-no question). There are problems solvable in *polynomial time* as $T(N) \leq p(N)$ with p a polynomial function. Finding the maximum value of a list, matrix multiplication, or the shortest path problem are examples of polynomial-time problems that we say belong to the class P [55]. However, there are much harder problems whose time complexity cannot be bounded by some polynomial and are said to be *intractable* [202]. TSP is one example of intractable problems as the search space grows exponentially (there are $(N-1)!/2$ possible tours). Imagine if we had access to some kind of oracle that would *guess* the answer to the TSP decision problem by providing a tentative tour. Having this tour, it is then possible to verify in polynomial time that its length is indeed $\leq D$. Because of the oracle, this kind of program is called *nondeterministic*. Moreover, as the verification of the guessed solution can be done in polynomial time, the algorithm is said to run in *nondeterministic polynomial time* (NP). Informally, we say that the problems that can be solved by nondeterministic polynomial time algorithms belong to the class NP [202].

SAT, NP-complete and NP-hard problems

The theory of NP-completeness provides powerful methods to study combinatorial problems stated in their decision version [202]. One of the most important tools is *polynomial transformation* or *reduction*. Informally, we say problem A transforms to problem B when there is a polynomial time algorithm that transforms any instance of A into an instance of B, denoted $A \propto B$. Importantly, it is transitive as if $A \propto B$ and $B \propto C$, then $A \propto C$ [202]. The foundation for this theory is the Theorem from Cook [203] which states that all problems in NP can be transformed into a single decision problem: Boolean Satisfiability (SAT). SAT can be described as follows. Consider a Boolean formula f_B :

$$f_B = C_1 \bigwedge C_2 \bigwedge ... \bigwedge C_{M-1} \bigwedge C_M \tag{4.1}$$

With each clause C_m expressed as the disjunction of *literals* such as $C_m = l_1^m \bigvee l_2^m ... \bigvee l_k^m$, and where literals consist of Boolean variables $l_j^m \in \{x_1, ..., x_N, \overline{x_1}, ..., \overline{x_N}\}$. The SAT problem consists of answering the question: "Is f_B satisfiable?"

Cook's Theorem points out an important property for SAT: it is *at least as hard* as any problem in NP. SAT was then qualified as *NP-complete*. One of the implications is that if SAT can be solved in polynomial time, then all problems in NP can be solved in polynomial time [202]. Using the transitivity property, many other decision problems have been proved NP-complete such as TSP, 0-1 integer programming, Knapsack, Max-cut, etc., and 21 of them are listed in seminal work from Karp [54].

Informally, an *NP-hard problem* is a problem for which any NP-complete problem can be transformed to it [202]. It is then *as hard* as all the NP-complete problems. For a combinatorial problem, the search and optimization versions are as hard as the decision version because finding a solution for the former answers the "yes-no" question. Hence, the optimization versions of the combinatorial problems (COPs) considered throughout this thesis are NP-hard. A polynomial transformation between the TSP decision problem and its optimization variant would be to loop the decision problem over decreasing D values until finding the optimal tour. Note that compared to NP-complete problems, an NP-hard problem does not necessarily belong to NP. This is the case for TSP in its optimization form because if someone provides a tour and claims that it is the optimal one, no polynomial-time algorithm can check if it is indeed the best solution.

4.1.2 ONN for solving NP-hard COPs

After the seminal works from Hopfield [119] and Tank [56], researchers have thought about solving NP-hard COPs using coupled oscillators that would implement a kind of Hopfield neurons. Endo [99] proposed to replace the neural activation with van der Pol oscillators having smooth transient dynamics and binary steady state (ON/OFF) for reading out the COP solution. For graph coloring, Wu [100] designed a coupled oscillator system whose phases at steady-state correspond to colors in the corresponding graph. Then, Hoppensteadt and Izhikevich [68] have further proposed an ONN implementation using a network of phase-locked-loop (PLL) behaving as an analog Hopfield neural network (HNN) in the phase domain.

More recently, using ONN for solving COPs was brought up to date thanks to the seminal

work from Wang et al. [162] who formally linked ONN to the Ising model. The Ising model was named after the physicist Ernst Ising who studied magnetic moments (or spins $S_i = \pm 1$) in ferromagnetic materials [51] whose Hamiltonian is expressed as:

$$H = -\sum_i \sum_j J_{ij} S_i S_j - \sum_i h_i S_i \tag{4.2}$$

where J_{ij} are interaction coefficients between spins S_i and S_j, and h_i is the external field applied to spin S_i. Essentially, both analog HNN and ONNs are relaxations of the Ising model in the sense that their energy functions are equal to the Ising Hamiltonian H when variables or phases are binary. As finding the ground states of the Ising Hamiltonian is an NP-hard problem [53], there is a great interest in designing oscillatory-based Ising machines (OIM) to solve any other NP problem using coupled oscillators. The OIM development has been mainly driven by technological progress as CMOS technology enables the integration of various OIMs [74, 90, 138] with up to 1968 oscillators demonstrated on-chip [76], and beyond-CMOS devices such as volatile memristors [86] or spintronic devices [87] promise a compact oscillator implementation.

On the conceptual side, Erementchouk et al. [204] have recently shown that ONNs can solve the NP-hard Max-cut problem even without binary phases and are linked to classical approximation algorithms for Max-cut. Landge et al. [113] proposed to solve the N-city TSP with N oscillators by encoding the TSP tour in the oscillator phase permutation. This extended the work from Duane [205] who first proposed using an array of N^2 oscillators for solving TSP, similar to the original Hopfield-Tank network [56]. Moreover, researchers have pushed the OIM boundaries and extended the binary phase encoding to multiple phase values for solving other NP-hard COPs such as TSP or Max-K-cut (Max-cut with K possible spin values) [206]. For Max-K-cut, authors in [206] propose extending real synaptic weights to complex weights $|W_{kl}|e^{if(\phi_k - \phi_l)}$ by adding a custom synaptic function $f(\phi_k - \phi_l)$, so that the ONN energy is minimized when two coupled oscillators are in different subsets defined by $f(\phi_k - \phi_l)$. They further constrain the ONN phases by injecting a harmonic signal at frequency $K f_{osc}$ to favor the K desired phases.

In [207], the authors further extended the OIM concept to hypergraphs involving high-order interactions between the variables, whereas the Ising model has only spin-to-spin quadratic interactions. This extension enables the direct mapping of many NP-hard problems such as Satisfiability to a coupled oscillator system without adding auxiliary oscillators. However, the practical implementation is more complicated as it involves hyper-edges, i.e. a simultaneous connection between more than two nodes.

In this chapter, we first study SKONN for solving Max-cut in a *free* regime, i.e. the phases can settle to any value. Then, inspired by the work from Landge et al. [113], we explore how Kuramoto-ONN can solve small-scale TSP by encoding the TSP tour in the ONN phase permutation. Finally, we harness ONN high-order interactions to solve the maximum-3-satisfiability problem (Max-3-SAT) by designing custom Lagrangian dynamics to escape local minima.

4.1.3 Contributions

The findings of this chapter are:

1. SKONN is closely related to the best-in-class Goemans-Williamson algorithm (GW) for solving the NP-hard Max-cut problem. Moreover, large-scale simulations using the G-SET benchmark indicate that SKONN is as good as GW while having a runtime that seems to scale only as $O(\log N)$.
2. Kuramoto-ONNs in free regime can find TSP tours with distance $\leq$1.4x the optimal distance for $N \leq 40$ cities.
3. A Lagrangian-ONN (*LagONN*) that combines both gradient descent and ascent can solve large-scale Max-3-SAT instances (simulations up to 200 variables and 860 clauses) by taking advantage of high-order interactions between oscillators.

4.2 ONN for Solving the NP-hard Max-cut Problem

Given a graph with a set V of N vertices connected by weighted edges $W_{ij} = W_{ji}$, the Max-cut problem consists in *cutting* the graph in two complementary subsets of vertices V_1 and V_2 such that the sum of weights between V_1 and V_2 is maximum. The Max-cut problem can be formulated as follows [111]:

$$\begin{aligned} &\text{Max} \frac{1}{2} \sum_{i,j} W_{ij}(1 - S_i S_j) \\ &\text{subject to:} \\ &S_i \in \{-1, +1\} \; \forall i \in V \end{aligned} \tag{4.3}$$

Solving the Max-cut problem is NP-hard and the best-known approximation algorithm is the semidefinite programming (SDP) algorithm found by Goemans and Williamson [111] and denoted GW throughout the paper. By relaxing the binary spins S_i to unit vectors v_i in $\mathbb{R}^N$, GW relaxes the NP-hard Max-cut problem to an SDP convex problem for which optimality can be found in polynomial time:

$$\begin{aligned} &\text{Max} \frac{1}{2} \sum_{i,j} W_{ij}(1 - v_i.v_j) \\ &\text{subject to:} \\ &v_i \in \mathbb{R}^N \\ &|v_i| = 1 \; \forall i \in V \end{aligned} \tag{4.4}$$

To compute the cut, the vectors are finally assigned to binary spins by splitting in two the N-dimensional sphere with a random hyperplan. Repeating this final rounding step provides a cut whose expectation is:

$$E[cut] = \frac{1}{2\pi} \sum_{i,j} W_{ij} \arccos(v_i.v_j) > 0.878 \, \text{Max-cut} \tag{4.5}$$

However, due to the high dimension of the problem relaxation ($\mathbb{R}^N$), GW is costly for large instances [58, 112] and alternative approaches using physical systems such as Quantum Annealers [208], coherent Ising machines [58], memristors [209] or coupled oscillators are being investigated [69, 76, 86, 139].

4.2.1 The Ising Approach

One of the most studied formalisms applied to ONN is from Ising which was initially derived to study magnetism in materials [51]. Given interaction coefficients $J_{ij} \in \mathbb{R}$ between particles that can have two spins $S_i \in \{-1;+1\}$, the particles relax to a state that minimizes the Ising Hamiltonian (we skip the external fields for simplicity):

$$H = -\frac{1}{2}\sum_{i,j}^{N} J_{ij} S_i S_j \tag{4.6}$$

Thanks to Lucas' seminal work [53], all Karp's 21 NP-complete problems can be mapped to the Ising formalism and the solutions can be approximated by any physical machine that minimizes the Ising Hamiltonian (eq.4.6). If SKONN phases take binary values $\phi_i = (1 - S_i)\pi/2 \in \{0, \pi\}$, its Lyapunov function (eq.3.8) becomes:

$$\begin{aligned} E =& \frac{\beta_0}{T}\sum_{i,j}^{N} Q_{ij}\,\text{triangle}\left(\frac{\pi}{2}(S_j - S_i)\right) \\ =& -\frac{\pi\beta_0}{2T}\sum_{i,j}^{N} Q_{ij} S_i S_j \\ \propto& H \end{aligned} \tag{4.7}$$

Each synaptic current spike can be thought of as a downward step (due to eq.3.11) in the energy landscape (eq.3.8) which corresponds to the Ising Hamiltonian (eq.4.7) if the final phases are binary. However, having binary phases is not guaranteed in general. To force phase binarization, it is common practice to inject into the oscillators a SHIL periodic signal at twice the oscillating frequency [69, 138] and described in Appendix B.5 for SKONN.

The Max-cut problem can easily be mapped to an oscillatory Ising machine (OIM) with spins corresponding to the graph vertices by setting $J_{ij} = -W_{ij}$ with W_{ij} the graph weights [53]. Then, the OIM performs the following minimization which is equivalent to the Max-cut (eq.4.4):

$$\begin{aligned} &\text{Min}\, H = \text{Max}\left(-\frac{1}{2}\sum_{i,j} W_{ij} S_i S_j\right) \\ &\text{subject to:} \\ &S_i \in \{-1,+1\}\ \forall i \in V \end{aligned} \tag{4.8}$$

The general strategy is to **1)** map the graph to the OIM, **2)** start the OIM while ramping up a 2-SHIL signal to binarize the phases, and **3)** read the stable phase state [69]. Forcing phase binarization is common practice as it maps the OIM Lyapunov function to the Ising Hamiltonian (eq.4.7). However, how to binarize is not straightforward. If the injected signal is too strong, it may "freeze" the phases to sub-optimal local minima [69, 137]. Whereas if the signal is too weak, it might increase the OIM computation time. As described next, we rather harness the free SKONN dynamics without SHIL to compute the Max-cut in this chapter.

4.2.2 A Rank-2 Relaxation Approach

Erementchouk et al. [204] have recently shown that the free OIM relaxation can be harnessed to solve the Max-cut problem. Their recent results demonstrate that letting a Kuramoto-ONN settle to analog phase values is equivalent to solving a rank-2 relaxation problem for the NP-hard Max-cut problem. Such phase dynamics are used in the CirCut solver [112]. Similarly to GW, the CirCut algorithm relaxes spins S_i to 2D unit vectors $x_i \in \mathbb{R}^2$ such that $x_i = \big(\cos(\phi_i)\ \sin(\phi_i)\big)$ that can take arbitrary values on the unit circle. The objective of the rank-2 relaxation is:

$$\begin{aligned} &\text{Max}\frac{1}{2}\sum_{i,j} W_{ij}(1 - x_i.x_j) \\ &= \text{Max}\frac{1}{2}\sum_{i,j} W_{ij}(1 - \cos(\phi_i - \phi_j)) \\ &\text{subject to:} \\ &x_i \in \mathbb{R}^2 \\ &|x_i| = 1\ \forall i \in V \end{aligned} \tag{4.9}$$

Then, a rounding procedure produces spins to compute the graph cut. Unfortunately, this rank-2 algorithm cannot guarantee a lower bound on the cut as it remains a non-convex optimization problem. Nevertheless, its accuracy is comparable to the GW algorithm in practical use [112].

4.2.3 SKONN's Approach and Link with GW

In this chapter, we only explore the relaxation approach where we let SKONN settle without forcing binarization. We will see that SKONN's phase binarization property (Proposition 4.2.1) is particularly useful in this case. For the Ising approach, we invite the reader to consult the excellent work from Wang and colleagues [69] as the reported dynamics are equivalent to SKONN's.

Moreover, it is worth mentioning that SKONN's energy landscape can be linked to the GW algorithm restricted to vectors of dimension 2 (GW2), as recently noticed in [210]. Indeed, SKONN's triangular interaction can be written as:

$$\text{triangle}(\phi_i - \phi_j) + \pi/2 = \arccos(x_i.x_j) \tag{4.10}$$

By setting SKONN's synapses as the negative of the graph weights $J_{ij} = -W_{ij}$, the energy becomes:

$$E \propto -\sum_{i,j} W_{ij} \arccos(x_i.x_j) + C \tag{4.11}$$

with C a real constant. Hence, SKONN's energy minimization is equivalent to GW2's maximization task (eq.4.5) as:

$$\text{Min}E = \text{Max}\sum_{ij} W_{ij} \arccos(x_i.x_j) \tag{4.12}$$

$$\begin{aligned} &\text{subject to:} \\ &x_i \in \mathbb{R}^2 \\ &|x_i| = 1\ \forall i \in V \end{aligned} \tag{4.13}$$

4.2.4 Link between the Graph Degree and SKONN Fixed Points

The 3-node case of Fig.4.1a reveals a different distribution of SKONN's phase fixed points compared to Kuramoto. A simple analysis of the dynamics for N oscillators gives a relationship between the connectivity of the network and the phase fixed points. Specifically, the next proposition links the degree of a SKONN neuron, i.e. the number of synaptic inputs, with the value of its phase fixed point.

Proposition 4.1. *Consider a neuron i of degree D, i.e. driven by D neurons j with weighted charges* $Q_{ij} \in \{-q, +q\}$ $q \neq 0$.

1. *If D is odd and* $d\phi_i/dt = 0$*, then there is at least one input neuron j such that* $(\phi_i - \phi_j)$ *is a multiple of* π.
2. *If D is even, then there is at least one* ϕ_i *and one set of input phase* ϕ_j *such that* $d\phi_i/dt = 0$ *and* $\forall j\,(\phi_i - \phi_j)$ *is not a multiple of* π.

The proof is shown in Appendix B.3.

Interestingly, odd-degree neurons will phase-lock in or out of phase with at least one input phase. The odd-degree property will be advantageous for solving some optimization problems (COPs) on graphs as it can prevent the use of SHIL to binarize phases. On another hand, an even number of inputs instead leads to a relaxed scenario where the neuron can settle into an infinite number of phases as seen in the 3-node case of Fig.4.1a. When SKONN has both odd and even numbers of input synapses, we heuristically find that most of the phases tend to binarize as illustrated with the 5-node graph in Fig.4.1b, although some phases (such as ϕ_2) can still converge to fixed points with arbitrary phase values. This aspect will be further discussed when solving larger graphs in section 4.2.7.

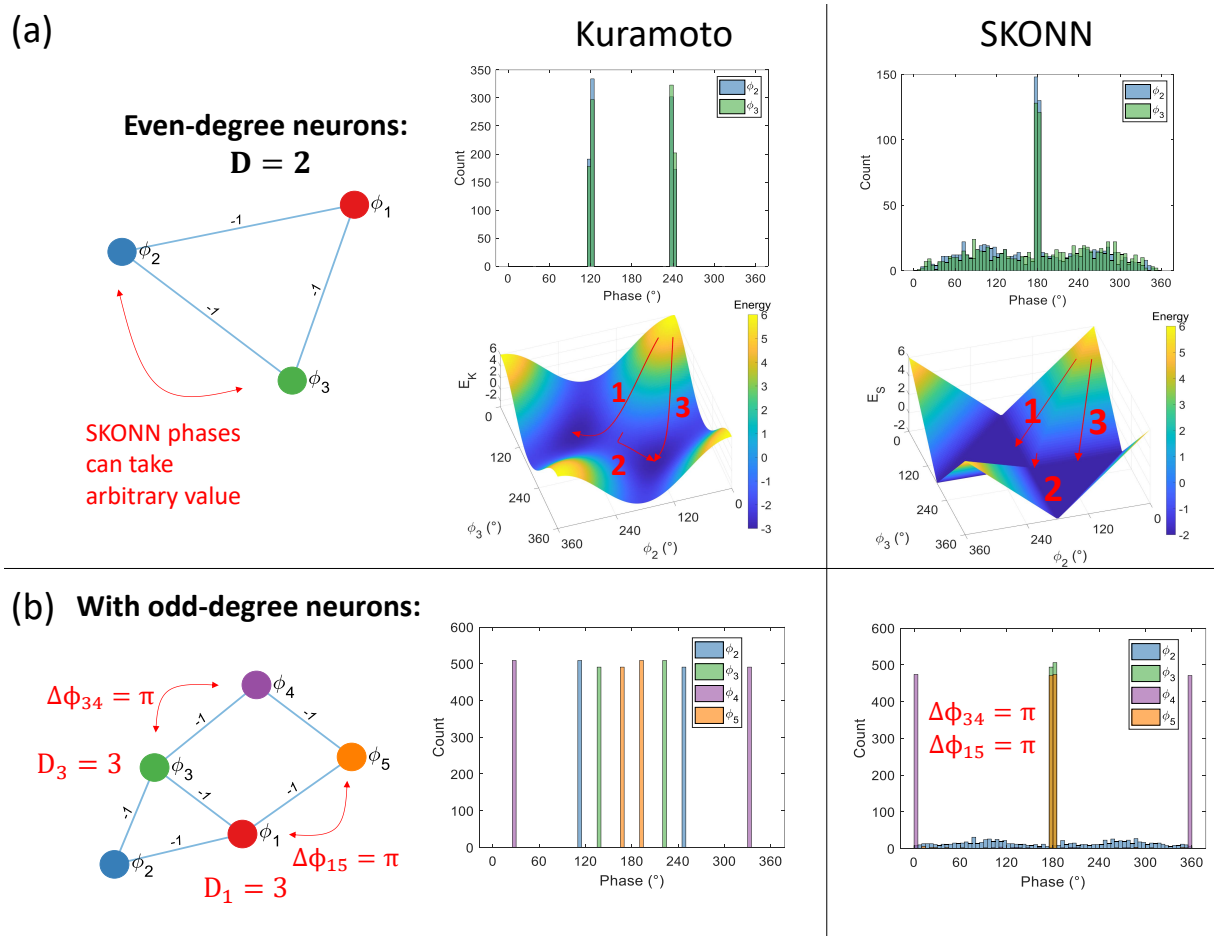


Figure 4.1: *a) Three oscillators coupled by negative weights. The right-hand side shows the Kuramoto and SKONN distributions of the final phases for 1000 random initializations (uniform distribution) and the corresponding Kuramoto and SKONN energy landscapes. The arrows represent three examples of trajectories with various initializations, highlighting Kuramoto's minima and saddle point. In this example, SKONN's energy minima consist of plateaus that are linked to the even-degree property (Proposition 4.1.2). b) Five coupled oscillators where D_1, D_3 and D_2, D_4, D_5 are odd and even, respectively. SKONN tends to binarize phases except for ϕ_2 which can rather converge to any phase value (Proposition 4.1).*

4.2.5 SKONN IC Experimental Results

Fig.4.2 shows the experimental setup to test SKONN IC. An FPGA programs the chip, i.e. loads the graph weights and initializes the phases prior to computation with a uniform distribution $\phi_0 \sim U[0, 2\pi[$. Then, the FPGA reads the phase differences at each oscillation cycle by detecting the rising edges of V_i^{out} with a precision of 5°. The results are then sent to a laptop (i7 Intel core @1.6 GHz and 32 GB of RAM) and post-processed with MATLAB to compute the cut. For each graph, we solve Max-cut by running GW using the CVX solver on MATLAB [211] that we consider as ground truth, i.e. SKONN's cuts are normalized by GW's best cut. We define the accuracy as the ratio cut/cut_{GW}.

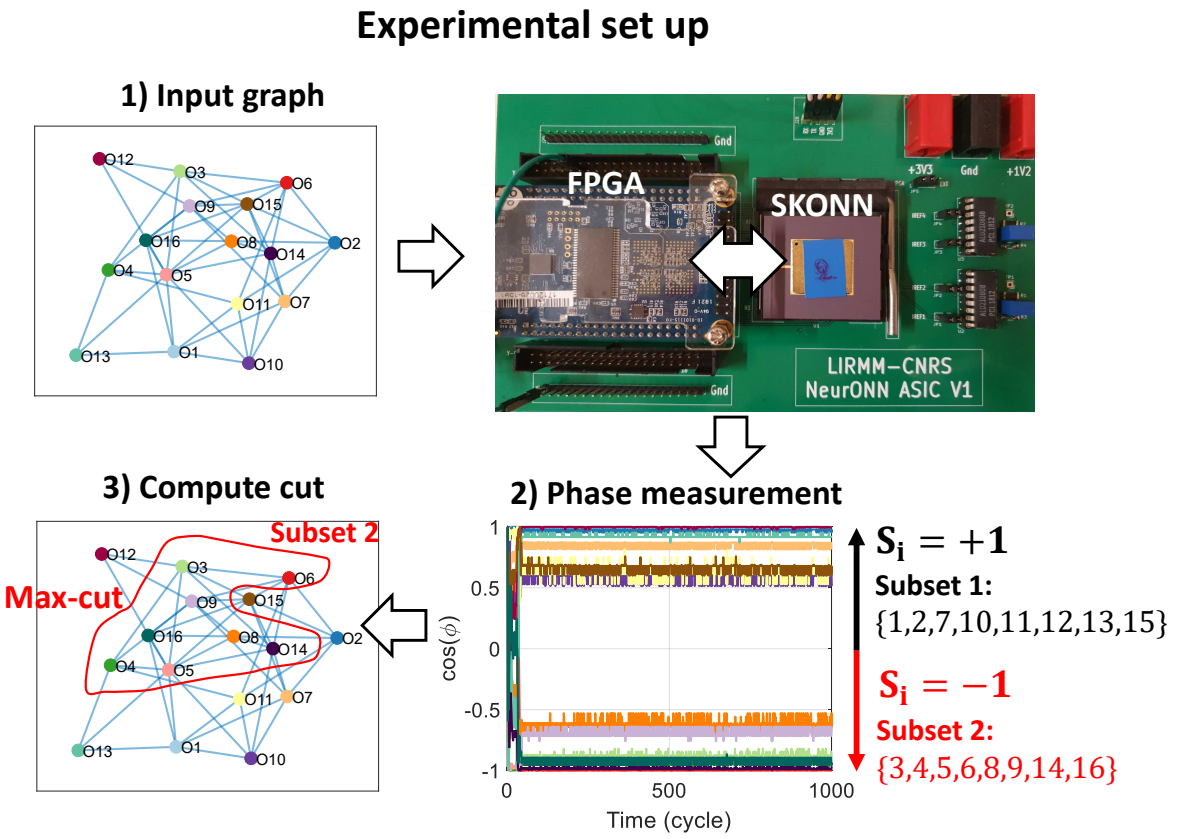


Figure 4.2: *Experimental setup for testing the IC. The FPGA programs the IC and measures the phases that naturally evolve in the analog domain without any external control. After a few tens of oscillation cycles, phases are sampled and assigned to Ising spins.*

Max-cut for Various Graph Sizes and Densities

Here, we compute cuts of random graphs with three different edge densities $d \in \{0.25, 0.5, 0.75\}$, uniform weights $W_{ij} = W_{ji} = +5$, and various sizes N. The graphs are generated such that in expectation, the number of graph edges is $M = dN(N-1)/2$. Data points of Fig.4.3a show the quartiles of cuts for 10 random graphs and 10 trials per graph when the phases are measured after 15 oscillation cycles (15μs). It appears that more than 25% of SKONN's cuts have accuracies higher than 0.95 for all sizes and for the three graph densities, although the accuracy is slightly lower for sparser graphs and for $N \geq 9$. We believe the accuracy drop for $N \geq 9$ could be due to the oscillators having undesired strong couplings found during the characterization (Table 3.5) and assigned for $N \geq 9$.

If phases are measured before 15 oscillation cycles, the accuracy drops as the ONN has not stabilized yet (Fig.4.3b). For longer measurement times, the accuracy is quasi-constant which indicates that the system remains stable when reaching a phase fixed point. Although the heuristic hardware does not always stabilize to binary phases (Fig.4.3c), SKONN favors binary phases (peaks at $\cos(\phi) = \pm 1$) due to its unique energy landscape (eq.3.8) as predicted analytically and confirmed here. In contrast, state-of-the-art ONNs often necessitate the injection of subharmonic signals (SHIL) to binarize the phases and emulate Ising spins [69, 138].

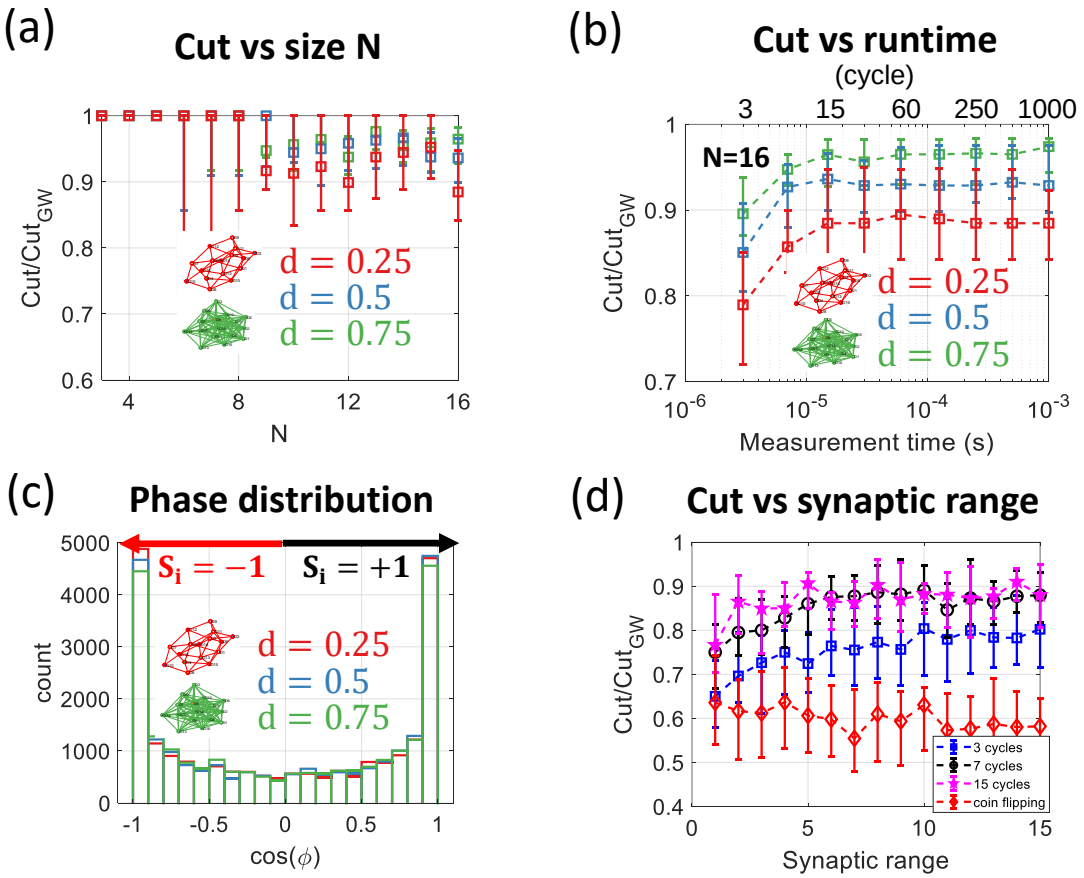


Figure 4.3: *a) Accuracy vs. size N for graph densities $d \in \{0.25, 0.5, 0.75\}$. b) Accuracy vs runtime (measurement time) for N=16. c) Phase distribution measured after 1000 oscillation cycles. c) Accuracy vs synaptic range SR. Graphs have weighted edges taking random values $W_{ij} \in \{1, ..., SR\}$. The star data points show the accuracy obtained after 15 oscillation cycles.*

Max-cut for various weight ranges

We set N=16, d=0.25 and explore random graphs with various weight ranges SR from 1 to 15, i.e. non-zero weights $W_{ij} \in \{1, 2, .., SR\}$ are chosen with probability 1/SR. Fig.4.3d shows the results for 10 random graphs per SR value. These graphs are harder to solve for SKONN as in general, only 25% of cuts have accuracies above 0.9. It also appears that SKONN's accuracy is reduced for $SR < 5$ which could be due to a too-weak coupling strength compared to the chip parasitics. As a sanity check, the obtained cuts are also compared to the simplest approximation algorithm labeled "coin-flipping" which randomly assigns each graph vertex to subsets V_1 or V_2 with 50% probability [111]. As the median cuts are only around 0.6 for the coin-flipping procedure, we are confident the chosen random instances are not trivial.

Runtime

Fig.4.3b presents the accuracy vs. measurement time extracted from the first experiment and for N=16. By measuring the phases after 15 oscillation cycles (15 μs), 25% of the cuts have accuracies higher than 0.95 for all graph densities. For N=16, GW's mean CPU execution time is 228 ms and our chip provides a runtime improvement of $1.5.10^4$x. Having oscillations at 100 MHz would further reduce the runtime by two orders of magnitude, at the expense of a higher power consumption.

4.2.6 Impact of SKONN Noise on Max-cut Solutions

Here, we study the impact of noise at the hardware level when solving the Max-cut task for various graph sizes. The objective is twofold: **1)** assess whether SKONN is robust to noise and **2)** quantify the maximum tolerated noise for practical implementation. Due to resource constraints during transient simulations, this study does not include every single noise source from each transistor. As an approximation, we perform stochastic simulations at the level of circuit equations by including the main thermal noise contributions at the oscillating input node (equations (B.4) derived in the Appendix B.1). To compensate for this approximation, the level of input noise is swept to 70x larger values than the calculated ASIC input noise. Note that the input noise also contributes to jitter noise at the output due to the stochasticity on the oscillator transition times, as shown by the 10 stochastic oscillator trajectories in Fig.4.4a.

Main Noise Contributions

Fig.4.4a presents the main thermal noise sources of the shaper block that perturb the analog input waveform. Assuming the noise amplitude is small (small-signal calculation [212]), we find that the main source of thermal noise originates from the current source transistors in our design that also carry noise from biasing stages. During the load capacitor charge, four transistors acting as current sources and four diode-connected transistors contribute to current noise at the input. Further assuming transistor noises are uncorrelated, the current noise power spectral density (PSD) at the input is expressed as:

$$S_N^2 = 4S_S^2 + 4S_D^2 + S_B^2 = \sigma_N^2 \tag{4.14}$$

where S_S^2, S_D^2, and S_B^2 are the PSDs of the current source, diode-connected, and bias transistors, respectively. Note that the noise contribution from the switches is negligible as their PSDs are divided by the high output impedance of the current sources. The transistor parameters used to compute the noise variance are listed in Table 4.1. The small-signal parameters are evaluated using the transistor model proposed by Enz, Krummenacher, and Vittoz (EKV) [201] where the inversion factor is defined as $I_F = I_{bias}/I_S$ with $I_S = 2\mu C_{ox}(W/L)U_T^2$ the specific current of the transistor, and $U_T = k_B T/q$ is the thermal voltage. The transconductance g_m is calculated as:

$$\frac{g_m}{I_{bias}} = \frac{1}{U_T}\frac{2}{1+\sqrt{4I_F+1}} \tag{4.15}$$

As the 1/f noise PSD is inversely proportional to the transistor dimensions, we assume that thermal noise is dominant here and express the current PSD of each transistor i as:

$$S_i^2 \approx 4k_B T g_m \gamma \tag{4.16}$$

with γ=1/2 or γ=2/3 when the transistor is in saturation and weak or strong inversion [201].

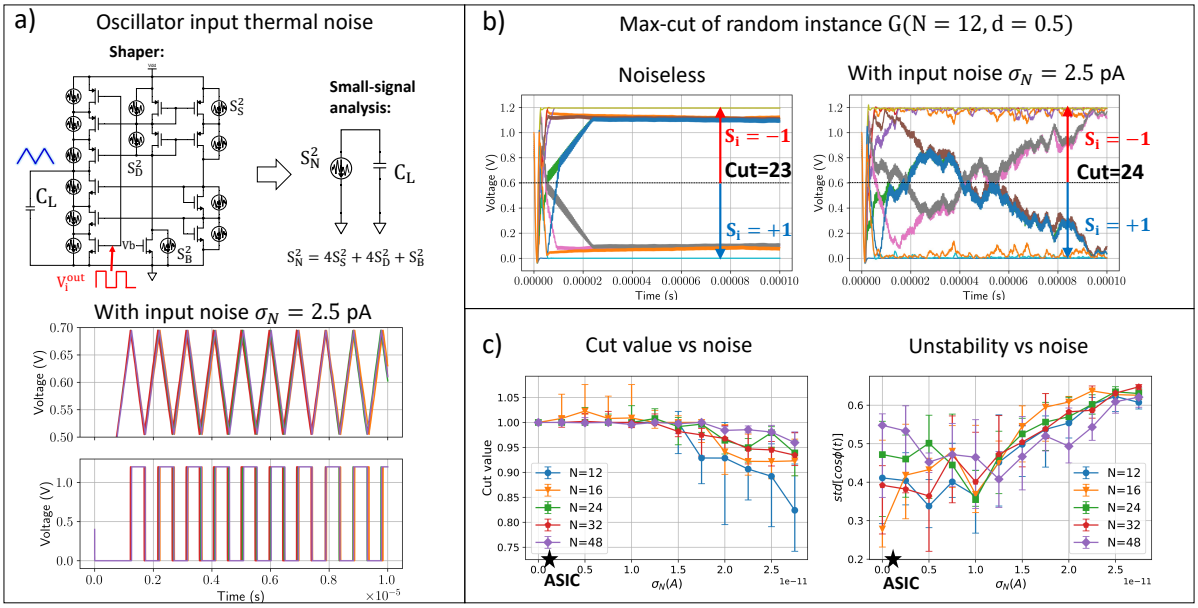


Figure 4.4: *a) Schematic of the shaper stage with current noise sources. The bottom figure shows 10 stochastic transient simulations with σ_N=2.5 pA. b) Example of phase dynamics with and without noise for a 12-node graph with 50% density. The Y-axis shows the output voltages V_f of each analog filter measuring the phases (see Fig.B.3). c) Left: cut-value vs standard deviation of noise σ_N. Right: measure of instability vs σ_N.*

Transistor	Length	Width	I_F	g_m or R	Noise standard deviation
Current source	6 μm	12 μm	0.5	$g_{m_S} = 5.6\,\mu$A/V	σ_S=0.25 pA
Diode-connected	6 μm	12 μm	0.5	$g_{m_D} = 5.6\,\mu$A/V	σ_D=0.25 pA
Chip current mirror	16 μm	2 μm	8	$g_{m_C} = 2.3\,\mu$A/V	σ_C=0.16 pA
External resistor	NA	NA	NA	$R = 1\,M\Omega$	σ_R=0.13 pA
Shaper bias	20 μm	10 μm	2	$g_{m_B} = 3.8\,\mu$A/V	$\sigma_B^2 = \sigma_R^2 + 4\sigma_C^2 + 0.18^2$ σ_B =0.4 pA
Total input current noise	$\sigma_N^2 = 4\sigma_S^2 + 4\sigma_D^2 + \sigma_B^2$ $\sigma_N = 0.8$ pA				

Table 4.1: *ASIC Transistor Parameters used in the Stochastic Simulations.*

To model noise in the time domain, we consider Gaussian white noise (GWN) which also has a similar PSD, i.e. a constant spectrum, with the additional assumption that the current noise samples follow a Gaussian distribution [213, 214]. Hence, we express the current noise in the time domain as:

$$I_N(t) = \sigma_N \xi(t) \tag{4.17}$$

where $\xi(t)$ is a GWN with $E[\xi(t)]$=0, and correlation $E[\xi(t+\tau)\xi(t)] = \delta(\tau)$ with $\delta(t)$ the Dirac delta function. It follows that $E[I_N(t+\tau)I_N(t)] = \sigma_N^2\,\delta(\tau)$ and the PSD is:

$$S_N^2 = \sigma_N^2 \tag{4.18}$$

This relation enables the numerical evaluation of the σ_N-values using the ASIC transistor parameters of Table 4.1 while running stochastic transient simulations. From the circuit of Fig.4.4a, we see that the current noise is integrated by the capacitor C_L during the dynamics. Overall, the addition of GWN turns the ODE system into a stochastic differential equation (SDE) system that is integrated by a dedicated Julia solver [215]. Especially, the ODE for the input voltage V_i^{in} of neuron i becomes the following SDE:

$$C_L dV_i^{in} = I_{bias}\big(1 - 2\frac{V_i^{out}}{V_{dd}}\big)dt + \sum_j C_{ij} dV_j^{out} + \sigma_N dW \tag{4.19}$$

where dW is a Wiener process defined as $\xi(t) = dW/dt$ [214].

Impact of Noise on Max-cut solutions

Fig.4.4c shows the cut values obtained for 10 random graphs G(N,d) of sizes $N \in \{12, 16, 24, 32, 48\}$, densities $d \in \{0.1, 0.2 \ldots 0.9, 1\}$ and various noise variances. For all graph instances, the cut value is evaluated after 100 oscillation cycles and normalized by the cut value obtained in the noiseless case. Note that from Table 4.1 we calculate the input current noise in the ASIC as $\sigma_N \approx$0.8 pA, which does not include the output noise from the hysteresis block. However, the simulations suggest robustness to larger input noise up to σ_N=15 pA which also contributes to output noise. For lower noise levels, noise can sometimes increase the accuracy as shown in the example of Fig.4.4b where a noisy SKONN with σ_N=2.5 pA finds a larger cut compared to the noiseless case. Furthermore, the stability of the system is estimated by analyzing the time evolution of the spin signals expressed as $S_i(t) = \cos\phi_i(t)$. Computing the standard deviation $std[S_i(t)]$ over time from t=0 to t=$100/f_{osc}$ indicates whether the spins settle or not. In the ideal case, $std[S_i(t)] \approx 0$ whereas in the worst case, the spin varies between -1 and +1 with $std[S_i(t)] \approx 1$. Note that the transient dynamics are also included in the metric $std[S_i(t)]$ which hence never reaches 0 in practice. Fig.4.4c shows a correlation between the decrease of cut value and the decrease of stability for $\sigma_N \geq$15 pA for all SKONN sizes.

From this study, we conclude that only large levels of noise have detrimental effects on SKONN performances. Moreover, moderate levels of noise can slightly increase the accuracy which we believe is linked to the escape from local minima, as already observed by Wang et al. [69] at the level of phase modeling.

4.2.7 Large-Scale Simulations and Benchmark

.

Weighted Max-cut of random graphs

To assess how SKONN's computational performances scale, we run large-scale simulations of random weighted Max-cut problems for N=8, 16, 32, 64, 128, 256, 512, and 1024 nodes. For each graph density d=0.25, 0.5, and 0.75, 10 random graphs $G(N,d)$ are generated such that the total number of edges $m = d\,N(N-1)/2$. The graph edges are randomly weighted with

positive values from 0 to 15 that correspond to the IC synaptic range. We use the IC parameters and solve SKONN's dynamics (eq.3.7) with MATLAB using the built-in ODE solver ode15s (see appendix C). For each graph instance, we run 10 trials with random phase initialization, for a total of 100 trials per graph size and density. As a ground truth, we consider the best solution Cut_{GW} provided by the Goemans-Williamson algorithm, out of 100 random projections defining the cut [111] and computed with the CVX solver on MATLAB [211]. The distance between the SDP cut Cut_{SDP} and Cut_{GW} is represented in Fig.4.5a. As $Cut_{SDP} \geq$Max-cut, the ratio Cut_{GW}/Cut_{SDP} gives a lower bound on the chosen GW cut. For all the trials, the results are compared with Kuramoto dynamics with the same parameters and initializations.

Fig.4.5b shows SKONN's and Kuramoto's phase distributions for each ONN size. Here again, it appears that SKONN phases tend to be clustered near 0° and 180° whereas Kuramoto phases seem more uniformly distributed. Fig.4.5c present the obtained cuts when considering the first oscillator as the reference, and normalized by Cut_{GW}. We first notice that the results are quite homogeneous with respect to the graph densities. Secondly, it appears that SKONN produces high-quality cuts as $Cut/Cut_{GW} \approx 1$ for all ONN sizes. In contrast, Kuramoto-ONNs have a lower accuracy for sizes between N=16 and N=256. Interestingly, the settling time (time to reach a steady phase state) seems to grow according to a logarithmic law with the ONN size. This result refines some previous scaling observations mentioning a quasi-constant settling time [69, 139]. It also confirms the high ONN parallelism and ability to compute in a few tens of cycles, even for large graphs.

Similarly to the CirCut algorithm [112] (rank-2 relaxation approach), we also investigate the Kuramoto accuracy when changing the reference oscillator and name it the *Kuramoto-CirCut* scheme. Fig.4.5d presents the case where the best Kuramoto cut is chosen out of N possible cuts, whereas the SKONN reference oscillator remains the first one. It can be seen that SKONN provides the same quality cut as GW and Kuramoto-CirCut. However, compared to GW and Kuramoto-CirCut, SKONN's cut is solely obtained by reading out the phases with respect to a single oscillator and does not need N different cut evaluations that linearly increase the time to solution.

G-set benchmark

The previous study concerned random graphs. Here, we benchmark SKONN for solving Max-cut using the G-set benchmark that includes various graph topologies [216]. Table 4.2 shows the cuts obtained for a single trial with SKONN and Kuramoto using the same random phase initialization, and considering the first oscillator as the phase reference. The cuts are compared against the state-of-the-art GW [111] and the Kuramoto-CirCut scheme [112]. GW's cut is the best cut obtained out of 100 random projections and computed with the CVX solver on MATLAB [211]. For graphs with $N > 3000$, GW values are taken from another state-of-the-art SDP solver [217] due to memory constraints. The Kuramoto-CirCut values correspond to the best cut extracted from the Kuramoto dynamics, out of N possible reference oscillators.

With a single run, Kuramoto-CirCut and SKONN solvers produce, on average, better results than the GW algorithm which picks up the best spin configuration out of 100 random projections. The average SKONN cut value is 94.6% of the best-known cuts [216] and the highest among the four methods. The accuracy obtained by simulating SKONN motivates its real hardware implementation as the time-to-solution could be drastically reduced compared to a CPU.

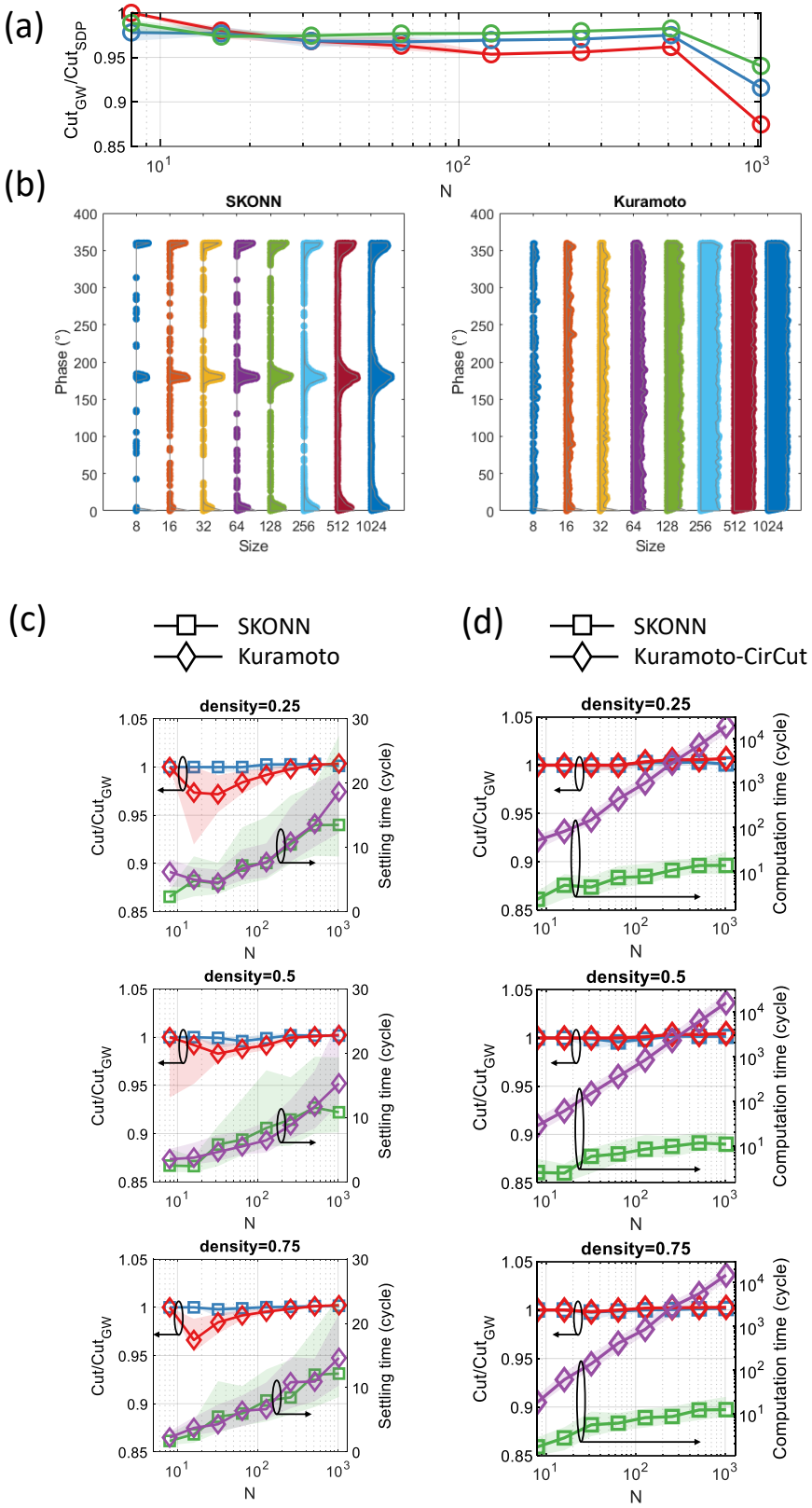


Figure 4.5: *a) Ratio between Cut_{GW} and Cut_{SDP} that gives a lower bound on Cut_{GW} as $Cut_{SDP} \geq$Max-cut. b) Phase distribution at steady state for SKONN (left) and Kuramoto (right) for various ONN sizes. c) Cuts obtained by SKONN and Kuramoto when the first oscillator is the phase reference and for various graph densities. d) Cut obtained when SKONN's reference is the first oscillator, compared to the case where the Kuramoto reference is changed N times, similarly to the CirCut algorithm [112].*

Table 4.2: *Comparison Between CirCut, Kuramoto, and SKONN solvers for G-SET Max-cut instances. The bold values indicate the best cut obtained among the four approaches. The number in parenthesis is the cut normalized by the best-known cut found in [216].*

Graph				Value					Settling time (cycles)	
Name	$\|(V\|, \|E\|)$	Weights	Type	GW	Kuramoto-CirCut	Kuramoto	SKONN	Best-known [216]	Kuramoto	SKONN
G11	(800, 1600)	-1,+1	toroidal	530 (0.940)	514 (0.911)	502 (0.890)	**540 (0.957)**	564	480	368
G12	(800, 1600)	-1,+1	toroidal	**532 (0.957)**	510 (0.917)	496 (0.892)	528 (0.950)	556	216	272
G13	(800, 1600)	-1,+1	toroidal	**554 (0.952)**	538 (0.924)	524 (0.900)	544 (0.935)	582	368	224
G14	(800, 4694)	+1	planar	2978 (0.972)	3005 (0.981)	2974 (0.970)	**3020 (0.986)**	3064	416	496
G15	(800, 4661)	+1	planar	2963 (0.971)	**2992 (0.981)**	2970 (0.974)	2979 (0.977)	3050	704	240
G20	(800, 4672)	-1,+1	planar	849 (0.902)	**871 (0.926)**	818 (0.869)	858 (0.912)	941	504	208
G21	(800, 4667)	-1,+1	planar	849 (0.911)	**868 (0.932)**	839 (0.901)	852 (0.915)	931	184	104
G22	(2000, 19990)	+1	random	12936 (0.968)	**13095 (0.980)**	13026 (0.975)	12988 (0.972)	13359	280	104
G23	(2000, 19990)	+1	random	12946 (0.970)	**13106 (0.982)**	13078 (0.980)	12950 (0.970)	13344	864	216
G24	(2000, 19990)	+1	random	12966 (0.972)	**13135 (0.985)**	13050 (0.978)	12997 (0.974)	13337	928	160
G30	(2000, 19990)	-1,+1	random	3014 (0.883)	**3200 (0.938)**	3175 (0.930)	3095 (0.907)	3413	376	192
G31	(2000, 19990)	-1,+1	random	2885 (0.872)	**3089 (0.933)**	3063 (0.925)	3015 (0.911)	3310	1424	456
G32	(2000, 4000)	-1,+1	toroidal	1290 (0.915)	1284 (0.911)	1280 (0.908)	**1332 (0.944)**	1410	616	520
G33	(2000, 4000)	-1,+1	toroidal	1266 (0.916)	1254 (0.907)	1244 (0.900)	**1294 (0.936)**	1382	424	248
G34	(2000, 4000)	-1,+1	toroidal	1274 (0.920)	1258 (0.909)	1224 (0.884)	**1306 (0.943)**	1384	704	456
G50	(3000, 6000)	-1,+1	toroidal	**5880 (1.00)**	5784 (0.983)	5764 (0.980)	5818 (0.989)	5880	544	1968
G56	(5000, 12498)	-1,+1	random	3634 (0.904)	3700 (0.921)	3622 (0.901)	**3733 (0.929)**	4017	712	296
G57	(5000, 10000)	-1,+1	toroidal	**3320 (0.950)**	3138 (0.898)	3054 (0.874)	3270 (0.936)	3494	792	576
G60	(7000, 17148)	+1	random	13610 (0.959)	**13730 (0.968)**	13669 (0.963)	13717 (0.967)	14188	824	384
G61	(7000, 17148)	-1,+1	random	5252 (0.906)	5322 (0.918)	5257 (0.907)	**5327 (0.919)**	5796	1648	312
G62	(7000, 14000)	-1,+1	toroidal	4612 (0.947)	4394 (0.902)	4358 (0.895)	**4642 (0.953)**	4870	520	1328
G64	(7000, 41459)	-1,+1	planar	7624 (0.871)	8046 (0.919)	7946 (0.908)	**8135 (0.930)**	8751	800	352
Average				0.934	0.938	0.923	**0.946**	1	623	431

For instance, solving the smallest graph G11 requires 12 s of GW runtime on a laptop (i7 Intel core @1.6 GHz and 32 GB of RAM). In contrast, SKONN's settling time does not vary much with the ONN size and could enable a large-scale cut computation in less than 431 cycles on average. With oscillators running at 1 MHz, the runtime per trial would only be 431 μs which is 2.8 10^4x faster than GW's execution. However, reaching excellent cuts such as 99.9% of the best-known cut requires more trials and annealing the ONN to escape local minima. Reaching the optimal solution probably takes exponential time as Max-cut is NP-hard. For more results using SHIL and various annealing schemes, we invite the reader to refer to [69].

4.3 Beyond Binary Phases with ONN

ONN dynamics are generally used as analog relaxations for COPs with binary variables such as the Ising problem. This means that ideally, the wished steady phases should be binary so that the relaxation is as accurate as possible. However, during transient, ONN phases are analog and can take non-binary values. Moreover, without further constraints such as harmonic injection, ONN phases can settle to arbitrary values. Similar to Erementchouk et al. [204] who proposed to exploit the analog phase values to solve Max-cut, we explore how a Kuramoto-ONN in a free regime, i.e. without harmonic injection, can approximate the traveling salesman problem (TSP).

4.3.1 Traveling Salesman Problem and Previous Work

TSP is one of the easiest COP to understand, yet one of the hardest to solve. Its NP-hard optimization version can be described as follows: *Given a list of cities and inter-city distances, find the shortest round-trip that visits each city only once* [55]. Here, we focus on the *symmetric* TSP, i.e. the distance between cities i and j is symmetric as $d_{ij} = d_{ji}$. The best-existing approximation algorithm for TSP instances satisfying the triangle inequality is the one from Christofides [218] that guarantees a 3/2 approximation ratio.

Hopfield and Tank were among the first to solve TSP with neural networks [56]. Their idea is to have an array of neurons such that a column corresponds to a city, and a line represents the position in the tour. With such encoding, one only needs to read the array line by line to know the corresponding tour. Of course, this is possible only if there is a single activated neuron per line and column. Such constraints are implemented using synaptic weights, among those that encode the inter-city distances. The concept is illustrated in Fig.4.6b for 5 cities. Note that there are 25 neurons required to solve this TSP instance. Overall, to solve an N-city TSP, there are N^2 neurons that evolve in a space of 2^{N^2} possible states.

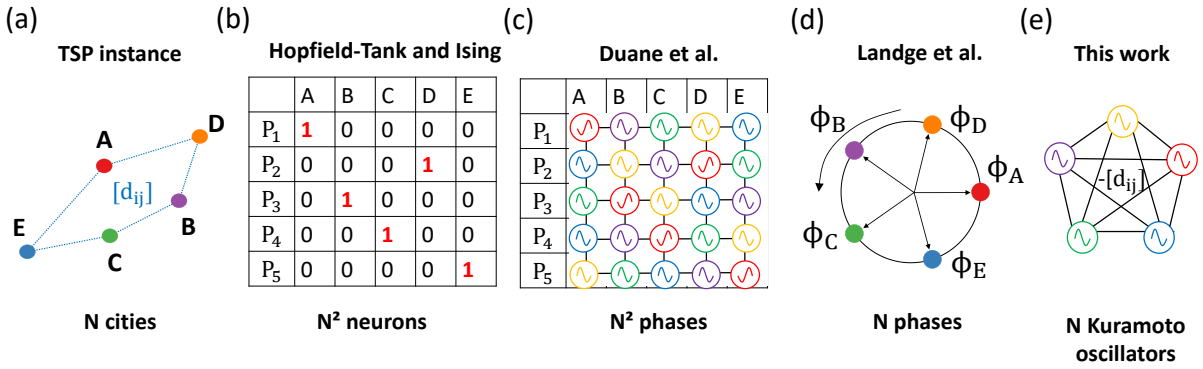


Figure 4.6: *a) Example of TSP instance with 5 cities. b) Original Hopfield-Tank mapping for solving TSP [56]. Note that the Ising approach is similar [53]. c) Architecture proposed by Duane et al. [205]. Each phase can take up to N values so all possible cyclic permutations are encoded (highlighted with different colors). d) Original idea from Landge et al. [113] to encode the TSP tour in the phase permutation of N oscillators. e) Proposed Kuramoto-ONN architecture to approximate TSP.*

In [205], authors have explored the same array architecture but using coupled oscillators with information encoded in the synchronization between them. Compared to Hopfield-Tank's approach, it allows for simultaneously representing all possible cyclic tour permutations, as illustrated in Fig.4.6c. As there are N possible cyclic permutations, each line/column sees N possible phase values. It is as if Hopfield-Tank binary variables were extended to N-spin variables. However, assuming that oscillators can take any phase value, there are now N^{N^2} possible states which further complexify the space search.

Recently, Landge et al. [113] brought back the idea of using coupled oscillators but having only N phase-based oscillators that encode the TSP tour in the phase permutation. Not only this approach reduces the number of neurons from N^2 to N, but it also avoids the tour degeneracy linked to the choices of starting city and tour direction. When phases are constrained to take N values, the search space has a size of N^N which is smaller than the two previous approaches. Inspired by this conceptual work, we propose a similar idea using a phase-based Kuramoto-ONN that could be implemented with any type of sinusoidal oscillator.

4.3.2 Approximating TSP with Kuramoto-ONN

The main idea behind the Kuramoto-ONN solver is that a negative synapse implements a repulsive force between two coupled oscillators. The stronger the negative coupling is, the less likely the two oscillators have similar phases. If the negative weight is very large, there is a high chance that the phase difference is π. Keeping in mind that we want to represent the TSP tour as a phase permutation on the unit circle, phases separated by π should correspond to cities that are far from each other in the tour, and most likely, cities that are separated by a long distance. Hence, it is natural to choose the ONN synaptic weights as:

$$W_{ij} = -d_{ij} \tag{4.20}$$

Overall, this mapping leads to an N-neuron fully-connected ONN with $N(N-1)/2$ negative weights. The corresponding energy function is then:

$$E = \frac{1}{2}\sum_i \sum_j d_{ij} \cos(\phi_j - \phi_i) \tag{4.21}$$

which is the standard ONN Lyapunov function [68]. Hence, the resulting ONN dynamics seek to minimize E as:

$$\frac{d\phi}{dt} = -\nabla_\phi E \tag{4.22}$$

The ONN minimization process can be alternatively written as:

$$\begin{aligned} &\min_\phi -\Big(\sum_{(i,j)|\Delta\phi_{ij}|\geq\pi/2} d_{ij}\,|\cos\Delta\phi_{ij}|\Big) + \Big(\sum_{(i,j)|\Delta\phi_{ij}|<\pi/2} d_{ij}\,|\cos\Delta\phi_{ij}|\Big) \\ =&\min_\phi -E_N + E_P \end{aligned} \tag{4.23}$$

The ONN only seeks to minimize E, thus long distances should be weighted by a negative sign and belong to E_N. Hence, those phases are pushed in the opposite direction on the unit circle and the corresponding cities are not consecutive in the tour. As this process happens for every pair of cities, the by-product effect consists of some distances d_{ij} that are positively weighted

(E_P terms). As they are positive, the ONN favors consecutive phases that have a small d_{ij} value to further minimize E. It is in E_P that some sort of TSP is embedded, with a summation over consecutive cities and distances weighted by $\cos\Delta\phi_{ij}$.

Hence, the ONN does not exactly embed TSP but rather executes a broader optimization task, as all possible d_{ij} intervene in the formulation eq.4.23. From a geometrical perspective, the ONN minimizes a sum of N(N-1) dot products as:

$$E = \frac{1}{2}\sum_i\sum_j \vec{v_{ij}}(\phi_i).\vec{v_{ji}}(\phi_j) \tag{4.24}$$

where vectors $\vec{v_{ij}}$ have length $\sqrt{d_{ij}}$ and angle ϕ_i. Note that it is possible to compensate for the undesired term $\cos\Delta\phi_{ij}$ that weights distance d_{ij} and has been proposed by Mallick et al. [206]. However, it requires using synaptic weights in the complex domain and harmonic injection at frequency Nf_{osc}. Next, we first assess the performances of this simpler ONN version.

4.3.3 Simulation Results

In Matlab, we generate random TSP instances where the city coordinates are randomly chosen according to a uniform distribution on a square map (Fig.4.7a). ONN phases are randomly initialized and sampled once the ONN stabilizes to a local minimum. From the measured phase permutation, we compute the final tour distance. Fig.4.7a shows some TSP instances and corresponding final tours. Visually, the proposed tours seem near-optimal. We further evaluate the ONN energy and TSP tour distance during the dynamics, as shown in Fig.4.7b. From these simple examples, we make two important observations.

1. By definition, the ONN energy is monotonously decreasing and rapidly reaches a local minimum. However, it appears that the tour distance can be non-monotonous, as shown in the second example. In this particular case, the ONN settles to a sub-optimal tour value, although its phase trajectory previously produced a better tour.
2. As shown in Fig.4.7c, the ONN dynamics are not uniformly distributed on the unit circle. Some phases are very close to each other, which can challenge the phase permutation measurement in a real implementation.

However, simulations enable high precision during the phase permutation measurement and thus we further investigate how the performance scales with the number of cities. Fig.4.8a presents the ONN tour distance versus the number of cities. It is normalized by the optimal tour found using an integer program in Matlab. Each data point consists of 10 random TSP instances and 10 trials per instance with random phase initialization. The shaded interval represents the maximum and minimum reached values. It appears that for $N \leq 17$, the ONN can find an optimal tour and median values are below 10% of the optimal solution. However, for larger instances, the ONN never finds the optimal solution, while having median tours below 50% of the optimal distance for $N < 45$. Fig.4.8b and c present instances from the TSPlib set [219]. For the 29 Bavarian cities, the ONN finds a 12% longer tour than the optimal one, while for the 48 US capitals, the tour is 36% longer. Visually, we notice that the ONN tour is kind of "circular", i.e. as if the ONN would seek a center of mass for all cities from which a tour is constructed. For instance, the particular southern path of the optimal 48 US tour has never been found by the ONN.

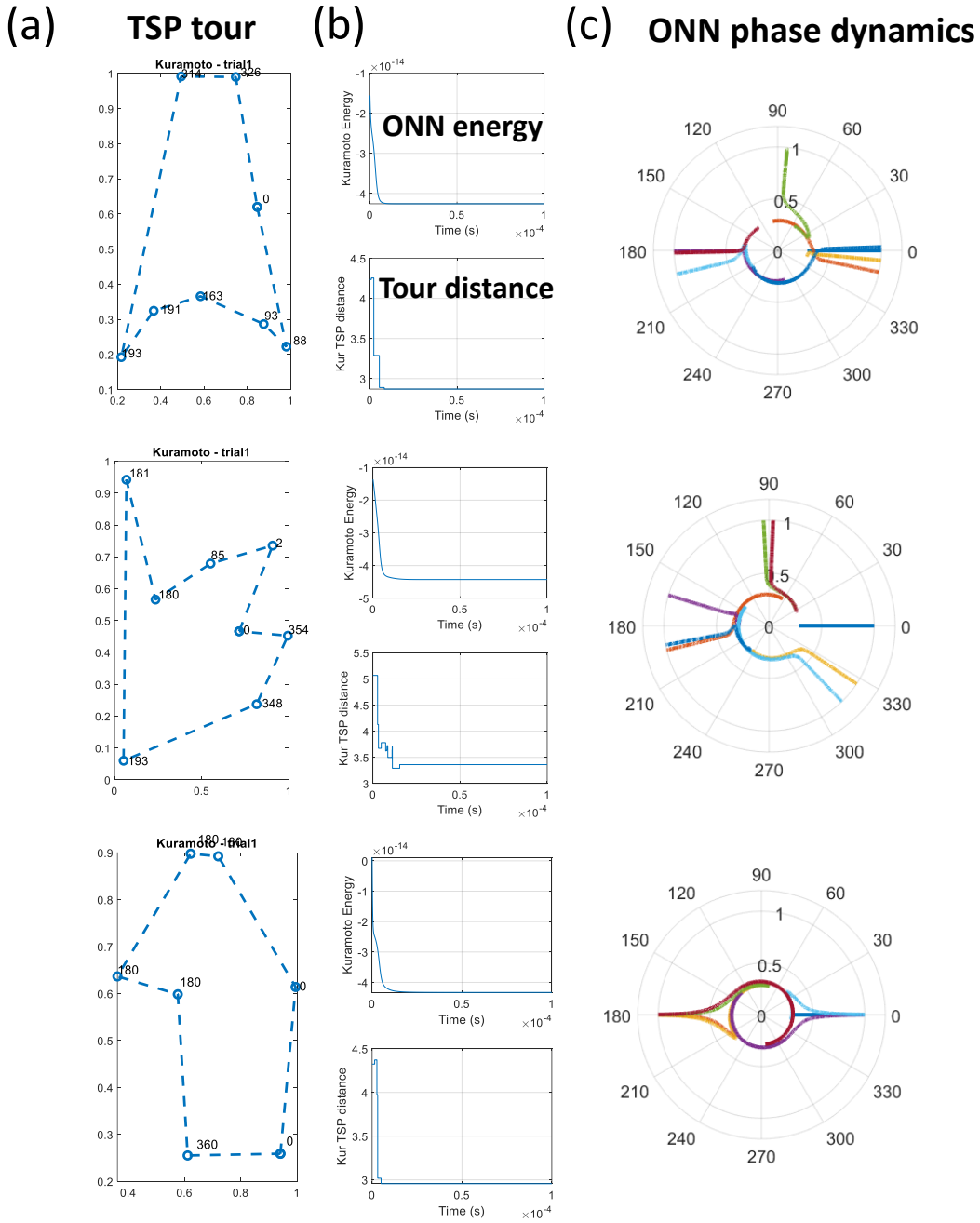


Figure 4.7: *a) Example of 8-city tours found by ONNs. Each point is a city randomly located on a square map. b) ONN energy and tour distance in real-time for each example. c) Corresponding ONN phase dynamics. The TSP tour is read from the phase permutation.*

To achieve the optimal solution for more than 17 cities, the ONN requires additional mechanisms to escape local minima such as annealing using noise [113]. However, these preliminary results are encouraging as ONN could be used in cluster-based TSP solvers. Indeed, one of the state-of-the-art strategies for solving large TSP instances is *divide and conquer* where smaller TSP tours are assembled to build a larger one [220]. One could think of having a modular architecture with an array of ONN modules that run in parallel and continuous time, and some

post-processing algorithm to assemble the sub-tours and further optimize the final TSP tour. However, as seen in Fig.4.7c, the architecture requires improvements in the phase separation as phases are often forming clusters, thus preventing reliable tour assessment.

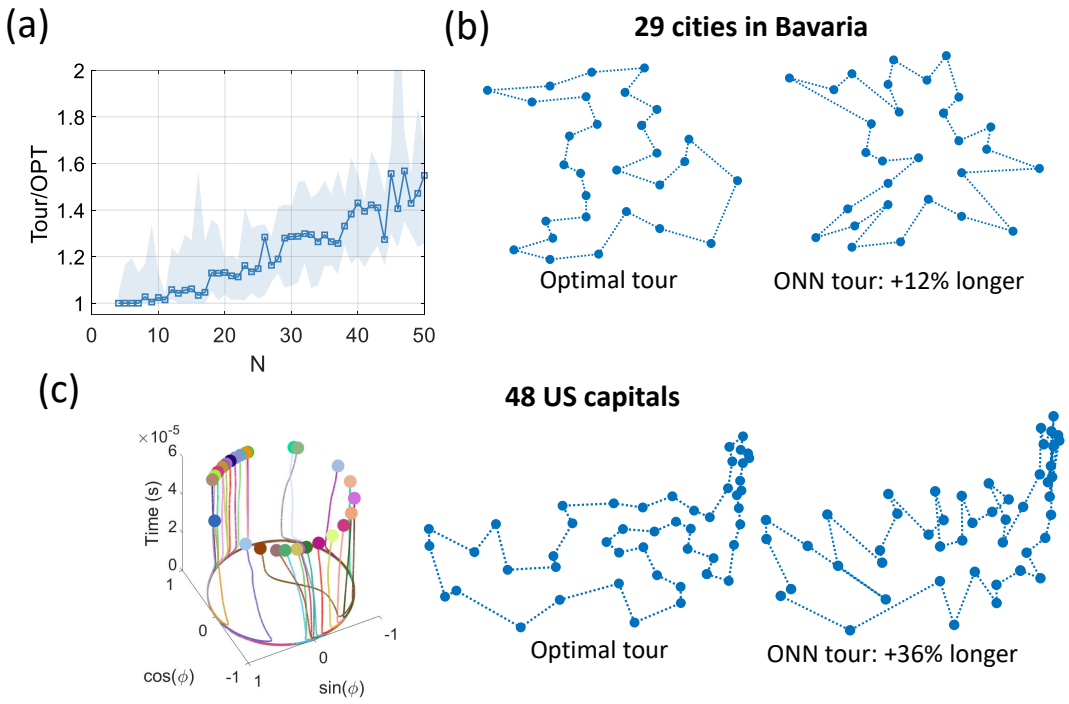


Figure 4.8: *a) Scaling of ONN tour vs. number of cities in random TSP instances. b) Instance of TSPlib [219] with 29 cities. c) TSP of the 48 capitals from the USA. The left-hand plot shows the evolution of the corresponding ONN unit circle.*

4.4 Constrained Optimization with Lagrangian-ONN

4.4.1 Mathematical Optimization

We have seen that ONNs are powerful to solve COPs thanks to their intrinsic gradient descent properties. However, minimization using gradient descent does not guarantee to reach one of the global minima and can be stuck at local minima, i.e. suboptimal COP solutions. One way of avoiding the system being stuck at local minima is to express conditions for optimality that must be reached before the system settles to a fixed point. In the field of Mathematical Optimization, the conditions are rather called *constraints*, and minimizing a function f subject to constraints g on **real variables** is known as the *General Programming Problem* [221, 222]:

$$\begin{aligned} &\min f(x) \\ &\text{subject to: } g(x) = 0 \\ &\qquad\qquad x \geq 0 \end{aligned} \tag{4.25}$$

where $f : x \in \mathbb{R}^N \longrightarrow \mathbb{R}$ is the scalar cost function to minimize along N real decision variables and $g : x \in \mathbb{R}^N \longrightarrow \mathbb{R}^M$ is the function expressing the M constraints to satisfy. Note that the constraints can be expressed in three equivalent formulations [221]:

$$(A) \begin{array}{r} g(x) = 0 \\ x \geq 0 \end{array} \iff (B) \begin{array}{r} g(x) + s \geq 0 \\ x \geq 0 \\ s \geq 0 \end{array} \iff (C)\ g'(x') \geq 0$$

where x' is an augmented vector including slack variables s, and g' is an augmented function comprising the non-negativity constraints on x'. Given a COP with constraints $g(x) = 0$ such as in formulation (A), the state-of-the-art approach using Ising machines [53] and ONNs is to transform the constrained optimization problem into an unconstrained problem. The idea is to include the constraints in an augmented cost function $f_P(x) = f(x) + P\, g^T(x) g(x)$ where P is a positive real term that penalizes the cost function if $g(x) \neq 0$. However, in the general case, reaching a minimum of $f_P(x)$ does not guarantee that the constraints are satisfied.

4.4.2 Lagrangian Multipliers

The method of *Lagrange multipliers* is a technique to solve constrained optimization problems named after the mathematician Joseph-Louis Lagrange. Similarly to the penalty method, it includes constraints weighted by *multipliers* λ in an augmented cost function $L : \mathbb{R}^{N+M} \longrightarrow \mathbb{R}$ defined as:

$$\begin{aligned} &\min f(x) \\ &\text{subject to: } g(x) = 0 \end{aligned} \longrightarrow L(x, \lambda) = f(x) - \lambda^T g(x) \tag{4.26}$$

Compared to the previous penalty method, it is as if the penalty terms P would become variables λ rather than fixed parameters. Suppose x is a minimum of $f(x)$ satisfying $g(x) = 0$. Then, the following conditions are necessary for local optimality (under some regularity conditions such

as gradients of constraints are linearly independent [222]):

$$\text{Lagrangian necessary conditions for local optimality:}$$
$$\exists\, \lambda \in \mathbb{R} \begin{cases} \nabla_\lambda L = 0 \\ \nabla_x L = 0 \end{cases} \iff \exists\, \lambda \in \mathbb{R} \begin{cases} g(x) = 0 \\ \nabla_x f(x) = \sum_{k=1}^{M} \lambda_k \nabla_x g_k(x) \end{cases} \tag{4.27}$$

where ∇_x is the gradient operator with respect to variable x. Note that the generalized conditions with inequality constraints are called the *Karush-Kuhn-Tucker conditions* (KKT) [223]. A geometrical interpretation of the Lagrangian necessary conditions is shown in Fig.4.9a with two variables and a single constraint. While moving on the line $g(x) = 0$, we are looking for a point where $f(x)$ cannot change in any direction of $g(x) = 0$. This situation can occur when a contour of $f(x) = c$ is tangent to the line $g(x) = 0$, i.e. $\exists \lambda \in \mathbb{R}\, \nabla_x f(x) = \lambda \nabla_x g(x)$, or when we reach an extremum of $f(x)$, i.e. $\nabla_x f(x) = 0$ and thus $\lambda = 0$. In the general case with multiple constraints, $\nabla_x f(x)$ is a linear combination of the constraints' gradients as expressed in eq.4.27.

We see from the definition of the Lagrange function (eq.4.26) that performing gradient descent of $L(x, \lambda)$ with respect to x minimizes the cost function $f(x)$, just like in the previous method. The KKT Theorem [222, 223] suggests how to find the λ values, which points out an interesting property of the optimal point that can be harnessed by any gradient system such as neural networks and ONNs.

Theorem 4.1 (Corollary of Karush-Kuhn-Tucker Theorem [222])**.** *Let $f(x)$ and $g(x)$ of the general nonlinear programming problem satisfy convexity and concavity conditions.*
x^* is global optimal solution $\iff$ $\exists\, \lambda^*$ $L(x^*, \lambda^*)$ is a saddle point.

The saddle point property linked to the KKT theorem is illustrated in Fig.4.9b. As having a saddle point is a necessary condition for local optimality, the search for a saddle point can drive some optimization algorithms and is the focus of this section.

4.4.3 Lagrangian Neural Networks

In literature, researchers have proposed various methods for solving mathematical optimization problems for real-time processing and to enhance convergence compared to digital approaches. Analog circuits were among the first accelerators to solve these problems by encoding them directly in the hardware with analog devices such as resistors, diodes, voltage, and current sources [221, 224]. The main idea is to have a system whose dynamics evolve to satisfy the necessary KKT conditions for optimality. Other approaches proposed to interpret the optimization problem as a Lagrangian Neural Network (LNN) where neurons x evolve to minimize the cost function $f(x)$, while Lagrangian neurons λ bring x in a feasible region where $g(x) = 0$. LNNs were first proposed in [114], where authors have defined custom dynamics for the neurons that seek a saddle point during the minimization process as:

$$\begin{cases} dx/dt = -\nabla_x L \\ d\lambda/dt = +\nabla_\lambda L \end{cases} \tag{4.28}$$

We see that any fixed point of the dynamics satisfies the Lagrangian necessary conditions for local optimality (eq.4.27). As illustrated in Fig.4.9c, LNNs seek solutions in real-time and do

not settle until $\nabla_\lambda L = g(x) = 0$, i.e. not before the constraints are met. Note that in the general case, there is no performance guarantee and the LNN can settle to a fixed point corresponding to a local minimum for $f(x)$. Nevertheless, if one expresses sufficient conditions for optimality as constraints, then the LNN is forced to find the optimal solution as we show next for the Satisfiability problem.

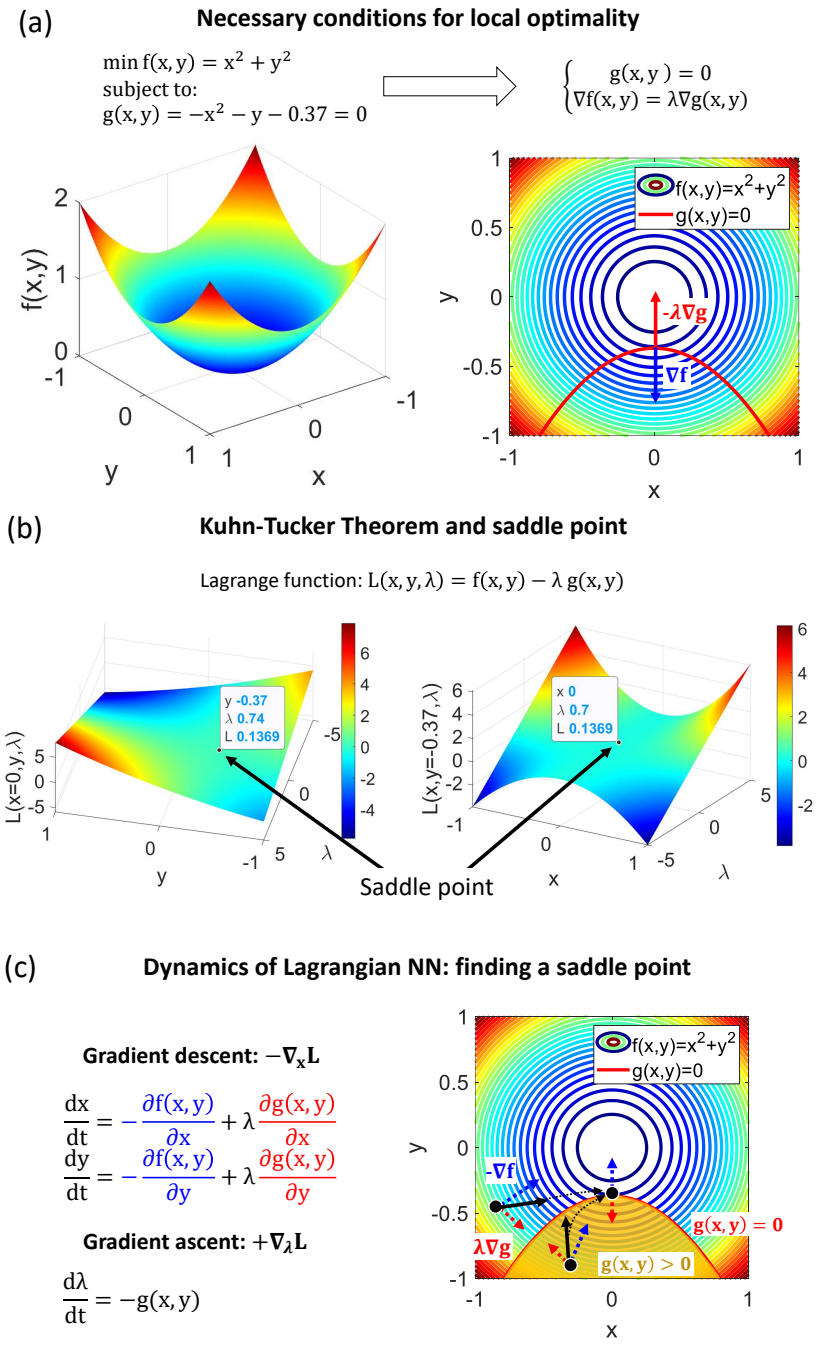


Figure 4.9: *a) Illustration of Lagrange necessary conditions for local optimality. b) Plots of the Lagrange function L for $x = 0$ (left) and $y = 0.37$ (right) highlighting the saddle point of L which corresponds to a local solution of the initial problem. c) Dynamics of a Lagrangian neural network that seeks a saddle point. The plain-line black arrows represent the trajectories of the neural network that converge to the local optimal point.*

4.4.4 Lagrangian-ONN (LagONN) for Solving Satisfiability Problems

Here, we propose a similar idea to Nagamatu et al. [225] for solving NP-hard satisfiability problems using an LNN with phase-based oscillatory neurons. Compared to most of the state-of-the-art models [225–228], we restrict ourselves to dynamics that are implementable by coupled phase oscillators. A crucial feature of analog-based approaches is whether the system has bounded variables or not. In [226] and [229], the authors proposed an analog model that provides an exponential speed-up at the expense of unbounded variables and hence a potential exponential power. Whereas limiting the growth of variable values generally induces an exponential computation time for hard instances [227]. However, note that there are promising models such as memcomputing machines that allow polynomial scaling for both convergence time and power [228]. Nevertheless, these models are not exempted from unstable dynamics such as limit cycles that can appear in the general case. As our goal is to have an implementable model with bounded variables, most likely the proposed ONN does not provide exponential speed-up in the general case. However, using oscillators guarantees a bounded circuit power as variables $\phi \in [0, 2\pi[$ and the synaptic amplitudes do not vary in the proposed implementation. Moreover, simulations up to N=200 suggest that our approach can find near-optimal solutions of hard max-3-satisfiability instances in polynomial time as $O(N^4)$.

The Max-3-Satisfiability Problem (Max-3-SAT)

The Max-3-SAT problem consists in finding the Boolean assignment of variable x that maximizes the number of TRUE clauses composed of three literals $C_m = l_1^m \bigvee l_2^m \bigvee l_3^m$ in the formula:

$$f_B = C_1 \bigwedge C_2 \bigwedge ... \bigwedge C_{M-1} \bigwedge C_M \tag{4.29}$$

With $l_j^m \in \{x_1, ..., x_N, \overline{x_1}, ..., \overline{x_N}\}$. The problem of assessing whether or not f_B is satisfiable is NP-complete [203], thus any problem in NP can be reduced to 3-SAT which motivates the development of hardware accelerators for solving Max-3-SAT.

Mapping a Boolean Clause to an ONN Module

Similar to Nagamatu et al. [225], we first express a clause C_i as an Ising energy term H_i which is null if and only if C_i is true. There are four kinds of clauses, depending on the number of complementary literals. We propose to map the four clauses as follows:

$$\begin{aligned}
C_0 &= X \vee Y \vee Z \longrightarrow H_0 = 1 + S_X S_Y + S_X S_Z + S_Y S_Z - (S_X + S_Y + S_Z) - S_X S_Y S_Z \\
C_1 &= \overline{X} \vee Y \vee Z \longrightarrow H_1 = 1 - S_X S_Y - S_X S_Z + S_Y S_Z - (-S_X + S_Y + S_Z) + S_X S_Y S_Z \\
C_2 &= \overline{X} \vee \overline{Y} \vee Z \longrightarrow H_2 = 1 + S_X S_Y - S_X S_Z - S_Y S_Z - (-S_X - S_Y + S_Z) - S_X S_Y S_Z \\
C_3 &= \overline{X} \vee \overline{Y} \vee \overline{Z} \longrightarrow H_3 = 1 + S_X S_Y + S_X S_Z + S_Y S_Z + (S_X + S_Y + S_Z) + S_X S_Y S_Z
\end{aligned} \tag{4.30}$$

where spins $S_j = \pm 1$ are the binary Ising variables. Writing the truth table for each clause, we find the following equivalence:

$$\begin{array}{l} C_i \text{ is TRUE} \\ C_i \text{ is FALSE} \end{array} \iff \begin{array}{l} H_i = 0 \\ H_i = 8 \end{array} \tag{4.31}$$

It is important to mention that this mapping leads to a cubic spin interaction $S_X S_Y S_Z$ that is not straightforward to implement in hardware. Note that it is also possible to map a clause to an Ising energy term with quadratic interactions only by adding auxiliary variables [60].

The next step is to map the Ising energies to phase-based ONNs. We propose to relax the binary Ising Hamiltonians to complex variables defined as:

$$\begin{aligned}
H_0 &\longrightarrow Z_0 = 1 + e^{i(\phi_X-\phi_Y)} + e^{i(\phi_X-\phi_Z)} + e^{i(\phi_Z-\phi_Y)} - (e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) - e^{i(\phi_X-\phi_Y+\phi_Z)} \\
H_1 &\longrightarrow Z_1 = 1 - e^{i(\phi_X-\phi_Y)} - e^{i(\phi_X-\phi_Z)} + e^{i(\phi_Z-\phi_Y)} - (-e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) + e^{i(\phi_X-\phi_Y+\phi_Z)} \\
H_2 &\longrightarrow Z_2 = 1 + e^{i(\phi_X-\phi_Y)} - e^{i(\phi_X-\phi_Z)} - e^{i(\phi_Z-\phi_Y)} - (-e^{i\phi_X} - e^{i\phi_Y} + e^{i\phi_Z}) - e^{i(\phi_X-\phi_Y+\phi_Z)} \\
H_3 &\longrightarrow Z_3 = 1 + e^{i(\phi_X-\phi_Y)} + e^{i(\phi_X-\phi_Z)} + e^{i(\phi_Z-\phi_Y)} + (e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) + e^{i(\phi_X-\phi_Y+\phi_Z)}
\end{aligned} \tag{4.32}$$

The complex relaxation Z_i is equal to the Hamiltonian H_i in some cases listed in the following proposition.

Proposition 4.2.

1. *If phases take binary values such as $\phi_j = k\pi$, then $Z_i = 0$ or $Z_i = 8$. Consequently $Z_i = H_i$.*
2. *If $Z_i = 8$, then all phases take binary values such as $\phi_j = k\pi$.*

The proof is detailed in Appendix C. Unfortunately, the reciprocal of Proposition 4.2.1 is false as for instance if $\phi_X = 3.73$, $\phi_Y = 5.17$, and $\phi_Z = 2.03$, then $Z_0 = 0$. This means that although the ONN reaches $Z_i = 0$, phases are not necessarily binary and there is no one-to-one correspondence with spin variables. Whereas Proposition 4.2.2 shows that an unsatisfied clause ($Z_i = 8$) has necessarily binary phases. To summarize, reaching the desired case where $Z_i = 0$ does not necessarily induce binary phases and is a similar situation to ONN solving Max-cut without SHIL: it opens the question of how to round phases to spins. However, we will see that when there are more than a few clauses, the stable phase states tend to be binary which mitigates rounding errors.

Using this property, the idea is to have an ONN module that minimizes $|Z_i|$ so that when $Z_i = 0$, the ONN phases ϕ_X, ϕ_Y, and ϕ_Z give the Boolean variables S_X, S_Y, and S_Z that satisfy clause C_i. Minimizing $|Z_i|$ can be expressed as the following unconstrained problem:

$$\min Z_i Z_i^* \tag{4.33}$$

where Z_i^* is the complex conjugate of Z_i. However, such formulation would involve multiple frequencies within the ONN and we rather choose to focus on phase-based ONNs with a uniform frequency.

Constraining the ONN using a Lagrangian Oscillator

Constraining the ONN for reaching $Z_i = 0$ can be expressed in an alternative manner using the Lagrange function $L_i(\phi, \lambda)$ as follows:

$$L_i(\phi, \lambda) = \lambda_1 Re[Z_i] + \lambda_2 Im[Z_i] \tag{4.34}$$

We indeed have $\nabla_\lambda L_i = (Re[Z_i], Im[Z_i])^T = 0 \iff Z_i = 0$. Note that there are only two constraints and no cost function f as in the general case (eq.4.27). Moreover, there are two possible interpretations for the Lagrangian multipliers leading to two different types of Lagrangian ONNs:

1. λ_1 **and** λ_2 **are synaptic elements**. This possibility opens up the question of how to implement synapses λ_1 and λ_2 that need to evolve in real-time while having a limited range for a real implementation.

2. λ_1 **and** λ_2 **are oscillatory variables**, i.e. neurons. This interpretation only involves oscillating neurons and synapses with fixed amplitude, potentially facilitating the hardware implementation.

Focusing on the second interpretation that we label *LagONN*, there is a particular choice for λ that simplifies the Lagrangian function L_i (eq.4.34). Consider a *Lagrangian Oscillator* with the same frequency as the other neurons, phase ϕ_λ, and amplitude of 1. While moving in the 2D plan, the corresponding unitary vector $\vec{u_\lambda}$ has coordinates $(\cos\phi_\lambda, \sin\phi_\lambda)$ that we assign to variables (λ_1, λ_2). Hence, the resulting Lagrangian function becomes the dot product between $\vec{u_\lambda}$ and the vector $\vec{Z_i} = (Re[Z_i], Im[Z_i])$ as:

$$L_i(\phi, \phi_\lambda) = \vec{u_\lambda}.\vec{Z_i} = \cos\phi_\lambda Re[Z_i] + \sin\phi_\lambda Im[Z_i] \tag{4.35}$$

This leads to a compact expression for L_i as shown here for $i = 0$:

$$\begin{aligned} L_0(\phi, \phi_\lambda) = &\cos\phi_\lambda \\ &- \cos(\phi_X - \phi_\lambda) - \cos(\phi_Y - \phi_\lambda) - \cos(\phi_Z - \phi_\lambda) \\ &+ \cos(\phi_X - \phi_Y - \phi_\lambda) + \cos(\phi_X - \phi_Z - \phi_\lambda) + \cos(\phi_Z - \phi_Y - \phi_\lambda) \\ &- \cos(\phi_X - \phi_Y + \phi_Z - \phi_\lambda) \end{aligned} \tag{4.36}$$

We recognize the Lyapunov function of 4 sinusoidal coupled oscillators $\phi_X, \phi_Y, \phi_Z, \phi_\lambda$ with high-order interactions up to four as shown by the last term. Interestingly, the synaptic amplitudes appear as fixed binary weights of the cosine terms. Hence, this ensures having bounded LagONN variables since only phases vary and are bounded by definition.

LagONN Fixed Points

For LagONN, the Lagrangian necessary conditions for optimality are written as:

$$\begin{cases} \nabla_{\phi_\lambda} L_i = 0 \\ \nabla_\phi L_i = 0 \end{cases} \iff \begin{cases} \vec{u'_\lambda}.\vec{Z_i} = 0 \\ \cos\phi_\lambda \nabla_\phi Re[Z_i] + \sin\phi_\lambda \nabla_\phi Im[Z_i] = \vec{0} \end{cases} \tag{4.37}$$

with $\vec{u'_\lambda} = (-\sin\phi_\lambda, \cos\phi_\lambda)$. Noticing that $|\vec{u'_\lambda}| \neq 0$ and using Proposition 4.2, we identify four possible types of LagONN fixed points that are described in Table 4.3 and illustrated in Fig.4.10b. It appears that LagONN fixed points are not necessarily inducing C_i =TRUE but instead, LagONN can settle to three spurious states. The first one is reached when $Z_i = 8 = H_i$, meaning that C_i is not satisfied. The second and third spurious states occur when at least one phase ϕ_j is non-binary. Then, ϕ_j does not correspond to any spin variable and we cannot decide

whether C_i is true or not. However, from State 3, it might be possible to retrieve the spin values that satisfy the clause by rounding the phases. The final desired state is when $Z_i = 0 = H_i$ with binary phases and C_i is TRUE. Evaluating the angle of $\vec{Z}_i$ with respect to $\vec{u_\lambda'}$ can indicate whether LagONN is in a spurious state or not. However, this would involve additional circuitry or post-processing. Instead, we propose to run several trials with different LagONN initial phases to maximize chances to reach State 4 if C_i is satisfiable.

Note that the undesired first state where $Z_i = 8$ will decrease the accuracy of the Max-SAT solver in general but can also help LagONN to settle if the Boolean formula is unsatisfiable. This contrasts with some other approaches where the network never stabilizes when the formula cannot be satisfied [225]. Our hypothesis for the existence of spurious states is that LagONN uses a single Lagrange variable ϕ_λ instead of λ_1 and λ_2 as in eq.4.34. In the latter, λ_1 would never settle until $Re[Z_i] = 0$. Nevertheless, we will see that this LagONN version performs well in simulations and would be easier to implement in practice given its simpler Lyapunov function (eq.4.36).

Spurious			Desired
State 1	State 2	State 3	State 4
$Z_i = 8$	$Z_i \neq 0$ $Z_i \neq 8$	$Z_i = 0$	$Z_i = 0$
$\vec{Z}_i \perp \vec{u_\lambda'}$ $\forall j\, \phi_j \in \{0, \pi\}$	$\vec{Z}_i \perp \vec{u_\lambda'}$ $\exists j\, \phi_j \notin \{0, \pi\}$	$\exists j\, \phi_j \notin \{0, \pi\}$	$\forall i\, \phi_i \in \{0, \pi\}$
Conclusion:			
C_i is not satisfied	Undecidable	Undecidable	C_i is satisfied

Table 4.3: *LagONN Fixed Points*

LagONN Network and Dynamics for Solving a Clause

Motivated by the saddle-point property, we set LagONN neurons' dynamics such that they minimize $L_i(\phi, \phi_\lambda)$ (Fig.4.10a). In parallel, the Lagrangian oscillator performs gradient ascent of $L_i(\phi, \phi_\lambda)$ until the constraints are met or LagONN reaches a spurious state. Overall, we propose the following dynamics for LagONN implementing a clause C_i:

$$\begin{cases} \tau \frac{d\phi_j}{dt} = -\nabla_{\phi_j} L_i(\phi, \phi_\lambda) = -\frac{\partial \vec{Z}_i}{\partial \phi_j}.\vec{u_\lambda} \\ \tau_\lambda \frac{d\phi_\lambda}{dt} = +\nabla_{\phi_\lambda} L_i(\phi, \phi_\lambda) = \vec{Z}_i.\vec{u_\lambda'} \end{cases} \tag{4.38}$$

where τ and τ_L are the time constants for standard neurons and the Lagrangian neuron. They set the relative speed between gradient descent and ascent. We will see later that for large problems, the performances increase when the Lagrange oscillator is slower than the other neurons. L_i's gradient descent causes $\vec{Z}_i$ to evolve in the opposite direction from $\vec{u_\lambda}$ as expressed here for $\tau = 1$:

$$\begin{aligned} \frac{d\vec{Z}_i}{dt}.\vec{u_\lambda} &= \frac{\partial \vec{Z}_i}{\partial \phi_X}.\vec{u_\lambda}\frac{d\phi_X}{dt} + \frac{\partial \vec{Z}_i}{\partial \phi_Y}.\vec{u_\lambda}\frac{d\phi_Y}{dt} + \frac{\partial \vec{Z}_i}{\partial \phi_Z}.\vec{u_\lambda}\frac{d\phi_Z}{dt} \\ &= -\left(\frac{\partial \vec{Z}_i}{\partial \phi_X}.\vec{u_\lambda}\right)^2 - \left(\frac{\partial \vec{Z}_i}{\partial \phi_Y}.\vec{u_\lambda}\right)^2 - \left(\frac{\partial \vec{Z}_i}{\partial \phi_Z}.\vec{u_\lambda}\right)^2 \\ &\leq 0 \end{aligned} \tag{4.39}$$

whereas L_i's gradient ascent tends to bring $\vec{u_\lambda}$ towards $\vec{Z}_i$ (Fig.4.10a). The dynamics can hence be thought of as a "race" between $\vec{Z}_i$ and $\vec{u_\lambda}$ where the desired outcome is reaching $Z_i = 0$ as a compromise (L_i's saddle point) between the two competitive dynamics.

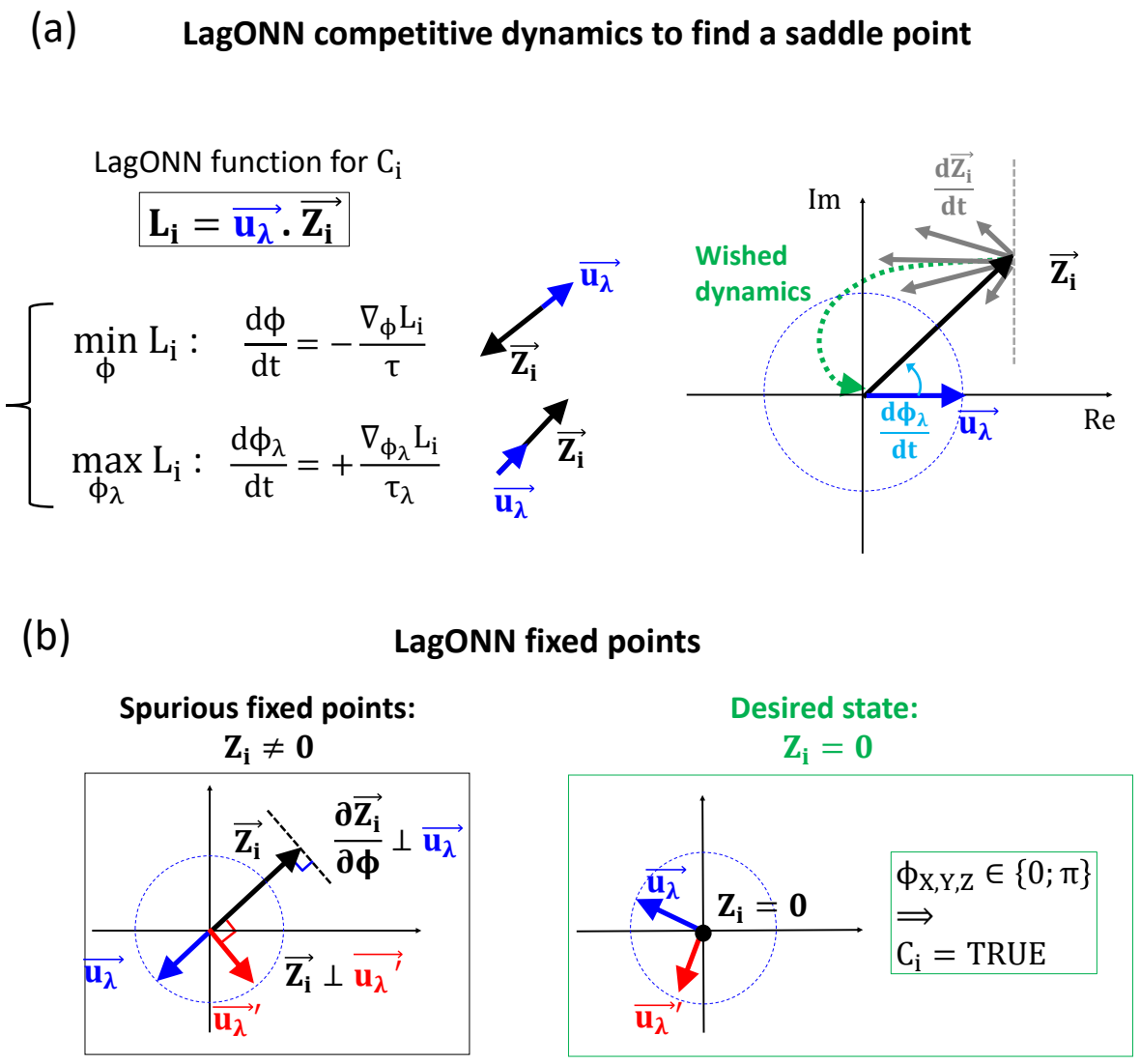


Figure 4.10: *a) Illustration of LagONN competitive dynamics mixing both gradient descent and ascent to satisfy a clause. b) Schematics of the possible fixed points for LagONN. The first type of fixed point corresponds to a spurious state as $Z_i \neq 0$ and C_i is either false or undecidable. If $Z_i = 0$ with binary phases, then C_i=TRUE.*

A LagONN network implementing such dynamics to satisfy $C_0 = X \bigvee Y \bigvee Z$ is shown in Fig.4.11c. Note that there is a reference oscillator to measure the phases and apply an external field to the Lagrangian oscillator. Thus, ϕ_j corresponds to $\phi_j - \phi_{REF}$ in practice. The LagONN network complexity mostly originates from the synaptic array that consists of delayed and weighted signals. In Fig.4.11c, we denote a synapse S_{ij} that modulates signals with weight W_{ij} and phase θ_{ij} as $S_{ij} = W_{ij}e^{i\theta_{ij}}$. This supposes having a mechanism to delay the synaptic input by θ_{ij}/ω_0 in real-time, which is the consequence of the high-order interaction

terms in the LagONN function (eq.4.36).

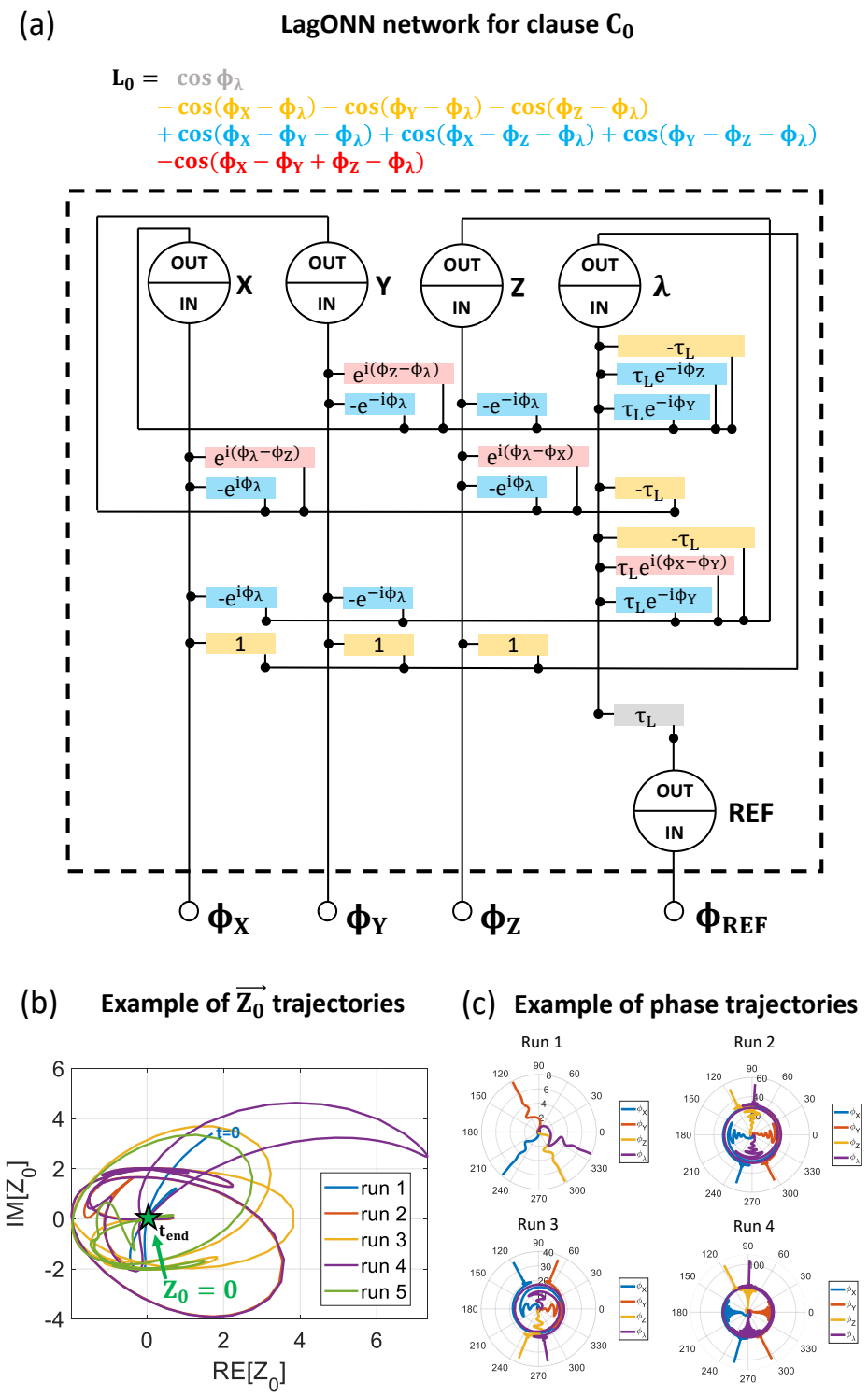


Figure 4.11: *a) LagONN network for solving clause C_0. b) Example of $\vec{Z_0}$ trajectories when phases are randomly initialized and with $\tau_L = \tau$. It appears that a single-clause LagONN network converges towards stable states where $Z_0 = 0$ every time (100 trials). c) Example of phase trajectories. As highlighted in Proposition 4.2, LagONN can satisfy $Z_i = 0$ while having non-binary phases. However, one can conclude that $C_i = TRUE$ here by rounding phases to the nearest multiple of π.*

Fig.4.11b shows examples of $\vec{Z}_i$ trajectories with $\tau_L = \tau$ that all converge to a stable state where $Z_i = 0$. The local stability can be evaluated by linearizing the system around the fixed point and computing the eigenvalues of the Jacobian matrix. As expected from Proposition 4.2, we find that with a single clause phases almost never settle to binary phase values, as shown in Fig.4.11c. Although the results for a single clause are not encouraging, we will see that the system becomes much more interesting at a larger scale as phases tend to relax to binary values.

LagONN Single Clause Stability

When the Lagrangian oscillator is OFF, i.e. $\forall\, t\ \phi_\lambda(t) = 0$, LagONN behaves as a standard ONN even though it has cubic interactions. As we have seen throughout this thesis, the ONN global stability can be proven by finding a bounded Lyapunov function such as L_i (eq.4.36) that is minimized through time due to the chosen dynamics (eq.4.47). Taking the clause C_0 as an example, the function $L_0(\phi, \phi_\lambda = 0)$ is minimized as:

$$\frac{d}{dt}L_0(\phi, \phi_\lambda = 0) = \sum_j \frac{\partial L_0}{\partial \phi_j}\frac{d\phi_j}{dt} = -\tau \sum_j \left(\frac{d\phi_j}{dt}\right)^2 \leq 0 \tag{4.40}$$

Hence the network is stable when the Lagrangian oscillator is OFF. The global stability is much harder to study when the Lagrangian oscillator is ON due to the gradient ascent operation. Authors in [114] have shown that global stability can be proved under convexity conditions along the dynamic path. They proposed the following Lyapunov function, adapted here for LagONN:

$$E_i(\phi, \phi_\lambda) = \frac{1}{2}\nabla_\phi L_i^T \nabla_\phi L_i + \frac{1}{2}\nabla_{\phi_\lambda} L_i^T \nabla_{\phi_\lambda} L_i \tag{4.41}$$

Note how $E_i(\phi, \phi_\lambda) \geq 0$ and $E_i(\phi, \phi_\lambda) = 0$ when LagONN reaches a fixed point. After some calculation, it can be shown that:

$$\frac{dE_i}{dt} = -\nabla_\phi L_i^T \nabla^2_{\phi\phi} L_i \, \nabla_\phi L_i \tag{4.42}$$

which is negative if the Hessian matrix $\nabla^2_{\phi\phi} L_i$ is positive definite. However, in our case, it is unlikely the energy landscape $L_i(\phi, \phi_\lambda)$ remains convex during the entire operation given that it is a sum of cosine terms (eq.4.36). Hence, it is possible that for some instances LagONN is unstable and never settles to a fixed point, although we have not observed such behavior in the following simulations.

4.4.5 LagONN Modular Architecture and Scaling

Consider a larger Boolean formula f_B with N Boolean variables and M clauses $C_m = l_1^m \bigvee l_2^m \bigvee l_3^m$ defined as:

$$f_B = C_1 \bigwedge C_2 \bigwedge \ldots \bigwedge C_{M-1} \bigwedge C_M \tag{4.43}$$

with $l_j^m \in \{x_1, \ldots, x_N, \overline{x_1}, \ldots, \overline{x_N}\}$. The 3-SAT instance can be mapped to LagONN modules where a module m corresponds to a clause C_m, and each literal l_j^m corresponds to an input port of a module m (Fig.4.12a). There is no need to implement the AND operation "$\bigwedge$" between two clauses as every Lagrangian oscillator evolves to satisfy its corresponding clause and thus, the

whole network tries to maximize the number of TRUE clauses in f_B. We connect the LagONN modules as follows:

1. Consider two clauses m and n that have a literal in common, i.e. $\exists (k,l) \in \{X,Y,Z\}^2 \, l_k^m = l_l^n$ or $l_k^m = \overline{l_l^n}$. Then, we connect input ports k and l together with a wire. This means that variables ϕ_k^m and ϕ_l^n are reduced to a single variable ϕ_{x_j}, $j \in \{1,...,N\}$ as the two oscillators have shorted inputs.
2. All reference oscillators are connected with a wire.

Repeating this procedure for every pair of clauses, the total LagONN function becomes:

$$L_T(\phi_x, \phi_\lambda) = \sum_{m=1}^{M} \vec{u_\lambda^m}.\vec{Z}_m(\phi_x) \tag{4.44}$$

where ϕ_x is now the vector of phases corresponding to the Boolean variables:

$$\phi_x = (\phi_{x_1}, \phi_{x_2}, ..., \phi_{x_{N-1}}, \phi_{x_N})^T \tag{4.45}$$

and ϕ_λ contains all the Lagrangian variables:

$$\phi_\lambda = (\phi_{\lambda_1}, \phi_{\lambda_2}, ..., \phi_{\lambda_{M-1}}, \phi_{\lambda_M})^T \tag{4.46}$$

Again, the objective is to reach $L_T = 0$ with $Z_m = 0$ for all clauses so that f_B is satisfied. Assembling the M LagONN modules leads to the following dynamics:

$$\begin{cases} \tau \frac{d\phi_x}{dt} = -\nabla_{\phi_x} L_T(\phi_x, \phi_\lambda) = -\sum_{m=1}^{M} \frac{\partial \vec{Z}_m}{\partial \phi_x}.\vec{u_\lambda^m} \\ \tau_\lambda \frac{d\phi_\lambda}{dt} = +\nabla_{\phi_\lambda} L_T(\phi_x, \phi_\lambda) = \sum_{m=1}^{M} \vec{Z}_m.\vec{u_\lambda^m} \end{cases} \tag{4.47}$$

It is important to mention that although there are only N phase variables and M Lagrange variables in the model, our modular construction leads to $5M$ oscillators in practice. As in general for Max-3-SAT, $M > N$, this approach leads to a high number of oscillators. However, it has the following advantages:

1. It avoids implementing large synaptic arrays as synapses are contained in modules.
2. It maintains the synaptic high-order interactions locally inside the modules.

In contrast, a programmable fully-connected design with $N+M$ oscillators would require a $(N+M)^2$ synaptic array to support any f_B instance. Moreover, it seems very challenging to propagate the high-order interaction throughout the whole synaptic array as not directly possible with a standard array laid out as a 2D grid. The comparison between the modular and fully connected LagONN architectures scaling is summarized in Table 4.4.

LagONN Architecture	Oscillators	Size of Synaptic Array	High-Order Interaction Worst Distance
Fully-Connected	$N+M+1$	$(N+M)^2$	$N+M$ (global)
Modular	$5M$	4^2/module (clause)	4 (local)

Table 4.4: *Comparison between Two Possible LagONN Architectures.*

An example of a modular LagONN network consisting of 6 modules is shown in Fig.4.12. There are 3 variables whose input lines are laid out vertically. Each module connects to the input lines according to its literals. As soon as the modules are turned on, each module acts on the input lines X_1, X_2, and X_3 to satisfy its corresponding clause C_m. The optimization process happens in real-time and in parallel until the modules find a common variable assignment that satisfies f_B or they reach a spurious fixed point. Fig.4.12b shows an example of simulation where the Lagrangian oscillators are initially 'OFF', and then 'ON' for $t \geq 2000$. Unlike the single-clause case of Fig.4.11c, activating the Lagrangian oscillator brings all the phases to binary values which satisfy f_B. Fig.4.12d indeed shows that all complex variables Z_m reach the desired state $Z_m = 0$, inducing $C_m = TRUE$ as ϕ_X, ϕ_Y, and ϕ_Z are binary (Proposition 4.2). When the number of clauses increases such as in this example, simulations almost always converge toward binary phase values for ϕ_X, ϕ_Y, and ϕ_Z.

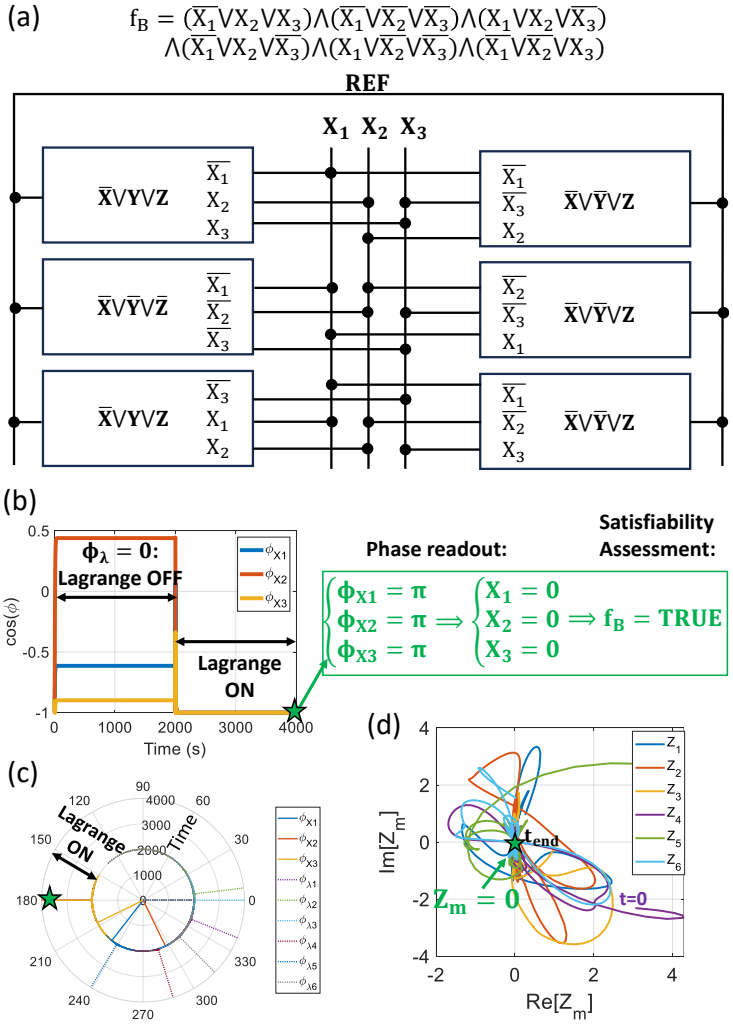


Figure 4.12: *a) Example of LagONN network for solving a formula f_B with 6 clauses. b) Simulation with $\tau_L = \tau$ where Lagrangian oscillators are OFF for $t < 2000$ and ON for $t \geq 2000$. Measuring the phases gives the Boolean assignment which satisfies f_B in this example. c) Same simulation showing the 6 Lagrangian phases. d) Evolution of the 6 corresponding $Z_m \to 0$.*

4.4.6 Simulation Set-up for SATlib

In the following sections, we benchmark LagONN on a set of random 3-SAT instances from SATlib ('uf') that are close to the solubility phase transition ($M/N \approx 4.3$ [230]) and thus are known to be very hard for most of the state-of-the-art SAT solvers [231].

Verilog-A Implementation

As LagONN is a modular architecture, we use a modular simulation approach based on Verilog-A modules to build circuits corresponding to 3-SAT instances from the SATlib library [232]. Each clause circuit is composed of Kuramoto oscillators and synapses with high-order interactions as shown in Fig.4.11 that overall satisfy equations (4.47). The Verilog-A codes and equivalent circuits are shown in Fig.4.13. The oscillator circuit integrates its input current and the result is buffered to its output node. The initial integration value is set by INIT. The synaptic block contains the sinusoidal interaction corresponding to Kuramoto dynamics. The two additional ports DIN, and DOUT, correspond to the additional phase delays coming from potential high-order interactions. Note that the simulated circuit voltages correspond to phases in radians. The synaptic currents do not have a direct meaning but could be thought of as contributing to the instantaneous oscillating frequency, i.e. the derivative of the phase. The oscillation frequency is set to 1 Hz so that a second corresponds to an oscillation cycle. Hence, the physical time can be estimated by multiplying the simulation time by the oscillating period of a given device.

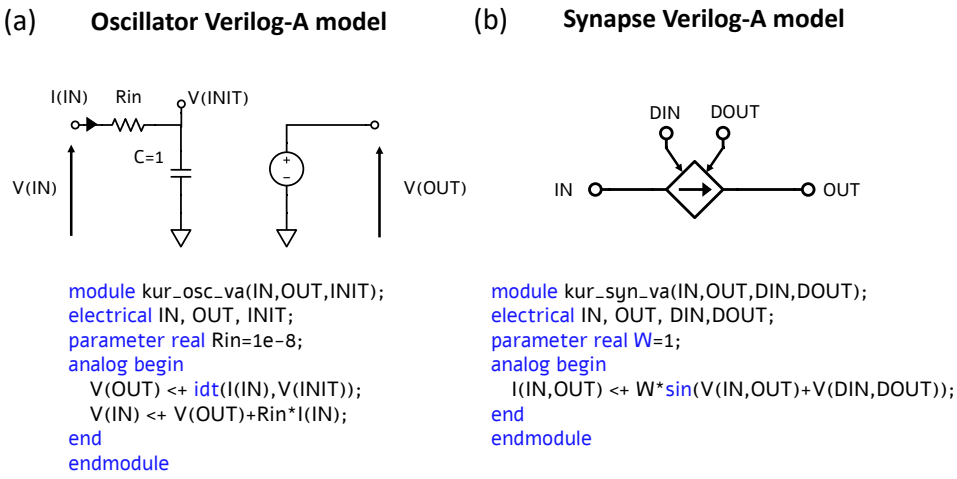


Figure 4.13: *a) Oscillator Verilog-A model. The small input resistance is used to avoid short warnings from the solver. b) Synapse Verilog-A model.*

Cost Function

From the SATlib ".cnf" files, we build a netlist of clause circuits interconnected according to the Boolean formula as shown in Fig.4.12. As there is not any straightforward Lyapunov function for the system, we monitor the number of unsatisfied clauses in real-time using a custom cost

function $\kappa(\phi)$ defined as:

$$f_B = C_1 \bigwedge C_2 \bigwedge \ldots \bigwedge C_{M-1} \bigwedge C_M$$
$$\longrightarrow \kappa(\phi) = K_1 + K_2 + \ldots + K_{M-1} + K_M$$

where $K_m(\phi) = l_1^m\, l_2^m\, l_3^m$ with literals $l_j^m = 0.5\left(1 \pm \tanh(\beta \cos\phi_j^m)\right) \in [0,1]$. The sign that weights the cosine term depends on whether the literal l_j^m corresponds to a positive x_j (-) or negated variable $\overline{x_j}$ (+). This way, $K_m(\phi) = 0$ if there is a variable assignment that satisfies the clause C_m. Consequently, f_B is true if $\kappa(\phi) = 0$. The tanh function is used to round the phases to spin values so we only need to know on which side of the unit circle each phase is to assign it to a Boolean variable. This is equivalent to rounding phases to the nearest multiples of π. For a sufficiently high β-value, we then have $\kappa(\phi) = N_{unsat}$, i.e. $\kappa(\phi)$ counts the number of unsatisfied clauses. Note that with the proposed rounding procedure, a clause C_m is true if and only if $K_m(\phi) < 0.5^3 = 0.125$ ($K_m(\phi) = 0.125$ when $\forall j\ \cos\phi_j^m = 0$). Thus, if $\kappa(\phi) < 0.125$, f_B is true and we use this value as a threshold to stop the LagONN search (Fig.4.14a).

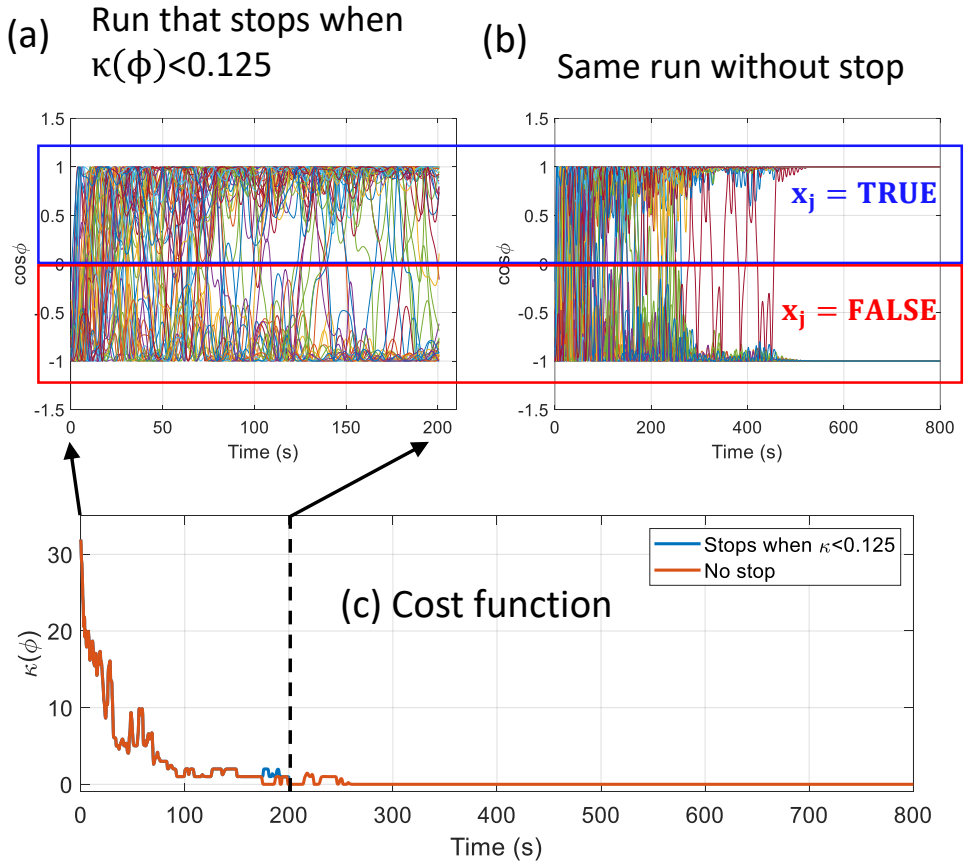


Figure 4.14: *Example of LagONN dynamics for SATlib instance 'cnf50-02' (50 variables and 218 clauses). a) The dynamics stop when $\kappa(\phi) < 0.125$. b) Same run without the stop. Waiting for the full phase convergence towards 0 and π is approximately twice as long as the time for reaching $\kappa(\phi) = 0.125$ in the first case. c) Cost function $\kappa(\phi)$ for both runs. We observe a trajectory deviation around 180 s which is probably due to the accumulation of numerical errors.*

Computation Time

At the time t^* where $\kappa(\phi) = 0.125$, LagONN phases are just about to reach the optimal sides of the unit circle that provide the optimal Boolean assignment. Thus, t^* is the minimum time for LagONN to find the solution. If the dynamics continue, we observe that phases settle to 0 or π after some time $t > t^*$ (Fig.4.14b). We chose t^* as the metric to assess LagONN's computation time as in practice, there would probably not be any circuit to monitor the cost in real-time, and a solution would rather be obtained by sampling phases at various times. If $\kappa(\phi) \geq 0.125$ (sub-optimal case), the simulation continues until it reaches the user-defined final time. In this case, we extract the computation time as the time needed for $\kappa(\phi)$ to settle within 0.5% of its final value.

Solver

We run LagONN transient simulations using the SPECTRE solver in the Virtuoso software from Cadence [233]. The numerical simulation employs the parameters listed in Table 4.5. They are the default ones used by the simulation tool, except for the integration method that we set to Euler's method to speed up simulations. At each time step, the solver checks that the difference of two successive Newton-Raphson (NR) iterations is bounded by $Tol_{NR} = abstol + reltol * Ref$. We choose Ref='pointlocal' so that the errors are evaluated "at each node relative to the current value of that node" [233] for a good accuracy. Finally, a new integrated point is accepted if the local truncation error (LTE) between the computed and predicted points (polynomial extrapolation) is smaller than $Tol_{LTE} = Tol_{NR} * LTEratio$. We further discuss the trade-off between speed and precision later in the chapter.

SPECTRE Solver							LagONN		
Integration	Maxstep	LTEratio	Reltol	Vabstol	Iabstol	Ref	T_{osc}	τ	τ_λ
Euler	$T_{sim}/10$	3.5	1m	1 μV	1 pA	pointlocal	1 s	1 s	10 s

Table 4.5: *LagONN Simulation Parameters.*

4.4.7 LagONN Accuracy and Runtime Scaling with SATlib

Here, we assess the performances of LagONN using hard random instances from SATlib [232] with various sizes $(N,M) \in \{(20,91),(50,218),(100,430),(200,860)\}$. For each (N,M), we select the 10 first 3-SAT instances from the library. Then, we run Monte Carlo simulations consisting of 10 transient simulations per instance with random initial phase values sampled according to a low-discrepancy sequence. Hence, there are 100 trials per (N,M). LagONN accuracy depends on individual instances as shown in Fig.4.15 with two examples of SATlib instances having 100 variables and 430 clauses. For 10 trials, the solver found 100% of optimal solutions for instance 'cnf100-06' whereas it settled to near-optimal solutions for 'cnf100-02' with at best $\kappa(\phi) = N_{unsat} = 1$. Moreover, some trials took up to 500k cycles to settle compared to 800 cycles for the 'simple' cnf100-06 instance.

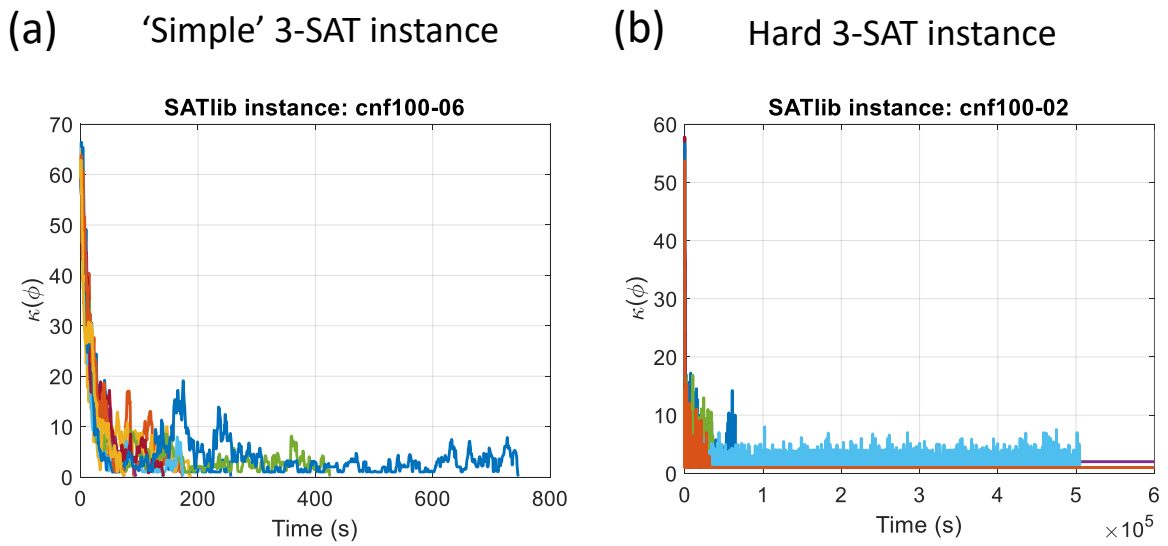


Figure 4.15: *Example of LagONN dynamics for SATlib instances a) 'cnf100-06' and b) 'cnf100-02' (both 100 variables and 430 clauses). For 10 trials, LagONN always finds an optimal solution for the first instance but never finds it for the second instance (one remaining unsatisfied clause in the best case).*

Fig.4.16a shows the cost $\kappa(\phi)$ and the computation time for each trial. Clearly, the computation time increases with N while the cost does not seem to vary much with N. In fact, for this experiment, Fig.4.16b indicates that for all sizes at least 25% of trials found an optimal solution and 75% of trials have $\kappa(\phi) = N_{unsat} \leq 1$. For the case $(N,M) = (200,860)$, this corresponds to 99.88% of satisfied clauses. Moreover, the constant cost highlights that at least for $N \leq 200$, LagONN undesired fixed points do not become dominant when increasing the size. Fig.4.16c presents the computation time against N in a semi-logarithmic plot. For this experiment, LagONN's computation time does not scale exponentially (true for the three quartiles) and the median values rather follow a polynomial trend as $O(N^4)$ as shown in Fig.4.16d. For some hard instances such as the one shown in Fig.4.15b, 10 trials with random initialization were not sufficient for finding an optimal solution. This suggests that for hard instances, many more trials are required and it is very likely that LagONN computation time scales exponentially in the general case. However, the best non-optimal solutions, i.e. with $N_{unsat} = 1$, can be found as fast as 10^5 cycles for $(N,M) = (200,860)$ in the median case. For oscillators running at 100 MHz, the runtime would be 1 ms only and suitable for real-time optimization.

Despite the preliminary results, we identified several important points regarding the solver's accuracy. First, LagONN dynamics are stiff as the solver time steps were often significantly reduced to sub-second values, sometimes down to 100 ms which corresponds to 1/10th of an oscillating period in the simulation. Second, we used the implicit Euler integration method to increase the simulation speed which is not the most accurate integration method, as shown in Fig.4.14c where the effect of numerical error is visible. As this method is known for showing heavy damping when simulating oscillatory behaviors such as resonators [233], it is possible that the simulated dynamics "smooth out" the real dynamics and mitigate some chaotic behavior. Furthermore, it is still an open question whether the solver's accuracy is related to LagONN's accuracy, i.e. if a higher accuracy would reduce the cost $\kappa(\phi)$, and what would be the impact on LagONN's computation time. As highlighted by Siegelmann and Fishman [49], there is no guarantee in the general case that the discretized version of a continuous system converges to the same fixed points. Hence, more studies are required to further assess LagONN's potential.

An immediate improvement would be to simulate the non-modular LagONN version (Table 4.4) having much fewer oscillators ($N+M+1$ compared to $5M$) to speed up simulations while using a better integration method such as a second-order one like Gear's.

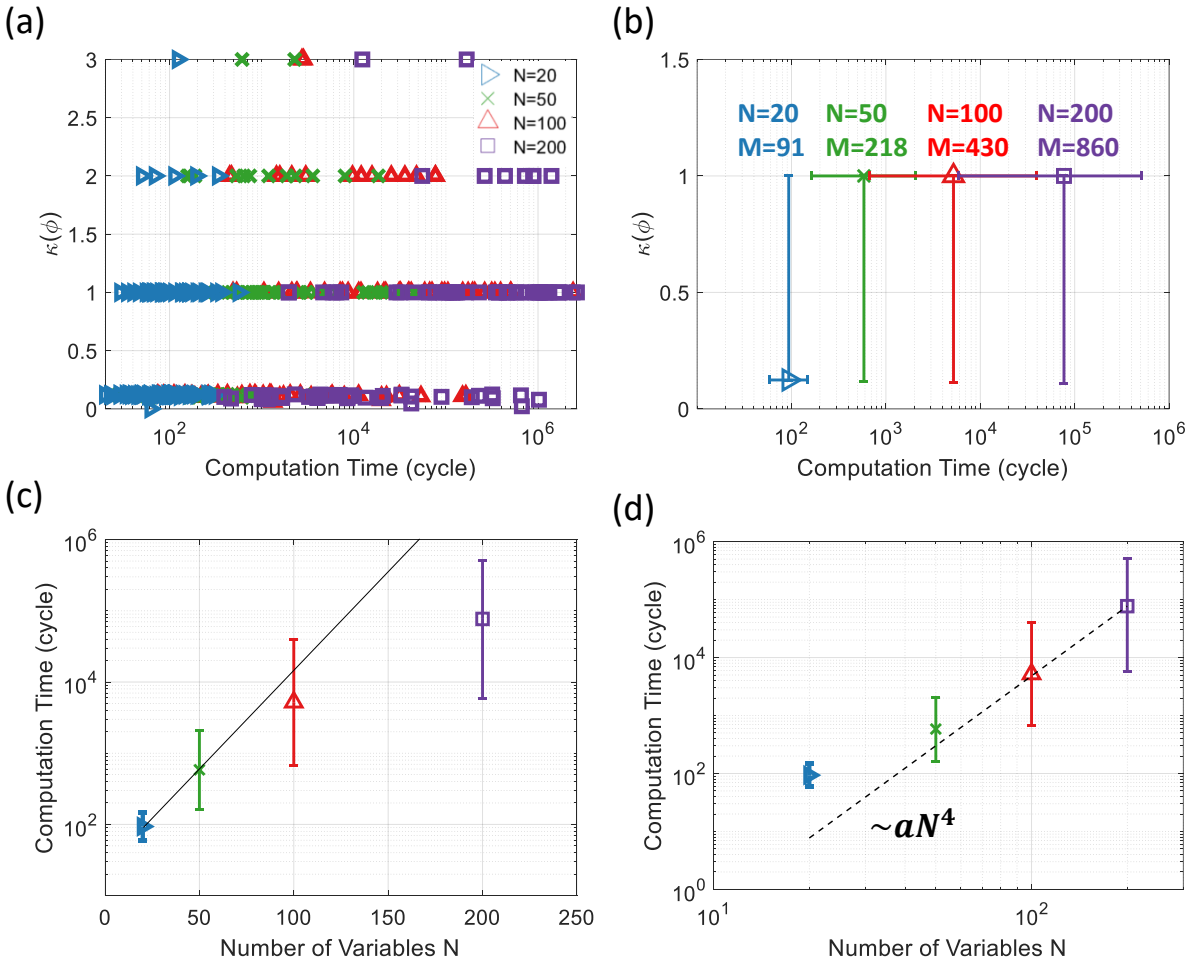


Figure 4.16: *LagONN scaling study with SATlib instances. a) Cost $\kappa(\phi)$ vs. computation time measured for each trial. b) Same results showing quartile data points. c) Computation time vs. N in a log-linear plot. The three quartile values do not scale exponentially. d) Same data in a log-log plot. The preliminary results suggest a polynomial scaling as $O(N^4)$. In the fitted curve aN^4, $a = 4.813\ 10^{-5}$.*

4.5 Discussions

4.5.1 SKONN, Max-cut, and 2D Goemans-Williamson

In this chapter, we first saw that SKONN has unique computational properties related to its phase binarization property (Proposition 4.1) and has a close relationship with the Goemans-Williamson (GW) algorithm for solving Max-cut. Independently, Erementchouk et al. [210] recently proposed an Ising machine model labeled GW2, as "2D Goemans-Williamson" whose cost function is proportional to SKONN's energy landscape. Essentially, SKONN's energy has the same expression as GW's expected cut value except that variables are 2-dimensional rather than N-dimensional for GW. As highlighted by the authors in [210] for GW2 and adapted here for SKONN, given a phase state ξ, the energy "can be regarded as the average size of cut obtained by random rounding of ξ". Hence, according to this interpretation, the deeper the energy minimum is, the more likely it is that its corresponding phase state produces a high-quality cut when rounding phases to the nearest multiple of π.

Interestingly, the authors have derived a particularly interesting property expressed in Theorem 3.3 [210] which states that "rounding a non-binary critical point with respect to different rounding centers produces binary states yielding the same cut". Hence according to the authors, GW2 intrinsically produces the best rounding of a fixed point which means for SKONN that the best rounding *does not* depends on the reference oscillator's choice. Intuitively, we think that the observed "plateaus" in SKONN's energy landscape are a consequence of this property since moving on the plateau does not change the cut value (see for instance Fig.4.1). As a future work, it would be interesting to validate this property experimentally by checking that a SKONN phase fixed point does not produce a better cut when changing the reference oscillator. If true, it would speed up the time-to-solution and prevent checking various possible references before picking up the best cut as it is done in the CirCut algorithm [112]. Moreover, the authors in [210] have suggested that running GW2 after adding small perturbations to a binary state cannot worsen the cut value. This mechanism could constitute an interesting local search technique to improve SKONN's accuracy.

4.5.2 Lagrangian ONN

Our study with Lagrangian oscillators performing *gradient ascent* highlighted a promising deterministic mechanism to escape local minima in ONNs. However, in principle, Lagrangian neural networks (LNN) do not stop until reaching a state that satisfies the constraints [114, 225]. Hence, the proposed LagONN architecture is a "pseudo" LNN in the sense that LagONN can settle to spurious fixed points where constraints are not met ($N_{unsat} \geq 1$), as highlighted in Table 4.3. One way of forcing the ONN to stop if and only if constraints are met is to introduce a second Lagrangian variable at the expense of a more complicated circuit. Nevertheless, the preliminary simulations indicate that with a single Lagrangian oscillator, LagONN can already find Boolean assignments such that $N_{unsat} \leq 1$ in 75% of trials for hard SATlib instances up to 200 variables and 860 clauses. Intuitively, we have set the Lagrangian oscillator to evolve slower than the remaining oscillators so that the latter "have time" to relax to minima in the augmented energy landscape which is shaped in real-time by the Lagrangian phase variable with speed set by τ_λ. Thus, we used $\tau_\lambda = 10\,\tau$ for all our SATlib simulations. However, it is an open question

whether there is an optimal value for τ_λ and if it should be scaled with the 3-SAT instance size.

Finally, our 3-SAT mapping to LagONN used high-order interactions between oscillators. How to implement them remains an open question too but luckily, these challenging coupling elements are locally confined in LagONN modules, i.e. there is no global high-order connection in the system. For future work, we are investigating if we can generalize SKONN's pairwise synaptic digital propagation to high-order interactions using elementary digital circuits like latches.

4.6 Conclusion

This chapter explored how to solve NP-hard combinatorial optimization problems (COPs) using ONNs. After introducing the ONN-to-Ising mapping for binary phases, we further discussed how to solve the NP-hard Max-cut problem using the free ONN dynamics, i.e. without forced phase binarization. Particularly, we highlighted the intimate link between SKONN, the architecture developed in Chapter 3, and the best approximation algorithm found by Goemans and Williamson (GW). Moreover, we analytically showed that for neurons with an odd number of synapses, SKONN dynamics tend to binarize phases when reaching a fixed point which facilitates rounding phases to binary variables. The phase binarization property was experimentally validated using our 16-neuron SKONN IC which produces median cut values larger than 0.9x Cut_{GW}. Moreover, experiments indicated that the IC finds its best cut after 15 oscillation cycles only, which constitutes a runtime improvement of more than 4 orders of magnitude at only f_{osc}=1 MHz compared to GW executed on a laptop. Large-scale simulations using random graphs and the G-set benchmark for Max-cut suggested SKONN's runtime scales as $O(\log N)$ while having a similar or better accuracy than GW.

Next, we further explored how the ONN free dynamics can be harnessed to solve the NP-hard N-city traveling salesman problem (TSP). Compared to the state-of-the-art Hopfield/Ising approach having N^2 variables, we proposed a simple fully-connected ONN with N oscillators where inter-city distances are mapped to negative coupling coefficients, and where the ONN phase permutation gives the TSP tour. While the approach does not accurately embed the TSP tour distance in the ONN energy function, simulations indicate that the ONN can find solutions at most 50% longer than the optimal tour distance for $N < 45$. However, in many instances, the phases are clustered which would challenge the phase readout in a real implementation.

Finally, we studied how to escape local energy minima in ONNs using the technique of Lagrange multipliers from the field of constrained optimization. Taking the NP-hard Max-3-SAT problem as an example, we constructed a dedicated Lagrangian-ONN (LagONN) architecture to escape local minima in a deterministic manner. Each 3-SAT clause corresponds to a LagONN module of 5 oscillators locally connected with high-order synapses. Overall, the architecture is modular where modules are interconnected to represent the global 3-SAT formula. Using one of the hardest sets of instances from the SATlib benchmark, LagONN simulations for up to 200 variables and 860 clauses indicate that 75% of trials find a Boolean assignment with at most a single unsatisfied clause. The simulation results further suggest a polynomial scaling of LagONN's computation time as $O(N^4)$, with a median runtime of 100k oscillation cycles for $N = 200$. For a physical LagONN hardware with oscillators running at 100 MHz, the corresponding runtime would be around 1 ms only.

CHAPTER 5

CONCLUSION AND OUTLOOK

5.1 Conclusion

HUMANITY is currently producing massive amounts of information that often need to be processed in real-time. To face this "data deluge", technological innovation is key for accelerating digital computing and improving its efficiency. Despite the mind-blowing progress of semiconductor technology and the exponential growth of the number of transistors per chip, digital computers still have trouble solving particular tasks such as optimization problems and running artificial intelligence algorithms (AI) that are now ubiquitous. The main limitation originates from the architecture of digital computers that separate memory and processing units, thus producing undesirable data transport and energy waste. Another constraint concerns digital computing itself which computes via the combination of many sequential binary operations that are implemented by arithmetic circuits with a high number of transistors. To overcome these limitations, researchers are developing alternative computing paradigms such as brain-inspired neuromorphic computing where **1)** processor and memory are interlinked (neurons and synapses) and **2)** computation occurs "naturally" as a consequence of physical laws rather than arithmetic circuits. This dissertation explored an unconventional neuromorphic computing paradigm based on analog oscillatory neural networks (ONN) that compute in the phase domain to solve AI and combinatorial optimization problems (COP). Although ONNs have been studied for a few decades, it remains several challenges that have been the focus of this thesis:

- **How to map conceptual models to physical ONNs?** Despite the simplicity of ONN architectures, their dynamics can be very complex and it is not always straightforward how to map a given computational model such as a Hopfield neural network (HNN) to physical ONNs. Although it is known that two-coupled oscillators can converge to two different binary states like HNN, the relationship between the **1)** initial phase state, **2)** coupling element value, and **3)** final phase state is generally empirically determined. This is problematic for scaling up the ONN as it cannot be trained and programmed automatically with high accuracy. In Chapter 2, we proposed a mapping procedure between HNN to ONN based on vanadium dioxide oscillating neurons (VO_2) coupled by resistive devices. When increasing the network size, the mapping allows ONN training for pattern recognition and its performance assessment from device to architecture. Furthermore, this study highlighted a high sensitivity of the proposed resistively coupled VO_2 ONN

regarding coupling resistance variations, which suggests implementation challenges at a large scale.

Chapter 2 further underlined the challenge of controlling analog phase dynamics. To implement a phase-based ONN at a large scale, we believe it is crucial to understand the physical dynamics such as having an ONN phase model that is exploitable to compute with high accuracy, e.g. which provides the weight mapping to coupling values but also sheds light on the underlying ONN energy function. Analogous to neural activation functions in deep neural networks, the shape of the ONN energy function determines the ONN properties and performances for particular tasks. For instance, the custom ONN architecture proposed in Chapter 3 is particularly suited to solve the NP-hard Max-cut COP as its energy is closely linked to the Max-cut classical cost function.

- **How to design analog ONNs at a large scale?** Analog ONNs exchange continuous variables such as synaptic currents that can be challenging to propagate at a large scale without altering the signals, in contrast with digital information. Chapter 3 proposed a novel mixed-signal ONN architecture, labeled SKONN, that computes in the analog domain and propagates the information digitally to facilitate synaptic signal propagation. Feeding a neuron output square wave to synaptic capacitors induces the injection of charges at the oscillator input node, which in turn provoke controllable phase shifts. Two fully-connected SKONN prototypes were demonstrated: a 9-neuron SKONN on a printed circuit board and a 16-neuron SKONN integrated circuit using a 65 nm technology. The proof of concepts confirmed SKONN's intimate link with NP-hard COPs and in particular with the Max-cut problem and its best approximation algorithm found by Goemans and Williamson (GW). Chapter 4 further highlighted SKONN's potential for solving Max-cut at a large scale with iso-accuracy compared to GW, while suggesting an advantageous logarithmic scaling of the settling time with the network size.

- **How to escape local energy minima?** ONNs are also known for their close relationship with the Ising model which has a great computational interest because finding the ground states of the Ising Hamiltonian is NP-hard. If a machine can efficiently solve the Ising problem, then it can efficiently solve any problem in NP. ONNs programmed to solve the Ising problem are called oscillatory Ising machines (OIM) and are promising thanks to their high parallelism and low-energy footprint. However, the OIM energy function that embeds the Ising Hamiltonian is non-convex and contains local minima where the OIM can get stuck, thus limiting the accuracy. In Chapter 4, we applied a technique from mathematical optimization using Lagrange multipliers to force the ONN to escape local minima by going uphill in an augmented energy landscape. Harnessing high-order interactions between oscillators, we proposed a modular Lagrangian-ONN (LagONN) architecture to solve the NP-hard Max-3-SAT COP. Compared to other analog models, LagONN's power is bounded as synaptic weights are fixed, and phase variables are naturally bounded. Preliminary simulations with up to 200 variables and 860 clauses indicate that in 75% of trials, LagONN finds Boolean assignments for which at most a single clause remains unsatisfied while suggesting a median computation time that scales polynomially with the number of variables.

5.2 Future Research Directions

This dissertation only explored a subset of possible ONN computing by restricting the oscillators to have uniform frequencies and computing in the phase domain. However, some authors have shown that ONN can indeed compute in the frequency domain which is not based on the ONN energy minimization but rather by harnessing the ONN non-linearity and short-term memory to process time series [94, 109]. Thanks to its real-time processing and energy efficiency, there is potential for ONN to process time-dependent analog sensory data within devices that have limited resources, and avoid some of the energy-hungry analog-to-digital conversions.

Moreover, phase-based ONNs have shown promising results for solving NP-hard combinatorial optimization problems thanks to their intimate link with the Ising model. The community has pushed the boundaries of large-scale ONN implementation from 100 fully connected oscillators in 2018 [89] to 560 [74] and 600 [139] oscillators in 2021, up to 1440 [140] and 1968 [76] oscillators very recently. If this "Moore-like" ONN scaling goes on, ONN systems could potentially speed up the resolution of real-life large-scale problems intractable for digital computers. However, noticing that fully connected ONNs are not implementable at a large scale, we believe that only combining ONNs with more classical algorithmic approaches could be a game changer. Such a hybrid approach was recently proposed by Mallick et al. [139] where ONNs provide very fast partial or near-optimal solutions prior to final algorithmic optimization. For instance, ONNs could solve sub-problems in divide-and-conquer approaches such as for large-scale traveling salesman problems, or even be co-integrated with branch and bound algorithms.

Finally, there are some exciting research topics that we think could further push the ONN computational capabilities. Two are related to improving the ONN solution to an optimization problem that is generally stuck at a local energy minimum. First, it would be interesting to transpose the technique of simulated annealing (SA) [63] to phase-based ONN. Although it has been experimentally shown that modulating a sub-harmonic signal can act similarly to decreasing the temperature in SA [69, 138], a theoretical framework including stochasticity is missing and how to operate such modulation to stochastically explore the energy landscape before converging to a good local solution remains empirical.

Second, we think the ONN gradient property can be combined with gradient ascent in an augmented energy landscape to escape local minima, similarly to the dynamics of several analog solvers [226, 228]. Our first attempt using additional Lagrange oscillators (LagONN) has room for improvement since more Lagrange oscillators could be used to guarantee convergence towards optimal solutions while exploring other types of energy landscapes. Finally, harnessing high-order oscillator interactions enables mapping any combinatorial optimization problem to the hardware [207, 234], without adding auxiliary oscillators that potentially create energy minima. The implementation of such high-order synapses for ONN constitutes a promising research topic.

Appendices

CHAPTER A

VO_2-BASED ONN ARCHITECTURE

A.1 Single Oscillator Time Constants

Before studying the dynamics of two coupled oscillators to derive the transition function, we express the time constants of a single oscillator. The oscillating period T_{osc} is given by the sum of the charging and discharging time of the output capacitance:

$$T_{osc} = t_c + t_d \tag{A.1}$$

If we assume abrupt VO_2 transitions we obtain:

$$t_c = \tau_c \log\left[\frac{V^- - V_{std}^{met}}{V^+ - V_{std}^{met}}\right] \tag{A.2}$$

$$t_d = \tau_d \log\left[\frac{V^+ - V_{std}^{ins}}{V^- - V_{std}^{ins}}\right] \tag{A.3}$$

with:

$$V_{std}^{met} = V_{DD}\frac{R_S}{R_S + R_{met}} \tag{A.4}$$

$$V_{std}^{ins} = V_{DD}\frac{R_S}{R_S + R_{ins}} \tag{A.5}$$

and with the two oscillator's time constants given by:

$$\tau_c = \frac{R_S R_{met}}{R_S + R_{met}} C_P \tag{A.6}$$

$$\tau_d = \frac{R_S R_{ins}}{R_S + R_{ins}} C_P \tag{A.7}$$

A.2 Dynamics of Two Coupled Oscillators after Initialization

The dynamics of two coupled oscillators can be expressed as:

$$\begin{cases} C_P \frac{dV_{out1}(t)}{dt} = \left(V_{DD1}(t) - V_{out1}(t)\right) G^1_{VO_2}(t) - \frac{V_{out1}(t)}{R_S} + I_{c1} \\ C_P \frac{dV_{out2}(t)}{dt} = \left(V_{DD2}(t) - V_{out2}(t)\right) G^2_{VO_2}(t) - \frac{V_{out2}(t)}{R_S} + I_{c2} \end{cases} \tag{A.8}$$

To express the transition function $\zeta(R_C)$, we study the oscillators' dynamics until the first oscillator reaches the lower threshold V^- (before IMT). Then, the voltage difference between the two oscillators (set by the initial delay and R_C) determines the final phase state. We analytically solve (eq.A.8) for $t \geq t_{on}$ (when the two oscillators are coupled) until one output voltage reaches V^-. During our initialization procedure, we turn-on the two oscillators via V_{DD} and we couple them at time $t_{on} = \Delta t_{init} + t_c$. Therefore, both their output capacitors start to discharge. For this reason, at t_{on} the initial output voltages V^0 are given by the dynamics of the uncoupled oscillators $i \in \{1,2\}$ when both VO_2 devices are in the insulating state: $R^i_{VO_2} = R_{ins}$:

$$V^0 = \begin{bmatrix} V^0_{out}1 \\ V^0_{out}2 \end{bmatrix} = \begin{bmatrix} V_{DD}\frac{R_S}{R_S+R_{ins}}\left[1-\exp\left(-\frac{\Delta t_{init}}{\tau_d}\right)\right] + V^+\exp\left(-\frac{\Delta t_{init}}{\tau_d}\right) \\ V^+ \end{bmatrix} \tag{A.9}$$

It is convenient to rewrite (eq.A.8) in a matrix form to use the formalism from dynamical systems. As both oscillators are in insulating state, we can write the linear equation:

$$C\frac{dV}{dt} = G_A\, V + G_B\, V_{DD} \tag{A.10}$$

with

$$V = \begin{bmatrix} V_{out}1 \\ V_{out}2 \end{bmatrix} \tag{A.11}$$

$$G_A = \begin{bmatrix} -\frac{1}{R_{ins}} - \frac{1}{R_C} & \frac{1}{R_C} \\ \frac{1}{R_C} & -\frac{1}{R_{ins}} - \frac{1}{R_C} \end{bmatrix} \tag{A.12}$$

$$G_B = \begin{bmatrix} \frac{1}{R_{ins}} & 0 \\ 0 & \frac{1}{R_{ins}} \end{bmatrix} \tag{A.13}$$

$$C = \begin{bmatrix} C_P & 0 \\ 0 & C_P \end{bmatrix} \tag{A.14}$$

$$V_{DD} = \begin{bmatrix} V_{DD}1 \\ V_{DD}2 \end{bmatrix} \tag{A.15}$$

We solve the first-order differential equation (eq.A.10) given the initial conditions (eq.A.9). We consider t_{on} as the new time origin: $t - t_{on} \rightarrow t$ to express the solution as:

$$V = \left(\exp\left(C^{-1}G_A\, t\right) - I_d\right) G_A^{-1} G_B V_{DD} + \exp\left(C^{-1}G_A\, t\right) V^0 \tag{A.16}$$

While both VO_2 devices are in insulating state, computing (A.16) leads to:

$$\begin{cases} V_{out}1 = \frac{V^0_{out1}}{2}\left(\exp(-\frac{t}{\tau_d}) + \exp(-\frac{t}{\tau'})\right) + \frac{V^0_{out2}}{2}\left(\exp(-\frac{t}{\tau_d}) - \exp(-\frac{t}{\tau'})\right) \\ \qquad +V^{ins}_{std}\left(1-\exp(-\frac{t}{\tau_d})\right) \\ V_{out}2 = \frac{V^0_{out1}}{2}\left(\exp(-\frac{t}{\tau_d}) - \exp(-\frac{t}{\tau'})\right) + \frac{V^0_{out2}}{2}\left(\exp(-\frac{t}{\tau_d}) + \exp(-\frac{t}{\tau'})\right) \\ \qquad +V^{ins}_{std}\left(1-\exp(-\frac{t}{\tau_d})\right) \end{cases} \tag{A.17}$$

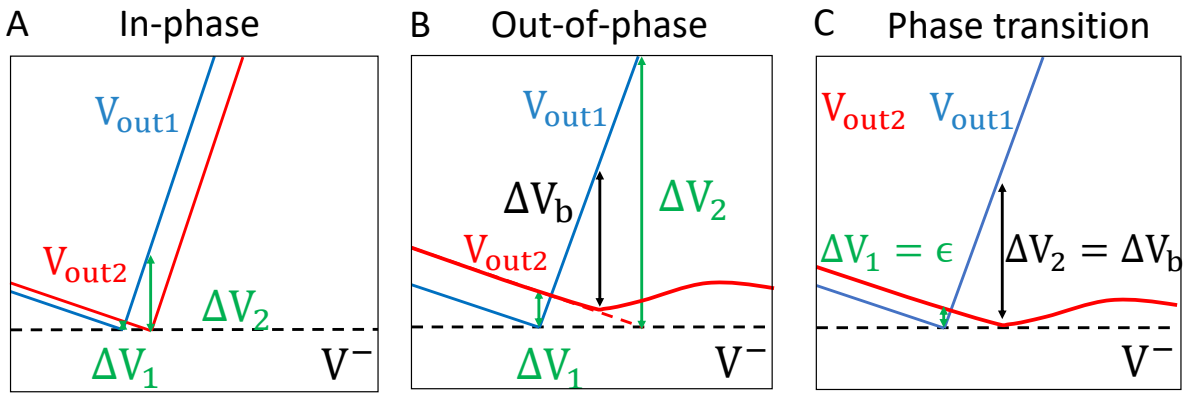


Figure A.1: *Output voltages near IMT threshold V^-. There are two final phase outcomes (A) in-phase state and (B) out-of-phase state. (C) The phase transition occurs when $\Delta V_2 = \Delta V_b$, and consequently when $\Delta V_1 = \varepsilon$.*

The attraction between the two oscillators is characterized by the time constant:

$$\tau' = \frac{C}{\frac{1}{R_S} + \frac{1}{R_{ins}} + \frac{2}{R_C}} \tag{A.18}$$

and we get a compact relation by expressing the difference:

$$\Delta V = V_{out2} - V_{out1} = \left(V^0_{out2} - V^0_{out1}\right) \exp\left(-\frac{t}{\tau'}\right) \tag{A.19}$$

A.3 Final Phase State and Transition Function

In simulations, when the first oscillator is about to reach V^- (Figure A.1), we observe a voltage ε such that:

$$\begin{cases} \Delta V < \varepsilon \rightarrow \Delta\phi_{out} = 0° \\ \Delta V \geq \varepsilon \rightarrow \Delta\phi_{out} = 180° \end{cases} \tag{A.20}$$

Using this observation, we derive the transition function ζ when both equations $\Delta V = \varepsilon$ and $V_{out1} = V^-$ are fulfilled, as ε represents the transition voltage threshold between the two phase domains. To derive ε, we consider the metallic state of the oscillator, i.e. when C_P charges. Fig.A.1A and B illustrate the two scenarios that lead to in-phase or out-of-phase states. ΔV_1 and ΔV_2 are the voltage differences when V_{out1} and V_{out2} reach V^-, respectively. ΔV_b is the minimum voltage difference that pulls up V_{out2} (via synaptic current) and prevents it from reaching the lower threshold V^-. This event is characterized by a "bump" appearing in the transient waveform, which postpones the IMT of oscillator 2. In this case, we have $dV_{out2}/dt = 0$, and using (eq.A.8), we obtain:

$$\Delta V_b = R_C \left(\frac{V^-}{R_{ins}} + \frac{V^-}{R_S} - \frac{V_{DD}}{R_{ins}}\right) \tag{A.21}$$

This configuration, shown in Fig.A.1C, represents the limit case between A and B and determines the phase transition. From this illustration, we express ε as:

$$\varepsilon \approx \frac{s_1}{s_2}\Delta V_b \tag{A.22}$$

with s_1, s_2 slopes of the discharge and charge curves, respectively. We approximate the slopes by considering the dynamic of a single oscillator, and we express them as:

$$\begin{cases} s_1 = \frac{V_{std}^{ins} - V^+}{\tau_d} \exp(\frac{-T_{osc}}{\tau_d}) \\ s_2 = \frac{V_{std}^{met} - V^+}{\tau_c} \end{cases} \tag{A.23}$$

Using (A.19), we express the time t_ε when $\Delta V = \varepsilon$ as:

$$t_\varepsilon = \tau' \log\left(\frac{V_{out2}^0 - V_{out1}^0}{\varepsilon}\right) \tag{A.24}$$

When the capacitor charge is much faster than its discharge ($s_2 >> s_1$), we obtain $\varepsilon \to 0^+$. Hence, we can write $V_{out2}(t_\varepsilon) = V_{out1}(t_\varepsilon) + \varepsilon \approx V_{out1}(t_\varepsilon)$. The transition between the in-phase and out-of-phase domain occurs when:

$$V_{out1}(t_\varepsilon) = V^- \to \frac{V_{out1}(t_\varepsilon) + V_{out2}(t_\varepsilon)}{2} \approx V^- \tag{A.25}$$

By combining this last equation with (eq.A.17) and (eq.A.24), we finally express coupling resistances that describe the phase transition curve as:

$$R_C = 2\frac{R_S R_{ins}}{R_S + R_{ins}} \frac{\log\left(\frac{V^- - V_{std}^{ins}}{V_{out1}^0/2 + V_{out2}^0/2 - V_{std}^{ins}}\right)}{\log\left(\frac{\varepsilon(R_C)}{V_{out2}^0 - V_{out1}^0}\right) - \log\left(\frac{V^- - V_{std}^{ins}}{V_{out1}^0/2 + V_{out2}^0/2 - V_{std}^{ins}}\right)} \tag{A.26}$$

CHAPTER B

SKONN ARCHITECTURE

B.1 SKONN Voltage and Current Dynamics

B.1.1 Neuron Voltage Dynamics

By denoting I_{ij} the input synaptic currents, the voltage dynamics of neuron i can be modeled as follows:

$$\begin{cases} C_L \frac{dV_i^{in}}{dt} = I_{bias}\left(1 - 2V_i^{out}/V_{DD}\right) + \sum_j I_{ij} \\ \tau_H \frac{dV_i^{out}}{dt} = V_{DD} f_H\left(V_i^{in}, V_i^{out}, V_L, V_H\right) - V_i^{out} \end{cases} \tag{B.1}$$

where C_L is the input capacitance, I_{bias} the current that charges and discharges C_L, V_{DD} is the amplitude of V_i^{out}, V_L and V_H are the lower and upper thresholds of the hysteresis block, and τ_H is the time constant linked to the output load of the hysteresis block. The term f_H expresses the output switching with hysteresis behavior. As in [158], one can model the hysteresis behavior using a tanh function with slope γ:

$$f_H = 0.5\left(1 + \tanh\left(\gamma\left(V_i^{in} - V_H - \frac{V_L - V_H}{V_{DD}} V_i^{out}\right)\right)\right) \tag{B.2}$$

When f_H=0, V_i^{out}=0 and C_L charges. When $f_H = 1$, V_i^{out}=V_{DD} and C_L discharges. The two switching occur when V_i^{in}=V_L and V_i^{in}=V_H, respectively. Fig.B.1 shows an example of a numerical solution for the equations (eq.B.1).

In SKONN, the digital output voltage V_j^{out} goes through the synaptic capacitance C_{ij} that creates current spikes holding the phase information ϕ_j. The synaptic spike train can be expressed as follows:

$$I_{ij} = C_{ij}\left(\frac{dV_j^{out}}{dt} - \frac{dV_i^{in}}{dt}\right) \tag{B.3}$$

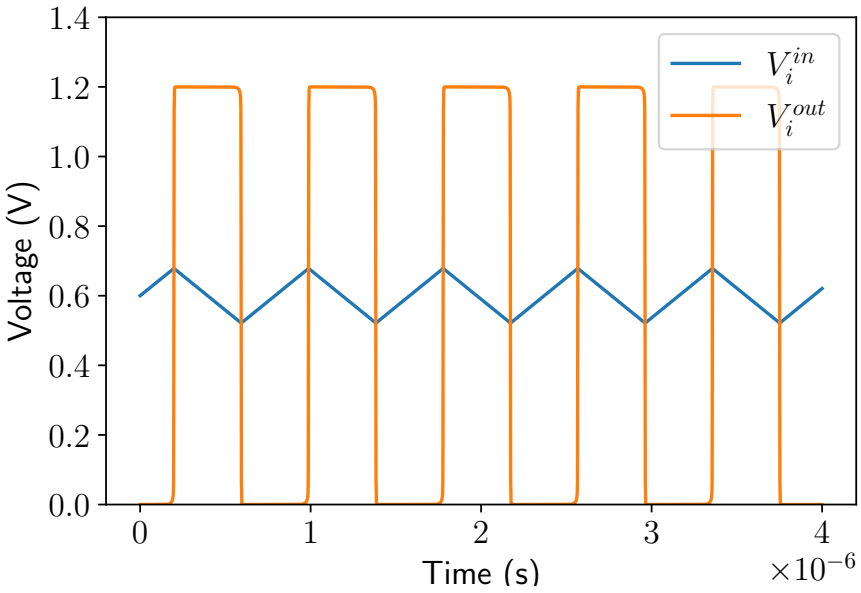


Figure B.1: *Voltage dynamics of a single neuron obtained by solving numerically (eq.B.1). In this example, C_L=500 fF, I_{bias}=200 nA, V_L=0.5 V, V_H=0.7 V, V_{DD}=1.2 V, γ=100 and τ_H=1 ns.*

B.1.2 Voltage dynamics of coupled neurons

By injecting the synaptic current expression in equation (eq.B.1), we obtain:

$$\begin{cases} C_{eq}\frac{dV_i^{in}}{dt} = I_{bias}\left(1-2V_i^{out}/V_{DD}\right)+\sum_j C_{ij}\frac{dV_j^{out}}{dt} \\ \tau_H\frac{dV_i^{out}}{dt} = V_{DD}\,f_H\left(V_i^{in},V_i^{out},V_L,V_H\right)-V_i^{out} \end{cases} \tag{B.4}$$

with C_{eq} the equivalent capacitance:

$$C_{eq} = C_L + \sum_j C_{ij} \tag{B.5}$$

Eq.B.5 indicates that the synaptic capacitances are added to the oscillator load and slow down the charge and discharge of the input node. Large synaptic capacitances could potentially induce heterogeneous frequencies within SKONN and still need to be explored.

B.2 SKONN Phase Perturbation Vector

The PPV is a T-periodic function $\vec{v}(t)$ that quantifies the phase shift of an oscillator subject to a perturbation occurring at time t [193, 194]. One way of computing $\vec{v}(t)$ is to inject a pulsed perturbation to the oscillator at time t, measure the induced phase shift, and normalize by the perturbation's strength [86]. In SKONN, the synaptic current I_{ij} perturbs the triangular oscillation V_i^{in} and the scalar PPV $v(t)$ can be derived by computing the phase shift $d\phi$ when injecting current pulses $I(t') = dQ\,\delta(t'-t)$ with $t \in [0;T[$. From Fig.B.2, we distinguish three cases:

1. $0 < t < T/2$: the perturbed oscillation is shifted toward the left by the same amount of time $-dt$.
2. $T/2 < t < T$: the perturbed oscillation is shifted toward the right by the same amount of time $+dt$.
3. $t \in \{0; T/2\}$: the time shift is undefined as V_i^{in}'s slope is undefined (not differentiable).

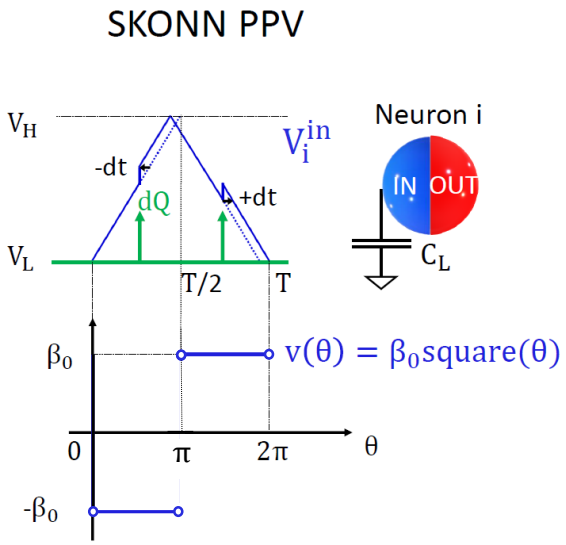


Figure B.2: *SKONN Phase Perturbation Vector (PPV). The injection of a charge dQ induces a time shift $\pm dt$, which in turn creates a phase shift $d\phi = 2\pi\, dt/T$. The phase shift sign changes when there is a change in V_i^{in}'s slope.*

Injecting dQ to C_L induces a voltage jump $dV = dQ/C_L$. This results in a time delay $-dt$ and $+dt$ when C_L charges and discharges, respectively, with $|dt| = C_L dV/I_{bias}$. The amount of phase shift can then be expressed as $d\phi = 2\pi\, dt/T$. The oscillation period T is expressed by $T = 2C_L\Delta V/I_{bias}$, with $\Delta V = V_H - V_L$. Finally, merging the equations leads to

$$\frac{d\phi}{dQ} = \pm\frac{\pi}{C_L\Delta V} = \pm\beta_0 \tag{B.6}$$

which is the phase shift caused by the injection of 1 coulomb. The phase shift sign depends on whether the charge is injected during the charge or discharge of the triangular waveform. Considering the three previous cases and changing the time variable t to phase θ, we express SKONN's PPV as follows:

$$v(\theta) = \beta_0\, \text{square}(\theta) \tag{B.7}$$

where:

$$\text{square}(\theta) = \begin{cases} -1, & \text{if } 0 < \theta < \pi \\ +1, & \text{if } \pi < \theta < 2\pi \end{cases} \tag{B.8}$$

B.3 SKONN Phase Dynamics

B.3.1 Two Coupled Oscillators

The phase dynamics of a single oscillator of frequency $\omega_0 = 2\pi/T$ are:

$$\frac{d}{dt}\phi(t) = \omega_0 \tag{B.9}$$

When the oscillator receives a pre-synaptic signal $\vec{b}(t)$, it undergoes a time shift $\alpha(t)$ associated with the perturbation $\vec{b}(t)$. If the variation of the oscillating amplitude remains small [193], $\alpha(t)$ dynamics can be expressed as follows:

$$\frac{d}{dt}\alpha(t) = \vec{v}(t+\alpha(t)).\vec{b}(t) \tag{B.10}$$

where $\vec{v}(t)$ is the T-periodic Phase Perturbation Vector (PPV) associated with the oscillator; and describes the phase sensitivity of the oscillator under injections at different nodes. In our case, we consider scalars $b(t)$ and $v(t)$ as the pre-synaptic signal $b(t)$ is injected into a unique input node. As $b(t)$ also oscillates at frequency ω_0 and with phase $\phi_b(t) = \omega_0 t$, we introduce $\Delta\phi(t) = \phi(t) - \phi_b(t) = \omega_0\alpha(t)$ that expresses the phase difference between post and presynaptic signals. The latter can be considered as the reference as it is driving the oscillator. To simplify equations, we define the 2π-periodic PPV and perturbation as $v^{2\pi}(\omega_0 t) = v(t)$ and $b^{2\pi}(\omega_0 t) = b(t)$, respectively, like in [69]. The phase dynamics become:

$$\frac{d}{dt}\Delta\phi = \omega_0\, v^{2\pi}\big(\phi_b + \Delta\phi\big)\, b^{2\pi}\big(\phi_b\big) \tag{B.11}$$

We assume that under weak coupling, the phase difference $\Delta\phi$ evolves slowly compared to the presynaptic phase ϕ_b and it is common practice to average out $\Delta\phi$ over one period [69, 86]:

$$\begin{aligned}&\frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{d}{dt}\Delta\phi\, d\phi_b \approx \frac{d}{dt}\Delta\phi \\ &= \frac{1}{T}\int_{-\pi}^{\pi} v^{2\pi}\big(\phi_b + \Delta\phi\big)\, b^{2\pi}\big(\phi_b\big)\, d\phi_b\end{aligned} \tag{B.12}$$

We saw previously that a pre-synaptic signal consists of current pulses that are aligned with the rising and falling edges of the digital pre-synaptic voltage. Then, we consider the case where $b^{2\pi}(\theta)$ consists of a train of Dirac pulses:

$$b^{2\pi}(\theta) = \sum_{n=0}^{\infty} p(\theta - n2\pi) \tag{B.13}$$

with:

$$p(\theta) = Q\big(\delta(\theta) - \delta(\theta-\pi)\big) \tag{B.14}$$

Under this assumption, (eq.B.12) becomes:

$$\frac{d}{dt}\Delta\phi = \frac{Q}{T}\Big(v^{2\pi}(\Delta\phi) - v^{2\pi}(\Delta\phi+\pi)\Big) \tag{B.15}$$

In SKONN, the analog input oscillation is a symetric triangular waveform that has a simple 2π-periodic PPV expressed as follows:

$$v^{2\pi}(\theta) = \beta_0 \,\text{square}(\theta) \tag{B.16}$$

where:

$$\text{square}(\theta) = \begin{cases} -1, & \text{if } 0 < \theta < \pi \\ +1, & \text{if } \pi < \theta < 2\pi \end{cases} \tag{B.17}$$

and β_0 is the phase shift induced by the injection of 1 coulomb to the oscillating node. Finally, we express the phase dynamics of the driven oscillator:

$$\frac{d}{dt}\Delta\phi = 2\beta_0 \frac{Q}{T} \,\text{square}(\Delta\phi) \tag{B.18}$$

We notice that the average of the phase dynamics are very similar to the Kuramoto model except that its sinusoidal interaction term is replaced by a *saturated* square function in our case.

Proposition B.1. *If the injected charge $Q \neq 0$ then the two SKONN oscillators admit a unique stable fixed-point $\Delta\phi^* = (\phi_i - \phi_j)^*$ such that:*

$$\Delta\phi^* = \begin{cases} 0, & \text{if } Q > 0 \\ \pi, & \text{if } Q < 0 \end{cases} \tag{B.19}$$

Proof. The proof consists in finding a Lyapunov function for the dynamics (eq.B.18). Consider the bounded continuous Lyapunov function:

$$E = 2\beta_0 \frac{Q}{T} \text{triangle}(\Delta\phi) \tag{B.20}$$

with:

$$\text{triangle}(\theta) = \begin{cases} \theta - \pi/2, & \text{if } 0 \leq \theta \leq \pi \\ 3\pi/2 - \theta, & \text{if } \pi \leq \theta \leq 2\pi \end{cases} \tag{B.21}$$

We have :

$$\frac{\partial E}{\partial \Delta\phi} = -2\beta_0 \frac{Q}{T} \text{square}(\Delta\phi) = -\frac{d\Delta\phi}{dt} \tag{B.22}$$

E is minimized through time as follows:

$$\frac{dE}{dt} = \frac{\partial E}{\partial \Delta\phi}\frac{d\Delta\phi}{dt} = -\left(\frac{d\Delta\phi}{dt}\right)^2 \leq 0 \tag{B.23}$$

1. If $Q > 0$, the minima of E are $\Delta\phi^* = 0\,[2\pi]$ and correspond to the phase fixed points of the dynamics (eq.B.18).
2. If $Q < 0$, the minima of E are $\Delta\phi^* = \pi\,[2\pi]$ and correspond to the phase fixed points of the dynamics (eq.B.18).

□

In other words, propagating a spike train that consists of positive and negative current spikes spaced in time by $T/2$ induces an in-phase or out-of-phase locking, depending on the polarity of Q. The latter can be set by choosing one of the two complementary digital post-synaptic voltages.

B.3.2 N Coupled Oscillators

When an oscillator i is perturbed by N other oscillators with same pulsation ω_0, (eq.B.10) can be generalized in the scalar case:

$$\frac{d}{dt}\alpha_i = \sum_{j=1}^{N} v_{ij}^{2\pi}\left(\phi_i\right) b_j^{2\pi}\left(\phi_j\right) \tag{B.24}$$

Similarly to the two-oscillators case, averaging out the previous equation along the fast variable ϕ_j leads to:

$$\frac{d}{dt}\alpha_i = \sum_{j=1}^{N} \frac{1}{2\pi}\int_{-\pi}^{\pi} v_{ij}^{2\pi}\left(\Delta\phi_{ij}+\phi_j\right) b_j^{2\pi}\left(\phi_j\right) d\phi_j \tag{B.25}$$

We use the spike train expression (eq.B.13) to obtain:

$$\frac{d}{dt}\alpha_i = \frac{1}{2\pi}\sum_{j=1}^{N}\left(v_{ij}^{2\pi}\left(\Delta\phi_{ij}\right) - v_{ij}^{2\pi}\left(\Delta\phi_{ij}+\pi\right)\right) \tag{B.26}$$

As we inject pre-synaptic signals to the same node, we have $v_{ij}^{2\pi} = v^{2\pi}$ and we use the SKONN oscillator PPV $v^{2\pi}$ (eq.B.16) to get finally:

$$\frac{d}{dt}\phi_i = \frac{2\beta_0}{T}\sum_{j=1}^{N} Q_{ij}\,\mathrm{square}\left(\phi_i-\phi_j\right) \tag{B.27}$$

Note that we omitted the term ω_0 in the right-hand side of (eq.B.27) as in practice we refer to the relative phase relationship between oscillators instead of the absolute values that linearly increase with $\omega_0 t$.

Considering SKONN's hardware implementation, we saw that β_0 is equal to:

$$\beta_0 = \frac{\pi}{\Delta V C_L} \tag{B.28}$$

and:

$$Q_{ij} = V_{DD} C_{ij} \tag{B.29}$$

where ΔV is the peak-to-peak triangular amplitude at the input, C_L is the neuron input capacitance, V_{DD} is the digital voltage swing, and C_{ij} is the synaptic capacitance value. SKONN's phase dynamics become:

$$\frac{d}{dt}\phi_i = \omega_0\frac{V_{DD}}{\Delta V}\sum_{j=1}^{N}\frac{C_{ij}}{C_L}\,\mathrm{square}\left(\phi_i-\phi_j\right) \tag{B.30}$$

SKONN has an interesting phase binarization property resumed in the following proposition:

Proposition B.2. *Consider a neuron i of degree D, i.e. driven by D neurons j with weighted charges $Q_{ij} \in \{-q,+q\}$ $q \neq 0$.*

1. *If D is odd and $d\phi_i/dt = 0$, then there is at least one input neuron j such that $(\phi_i - \phi_j)$ is a multiple of π.*
2. *If D is even, then there is at least one ϕ_i and one set of input phase ϕ_j such that $d\phi_i/dt = 0$ and $\forall j\,(\phi_i - \phi_j)$ is not a multiple of π.*

Proof. 1. By assuming the opposite, i.e. that $\forall j\,(\phi_i - \phi_j) \neq 0\ [\pi]$, it leads to $\forall j\,\text{square}(\phi_i - \phi_j) = \pm 1$, using (eq.B.17). Noticing that $D = m + l$ with $m \neq l$, and writing SKONN's phase dynamics (eq.B.30) leads to:

$$\frac{d\phi_i}{dt} = 0 \implies \sum_{j=1}^{D} \pm 1 = \sum_{j=1}^{m} 1 - \sum_{j=1}^{l} 1 = 0 \tag{B.31}$$

Which is not possible as $m \neq l$ and proves the proposition.

2. Consider the integers m and l such that there are m weights $Q_{ij} = +q$ and l weights $Q_{ij} = -q$, with $D = m + l = 2k$. We can choose:

   (a) $\phi_j = 0$ for the l (resp. m) input neurons.

   (b) $\phi_j = \pi$ for $k - l$ (resp. $k - m$) other input neurons.

   (c) ϕ_j=0 for the remaining k input neurons.

   From SKONN's phase dynamics (eq.B.30) we obtain:

$$\begin{aligned}\frac{T}{2\beta_0}\frac{d\phi_i}{dt} = &-q\sum_{j\leq l}\text{square}(\phi_i)\\ &+q\sum_{l<j\leq k}\text{square}(\phi_i-\pi)\\ &+q\sum_{k<j\leq 2k}\text{square}(\phi_i)\end{aligned}$$

   Assuming that $\phi_i \neq 0\ [\pi]$, it follows from (eq.B.17):

$$\begin{aligned}\frac{T}{2\beta_0}\frac{d\phi_i}{dt} &= -ql(\pm 1) - q(k-l)(\pm 1) + kq(\pm 1)\\ &= 0\end{aligned}$$

□

B.4 Comparison Between SKONN Phase Model and Circuit Dynamics

The phase dynamics defined in (eq.B.30) only provide the phases ϕ_i^m and do not include voltage and current equations. Here, we compare ϕ_i^m with ϕ_i^c, the phases extracted from the circuit dynamics expressed in (eq.B.4). For every oscillator i, the pair of output voltages V_i^{out} and V_1^{out} are fed to a 2-XOR gate which produces a signal V_i^{XOR} whose duty cycle is proportional to the phase between the first (reference) and the i-th oscillator (Fig.B.3a). Then, the voltages V_i^{XOR} are filtered out with a second-order low-pass filter to obtain the phase dynamics ϕ_i^c. Fig.B.3b and c show an example of circuit simulation for 4 fully-coupled oscillators with $W_{ij} = -1$ and a random initialization. Fig.B.3d shows the corresponding phases ϕ_i^c and ϕ_i^m (dashed lines) obtained from the circuit and the phase model, respectively. Note that the XOR-based phase measurement is insensitive to the phase sign and so ϕ_3^c departs from ϕ_3^m when $\phi_3^m > \phi_3^c = 180°$. However, computing $\cos(\phi_i^c)$ and $\cos(\phi_i^m)$ provide the spin dynamics for the two approaches and are shown at the bottom of Fig.B.3d. Overall, in the weak-coupling regime, we observe a good agreement between the phase and circuit models. This comparison motivates the use of the phase model (eq.B.30) as it only contains N ODEs for emulating an N-node SKONN, compared to at least 2N ODEs for the circuit approach (eq.B.4).

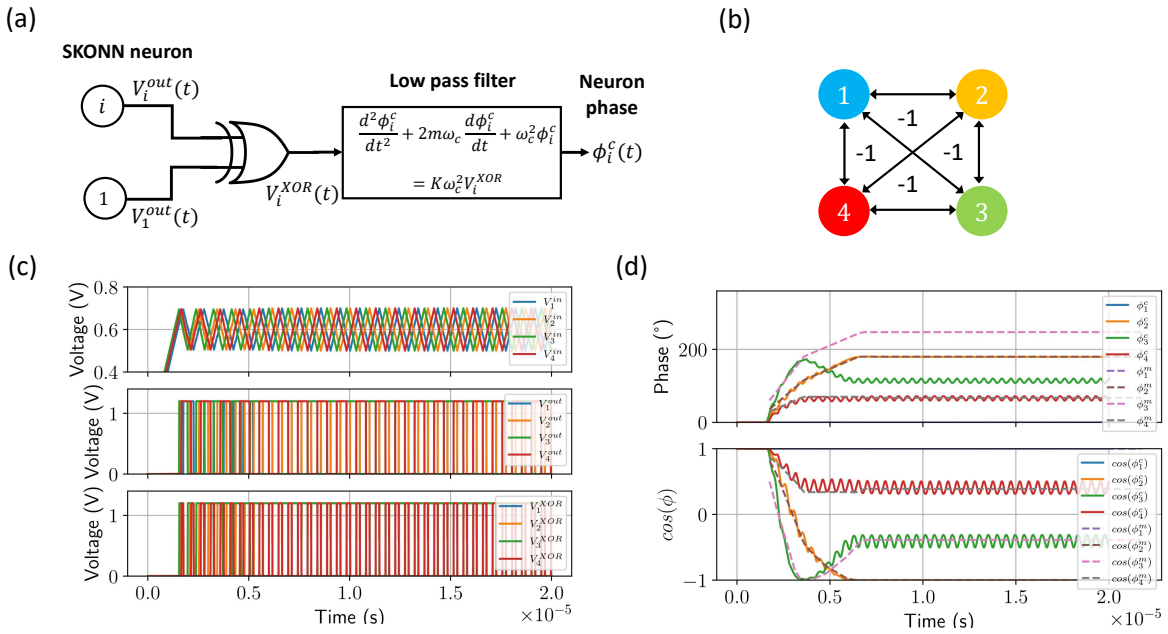


Figure B.3: *a) Proposed phase read-out to analyze the circuit phase dynamics. We set the cut-off frequency $\omega_c = \omega_0/2 = 500$ MHz and damping factor $m = \sqrt{2}/2$. The gain K converts voltage to phase as $K = \pi/V_{DD}$. Circuit parameters are C_L=500 fF, I_{bias}=200 nA, V_L=0.5 V, V_H=0.7 V, V_{DD}=1.2 V, γ=100 and τ_H=1 ns. b) Example of SKONN network. $|W_{ij}| = 1$ corresponds to a coupling capacitance C_{ij}=2.5 fF. c) Circuit dynamics obtained by solving numerically (eq.B.4). d) Comparison between SKONN's phase model ϕ_i^m (eq.B.30) and the circuit phase ϕ_i^c.*

B.5 Sub Harmonic Injection Locking in SKONN

To binarize the phases, one can inject a signal at twice the oscillating frequency $V_{SHIL}(t) = A\sin(4\pi\omega_0 t)$ to an oscillator's node for which its scalar PPV contains a second-order harmonic $P_2 \neq 0$ in its Fourier decomposition [193]. In SKONN, we cannot inject the 2-SHIL signal to the input oscillating node $V_i^{in}(t)$ as the associated scalar PPV only contains odd harmonics (square PPV (eq.B.7)). In practice, we inject the 2-SHIL signal to a biasing node that allows SKONN binary phase locking. In this case, the SKONN Lyapunov function becomes

$$\begin{aligned} E = &\frac{\beta_0}{T}\sum_{i,j}^{N} Q_{ij}\,\text{triangle}\big(\phi_i - \phi_j\big) \\ &+\sum_{i}^{N} A_i P_2 \cos(2\phi_i) \end{aligned} \tag{B.32}$$

When SHIL amplitudes A_i are large enough, phases are binarized $\phi_i = (1-S_i)\pi/2 \in \{0,\pi\}$ and the SKONN Lyapunov function corresponds to the Ising Hamiltonian H with an additional offset:

$$\begin{aligned} E &= -\frac{\pi\beta_0}{2T}\sum_{i,j}^{N} Q_{ij}S_iS_j + \sum_{i}^{N} A_i P_2 \\ &= H + \text{constant} \end{aligned} \tag{B.33}$$

B.6 Impact of SKONN's Limited Bandwidth

When SKONN is implemented in hardware, we observe some phase deviation with respect to the theoretical phase fixed points, as shown in Fig.B.4. The main reason is the hysteresis block that does not switch instantaneously when the synaptic current spikes induce voltage jumps above or below the hysteresis thresholds V_H and V_L. To better understand this phenomenon, we ran two transistor-level simulations of two coupled neurons in feed-forward mode with a strong weight W_{21}=+15 and different frequencies (Fig.B.4). When the oscillation frequency is low (300 kHz), the hysteresis switching delay is negligible and there is only a small phase deviation $\delta\phi$=4°. However, when the oscillation frequency increases to 1.2 MHz, the limited bandwidth of the hysteresis circuit causes a switching delay and a larger phase deviation $\delta\phi$=13°.

For a given phase precision required by the application, this error could be mitigated by increasing the bandwidth of the hysteresis circuit or slowing down the oscillators which constitutes a trade-off with the energy consumption. Interestingly, we have observed in experiments that having recurrent synapses $W_{ij} = W_{ji}$ compensate the hysteresis delay induced in both neurons, and the theoretical phase fixed point is reached in that case (see Fig.3.6c and d).

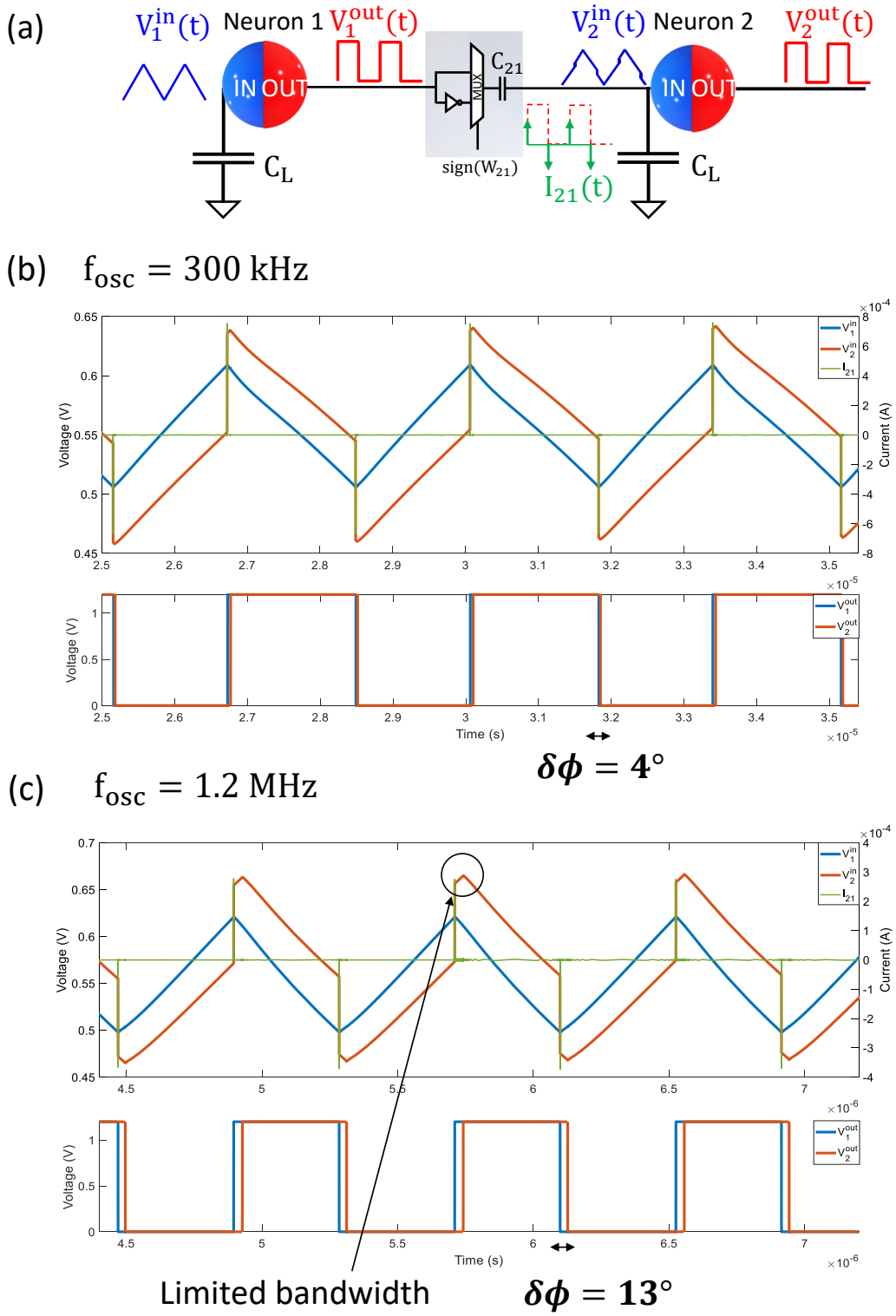


Figure B.4: *a) Two coupled oscillator in feed-forward mode with $+C_{21}$=7.5%C_L which implements W_{21}=+15 in the ASIC. b) Transistor-level simulation with C_L=2pF to decrease the frequency to f_{osc}=300 kHz. c) Transistor level simulation in nominal case where C_L=500 fF and f_{osc}=1.2 MHz.*

CHAPTER C

ONN FOR COMBINATORIAL OPTIMIZATION

C.1 MATLAB code for simulating Kuramoto-ONN and SKONN

C.1.1 Kuramoto ONN

The following function defines Kuramoto's dynamics:

Listing C.1: *kuramoto_dynamics.m*

```
function dphidt = kuramoto_dynamics(phase, N, C, Ashil, Cshil,
    shil_order, osc_parameters)

dphidt=zeros(N,1);

w0=osc_parameters.w0; %nominal frequency
Vdd=osc_parameters.Vdd; % supply voltage
CL=osc_parameters.CL; % oscillator load capacitance
dV=osc_parameters.dV; % oscillator peak-to-peak amplitude
w=ones(N,1)*w0; % oscillator frequency

for i=1:N
    interaction=sin(phase-phase(i));
    dphidt(i)=w(i)*C(i,:)*Vdd/(dV*CL)*interaction
    +Ashil*w(i)*Cshil*Vdd/(dV*CL*shil_order)*cos(shil_order*(phase(i)));
end
```

Providing the Jacobian matrix of the phase dynamics can speed up the simulations:

Listing C.2: *kuramoto_Jacobian.m*

```
function J = kuramoto_Jacobian(phase, N, C, Ashil, Cshil, shil_order,
    osc_parameters)

J=zeros(N,N);

w0=osc_parameters.w0;
Vdd=osc_parameters.Vdd;
CL=osc_parameters.CL;
```

```
dV=osc_parameters.dV;
w=ones(N,1)*w0;


for i=1:N
    for j=1:N
        if i==j
            interaction=cos(phase-phase(i));
            J(i,j)=w(i)*C(i,:)*Vdd/(dV*CL)*(-interaction)
            +Ashil*w(i)*Cshil*Vdd/(dV*CL*shil_order)*(-sin(shil_order*(
   phase(i))));
        else
            interaction=cos(phase(i)-phase(j));
            J(i,j)=w(i)*C(i,j)*Vdd/(dV*CL)*interaction;
        end
    end
end
```

C.1.2 SKONN

In simulations, SKONN's square interaction term is approximated as $\text{square}(\theta) = -\tanh\big(\alpha \sin(\theta)\big)$ with $\alpha = 50$. This allows making $\text{square}(\theta)$ differentiable everywhere and speed-up simulations by defining SKONN's Jacobian. SKONN's dynamics are defined in MATLAB as follows:

Listing C.3: *skonn_dynamics.m*

```
function dphidt = skonn_dynamics(phase, N, C, Ashil, Cshil, shil_order,
   osc_parameters)

dphidt=zeros(N,1);

w0=osc_parameters.w0; % nominal frequency
Vdd=osc_parameters.Vdd; % supply voltage
CL=osc_parameters.CL; % oscillator load capacitance
dV=osc_parameters.dV; % oscillator input peak-to-peak amplitude
alpha=osc_parameters.alpha; % slope of tanh interaction
w=ones(N,1)*w0;

for i=1:N
    interaction=-tanh(alpha*sin(phase-phase(i)));
    dphidt(i)=w(i)*C(i,:)*Vdd/(dV*CL)*(-interaction)
    +Ashil*w(i)*Cshil*Vdd/(dV*CL*shil_order)*cos(shil_order*(phase(i)));
end
```

SKONN's Jacobian can hence be derived as follows:

Listing C.4: *skonn_Jacobian.m*

```
function J = skonn_Jacobian(phase, N, C, Ashil, Cshil, shil_order,
   osc_parameters)

J=zeros(N,N);

w0=osc_parameters.w0;
Vdd=osc_parameters.Vdd;
```

```
CL=osc_parameters.CL;
dV=osc_parameters.dV;
alpha=osc_parameters.alpha;
w=ones(N,1)*w0;


for i=1:N
    for j=1:N
        if i==j
            interaction=(tanh(alpha*sin(phase(i)-phase)).^2-1)
            .*(alpha*cos(phase(i)-phase));
            J(i,j)=w(i)*C(i,:)*Vdd/(dV*CL)*interaction
            +Ashil*w(i)*Cshil*Vdd/(dV*CL*shil_order)*(-sin(shil_order*(
   phase(i))));
        else
            interaction=(tanh(alpha*sin(phase(j)-phase(i))).^2-1)
            .*(-alpha*cos(phase(j)-phase(i)));
            J(i,j)=w(i)*C(i,j)*Vdd/(dV*CL)*interaction;
        end
    end
end
```

C.2 Lagrangian-ONN

C.2.1 Properties of the Relaxed Complex Function

Given 3-SAT Boolean clauses C_i, we express them in an equivalent Ising formulation H_i as:

$$\begin{aligned}
C_0 = X \vee Y \vee Z &\longrightarrow H_0 = 1 + S_X S_Y + S_X S_Z + S_Y S_Z - (S_X + S_Y + S_Z) - S_X S_Y S_Z \\
C_1 = \overline{X} \vee Y \vee Z &\longrightarrow H_1 = 1 - S_X S_Y - S_X S_Z + S_Y S_Z - (-S_X + S_Y + S_Z) + S_X S_Y S_Z \\
C_2 = \overline{X} \vee \overline{Y} \vee Z &\longrightarrow H_2 = 1 + S_X S_Y - S_X S_Z - S_Y S_Z - (-S_X - S_Y + S_Z) - S_X S_Y S_Z \\
C_3 = \overline{X} \vee \overline{Y} \vee \overline{Z} &\longrightarrow H_3 = 1 + S_X S_Y + S_X S_Z + S_Y S_Z + (S_X + S_Y + S_Z) + S_X S_Y S_Z
\end{aligned} \tag{C.1}$$

with spins $S_i = \pm 1$ the binary Ising variables. There is the following equivalence:

$$\begin{array}{l} C_i \text{ is TRUE} \\ C_i \text{ is FALSE} \end{array} \iff \begin{array}{l} H_i = 0 \\ H_i = 8 \end{array} \tag{C.2}$$

The next step is to map the Ising energies to phase-based ONNs. We propose to relax the binary Ising Hamiltonians to complex variables defined as:

$$\begin{aligned}
H_0 &\longrightarrow Z_0 = 1 + e^{i(\phi_X - \phi_Y)} + e^{i(\phi_X - \phi_Z)} + e^{i(\phi_Z - \phi_Y)} - (e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) - e^{i(\phi_X - \phi_Y + \phi_Z)} \\
H_1 &\longrightarrow Z_1 = 1 - e^{i(\phi_X - \phi_Y)} - e^{i(\phi_X - \phi_Z)} + e^{i(\phi_Z - \phi_Y)} - (-e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) + e^{i(\phi_X - \phi_Y + \phi_Z)} \\
H_2 &\longrightarrow Z_2 = 1 + e^{i(\phi_X - \phi_Y)} - e^{i(\phi_X - \phi_Z)} - e^{i(\phi_Z - \phi_Y)} - (-e^{i\phi_X} - e^{i\phi_Y} + e^{i\phi_Z}) - e^{i(\phi_X - \phi_Y + \phi_Z)} \\
H_3 &\longrightarrow Z_3 = 1 + e^{i(\phi_X - \phi_Y)} + e^{i(\phi_X - \phi_Z)} + e^{i(\phi_Z - \phi_Y)} + (e^{i\phi_X} + e^{i\phi_Y} + e^{i\phi_Z}) + e^{i(\phi_X - \phi_Y + \phi_Z)}
\end{aligned} \tag{C.3}$$

The complex relaxation Z_i is equal to the Hamiltonian H_i in some cases listed in the following proposition.

Proposition C.1. *1. If phases take binary values such as $\phi_j = k\pi$, then $Z_i = 0$ or $Z_i = 8$. Consequently $Z_i = H_i$.*

2. If $Z_i = 8$, then all phases take binary values such as $\phi_j = k\pi$.

Proof. 1. For every clause i, expressing all possible phase binary values $\phi_j(S_j) \in \{0, \pi\}$ leads to $e^{i\phi_j(S_j)} = \pm 1$ and $Z_i(\phi) = H_i(S) \in \{0, 8\}$.

2. By contraposition, suppose $\exists\, j\, \phi_j \notin \{0, \pi\}$. We consider the four possible clauses:

   (a) Clause C_0 and C_1: consider phases $\phi_X = \pi/2$ and $\phi_Y = k_1\pi$, $\phi_Z = k_2\pi$ with $k_1, k_2 \in \mathbb{Z}$. Then, the real part of Z_0 and Z_1 becomes:

$$\begin{aligned} Re[Z_{0-1}] &= 1 \pm \cos(\pi/2 - k_1\pi) \pm \cos(\pi/2 - k_2\pi) + \cos(k_2\pi - k_1\pi) \\ &\quad \mp \cos(\pi/2) - \cos k_1\pi - \cos k_2\pi \\ &\quad \mp \cos(\pi/2 - k_1\pi + k_2\pi) \\ &= 1 + \cos((k_2 - k_1)\pi) - \cos k_1\pi - \cos k_2\pi \\ &= 4 \text{ if } k_1, k_2 \text{ are odd.} \\ &= 0 \text{ otherwise.} \\ &\neq 8 \end{aligned}$$

   (b) Clause C_2: consider phases $\phi_X = \pi/2$ and $\phi_Y = k_1\pi$, $\phi_Z = k_2\pi$ with $k_1, k_2 \in \mathbb{Z}$. Then, the real part of Z_2 becomes:

$$\begin{aligned} Re[Z_2] &= 1 + \cos(\pi/2 - k_1\pi) - \cos(\pi/2 - k_2\pi) - \cos(k_2\pi - k_1\pi) \\ &\quad + \cos(\pi/2) + \cos k_1\pi - \cos k_2\pi \\ &\quad - \cos(\pi/2 - k_1\pi + k_2\pi) \\ &= 1 - \cos((k_2 - k_1)\pi) + \cos k_1\pi - \cos k_2\pi \\ &= 4 \text{ if } k_1, k_2 \text{ are even and odd, resp.} \\ &= 0 \text{ otherwise.} \\ &\neq 8 \end{aligned}$$

   (c) Clause C_3: consider phases $\phi_X = \pi/2$ and $\phi_Y = k_1\pi$, $\phi_Z = k_2\pi$ with $k_1, k_2 \in \mathbb{Z}$. Then, the real part of Z_3 becomes:

$$\begin{aligned} Re[Z_3] &= 1 + \cos(\pi/2 - k_1\pi) + \cos(\pi/2 - k_2\pi) + \cos(k_2\pi - k_1\pi) \\ &\quad + \cos(\pi/2) + \cos k_1\pi + \cos k_2\pi \\ &\quad + \cos(\pi/2 - k_1\pi + k_2\pi) \\ &= 1 + \cos((k_2 - k_1)\pi) + \cos k_1\pi - \cos k_2\pi \\ &= 4 \text{ if } k_1, k_2 \text{ are even.} \\ &= 0 \text{ otherwise.} \\ &\neq 8 \end{aligned}$$

□

Corentin Delacour

corentin.delacour@lirmm.fr +33(0)648888421 161 Rue Ada, Montpellier, 34095, France
linkedin.com/in/corentindelacour
orcid.org/0000-0001-6386-9588

Research Interest

Physical and Analog Computing, Neuromorphic Computing, Circuit Design, Mathematical Optimization.

Research Experience

2020 – 2023 **LIRMM, University of Montpellier, CNRS, France**
Ph.D. program funded by the H2020 European project NeurONN.
Thesis title: Architecture Design for Analog Oscillatory Neural Networks.
Supervisors: Prof. Aida Todri-Sanial and Dr. Nadine Azemard-Crestani.

Mar. – Jun. 2020 **University of Geneva, Switzerland**
4-month Master thesis
Thesis title: Design of an Analogue Front-end for Silicon Photo Multipliers.
Supervisors: Dr. Adil Koukab and Dr. Matthieu Heller.
Manuscript Available at cds.cern.ch/record/2744108.

Jan. – Jun. 2019 **Pyxalis, Moirans, France**
6-month Research Internship
Title: Design of a JFET-based Image Sensor.
Supervisor: Dr. Julien Michelot.

Jun. – Jul. 2018 **University Space Center of Montpellier, France**
2-month Internship
Title: CubeSAT ROBUSTA-1B on-orbit Data Analysis.
Supervisor: Dr. Laurent Dusseau.

Education

2020–2023 **LIRMM, University of Montpellier, CNRS, France**
Ph.D Candidate in Automatic and Microelectronics Systems.

2018-2020 **Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland**
Master in Microelectronics Circuits and Systems.

2016-2018 **CentraleSupélec Paris, France**
Supélec Engineering Degree.

2013-2016 **University of Montpellier, France**
Bachelor in Electrical & Electronic Engineering and Automatic Control.

Research Publications

Peer Reviewed Journal Articles

1. **C. Delacour**, S. Carapezzi, M. Abernot, and A. Todri-Sanial, "Energy-performance assessment of oscillatory neural networks based on vo_2 devices for future edge ai computing," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2023. DOI: 10.1109/TNNLS.2023.3238473.

2. **C. Delacour**, S. Carapezzi, G. Boschetto, *et al.*, "A mixed-signal oscillatory neural network for scalable analog computations in phase domain," *Neuromorphic Computing and Engineering*, vol. 3, no. 3, p. 034 004, Aug. 2023. DOI: 10.1088/2634-4386/ace9f5.

3. S. Carapezzi, **C. Delacour**, A. Plews, A. Nejim, S. Karg, and A. Todri-Sanial, "Role of ambient temperature in modulation of behavior of vanadium dioxide volatile memristors and oscillators for neuromorphic applications," *Scientific Reports*, vol. 12, no. 1, p. 19 377, Nov. 2022, ISSN: 2045-2322. DOI: 10.1038/s41598-022-23629-4.

4. A. Todri-Sanial, S. Carapezzi, **C. Delacour**, *et al.*, "How frequency injection locking can train oscillatory neural networks to compute in phase," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 5, pp. 1996–2009, 2022. DOI: 10.1109/TNNLS.2021.3107771.

5. **C. Delacour** and A. Todri-Sanial, "Mapping hebbian learning rules to coupling resistances for oscillatory neural networks," *Frontiers in Neuroscience*, vol. 15, 2021, ISSN: 1662-453X. DOI: 10.3389/fnins.2021.694549.

6. S. Carapezzi, G. Boschetto, **C. Delacour**, *et al.*, "Advanced design methods from materials and devices to circuits for brain-inspired oscillatory neural networks for edge computing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 4, pp. 586–596, 2021. DOI: 10.1109/JETCAS.2021.3128756.

Peer Reviewed Conference Proceedings

1. **C. Delacour**, S. Carapezzi, M. Abernot, *et al.*, "Oscillatory neural networks for edge ai computing," in *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2021, pp. 326–331. DOI: 10.1109/ISVLSI51109.2021.00066.

2. S. Carapezzi, **C. Delacour**, G. Boschetto, *et al.*, "Multi-scale modeling and simulation flow for oscillatory neural networks for edge computing," in *2021 19th IEEE International New Circuits and Systems Conference (NEWCAS)*, 2021, pp. 1–5. DOI: 10.1109/NEWCAS50681.2021.9462761.

3. E. Corti, **C. Delacour**, A. Todri-Sanial, and S. Karg, "Frequency injection locking-controlled oscillations for synchronized operations in vo2 crossbar devices," in *2021 Device Research Conference (DRC)*, 2021, pp. 1–2. DOI: 10.1109/DRC52342.2021.9467129.

Conference Without Proceedings

1. **C. Delacour**, N. Azemard, and A. Todri-Sanial, "Analog Oscillatory Neural Networks for Energy Efficient Physics-based Computing", Poster presentation at *the Design and Automation Conference (DAC)*, San Francisco, 2023.

2. **C. Delacour**, N. Azemard, and A. Todri-Sanial, "Phase-based Oscillatory Neural Networks for Energy Efficient Neuromorphic Computing", Poster presentation at *the Design, Automation and Test Conference (DATE)*, Antwerpen, 2023.

3. **C. Delacour**, N. Azemard, and A. Todri-Sanial, "Solving the Travelling Salesman Problem in Continuous Phase Domain with Neuromorphic Oscillatory Neural Networks", Poster presentation at *NatConfAI*, Bonn, 2022.

4. **C. Delacour**, N. Azemard, and A. Todri-Sanial, "Phase-based Oscillatory Neural Network for Associative Neuromorphic Computing", Poster presentation at the *Colloque National du GDR SOC2*, Strasbourg, 2022.

Talks

Oct. 2023 — **Analog Computing based on Oscillatory Neural Networks for Solving Combinatorial Optimization Problems.**
Invited talk at TSMC's 2023 OIP Ecosystem Forum in Amsterdam, The Netherlands.

Talks (continued)

Jul. 2022 — **VO_2-based Oscillatory Ising Machine: The Role of External Temperature on Performance.**
Invited talk at the IEEE NANO 2022 conference in Palma de Mallorca, Spain.

Mar. 2022 — **Analog Oscillatory Neural Networks for Energy-Efficient Computing at the Edge.**
Invited talk (online) at the 2nd NeurONN workshop (DATE22 conference).

Jul. 2021 — **Energy Efficient Neuromorphic Computing with beyond-CMOS Oscillatory Neural Networks.**
Online talk at the International Conference on Neuromorphic Systems (ICONS).

May 2021 — **Neural and Synaptic Circuits for Oscillatory Neural Networks.**
Invited talk (online) at the 1st NeurONN workshop organized by IBM.

Skills

Languages	French, English.
Programming	Matlab, Python, C, Java.
EDA and TCAD Tools	SPICE, Cadence Virtuoso, Synopsys Calibre PEX, Sentaurus TCAD, Unix.

BIBLIOGRAPHY

[1] N. Jones, "Data centres are chewing up vast amounts of energy." https://www.nature.com/articles/d41586-018-06610-y, 2018. [Online; accessed 14-September-2023]. 1

[2] C. Freitag, M. Berners-Lee, K. Widdicks, B. Knowles, G. S. Blair, and A. Friday, "The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations," *Patterns*, vol. 2, p. 100340, Sept. 2021. 1

[3] A. Ross and L. Christie, "Energy consumption of ict." https://post.parliament.uk/research-briefings/post-pn-0677/, 2022. [Online; accessed 24-September-2023]. 1

[4] D. Amodei and D. Hernandez, "Ai and compute." https://openai.com/research/ai-and-compute, 2018. [Online; accessed 24-September-2023]. 1

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," July 2020. arXiv:2005.14165 [cs]. 1, 4

[6] AMD, "Amd instinct mi200 series accelerator datasheet." https://www.amd.com/system/files/documents/amd-instinct-mi200-datasheet.pdf, 2021. [Online; accessed 24-September-2023]. 1

[7] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, 1965. 1

[8] TSMC, "3nm technology." https://www.tsmc.com/english/dedicatedFoundry/technology/logic/l3nm, 2023. [Online; accessed 24-September-2023]. 1

[9] S. Sudhakar, V. Sze, and S. Karaman, "Data Centers on Wheels: Emissions From Computing Onboard Autonomous Vehicles," *IEEE Micro*, vol. 43, pp. 29–39, Jan. 2023. 2

[10] M. Helmstaedter, "Cellular-resolution connectomics: challenges of dense neural circuit reconstruction," *Nature Methods*, vol. 10, pp. 501–507, June 2013. 2

[11] NVIDIA, "Nvidia h100 tensor core gpu datasheet." https://resources.nvidia.com/en-us-tensor-core/nvidia-tensor-core-gpu-datasheet, 2023. [Online; accessed 24-September-2023]. 2

[12] Dictionary, "Definition of compute." https://dictionary.cambridge.org/dictionary/english/compute, 2023. [Online; accessed 12-October-2023]. 2

[13] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From single neurons to networks and models of cognition*, ch. 17.2. Cambridge University Press, 2014. 2, 3, 30

[14] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of Physiology*, vol. 117, no. 4, pp. 500–544, 1952. 2, 21

[15] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, Dec 1943. 2, 3, 5

[16] F. Rosenblatt, "The perceptron - a perceiving and recognizing automaton," Tech. Rep. 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York, January 1957. 3

[17] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, ch. 5. Spartan Books, 1962. 4, 5

[18] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, Jan. 1989. 4

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015. 4

[20] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, "Learning representations by back-propagating errors," 1986. 4

[21] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015. arXiv:1409.1556 [cs]. 4, 19

[22] Q. Cappart, D. Chételat, E. Khalil, A. Lodi, C. Morris, and P. Veličković, "Combinatorial optimization and reasoning with graph neural networks," Sept. 2022. arXiv:2102.09544 [cs, math, stat]. 4

[23] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, pp. 583–589, Aug. 2021. 4

[24] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, A. Novikov, F. J. R. Ruiz, J. Schrittwieser, G. Swirszcz, D. Silver, D. Hassabis, and P. Kohli, "Discovering faster matrix multiplication algorithms with reinforcement learning," *Nature*, vol. 610, pp. 47–53, Oct. 2022. 4

[25] J. von Neumann, "First draft of a report on the edvac," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993. 4

[26] A. M. Turing, "On computable numbers, with an application to the entscheidungsproblem," *Proceedings of the London Mathematical Society*, vol. s2-42, no. 1, pp. 230–265, 1936. 4, 6

[27] T.-J. Yang, Y.-H. Chen, J. Emer, and V. Sze, "A method to estimate the energy consumption of deep neural networks," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, (Pacific Grove, CA, USA), pp. 1916–1920, IEEE, Oct. 2017. 4

[28] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature Nanotechnology*, vol. 15, pp. 529–544, July 2020. 5

[29] S. Yu, H. Jiang, S. Huang, X. Peng, and A. Lu, "Compute-in-Memory Chips for Deep Learning: Recent Trends and Prospects," *IEEE Circuits and Systems Magazine*, vol. 21, no. 3, pp. 31–56, 2021. 5

[30] J. D. Kendall and S. Kumar, "The building blocks of a brain-inspired computer," *Applied Physics Reviews*, vol. 7, p. 011305, Mar. 2020. 5

[31] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE*, vol. 78, pp. 1629–1636, Oct. 1990. 5

[32] D. V. Christensen, R. Dittmann, B. Linares-Barranco, A. Sebastian, M. Le Gallo, A. Redaelli, S. Slesazeck, T. Mikolajick, S. Spiga, S. Menzel, I. Valov, G. Milano, C. Ricciardi, S.-J. Liang, F. Miao, M. Lanza, T. J. Quill, S. T. Keene, A. Salleo, J. Grollier, D. Marković, A. Mizrahi, P. Yao, J. J. Yang, G. Indiveri, J. P. Strachan, S. Datta, E. Vianello, A. Valentian, J. Feldmann, X. Li, W. H. P. Pernice, H. Bhaskaran, S. Furber, E. Neftci, F. Scherr, W. Maass, S. Ramaswamy, J. Tapson, P. Panda, Y. Kim, G. Tanaka, S. Thorpe, C. Bartolozzi, T. A. Cleland, C. Posch, S. Liu, G. Panuccio, M. Mahmud, A. N. Mazumder, M. Hosseini, T. Mohsenin, E. Donati, S. Tolu, R. Galeazzi, M. E. Christensen, S. Holm, D. Ielmini, and N. Pryds, "2022 roadmap on neuromorphic computing and engineering," *Neuromorphic Computing and Engineering*, vol. 2, p. 022501, June 2022. 5, 6

[33] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, pp. 1659–1671, Dec. 1997. 5

[34] S. Davidson and S. B. Furber, "Comparison of Artificial and Spiking Neural Networks on Digital Hardware," *Frontiers in Neuroscience*, vol. 15, p. 651141, Apr. 2021. 5

[35] M. Davies, A. Wild, G. Orchard, Y. Sandamirskaya, G. A. F. Guerra, P. Joshi, P. Plank, and S. R. Risbud, "Advancing Neuromorphic Computing With Loihi: A Survey of Results and Outlook," *Proceedings of the IEEE*, vol. 109, pp. 911–934, May 2021. 5, 6

[36] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems," *Frontiers in Neuroscience*, vol. 15, p. 212, 2021. 5, 41, 42

[37] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 1943–1953, Aug. 2013. 5

[38] Y. Stradmann, S. Billaudelle, O. Breitwieser, F. L. Ebert, A. Emmel, D. Husmann, J. Ilmberger, E. Muller, P. Spilger, J. Weis, and J. Schemmel, "Demonstrating Analog Inference on the BrainScaleS-2 Mobile System," *IEEE Open Journal of Circuits and Systems*, vol. 3, pp. 252–262, 2022. 5, 44

[39] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J.-M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations," *Proceedings of the IEEE*, vol. 102, pp. 699–716, May 2014. 5, 44

[40] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses," *Frontiers in Neuroscience*, vol. 9, Apr. 2015. 6

[41] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, pp. 1537–1557, Oct. 2015. 6, 44

[42] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C.-K. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y.-H. Weng, A. Wild, Y. Yang, and H. Wang, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," *IEEE Micro*, vol. 38, pp. 82–99, Jan. 2018. 6

[43] T. Taunyazov, W. Sng, B. Lim, H. Hian See, J. Kuan, A. Fatir Ansari, B. Tee, and H. Soh, "Event-Driven Visual-Tactile Sensing and Learning for Robots," in *Robotics: Science and Systems XVI*, Robotics: Science and Systems Foundation, July 2020. 6

[44] W. Maass, "To Spike or Not to Spike: That Is the Question," *Proceedings of the IEEE*, vol. 103, pp. 2219–2224, Dec. 2015. 6

[45] D. Deutsch, "Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer," *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 400, no. 1818, pp. 97–117, 1985. 6

[46] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, (Santa Fe, NM, USA), pp. 124–134, IEEE Comput. Soc. Press, 1994. 6

[47] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin,

D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, pp. 505–510, Oct. 2019. 6

[48] V. Brattka and P. Hertling, eds., *Handbook of Computability and Complexity in Analysis*. Theory and Applications of Computability, Cham: Springer International Publishing, 2021. 7

[49] H. T. Siegelmann and S. Fishman, "Analog computation with dynamical systems," *Physica D: Nonlinear Phenomena*, vol. 120, pp. 214–235, Sept. 1998. 7, 138

[50] N. Mohseni, P. L. McMahon, and T. Byrnes, "Ising machines as hardware solvers of combinatorial optimization problems," *Nature Reviews Physics*, vol. 4, pp. 363–379, Jun 2022. 8

[51] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, vol. 31, pp. 253–258, Feb 1925. 8, 9, 104, 106

[52] F. Barahona, "On the computational complexity of Ising spin glass models," *Journal of Physics A: Mathematical and General*, vol. 15, pp. 3241–3253, Oct. 1982. 8

[53] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, 2014. 8, 104, 106, 118, 123

[54] R. M. Karp, "Reducibility among Combinatorial Problems," in *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department* (R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, eds.), pp. 85–103, Boston, MA: Springer US, 1972. 8, 103

[55] H. H. Hoos and T. Stützle, "1 - introduction," in *Stochastic Local Search* (H. H. Hoos and T. Stützle, eds.), The Morgan Kaufmann Series in Artificial Intelligence, pp. 13–59, San Francisco: Morgan Kaufmann, 2005. 8, 102, 118

[56] D. Tank and J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Transactions on Circuits and Systems*, vol. 33, pp. 533–541, May 1986. 8, 14, 103, 104, 118

[57] T. Inagaki, Y. Haribara, K. Igarashi, T. Sonobe, S. Tamate, T. Honjo, A. Marandi, P. L. McMahon, T. Umeki, K. Enbutsu, O. Tadanaga, H. Takenouchi, K. Aihara, K.-i. Kawarabayashi, K. Inoue, S. Utsunomiya, and H. Takesue, "A coherent Ising machine for 2000-node optimization problems," *Science*, vol. 354, pp. 603–606, Nov. 2016. 8

[58] T. Honjo, T. Sonobe, K. Inaba, T. Inagaki, T. Ikuta, Y. Yamada, T. Kazama, K. Enbutsu, T. Umeki, R. Kasahara, K.-i. Kawarabayashi, and H. Takesue, "100,000-spin coherent Ising machine," *Science Advances*, vol. 7, p. eabh0952, Oct. 2021. 8, 10, 106

[59] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi,

E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose, "Quantum annealing with manufactured spins," *Nature*, vol. 473, pp. 194–198, May 2011. 8

[60] K. Y. Camsari, R. Faria, B. M. Sutton, and S. Datta, "Stochastic p-bits for Invertible Logic," *Physical Review X*, vol. 7, p. 031014, July 2017. arXiv:1610.00377 [cond-mat]. 8, 127

[61] J. Kaiser, S. Datta, and B. Behin-Aein, "Life is probabilistic — why should all our computers be deterministic? Computing with p-bits: Ising Solvers and Beyond," in *2022 International Electron Devices Meeting (IEDM)*, (San Francisco, CA, USA), pp. 21.4.1–21.4.4, IEEE, Dec. 2022. 8

[62] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive Science*, vol. 9, no. 1, pp. 147–169, 1985. 8

[63] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983. 8, 145

[64] N. A. Aadit, A. Grimaldi, M. Carpentieri, L. Theogarajan, J. M. Martinis, G. Finocchio, and K. Y. Camsari, "Massively Parallel Probabilistic Computing with Sparse Ising Machines," *Nature Electronics*, vol. 5, pp. 460–468, June 2022. arXiv:2110.02481 [cond-mat]. 8

[65] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, vol. 21, pp. 1087–1092, June 1953. 8

[66] A. Grimaldi, L. Sánchez-Tejerina, N. Anjum Aadit, S. Chiappini, M. Carpentieri, K. Camsari, and G. Finocchio, "Spintronics-compatible Approach to Solving Maximum-Satisfiability Problems with Probabilistic Computing, Invertible Logic, and Parallel Tempering," *Physical Review Applied*, vol. 17, p. 024052, Feb. 2022. 8

[67] G. Csaba and W. Porod, "Coupled oscillators for computing: A review and perspective," *Applied Physics Reviews*, vol. 7, p. 011302, Mar. 2020. 8, 9, 60

[68] F. C. Hoppensteadt and E. M. Izhikevich, "Pattern recognition via synchronization in phase-locked loop neural networks," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 734–738, 2000. 8, 9, 10, 11, 18, 21, 22, 33, 46, 53, 67, 103, 119

[69] T. Wang, L. Wu, P. Nobel, and J. Roychowdhury, "Solving combinatorial optimisation problems using oscillator based ising machines," *Natural Computing*, vol. 20, pp. 287–306, Jun 2021. 8, 9, 10, 11, 20, 21, 64, 67, 71, 106, 107, 110, 114, 115, 117, 145, 156

[70] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons.," *Proceedings of the National Academy of Sciences*, vol. 81, pp. 3088–3092, May 1984. 9

[71] M. A. Cohen and S. Grossberg, "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, pp. 815–826, Sept. 1983.

[72] Y. Abu-Mostafa and J. St. Jacques, "Information capacity of the Hopfield model," *IEEE Transactions on Information Theory*, vol. 31, pp. 461–464, July 1985.

[73] J. Šíma and P. Orponen, "Continuous-Time Symmetric Hopfield Nets Are Computationally Universal," *Neural Computation*, vol. 15, pp. 693–733, Mar. 2003. 9

[74] I. Ahmed, P.-W. Chiu, W. Moy, and C. H. Kim, "A probabilistic compute fabric based on coupled ring oscillators for solving combinatorial optimization problems," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 9, pp. 2870–2880, 2021. 9, 10, 20, 21, 45, 104, 145

[75] E. Izhikevich, "Weakly pulse-coupled oscillators, FM interactions, synchronization, and oscillatory associative memory," *IEEE Transactions on Neural Networks*, vol. 10, pp. 508–526, May 1999. 9

[76] W. Moy, I. Ahmed, P.-w. Chiu, J. Moy, S. S. Sapatnekar, and C. H. Kim, "A 1,968-node coupled ring oscillator circuit for combinatorial optimization problem solving," *Nature Electronics*, vol. 5, pp. 310–317, May 2022. 9, 20, 21, 22, 45, 60, 61, 74, 104, 106, 145

[77] C. Huygens, *Oeuvres complètes de Christiaan Huygens. Publiées par la Société hollandaise des sciences*, vol. 1. La Haye, M. Nijhoff, 1888-1950. 9, 82

[78] J. von Neumann, "Non-linear capacitance or inductance switching, amplifying and memory devices." 1954. 9, 17

[79] E. Goto, "The parametron, a digital computing element which utilizes parametric oscillation," *Proceedings of the IRE*, vol. 47, no. 8, pp. 1304–1316, 1959. 9, 17, 61, 96

[80] J. Roychowdhury, "Boolean Computation Using Self-Sustaining Nonlinear Oscillators," *Proceedings of the IEEE*, vol. 103, pp. 1958–1969, Nov. 2015. 9, 10

[81] M. J. Avedillo, J. M. Quintana, and J. Nunez, "Phase Transition Device for Phase Storing," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 107–112, 2020. 9, 10

[82] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities.," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982. 9, 11, 13, 35

[83] T. Aoyagi, "Network of neural oscillators for retrieving phase information," *Physical review letters*, vol. 74, pp. 4075–4078, 06 1995. 9

[84] Y. Kuramoto, *Chemical Oscillations, Waves, and Turbulence*, vol. 19 of *Springer Series in Synergetics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984. 9, 10, 21

[85] E. Corti, J. A. Cornejo Jimenez, K. M. Niang, J. Robertson, K. E. Moselund, B. Gotsmann, A. M. Ionescu, and S. Karg, "Coupled vo2 oscillators circuit as analog first layer filter in convolutional neural networks," *Frontiers in Neuroscience*, vol. 15, p. 19, 2021. 10, 13, 19, 21, 22, 23, 39, 51, 60

[86] S. Dutta, A. Khanna, A. S. Assoa, H. Paik, D. G. Schlom, Z. Toroczkai, A. Raychowdhury, and S. Datta, "An ising hamiltonian solver based on coupled stochastic phase-transition nano-oscillators," *Nature Electronics*, vol. 4, pp. 502–512, Jul 2021. 10, 13, 21, 22, 24, 60, 61, 74, 104, 106, 154, 156

[87] J. Grollier, D. Querlioz, K. Y. Camsari, K. Everschor-Sitte, S. Fukami, and M. D. Stiles, "Neuromorphic spintronics," *Nature Electronics*, vol. 3, pp. 360–370, Jul 2020. 10, 21, 22, 60, 104

[88] P. Maffezzoni, B. Bahr, Z. Zhang, and L. Daniel, "Analysis and design of boolean associative memories made of resonant oscillator arrays," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1964–1973, 2016. 10, 60

[89] T. Jackson, S. Pagliarini, and L. Pileggi, "An oscillatory neural network with programmable resistive synapses in 28 nm cmos," in *2018 IEEE International Conference on Rebooting Computing (ICRC)*, pp. 1–7, 2018. 10, 18, 39, 45, 60, 61, 74, 145

[90] M. Graber and K. Hofmann, "A versatile adjustable 400 node cmos oscillator based ising machine to investigate and optimize the internal computing principle," in *2022 IEEE 35th International System-on-Chip Conference (SOCC)*, pp. 1–6, 2022. 10, 20, 45, 60, 61, 74, 98, 104

[91] D. E. Nikonov and I. A. Young, "Benchmarking delay and energy of neural inference circuits," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 5, no. 2, pp. 75–84, 2019. 10, 41, 43

[92] D. Vodenicarevic, N. Locatelli, F. Abreu Araujo, J. Grollier, and D. Querlioz, "A Nanotechnology-Ready Computing Scheme based on a Weakly Coupled Oscillator Network," *Sci Rep*, vol. 7, p. 44772, Apr. 2017. 10

[93] J. Torrejon, M. Riou, F. A. Araujo, S. Tsunegi, G. Khalsa, D. Querlioz, P. Bortolotti, V. Cros, K. Yakushiji, A. Fukushima, H. Kubota, S. Yuasa, M. D. Stiles, and J. Grollier, "Neuromorphic computing with nanoscale spintronic oscillators," *Nature*, vol. 547, pp. 428–431, Jul 2017. 19, 22, 60

[94] S. Dutta, A. Khanna, W. Chakraborty, J. Gomez, S. Joshi, and S. Datta, "Spoken vowel classification using synchronization of phase transition nano-oscillators," in *2019 Symposium on VLSI Circuits*, pp. T128–T129, 2019. 10, 11, 19, 23, 60, 145

[95] D. Nikonov, P. Kurahashi, J. Ayers, H. Li, T. Kamgaing, G. Dogiamis, H.-J. Lee, Y. Fan, and I. Young, "Convolution inference via synchronization of a coupled cmos oscillator array," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. PP, pp. 1–1, 12 2020. 10, 11, 19, 45, 60

[96] N. Shukla, A. Parihar, M. Cotter, M. Barth, X. Li, N. Chandramoorthy, H. Paik, D. G. Schlom, V. Narayanan, A. Raychowdhury, and S. Datta, "Pairwise coupled hybrid vanadium dioxide-mosfet (hvfet) oscillators for non-boolean associative computing," in *2014 IEEE International Electron Devices Meeting*, pp. 28.7.1–28.7.4, 2014. 10, 11, 18, 23, 60

[97] D. E. Nikonov, G. Csaba, W. Porod, T. Shibata, D. Voils, D. Hammerstrom, I. A. Young, and G. I. Bourianoff, "Coupled-Oscillator Associative Memory Array Operation for Pattern Recognition," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 1, pp. 85–93, Dec. 2015. 11

[98] C. Delacour and A. Todri-Sanial, "Mapping hebbian learning rules to coupling resistances for oscillatory neural networks," *Frontiers in Neuroscience*, vol. 15, p. 1489, 2021. 10, 46

[99] T. Endo, L. Chua, and K. Takeyama, "A neural network using oscillators," in *1991., IEEE International Sympoisum on Circuits and Systems*, (Singapore), pp. 782–785, IEEE, 1991. 10, 18, 103

[100] Chai Wah Wu, "Graph coloring via synchronization of coupled oscillators," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 45, pp. 974–978, Sept. 1998. 10, 103

[101] F. Dörfler and F. Bullo, "Synchronization in complex networks of phase oscillators: A survey," *Automatica*, vol. 50, pp. 1539–1564, June 2014. 10

[102] A. T. Winfree, "Biological rhythms and the behavior of populations of coupled oscillators," *Journal of Theoretical Biology*, vol. 16, pp. 15–42, July 1967. 10

[103] Z. Néda, E. Ravasz, T. Vicsek, Y. Brechet, and A. L. Barabási, "Physics of the rhythmic applause," *Physical Review E*, vol. 61, pp. 6987–6992, June 2000. 10

[104] Y. Wu, Z. Zheng, L. Tang, and C. Xu, "Synchronization dynamics of phase oscillator populations with generalized heterogeneous coupling," *Chaos, Solitons & Fractals*, vol. 164, p. 112680, Nov. 2022. 10

[105] J. A. Acebrón, L. L. Bonilla, C. J. Pérez Vicente, F. Ritort, and R. Spigler, "The Kuramoto model: A simple paradigm for synchronization phenomena," *Reviews of Modern Physics*, vol. 77, pp. 137–185, Apr. 2005. 10

[106] A. A. Sharma, Y. Kesim, M. Shulaker, C. Kuo, C. Augustine, H.-P. Wong, S. Mitra, M. Skowronski, J. Bain, and J. Weldon, "Low-power, high-performance S-NDR oscillators for stereo (3D) vision using directly-coupled oscillator networks," pp. 1–2, June 2016. ISSN: 2158-9682. 11, 18

[107] N. Shukla, W.-Y. Tsai, M. Jerry, M. Barth, V. Narayanan, and S. Datta, "Ultra low power coupled oscillator arrays for computer vision applications," in *2016 IEEE Symposium on VLSI Technology*, pp. 1–2, 2016. 11, 23, 51

[108] E. Vassilieva, G. Pinto, J. Acacio De Barros, and P. Suppes, "Learning Pattern Recognition Through Quasi-Synchronization of Phase Oscillators," *IEEE Transactions on Neural Networks*, vol. 22, pp. 84–95, Jan. 2011. 11

[109] M. Romera, P. Talatchian, S. Tsunegi, F. Abreu Araujo, V. Cros, P. Bortolotti, J. Trastoy, K. Yakushiji, A. Fukushima, H. Kubota, S. Yuasa, M. Ernoult, D. Vodenicarevic, T. Hirtzlin, N. Locatelli, D. Querlioz, and J. Grollier, "Vowel recognition with four coupled spin-torque nano-oscillators," *Nature*, vol. 563, pp. 230–234, Nov. 2018. Number: 7730 Publisher: Nature Publishing Group. 11, 19, 74, 145

[110] E. Corti, A. Khanna, K. Niang, J. Robertson, K. E. Moselund, B. Gotsmann, S. Datta, and S. Karg, "Time-delay encoded image recognition in a network of resistively coupled vo on si oscillators," *IEEE Electron Device Letters*, vol. 41, no. 4, pp. 629–632, 2020. 13, 22, 43

[111] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. ACM*, vol. 42, pp. 1115–1145, 1995. 14, 105, 111, 115

[112] S. Burer, R. Monteiro, and Y. Zhang, "Rank-two relaxation heuristics for max-cut and other binary quadratic programs," *SIAM Journal on Optimization*, vol. 12, 07 2001. 14, 106, 107, 115, 116, 140

[113] S. Landge, V. Saraswat, S. F. Singh, and U. Ganguly, "n-Oscillator Neural Network based Efficient Cost Function for n-city Traveling Salesman Problem," in *2020 International Joint Conference on Neural Networks (IJCNN)*, (Glasgow, United Kingdom), pp. 1–8, IEEE, July 2020. 14, 104, 118, 119, 121

[114] S. Zhang and A. Constantinides, "Lagrange programming neural networks," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, pp. 441–452, July 1992. 14, 124, 132, 140

[115] DeLiang Wang and D. Terman, "Locally excitatory globally inhibitory oscillator networks: theory and application to pattern segmentation," in *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*, pp. 136–145, IEEE, 1994. 17

[116] J. Cosp, J. Madrenas, and J. Cabestany, "A VLSI implementation of a neuromorphic network for scene segmentation," in *Proceedings of the Seventh International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems*, pp. 403–408, Apr. 1999. 18

[117] J. Cosp, J. Madrenas, and D. Fernández, "Design and basic blocks of a neuromorphic VLSI analogue vision system," *Neurocomputing*, vol. 69, pp. 1962–1970, Oct. 2006. 18

[118] M. J. Cotter, Y. Fang, S. P. Levitan, D. M. Chiarulli, and V. Narayanan, "Computational architectures based on coupled oscillators," in *2014 IEEE Computer Society Annual Symposium on VLSI*, pp. 130–135, 2014. 18, 46

[119] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982. 18, 22, 103

[120] A. J. Noest, "Associative memory in sparse phasor neural networks," *EPL*, vol. 6, pp. 469–474, 1988. 18

[121] F. C. Hoppensteadt and E. M. Izhikevich, *Weakly Connected Oscillators*, ch. 9, pp. 247–293. New York, NY: Springer New York, 1997. 18, 67

[122] E. P. Frady and F. T. Sommer, "Robust computation with rhythmic spike patterns," *Proceedings of the National Academy of Sciences*, vol. 116, pp. 18050–18059, Sept. 2019. 18

[123] T. C. Jackson, A. A. Sharma, J. A. Bain, J. A. Weldon, and L. Pileggi, "Oscillatory neural networks based on tmo nano-oscillators and multi-level rram cells," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 5, no. 2, pp. 230–241, 2015. 18, 60

[124] R. Shi, T. C. Jackson, B. Swenson, S. Kar, and L. Pileggi, "On the design of phase locked loop oscillatory neural networks: Mitigation of transmission delay effects," in *2016 International Joint Conference on Neural Networks (IJCNN)*, (Vancouver, BC, Canada), pp. 2039–2046, IEEE, July 2016. 18

[125] M. Abernot, T. Gil, M. Jiménez, J. Núñez, M. J. Avellido, B. Linares-Barranco, T. Gonos, T. Hardelin, and A. Todri-Sanial, "Digital implementation of oscillatory neural network for image recognition applications," *Frontiers in Neuroscience*, vol. 15, p. 1095, 2021. 18, 36

[126] F. C. Hoppensteadt and E. M. Izhikevich, "Oscillatory Neurocomputers with Dynamic Connectivity," *Physical Review Letters*, vol. 82, pp. 2983–2986, Apr. 1999. 18

[127] M. Itoh and L. O. Chua, "Star cellular neural networks for associative and dynamic memories," *International Journal of Bifurcation and Chaos*, vol. 14, pp. 1725–1772, May 2004. 18, 19

[128] F. Corinto, M. Bonnin, and M. Gilli, "Weakly connected oscillatory network models for associative and dynamic memories," *International Journal of Bifurcation and Chaos*, vol. 17, pp. 4365–4379, Dec. 2007. Publisher: World Scientific Publishing Co. 18, 21

[129] D. I. Albertsson and A. Rusu, "Highly reconfigurable oscillator-based Ising Machine through quasiperiodic modulation of coupling strength," *Scientific Reports*, vol. 13, p. 4005, Mar. 2023. 19

[130] B. Scellier and Y. Bengio, "Equilibrium Propagation: Bridging the Gap between Energy-Based Models and Backpropagation," *Frontiers in Computational Neuroscience*, vol. 11, p. 24, May 2017. 19, 96

[131] G. Zoppo, F. Marrone, and F. Corinto, "Equilibrium Propagation for Memristor-Based Recurrent Neural Networks," *Frontiers in Neuroscience*, vol. 14, p. 240, Mar. 2020. 19, 96

[132] A. Laborieux and F. Zenke, "Holomorphic Equilibrium Propagation Computes Exact Gradients Through Finite Size Oscillations," Sept. 2022. arXiv:2209.00530 [cs]. 19

[133] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks*, vol. 115, pp. 100–123, July 2019. 19

[134] H. Jaeger, "The" echo state" approach to analysing and training recurrent neural networks-with an erratum note'," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, 01 2001. 19

[135] M. Riou, J. Torrejon, B. Garitaine, F. Abreu Araujo, P. Bortolotti, V. Cros, S. Tsunegi, K. Yakushiji, A. Fukushima, H. Kubota, S. Yuasa, D. Querlioz, M. Stiles, and J. Grollier, "Temporal Pattern Recognition with Delayed-Feedback Spin-Torque Nano-Oscillators," *Physical Review Applied*, vol. 12, p. 024049, Aug. 2019. Publisher: American Physical Society. 19

[136] D. Marković, N. Leroux, M. Riou, F. A. Araujo, J. Torrejon, D. Querlioz, A. Fukushima, S. Yuasa, J. Trastoy, P. Bortolotti, and J. Grollier, "Reservoir computing with the frequency, phase and amplitude of spin-torque nano-oscillators," *Applied Physics Letters*, vol. 114, p. 012409, Jan. 2019. arXiv:1811.00309 [cond-mat, physics:physics]. 19

[137] M. K. Bashar, Z. Lin, and N. Shukla, "Stability of oscillator ising machines: Not all solutions are created equal," 2023. 20, 72, 106

[138] M. K. Bashar, A. Mallick, and N. Shukla, "Experimental Investigation of the Dynamics of Coupled Oscillators as Ising Machines," *IEEE Access*, vol. 9, pp. 148184–148190, 2021. 20, 21, 45, 60, 71, 74, 75, 104, 106, 110, 145

[139] A. Mallick, M. K. Bashar, D. S. Truesdell, B. H. Calhoun, and N. Shukla, "Overcoming the Accuracy vs. Performance Trade-off in Oscillator Ising Machines," in *2021 IEEE International Electron Devices Meeting (IEDM)*, (San Francisco, CA, USA), pp. 40.2.1–40.2.4, IEEE, Dec. 2021. 20, 60, 61, 71, 98, 106, 115, 145

[140] M. Graber and K. Hofmann, "A Coupled Oscillator Network to Solve Combinatorial Optimization Problems with Over 95% Accuracy," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, (Monterey, CA, USA), pp. 1–5, IEEE, May 2023. 20, 21, 145

[141] M. Graber, M. Wesp, and K. Hofmann, "A Fast Graph Minor Embedding Heuristic for Oscillator Based Ising Machines," in *2022 Austrochip Workshop on Microelectronics (Austrochip)*, (Villach, Austria), pp. 41–44, IEEE, Oct. 2022. 20

[142] A. A. Sharma, J. A. Bain, and J. A. Weldon, "Phase Coupling and Control of Oxide-Based Oscillators for Neuromorphic Computing," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 1, pp. 58–66, Dec. 2015. 21, 22

[143] M. Weiher, M. Herzig, R. Tetzlaff, A. Ascoli, T. Mikolajick, and S. Slesazeck, "Improved Vertex Coloring With NbO Memristor-Based Oscillatory Networks," *IEEE Trans. Circuits Syst. I*, vol. 68, pp. 2082–2095, May 2021. 21, 22

[144] S. Lashkare, P. Kumbhare, V. Saraswat, and U. Ganguly, "Transient Joule Heating-Based Oscillator Neuron for Neuromorphic Computing," *IEEE Electron Device Lett.*, vol. 39, pp. 1437–1440, Sept. 2018. 21, 22

[145] H. Eslahi, T. J. Hamilton, and S. Khandelwal, "Energy-Efficient Ferroelectric Field-Effect Transistor-Based Oscillators for Neuromorphic System Design," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 6, pp. 122–129, Dec. 2020. 21, 22

[146] E. Izhikevich and Y. Kuramoto, "Weakly coupled oscillators," in *Encyclopedia of Mathematical Physics*, pp. 448–453, Academic Press, 2006. 21, 61, 63, 64, 67

[147] B. van der Pol Jun. D.Sc, "Lxxxviii. on "relaxation-oscillations"," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 978–992, 1926. 21

[148] R. Andrawis and K. Roy, "A new oscillator coupling function for improving the solution of graph coloring problem," *Physica D: Nonlinear Phenomena*, vol. 412, p. 132617, Nov. 2020. 21

[149] A. Parihar, N. Shukla, M. Jerry, S. Datta, and A. Raychowdhury, "Vertex coloring of graphs via phase dynamics of coupled oscillatory networks," *Scientific Reports*, vol. 7, p. 911, Apr 2017. 21, 56

[150] S. Carapezzi, G. Boschetto, C. Delacour, E. Corti, A. Plews, A. Nejim, S. Karg, and A. Todri-Sanial, "Advanced design methods from materials and devices to circuits for

brain-inspired oscillatory neural networks for edge computing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 4, pp. 586–596, 2021. 21, 22, 39, 45, 55, 60

[151] G. Csaba and W. Porod, "Computational Study of Spin-Torque Oscillator Interactions for Non-Boolean Computing Applications," *IEEE Trans. Magn.*, vol. 49, pp. 4447–4451, July 2013. 22

[152] R. Lebrun, S. Tsunegi, P. Bortolotti, H. Kubota, A. S. Jenkins, M. Romera, K. Yakushiji, A. Fukushima, J. Grollier, S. Yuasa, and V. Cros, "Mutual synchronization of spin torque nano-oscillators through a long-range and tunable electrical coupling scheme," *Nat Commun*, vol. 8, p. 15825, June 2017. 22

[153] J. Vaidya, R. S. Surya Kanthi, and N. Shukla, "Creating electronic oscillator-based Ising machines without external injection locking," *Sci Rep*, vol. 12, p. 981, Jan. 2022. 22

[154] W.-Y. Tsai, X. Li, M. Jerry, B. Xie, N. Shukla, H. Liu, N. Chandramoorthy, M. Cotter, A. Raychowdhury, D. M. Chiarulli, S. P. Levitan, S. Datta, J. Sampson, N. Ranganathan, and V. Narayanan, "Enabling New Computation Paradigms with HyperFET - An Emerging Device," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, pp. 30–48, Jan. 2016. 22

[155] S. Carapezzi, C. Delacour, A. Plews, A. Nejim, S. Karg, and A. Todri-Sanial, "Role of ambient temperature in modulation of behavior of vanadium dioxide volatile memristors and oscillators for neuromorphic applications," *Scientific Reports*, vol. 12, p. 19377, Nov 2022. 22

[156] A. Zimmers, L. Aigouy, M. Mortier, A. Sharoni, S. Wang, K. G. West, J. G. Ramirez, and I. K. Schuller, "Role of thermal heating on the voltage induced insulator-metal transition in vo_2," *Phys. Rev. Lett.*, vol. 110, p. 056601, Jan 2013. 23

[157] Z. Yang, C. Ko, and S. Ramanathan, "Oxide electronics utilizing ultrafast metal-insulator transitions," *Annual Review of Materials Research*, vol. 41, no. 1, pp. 337–367, 2011. 23

[158] P. Maffezzoni, L. Daniel, N. Shukla, S. Datta, and A. Raychowdhury, "Modeling and simulation of vanadium dioxide relaxation oscillators," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 9, pp. 2207–2215, 2015. 24, 34, 55, 153

[159] E. Corti, B. Gotsmann, K. Moselund, I. Stolichnov, A. Ionescu, and S. Karg, "Resistive coupled vo2 oscillators for image recognition," in *2018 IEEE International Conference on Rebooting Computing (ICRC)*, pp. 1–7, 2018. 24, 27

[160] O. H. Schmitt, "A thermionic trigger," *Journal of Scientific Instruments*, vol. 15, pp. 24–26, jan 1938. 27, 75

[161] J. Núñez, M. J. Avedillo, M. Jiménez, J. M. Quintana, A. Todri-Sanial, E. Corti, S. Karg, and B. Linares-Barranco, "Oscillatory neural networks using vo2 based phase encoded logic," *Frontiers in Neuroscience*, vol. 15, p. 442, 2021. 29

[162] T. Wang, L. Wu, and J. Roychowdhury, "New computational results and hardware prototypes for oscillator-based ising machines," in *Proceedings of the 56th Annual Design Automation Conference 2019*, DAC '19, (New York, NY, USA), Association for Computing Machinery, 2019. 38, 45, 60, 61, 104

[163] S. Carapezzi, C. Delacour, G. Boschetto, E. Corti, M. Abernot, A. Nejim, T. Gil, S. Karg, and A. Todri-Sanial, "Multi-scale modeling and simulation flow for oscillatory neural networks for edge computing," in *2021 19th IEEE International New Circuits and Systems Conference (NEWCAS)*, pp. 1–5, 2021. 39, 40, 55

[164] M. S. Mian, K. Okimura, and J. Sakai, "Self-oscillation up to 9mhz based on voltage triggered switching in vo2/tin point contact junctions," *Journal of Applied Physics*, vol. 117, no. 21, p. 215305, 2015. 43

[165] P. Chen, F. Zhang, Z. Zong, S. Hu, T. Siriburanon, and R. B. Staszewski, "A 31- μ w, 148-fs step, 9-bit capacitor-dac-based constant-slope digital-to-time converter in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 11, pp. 3075–3085, 2019. 43, 44

[166] S. Dutta, A. Khanna, A. S. Assoa, H. Paik, D. G. Schlom, Z. Toroczkai, A. Raychowdhury, and S. Datta, "An ising hamiltonian solver based on coupled stochastic phase-transition nano-oscillators - supplementary material," *Nature Electronics*, vol. 4, pp. 502–512, Jul 2021. 43

[167] E. Stromatias, F. Galluppi, C. Patterson, and S. Furber, "Power analysis of large-scale, real-time neural networks on SpiNNaker," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, (Dallas, TX, USA), pp. 1–8, IEEE, Aug. 2013. 44

[168] G. Indiveri, F. Corradi, and N. Qiao, "Neuromorphic architectures for spiking deep neural networks," in *2015 IEEE International Electron Devices Meeting (IEDM)*, (Washington, DC, USA), pp. 4.2.1–4.2.4, IEEE, Dec. 2015. 44

[169] A. Lines, P. Joshi, R. Liu, S. McCoy, J. Tse, Y.-H. Weng, and M. Davies, "Loihi Asynchronous Neuromorphic Research Chip," in *2018 24th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, (Vienna), pp. 32–33, IEEE, May 2018. 44

[170] Y. Kuang, X. Cui, Y. Zhong, K. Liu, C. Zou, Z. Dai, Y. Wang, D. Yu, and R. Huang, "A 64K-Neuron 64M-1b-Synapse 2.64pJ/SOP Neuromorphic Chip With All Memory on Chip for Spike-Based Models in 65nm CMOS," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, pp. 2655–2659, July 2021. 44

[171] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, "Diannao: a small-footprint high-throughput accelerator for ubiquitous machine-learning," *ACM SIGPLAN Notices*, vol. 49, pp. 269–284, 04 2014. 44, 45

[172] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "ShiDianNao: shifting vision processing closer to the sensor," in *Proceedings of the 42nd Annual International Symposium on Computer Architecture*, (Portland Oregon), pp. 92–104, ACM, June 2015. 44

[173] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, and O. Temam, "DaDianNao: A Machine-Learning Supercomputer," in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, (Cambridge, United Kingdom), pp. 609–622, IEEE, Dec. 2014. 44

[174] L. Cavigelli and L. Benini, "Origami: A 803 GOp/s/W Convolutional Network Accelerator," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, pp. 2461–2475, Nov. 2017. arXiv:1512.04295 [cs]. 44

[175] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, vol. 52, pp. 127–138, Jan. 2017. 44

[176] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "14.5 Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable Convolutional Neural Network processor in 28nm FDSOI," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, (San Francisco, CA, USA), pp. 246–247, IEEE, Feb. 2017. 44

[177] Google, "System architecture." https://cloud.google.com/tpu/docs/system-architecture-tpu-vm?hl=fr#tpu_v4, 2023. [Online; accessed 11-October-2023]. 44

[178] B. Barry, C. Brick, F. Connor, D. Donohoe, D. Moloney, R. Richmond, M. O'Riordan, and V. Toma, "Always-on Vision Processing Unit for Mobile Applications," *IEEE Micro*, vol. 35, pp. 56–66, Mar. 2015. 44

[179] C. Delacour, S. Carapezzi, G. Boschetto, M. Abernot, T. Gil, N. Azemard, and A. Todri-Sanial, "A mixed-signal oscillatory neural network for scalable analog computations in phase domain," *Neuromorphic Computing and Engineering*, vol. 3, p. 034004, aug 2023. 45

[180] H. Spontón and J. Cardelino, "A Review of Classic Edge Detectors," *Image Processing On Line*, vol. 5, pp. 90–123, 2015. 46, 47

[181] M. Abernot, T. Gil, and A. Todri-Sanial, "Oscillatory Neural Network as Hetero-Associative Memory for Image Edge Detection," in *NICE 2022 - 9th Neuro-Inspired Computational Elements Workshop*, (New York (Virtual), United States), p. In press, ACM, 2022. 46

[182] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986. 47

[183] C. Lopez-Molina, B. De Baets, and H. Sola, "Quantitative error measures for edge detection," *Pattern Recognition*, vol. 46, pp. 1125 – 1139, 2013. 47

[184] Z. Musoromy, S. Ramalingam, and N. Bekooy, "Edge detection comparison for license plate detection," in *2010 11th International Conference on Control Automation Robotics Vision*, pp. 1133–1138, 2010. 47

[185] Şaban Öztürk and B. Akdemir, "Comparison of edge detection algorithms for texture analysis on glass production," *Procedia - Social and Behavioral Sciences*, vol. 195, pp. 2675–2682, 2015. World Conference on Technology, Innovation and Entrepreneurship. 47

[186] P. Jaccard, "The distribution of the flora in the alpine zone," *The New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912. 47

[187] A. A. Taha and A. Hanbury, "Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool," *BMC Medical Imaging*, vol. 15, no. 1, p. 29, 2015. 47

[188] Departamento de Ciencias de la Computación e Inteligencia Artificial - Universidad de Grenada, "Dataset of standard 512x512 grayscale test images." https://ccia.ugr.es/cvg/CG/base.htm. [Online; accessed 13-October-2023]. 47, 48

[189] J. Lee, H. Tang, and J. Park, "Energy efficient canny edge detector for advanced mobile vision applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 4, pp. 1037–1046, 2018. 48

[190] L. B. Soares, J. Oliveira, E. A. C. da Costa, and S. Bampi, "An energy-efficient and approximate accelerator design for real-time canny edge detection," *Circuits, Systems, and Signal Processing*, vol. 39, no. 12, pp. 6098–6120, 2020. 48

[191] A. Parihar, N. Shukla, S. Datta, and A. Raychowdhury, "Synchronization of pairwise-coupled, identical, relaxation oscillators based on metal-insulator phase transition devices: A model study," *Journal of Applied Physics*, vol. 117, no. 5, p. 054902, 2015. 56

[192] F. Hoppensteadt and E. Izhikevich, "Synchronization of mems resonators and mechanical neurocomputing," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 2, pp. 133–138, 2001. 60

[193] A. Neogy and J. Roychowdhury, "Analysis and design of sub-harmonically injection locked oscillators," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1209–1214, 2012. 60, 154, 156, 161

[194] P. Bhansali and J. Roychowdhury, "Gen-adler: The generalized adler's equation for injection locking analysis in oscillators," in *2009 Asia and South Pacific Design Automation Conference*, pp. 522–527, 2009. 60, 66, 154

[195] J. Chou, S. Bramhavar, S. Ghosh, and W. Herzog, "Analog coupled oscillator based weighted ising machine," *Scientific Reports*, vol. 9, p. 14786, Oct 2019. 60, 61

[196] P. Erdös, A. Rényi, *et al.*, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960. 71

[197] B. Dokić, "CMOS schmitt triggers," *IEE Proceedings G (Electronic Circuits and Systems)*, vol. 131, no. 5, p. 197, 1984. 75

[198] R. Gregorian, *Introduction to CMOS OP-AMPs and Comparators*. Wiley, 1999. 75

[199] F. Yuan, "A high-speed differential CMOS Schmitt trigger with regenerative current feedback and adjustable hysteresis," *Analog Integrated Circuits and Signal Processing*, vol. 63, pp. 121–127, Apr. 2010. 75

[200] E. A. Vittoz, "Analog vlsi signal processing: Why, where, and how?," *Analog Integrated Circuits and Signal Processing*, vol. 6, pp. 27–44, Jul 1994. 77

[201] C. C. Enz, F. Krummenacher, and E. A. Vittoz, "An analytical mos transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications," *Analog Integrated Circuits and Signal Processing*, vol. 8, pp. 83–114, Jul 1995. 77, 112, 113

[202] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co., 1990. 102, 103

[203] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, (New York, NY, USA), p. 151–158, Association for Computing Machinery, 1971. 103, 126

[204] M. Erementchouk, A. Shukla, and P. Mazumder, "On computational capabilities of ising machines based on nonlinear oscillators," *Physica D: Nonlinear Phenomena*, vol. 437, p. 133334, 2022. 104, 107, 118

[205] G. S. Duane, "A "cellular neuronal" approach to optimization problemsa)," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 19, p. 033114, 07 2009. 104, 118, 119

[206] A. Mallick, M. K. Bashar, Z. Lin, and N. Shukla, "Computational Models Based on Synchronized Oscillators for Solving Combinatorial Optimization Problems," *Physical Review Applied*, vol. 17, p. 064064, June 2022. 104, 120

[207] M. K. Bashar and N. Shukla, "Designing Ising machines with higher order spin interactions and their application in solving combinatorial optimization," *Scientific Reports*, vol. 13, p. 9558, June 2023. 104, 145

[208] R. Hamerly, T. Inagaki, P. L. McMahon, D. Venturelli, A. Marandi, T. Onodera, E. Ng, C. Langrock, K. Inaba, T. Honjo, K. Enbutsu, T. Umeki, R. Kasahara, S. Utsunomiya, S. Kako, K. ichi Kawarabayashi, R. L. Byer, M. M. Fejer, H. Mabuchi, D. Englund, E. Rieffel, H. Takesue, and Y. Yamamoto, "Experimental investigation of performance differences between coherent ising machines and a quantum annealer," *Science Advances*, vol. 5, no. 5, p. eaau0823, 2019. 106

[209] F. Cai, S. Kumar, T. Van Vaerenbergh, X. Sheng, R. Liu, C. Li, Z. Liu, M. Foltin, S. Yu, Q. Xia, J. J. Yang, R. Beausoleil, W. D. Lu, and J. P. Strachan, "Power-efficient combinatorial optimization using intrinsic noise in memristor hopfield neural networks," *Nature Electronics*, vol. 3, pp. 409–418, Jul 2020. 106

[210] M. Erementchouk, A. Shukla, and P. Mazumder, "Self-contained relaxation-based dynamical Ising machines," May 2023. arXiv:2305.06414 [cs, math]. 107, 140

[211] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," Mar. 2014. 109, 115

[212] W. Sansen, *Noise performance of elementary transistor stages*, pp. 117–147. Boston, MA: Springer US, 2006. 112

[213] C. Penski, "A new numerical method for sdes and its application in circuit simulation," *Journal of Computational and Applied Mathematics*, vol. 115, no. 1, pp. 461–470, 2000. 113

[214] T. Sickenberger and R. Winkler, "Efficient transient noise analysis in circuit simulation," *PAMM*, vol. 6, no. 1, pp. 55–58, 2006. 113, 114

[215] C. Rackauckas and Q. Nie, "DifferentialEquations.jl–a performant and feature-rich ecosystem for solving differential equations in Julia," *Journal of Open Research Software*, vol. 5, no. 1, 2017. 114

[216] D. Martí, "G-set data for max-cut." https://grafo.etsii.urjc.es/optsicom/maxcut.html, 2009. [Online; accessed 13-October 2023]. 115, 117

[217] C. Choi and Y. Ye, "Solving sparse semidefinite programs using the dual scaling algorithm with an iterative solver," *Manuscript, Department of Management Sciences, University of Iowa, Iowa City, IA*, vol. 52242, 2000. 115

[218] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976. 118

[219] G. Reinelt, "Tsplib." http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/, 2002. [Online; accessed 11-October-2023]. 120, 122

[220] R. Mariescu-Istodor and P. Fränti, "Solving the Large-Scale TSP Problem in 1 h: Santa Claus Challenge 2020," *Frontiers in Robotics and AI*, vol. 8, p. 689908, Oct. 2021. 121

[221] J. B. Dennis, *Mathematical Programming and Electrical Networks*. Technology Press of the Massachusetts Institute of Technology, 1959. 123, 124

[222] G. R. Walsh, *Methods of Optimization*. Wiley, 1975. 123, 124

[223] H. W. Kuhn and W. Tucker, "Nonlinear programming," *Berkeley Symp. on Math. Statist. and Prob.*, pp. 481–492, 1951. 124

[224] L. Chua and Gui-Nian Lin, "Nonlinear programming without computation," *IEEE Transactions on Circuits and Systems*, vol. 31, pp. 182–188, Feb. 1984. 124

[225] M. Nagamatu and T. Yanaru, "On the stability of lagrange programming neural networks for satisfiability problems of prepositional calculus," *Neurocomputing*, vol. 13, no. 2, pp. 119–133, 1996. Soft Computing. 126, 129, 140

[226] M. Ercsey-Ravasz and Z. Toroczkai, "Optimization hardness as transient chaos in an analog approach to constraint satisfaction," *Nature Physics*, vol. 7, pp. 966–970, Dec. 2011. 126, 145

[227] B. Molnár and M. Ercsey-Ravasz, "Asymmetric Continuous-Time Neural Networks without Local Traps for Solving Constraint Satisfaction Problems," *PLoS ONE*, vol. 8, p. e73400, Sept. 2013. 126

[228] F. L. Traversa and M. Di Ventra, "Polynomial-time solution of prime factorization and NP-complete problems with digital memcomputing machines," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, p. 023107, Feb. 2017. 126, 145

[229] B. Molnár, F. Molnár, M. Varga, Z. Toroczkai, and M. Ercsey-Ravasz, "A continuous-time MaxSAT solver with high analog performance," *Nature Communications*, vol. 9, p. 4864, Nov. 2018. 126

[230] D. Mitchell, B. Selman, and H. Levesque, "Hard and easy distributions of sat problems," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, AAAI'92, p. 459–465, AAAI Press, 1992. 135

[231] H. H. Hoos and T. Stützle, "Local search algorithms for sat: An empirical evaluation," *Journal of Automated Reasoning*, vol. 24, pp. 421–481, May 2000. 135

[232] H. Hoos, "Satlib - benchmark problems." https://www.cs.ubc.ca/~hoos/SATLIB/benchm.html, 2000. [Online; accessed 13-October 2023]. 135, 137

[233] Cadence, "Spectre circuit simulator user guide." https://www.cadence.com/en_US/home/tools/custom-ic-analog-rf-design/circuit-simulation.html. [Online; accessed 13-October 2023]. 137, 138

[234] C. Bybee, D. Kleyko, D. E. Nikonov, A. Khosrowshahi, B. A. Olshausen, and F. T. Sommer, "Efficient optimization with higher-order ising machines," *Nature Communications*, vol. 14, p. 6033, Sept. 2023. 145