Dynamics of Complex Autonomous Boolean Networks

vorgelegt von
Physiker
David Rosin (MSc)
aus Berlin

von der Fakultät II — Mathematik und Naturwissenschaften

der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
— Dr. rer. nat. —

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Thomas Möller
Berichter/Gutachter:	Prof. Dr. Eckehard Schöll, PhD
Berichter/Gutachter:	Prof. Daniel J. Gauthier, PhD

Tag der wissenschaftlichen Aussprache: 20. Juni 2014

Berlin 2014
D 83

The research that lead to this thesis was mainly carried out at Duke University.

ABSTRACT

Network science provides a powerful framework for analyzing complex systems found in physics, biology, and social sciences. One way of studying the dynamics of networks is to engineer and measure them in the laboratory, which is particularly difficult with established approaches. In this thesis, I approach this problem using a hardware device with time-delay elements executing Boolean functions that can be connected to autonomous Boolean networks with chaotic, periodic, or excitable dynamics. I am able to make scientific discoveries for networks with each of these three different node dynamics, driven by the large flexibility and the non-ideal effects of the experiment complemented by analytical and numerical investigations.

Using network realizations with periodic Boolean oscillators, I study so-called chimera states and find that they can disappear and reappear—the resurgence of chimera states. I measure the transient times of chimera states and find a power-law relationship between the average transient time and the phase space volume with an exponent of $\kappa = -0.28 \pm 0.10$.

I also study cluster synchronization in networks of coupled excitable systems. In these artificial neural networks, I find a breakdown of an established theoretical tool when the heterogeneity of the link time delays is greater than the neural refractory period. This phenomenon is used to derive a control scheme for spiking patterns generated by neural networks.

Experimental implementations of these systems take advantage of the fast timescale of electronic logic gates, large scalability, and low price. These properties make the system attractive for technological applications, as I demonstrate by realizing a physical random number generator that has an ultra-high bitrate of 12.8 Gbit/s and a silicon neuron that is a thousand times faster than the fastest preceding silicon neuron. For the study of coupled oscillator networks, I develop a phase-locked loop allowing for multiple drivers that may be advantageous for clock synchronization. Instead of the common topologies with one driver per oscillator, it allows for heavily connected clock networks to increase robustness against failure.

ZUSAMMENFASSUNG

Netzwerkforschung hat stark zu neuen Erkenntnissen in Physik, Biologie und den Sozialwissenschaften beigetragen. Die Dynamik von Netzwerken kann im Labor untersucht werden, jedoch ist dies mit etablierten Versuchsaufbauten schwierig. Die in dieser Arbeit verwendete Untersuchungsmethode basiert auf einem Logikchip, auf dem Zeitverzögerungselemente mit Boolescher Dynamik zu sogenannten autonomen Booleschen Netzwerken verbunden werden. Ich zeige, dass in geeigneten Schaltkreisen chaotische, periodische und erregbare Dynamiken unterschieden werden können. Mit Hilfe dieser dynamischen Systeme können wiederum weitere Netzwerke konstruiert und untersucht werden. In meiner Arbeit fasse ich die wissenschaftlichen Erkenntnisse zusammen, die ich in Netzwerken jeder dieser drei Dynamiken gefunden habe. Die große Flexibilität der experimentellen Methode und die nicht-idealen Effekte der Logikbausteine helfen mir, neue wissenschaftliche Erkenntnisse zu erlangen. Die experimentellen Ergebnisse werden ergänzt durch numerische Simulationen und analytische Untersuchungen.

In Netzwerken Boolescher periodischer Oszillatoren untersuche ich Chimera-Zustände und entdecke eine neue Dynamik, die ich wiederauferstehende Chimera-Zustände nenne. Die Untersuchung des transienten Verhaltens dieser dynamischen Zustände ergibt ein Potenzgesetz zwischen der durchschnittlichen Lebensdauer und dem Phasenraumvolumen mit einem Exponenten von $\kappa = -0.28 \pm 0.10$.

Netzwerke Boolescher erregbarer Systeme, sogenannte Boolesche Neuronen, zeigen Gruppen-Synchronisation. Ich finde, dass eine etablierte Theorie für diese Dynamik dann nicht gilt, wenn die Heterogenität der Zeitverzögerungen im Netzwerk größer als die neuronale Refraktärzeit ist. Mit Hilfe dieses Phänomens können neuronale Synchronisationszustände kontrolliert werden.

Die von mir verwendete Untersuchungsmethodik weist erhebliche Vorteile auf. Experimentelle Realisierungen dieser Systeme funktionieren auf schnellen Zeitskalen, erlauben massive Parallelisierung und sind günstig herzustellen. Diese Eigenschaften sind attraktiv für vielfältige Anwendungen, wie die Implementierung eines physikalischen Zufallszahlgenerators mit einer hohen Rate von 12.8 Gbit/s unter Verwendung eines Netzwerkes chaotischer Dynamik. Außerdem identifiziere ich mögliche Anwendungen der Booleschen Neuronen, die tausendmal schneller als etablierte künstliche Neuronen sind, und der Booleschen Oszillatoren, die die Robustheit von Netzwerken synchronisierter Uhren erhöhen können.

PUBLICATIONS

Most of the results in this thesis appeared previously in the following publications:

- D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll. Control of synchronization patterns in neural-like Boolean networks. *Phys. Rev. Lett.*, **110**, 104102 (2013).
- D. P. Rosin, D. Rontani, and D. J. Gauthier. Ultrafast physical generation of random numbers using hybrid Boolean networks. *Phys. Rev. E* **87**, 040902(R) (2013).
- D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll. Excitability in autonomous Boolean networks. *Europhys. Lett.* **100**, 30003 (2012).
- D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll. Experiments on autonomous Boolean networks. *Chaos* **23**, 025102 (2013).
- D. P. Rosin, D. Rontani, and D. J. Gauthier. Synchronization of coupled Boolean phase oscillators. *Phys. Rev. E* **89**, 042907 (2014).
- D. P. Rosin, D. Rontani, E. Schöll, and D. J. Gauthier. Transient scaling and resurgence of chimera states in coupled Boolean phase oscillators. *Submitted for publication*, arXiv:1405.1950 (2014).
- D. Rontani, D. P. Rosin, D. J. Gauthier, and E. Schöll. Autonomous time-delayed Boolean networks using FPGAs. *Proc. 2012 Internat. Symposium on Nonlinear Theory and its Applications (NOLTA2012)*, 391-394 (2012).

Furthermore, the following publications include my previous work essential for the study of excitable autonomous Boolean networks in this thesis:

- D. P. Rosin, K. E. Callan, D. J. Gauthier, and E. Schöll. Pulse-train solutions and excitability in an optoelectronic oscillator. *Europhys. Lett.*, **96**, 34001 (2011).
- A. Panchuk, D. P. Rosin, P. Hövel, and E. Schöll. Synchronization of coupled neural oscillators with heterogeneous delays. *Int. J. Bif. Chaos*, **23**, 1330039 (2013).

CONTENTS

1 INTRODUCTION 1
- 1.1 Network Description of Complex Systems 1
- 1.2 Dynamics of Complex Networks 2
- 1.3 Challenges and Rewards of Experimental Network Realizations 4
- 1.4 Boolean Networks 5
- 1.5 Overview 6

2 PREVIOUS WORK ON BOOLEAN NETWORKS 8
- 2.1 Abstract 8
- 2.2 Synchronous and Autonomous Boolean Networks 8
- 2.3 Boolean Network Models 9
  - 2.3.1 Kauffman Networks 9
  - 2.3.2 Boolean Delay Equations 11
  - 2.3.3 Piecewise-Linear Differential Equations 12
  - 2.3.4 Overview of Boolean Network Models 14
- 2.4 Electronic realizations of Boolean Networks 15
  - 2.4.1 Autonomous Boolean Network by Zhang and Collaborators 16
  - 2.4.2 Boolean Chaos 17
- 2.5 Conclusion 18

3 AUTONOMOUS BOOLEAN NETWORKS ON ELECTRONIC CHIPS 19
- 3.1 Abstract 19
- 3.2 Field-Programmable Gate Arrays 19
  - 3.2.1 Architecture of Field-Programmable Gate Arrays 20
  - 3.2.2 Autonomous Mode of Operation 22
  - 3.2.3 Non-Ideal Effects of Autonomous Logic Gates 22
- 3.3 Design Flow of Implementing Autonomous Boolean Networks on Electronic Chips 23
  - 3.3.1 Hardware Description for Autonomous Boolean Networks 24
  - 3.3.2 Chip Placement of Autonomous Boolean Networks 26
  - 3.3.3 Resulting Dynamics 26
- 3.4 Conclusion 27

4 CHAOTIC DYNAMICS OF AUTONOMOUS BOOLEAN NETWORKS 28
- 4.1 Abstract 28
- 4.2 Introduction to Deterministic Chaos 28
  - 4.2.1 Lyapunov Exponent 30
  - 4.2.2 Strong and Weak Chaos 30
- 4.3 Delayed-Feedback XNOR Oscillator 32

4.3.1 Motivation for Developing a New Chaotic Oscillator 32
4.3.2 Search for a Simplified Network Topology 32
4.3.3 Setup of the Delayed-Feedback XNOR Oscillator 37
4.4 Dynamics of the Delayed-Feedback XNOR Oscillator 38
4.4.1 Dynamics Measured from the Experiment 38
4.4.2 Boolean Network Model for the Chaotic Dynamics 41
4.4.3 Transition to Chaos 45
4.5 Conclusion 47
5 ULTRA-FAST PHYSICAL GENERATION OF RANDOM NUMBERS USING HYBRID BOOLEAN NETWORKS 48
5.1 Abstract 48
5.2 Introduction to Random Number Generation 48
5.2.1 Application of Random Numbers to Private Communication 49
5.2.2 Pseudorandom and Physical Random Number Generation 50
5.2.3 Desired Statistical Properties of Random Numbers and Post-Processing 52
5.2.4 Utilization of Autonomous Boolean networks for Random Number Generation 54
5.3 Hybrid Boolean Network Approach 56
5.3.1 Dynamics of the XOR Ring Network 56
5.3.2 Characterization of Boolean Complexity 59
5.3.3 Modeling Results for the Dynamics of the XOR Ring Network 62
5.3.4 Modeling Results for Boolean Complexity 65
5.3.5 The Synchronous Part of the Hybrid Boolean Network 65
5.4 Utilization as a Physical Random Number Generator 66
5.4.1 Testing the Physical Random Number Generator 66
5.4.2 Parallelization to Increase the Bitrate 67
5.5 Conclusion 70
6 PERIODIC DYNAMICS IN AUTONOMOUS BOOLEAN NETWORKS 72
6.1 Abstract 72
6.2 Introduction to Periodic Dynamical Systems 72
6.2.1 Historical Perspective on Synchronization of Periodic Oscillators 73
6.2.2 Motivation for the Study of Synchronization of Periodic Oscillators 73
6.2.3 Notation of a Periodic Oscillator, Phase, and Synchronization 73
6.2.4 Dynamical Systems with Periodic Dynamics 75
6.2.5 Previous Work on Periodic Autonomous Boolean Networks 78
6.3 Coupling of Modified Ring Oscillators 80

6.3.1 Unidirectional Coupling of Modified Ring Oscillators 81
6.3.2 Mutual Coupling of Modified Ring Oscillators 82
6.3.3 Model for Modified Ring oscillators 84
6.3.4 Discussion 85
6.4 Boolean Oscillators with Variable Coupling Strength 85
6.4.1 Boolean Phase oscillator 86
6.4.2 Unidirectional Synchronization of Boolean Phase Oscillators and Weak Coupling Analogy 88
6.4.3 Model for Boolean Phase Oscillator 91
6.4.4 Synchronization in a Bidirectional Coupling Configuration 93
6.4.5 Discussion 95
6.5 Conclusion 96
7 CHIMERA DYNAMICS IN NETWORKS OF BOOLEAN PHASE OSCILLATORS 97
7.1 Abstract 97
7.2 Introduction to Chimera states in Theory and Experiment 98
7.2.1 Kuramoto Model 98
7.2.2 Chimera States in Theoretical Models 100
7.2.3 Transient Behavior of Chimera States in Finite-Size Networks 103
7.2.4 Chimera States in Experiments 104
7.3 Boolean Phase Oscillator Networks 105
7.3.1 Setup of Boolean Phase Oscillators with Large In-Degree 105
7.3.2 Derivation of a Phase Model for Boolean Phase Oscillators 106
7.3.3 Electronic Implementation of Boolean Phase Oscillators 108
7.4 Complex Dynamics in Boolean Phase Oscillator Networks 110
7.4.1 Chimera Dynamics in Boolean Phase Oscillator Networks 110
7.4.2 Resurgence of Chimera States 111
7.4.3 Determining the Fraction of Chimera States in the Transient 114
7.4.4 Nearly Synchronized State 115
7.5 Transient Scaling of Boolean Oscillator Networks 116
7.5.1 Scaling of the Transient Time 116
7.5.2 Scaling of the Average Transient Time 117
7.6 Numerical Simulation of Networks of Boolean Oscillators 118
7.6.1 Chimera States in the Model of Boolean Phase Oscillators 119
7.6.2 Differences between Modeled and Experimental Dynamics 119
7.7 Conclusion 121

8 EXCITABLE DYNAMICS IN AUTONOMOUS BOOLEAN NETWORKS 122
8.1 Abstract 122
8.2 Introduction to Excitability 122
8.2.1 Neurons 123
8.2.2 Hodgkin Classification 124
8.2.3 Key Components of Excitability 125
8.2.4 Dynamics of Two Delay-Coupled Neurons 125
8.2.5 Artificial Neural Networks 128
8.3 Design of a Boolean Neuron 129
8.3.1 Setup of Boolean Neurons 129
8.3.2 Model for Boolean Neurons 132
8.4 Dynamics of Network Motifs of Boolean Neurons 133
8.4.1 Dynamics of One Boolean Neuron 133
8.4.2 Dynamics of Two Delay-Coupled Boolean Neurons 136
8.4.3 Simulation of the Dynamics of Network Motifs of Boolean Neurons 137
8.5 Conclusion 138
9 CLUSTER SYNCHRONIZATION IN BOOLEAN NEURAL NETWORKS 140
9.1 Abstract 140
9.2 Dynamics of Spiking Neural Networks 140
9.2.1 Zero-Lag Cluster Synchronization 141
9.2.2 Neural Topologies of Connected Ring Networks 142
9.3 Theoretical Tools to Determine Cluster Synchronization 143
9.3.1 Master Stability Function for Network Synchronization 143
9.3.2 Greatest Common Divisor for Cluster Synchronization 146
9.4 Observation of Cluster Synchronization 148
9.5 Breakdown of Cluster Synchronization 149
9.6 Altered Cluster Synchronization Patterns 150
9.7 Control of Synchronization Patterns 152
9.8 Numerical Simulation of Boolean Neural Networks 153
9.9 Conclusion 154
10 SUMMARY AND OUTLOOK 156
A DELAY LINES REALIZED WITH ELECTRONIC LOGIC CIRCUITS 158
A.1 Implementation of Delay Lines 158
A.2 Measurement of the Gate Propagation Delay 160
A.3 Difference between Copier- and Inverter-Based Delay Lines 160
A.4 Delay of Different Logic Gates 162
B HARDWARE DESCRIPTIONS AND NUMERICAL ALGORITHMS 163
B.1 Inverter-Based Delay Lines 163
B.2 Delayed-Feedback XNOR Oscillator 164
B.3 Random Number Generator 164
B.3.1 XOR Ring Oscillator 164

B.3.2 Transfer of Random Numbers to a computer 166
B.3.3 XOR Ring Networks in Parallel 167
B.4 Modified Ring Oscillators 167
B.5 Network Motif of Modified Ring Oscillators 168
B.6 Boolean Phase Oscillators 168
B.6.1 Boolean Phase Oscillator with In-Degree One 168
B.6.2 Boolean Phase Oscillator with Large In-Degree 169
B.6.3 Non-local Network of Boolean Phase Oscillators 172
B.7 Boolean Neurons 173
B.7.1 Pulse Generator 174
B.7.2 Boolean Neuron 174
B.7.3 Network Motif of Two Bidirectionally-Coupled Boolean Neurons 175
B.7.4 Connected Ring Network of Boolean Neurons 175
B.8 Numerical Simulation with Adams-Bashforth method 178
C ADDITIONAL MATERIAL 179
C.1 Bias Reduction of the XOR Operation 179
C.2 Chimera States with Randomized Initial Conditions 180
C.3 Cluster Synchronization in Coupled Neural Populations 181

REFERENCES 183

1 INTRODUCTION

1.1 NETWORK DESCRIPTION OF COMPLEX SYSTEMS

In a *network description*, multi-agent systems, such as collections of interacting genes, assemblies of interconnected neurons, or social communities, are approximated as sets of nodes and edges. Nodes interact with each other if they share an edge [New10, Alb02a]. The network description has been beneficial to improve the understanding of a wide range of systems in nature and society, such as biological cells [Jeo00, Kau93, Kur84, Pom09, Win80], phone call interactions [Abe99, Gon08], and the World Wide Web [Alb99, Kum99].

An important early study on networks is the small-world experiment [Mil67], where Milgram studied social networks. He measured the average path length in the social graph of the people of the United States, where the nodes of the network are people and the edges are social links. In his experiments, Milgram found that the average shortest path length—the average degree of separation—between two randomly chosen persons is close to six. With this study, Milgram was the first to report on the existence of short average path lengths in large networks. Today, such networks, which are known as *small world networks*, have been identified for many systems in nature and technology, for example, for the World Wide Web, the power grid of the western United States, and the collaboration graph of film actors [Alb99, Wat98].

The study of network has led to important insight regarding the robustness of global systems, such as the internet and airline transportation graphs. Specifically, the short average path length in small-world networks is often ensured by an underlying *scale-free network* topology, where the *degree of a nodes*, *i.e.*, the number of links per node, follows a power law. As a result, a small fraction of the nodes, *e.g.*, 20%, have a large fraction the links, *e.g.*, 80% [New05], and a very few number of nodes, so-called hubs, have a crucial number of links. These hubs are, in the example of the internet, websites like `Google.com`. The hubbed structure of scale-free networks can make them vulnerable to targeted attacks, which is a concern for the World Wide Web and other computer networks when target of cyber-attacks [Alb00]. However, such scale-free networks are robust against accidental failures, which could explain their prevalence in nature [Bar03a].

As a result, scale-free airline transportation graphs allow for fast and efficient travel with only a few layovers, but also accelerate the spread of epidemic diseases [Alb02a, Mel11, Pas01].

Examples of networks in nature, in addition to the ones mentioned above, are food webs [Mon02, Wil00], where the nodes are species and the edges represent their predator-prey relationship, science collaboration graphs [New01c], where the nodes are scientists and edges represent co-authored articles, and neural networks, where the nodes are neurons or populations of neurons and edges are synapses or gap junctions. For example, the neural network of the worm *C. elegans* consists of 282 neurons with a known network topology [Wat98].

While there is much effort in studying the topology of the various networks in nature, such as the degree distribution, the clustering, and the community structure of networks [Alb02a, Muc10], another branch of network science focuses on the *dynamics of and on networks* and the relation between dynamics and topology.

1.2 DYNAMICS OF COMPLEX NETWORKS

A well-studied network dynamics is *global synchronization* or simply synchronization. Synchronization of coupled periodic oscillators was first documented in the 17th century by Huygens in two mechanically coupled pendulum clocks [Huy86]. Later, synchronization of oscillators has been found throughout nature with popular examples of synchronization of flashing fireflies and synchronization of walking crowds on the Millennium Bridge in London [Mir90a, Smi35, Str93, Str05a]. Synchronization has not only been observed for periodic systems but also for chaotic and excitable systems [Oht90, Pec90].

A useful mathematical tool to study network synchronization is the *master stability function*, which separates the influence of the network topology and the influence of the individual node dynamics to the overall synchronization dynamics [Pec98]. The master stability function has been successfully applied to study synchronization in various delay-coupled networks of different node dynamics [Dah12, Leh11, Kin09, Cho09, Flu10b, Hei11, Kea12, Wil13, Lad13, Bla13, Sch13, Cho14].

Astonishingly, even when time delays exist along the links, networks can synchronize with zero time lag, which is emphasized with the expression *zero-lag synchronization*. This phenomenon has been observed in the brain, where, even between distant neural populations, zero-lag synchronized neural activity has been observed [Vic08, Fri97a, Rod99, Roe97, Sch06i, Var01] and found to be associated with perception and neurological diseases [Sch11e, Uhl06]. The time delays in neural networks result from propagation of neuronal pulses along the axons introducing several tens

of milliseconds of latency, which is significantly larger than the duration of the action potential ($\lesssim 1\,\mathrm{ms}$) [Rin94]. Zero-lag synchronization occurs also in coupled lasers, where signals take a significant amount of time to be exchanged in the network due to spatial distance and the finite speed of light [Eng10, Fis06, Loc02, Mas01, Sor13].

One striking extension of this dynamics is *zero-lag cluster synchronization*, where the network separates into groups of nodes (the clusters), which are individually zero-lag synchronized. This dynamics has been observed in networks of coupled lasers [Nix11, Nix12], neural elements [Var12, Var12a], and optoelectronic oscillators [Wil13].

The number of dynamical clusters in the networks can under certain conditions be calculated from the network topology using a measure termed the greatest common divisor of a network topology [Kan11, Kan11a] or with the master stability function [Dah12, Sor07]. These analytic theories have been applied to networks with periodic [Bla13, Cho09, Nix12], chaotic [Kan11, Pec98] and excitable node dynamics [Dah12, Kan11a].

Another network dynamics is partial synchronization, where a fraction of the nodes is synchronized and the remaining nodes are desynchronized. The latter dynamics occurs in networks of oscillators when the distribution of their natural frequencies is broad and the coupling is weak [Kur84, Mar13, Win67, Cak14].

Coexistence of coherence (synchronization) and incoherence (desynchronization) in networks of coupled oscillators can occur even when the oscillators are identical and the network topology is homogeneous [Abr04, Kur02a]. As a requirement to observe this dynamics, the network has to have, in most studies, a non-local network topology and phase-lag or time delay along the links. To highlight the occurrence of two very different domains of synchronization and desynchronization, this dynamics was named after the chimera creature in Greek mythology, which is composed of different animals [Abr04]. *Chimera states* have been found in networks of various node dynamics described by a wide range of theoretical models [Ome10a, Ome11, Ome13, Pan13, Zak14] and also observed in experimental setups using coupled optical systems [Hag12], electrical [Lar13], chemical [Tin12, Nko13, Wic13, Sch14a], and mechanical oscillators [Mar13].

In addition to observing the dynamics of networks, some recent work focuses on its control. For example, researchers are interested in which network topologies can be controlled [Liu11, Nep12, Sie14, Flu13].

1.3 CHALLENGES AND REWARDS OF EXPERIMENTAL NETWORK REALIZATIONS

Networks of dynamical systems have also been realized in experimental setups with three major goals. First, such realizations are needed to show that the various network dynamics are robust enough to occur in an experimental system because experiments include noise and heterogeneity, which are often not accounted for in models and even unexpected differences between models and experiment can exist. Second, an experimenter's perspective on network dynamics is different from the dominant theoretician's point of view, possibly allowing for game-changing innovations. Third, dynamics generated by physical networks have important applications. For example, hardware neural networks are used to develop new computing structures inspired by the human brain [Boa00, Boa05, Ind11], opto-electronic networks are used to ensure private communication [Arg05, Col94, Ron09], and chaotic dynamics of lasers is used to realize physical random number generators [Rei09, Oli11, Uch08], which are vital for cybersecurity [Dep14, Jun99].

While research on physical realizations of networks can be very rewarding, it is also challenging because multiple systems have to be set up and coupled. This is especially true for networks that generate complex dynamics, such as chimera states or cluster synchronization, because they have to include a large number of highly-connected nodes. For this, most of the traditional experimental systems from nonlinear dynamics research are not feasible, such as classical analog electronic circuits and opto-electronic circuits [Rul95, Ill11, Hei01b]. Recent studies on coupled lasers, however, allowed to couple as many as 100 nodes [Dav12, Fri10, Nix12, Ama12]. In studies on experimental chimera states, namely in Ref. [Hag12, Lar13, Mar13, Nko13, Tin12, Wic13, Sch14a], workarounds allow to couple many nodes into complex topologies. For example, these studies either use computer algorithms to manage the coupling or they are restricted to simple network topologies or a small number of nodes [Hag12, Mar13, Tin12, Wic13]. Laurent Larger and collaborators use an elegant mapping from node number to time domain that allowed him to realize a virtual network with a single time-delayed feedback electronic circuit [App11, Lar13]. However, there has not yet been an all-physical realization of chimera states with a network that has a similar topology to the originally studied network in Refs. [Abr04, Kur02a] and includes more than 30 nodes.

In this thesis, I pursue a novel approach to experimental network realizations using *Boolean networks* built with electronic logic circuits. I take advantage of recent developments in very-large-scale integration (VLSI) of digital electronics that allow me to realize large networks with hundreds of highly-connected nodes, beyond reach of traditional setups. Furthermore,

their fast timescale on the order of 100 ps means that the physical networks have several potential applications, such as network-based information processing [Jae04, Maa02]. The study of Boolean networks is also interesting from a fundamental point of view because these systems are a popular generic model in complex systems theory, as is discussed next.

1.4 BOOLEAN NETWORKS

Boolean networks were first proposed in 1965 by Walker and Ashby [Wal65] as a general, interesting complex system; in a groundbreaking paper from 1969, Kauffman popularized Boolean networks as a model for genetic circuits [Kau69]. These so-called *Kauffman networks* were extended by Ghil and Mullhaupt in 1984 with *Boolean delay equations* by including time delays [Dee84, Ghi85]. Later, Glass and collaborators popularized *piecewise linear differential equations* with a Boolean switching term as another approach to Boolean networks [Mes96, Gla98].

Boolean network descriptions are commonly used today to model complex systems that exhibit threshold behavior, have multiple feedbacks, and multiple time delays [Alb00b, Ghi08, Kau69, Kau03, Soc03]. Boolean networks are seen as a way towards understanding large coupled systems that are too complex to be modeled in every detail, especially including amplitude-specific interactions [Ghi08, Nor07, Rib08, Soc03, Sun13]. For example, simple models such as Boolean networks are helpful in the fields of life sciences and geosciences. In biology, Boolean networks are used to model genetic regulatory networks, where genes interact with each other via transcriptional factors [Alb00b, Cha05, Kau69, Kau93, Kau03, Kau04]. The study on Boolean networks led to the idea that the attractors in genetic networks represent different cell expressions [Kau69, Kau93]. In geosciences, Boolean networks have been used as an idealized climate model for a wide range of timescales ranging from climate change on interanual to paleoclimatic timescales [Dar93, Ghi87, Ghi08, Woh95]; they have also been used in seismicity for earthquake modeling and prediction [Ghi08, Zal03, Zal03a].

Applications of Boolean networks include neural network models, which are needed for novel approaches to computing, and systems biology, for example with the mathematical description of Kauffman networks [Che10, Hop82, Mcc43, Shm02, Sny12]. Last but not least, with processors, most of the modern-day electronic equipment is based on Boolean algebra. In the development of these systems, the logic designs are usually simulated extensively with Boolean delay models known as timing simulation [Bro08].

In addition to a large body of theoretical work on Boolean networks, they have been used to build physical systems that can potentially be applied in signal processing because of their fast timescale and very complex dy-

namics. A so-called *autonomous Boolean networks* was built by Zhang and collaborators in 2009 [Zha09a], which is the starting point for the research in this thesis.

1.5 OVERVIEW

The thesis is organized in ten chapters, where Ch. 2 and 3 introduce autonomous Boolean networks and their experimental implementation. Chapters 4 and 5 focus on chaotic autonomous Boolean networks, where each node executes a Boolean function. In Ch. 6 and 8, on the other hand, I develop autonomous Boolean networks with periodic and excitable dynamics that are used in Ch. 7 and 9 to construct meta-networks of these systems. There, I consider the periodic or excitable dynamical systems as nodes and the meta-networks simply as a networks. The resulting scientific findings are summarized in Ch. 10.

Specifically, Ch. 2 introduces Boolean network models and preceding work on experimental realizations of Boolean networks with electronic logic gates, especially the electronic realization of a chaotic autonomous Boolean network by Zhang and collaborators [Zha09a]. Chapter 3 discusses a new experimental platform used for the experimental implementation of networks in this thesis. The design flow of implementing networks is discussed. The experimental platform allows to realize large networks of hundreds of nodes.

Chapter 4 focuses on chaotic dynamics in autonomous Boolean networks with a system that I term delayed-feedback XNOR oscillator. This chaotic dynamic system, motivated by an early theoretical study on Boolean networks, consists of only a single dynamical node with time-delayed feedback. In Ch. 5, the chaotic dynamics are applied to physical random number generation. I characterize how the network dynamics changes as a function of the network size and characterize its complexity with common measures, such as the entropy and the autocorrelation. I find that a hybrid Boolean network generates high-quality physical random numbers with a record bitrate of 12.8 GHz.

Chapter 6 focuses on periodic dynamics of autonomous Boolean networks. I introduce and study two concepts to realizing periodic Boolean oscillators: modified ring oscillators and Boolean phase oscillators. The latter is based on all-digital phase-locked loops, allowing for weak coupling. With both concepts I study small network motifs of coupled periodic Boolean oscillators and study the synchronization regimes. In Ch. 7, the Boolean phase oscillators are applied to studying large networks in a nonlocal coupling topology. I find that these networks show intriguing network dynamics known as chimera states and discover a new dynamics that I call

resurgence of chimera states. I also find that the dynamics is transient with a transient time that follows a power law of the phase space volume.

In Ch. 8, I propose and characterize an autonomous Boolean network that shows excitable dynamics and is a particularly fast silicon neuron. Its spiking dynamics is characterized in small network motifs of two silicon neurons and is used to confirm experimentally previous theory results on the dynamics of biological neural systems. The silicon neurons are used in Ch. 9 to study cluster synchronization in directed, interconnected ring networks. I find that an established theory for the neurodynamics breaks down when the time-delay heterogeneity is larger than the neural refractory period.

I summarize the results of the thesis in Chapter 10, where I also give an outlook over possible continuation of my work.

2

PREVIOUS WORK ON BOOLEAN NETWORKS

2.1 ABSTRACT

This chapter summarizes both previous theoretical and expeirmental work on Boolean networks. I distinguish between synchronous and autonomous Boolean networks in Sec. 2.2 and introduce Boolean network models and preceding experimental work with electronic circuits in Secs. 2.3 and 2.4.

2.2 SYNCHRONOUS AND AUTONOMOUS BOOLEAN NETWORKS

A Boolean network is a composition of nodes that can be in one of two Boolean states—either "on" or "off", "1" or "0"—and links that connect the nodes [Wal65, Kau69]. The network dynamics is determined by Boolean functions of the Boolean states, processing delays, and, especially, the updating method of the Boolean states.

I distinguish between two forms of Boolean networks depending on the updating method: *synchronous* and *autonomous Boolean networks*. Synchronous Boolean networks evolve in discrete time steps, mathematically described by iterated maps and experimentally realized with clocked logic circuits. The processing delays are then given by one iteration step of the map. Autonomous Boolean networks evolve in continuous time, mathematically described by differential equations or Boolean delay equations and experimentally realized with unclocked logic circuits. The processing delays in autonomous Boolean networks originate from processing times of the nodes and propagation delays along the links.

One important example for an autonomous Boolean network is a synthetic biological circuit termed the "repressilator," which is similar to naturally occurring biological circuits that function as biological clocks [Jeo00]. This circuit includes three transcriptional repressors that inhibit each other in a cyclic way, leading to oscillations [El000, Nor09].

A simplified network topology of the repressilator is shown in Fig. 1. It consists of three autonomous nodes connected in a directional ring as shown in Fig. 1(a). In this example, each node executes the inversion Boolean function, hence, it adjusts its Boolean state to be the opposite of the state

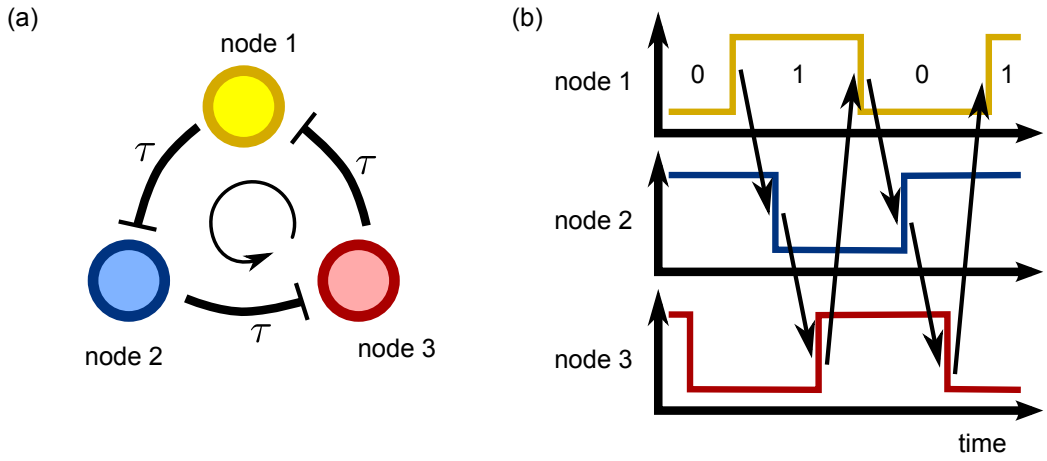


FIGURE 1: (a) Example of an autonomous Boolean network with three nodes. Each node inhibits one neighbor as indicated by arrows. (b) Schematic of the resulting dynamics. The Boolean states are indicated by "0" and "1" in the first waveform.

of the input node, which is referred to as inhibition in a biological context. The specific dynamics of the network depends on the underlying modeling framework and corresponding parameters, which I discuss below, but, for large enough processing delays, the states of the nodes oscillate. This oscillation is a result of an odd number of inversion operations and processing delays of the nodes, as illustrated in Fig. 1(b). A transition in the first node results in a transition in the second node after one processing delay; after three processing delays, the first node displays another transition, resulting in an oscillation period of six processing delays.

2.3 BOOLEAN NETWORK MODELS

In this section, I describe three standard Boolean network models, known as Kauffman networks, Boolean delay equations, and piecewise-linear differential equations by Glass and collaborators. The first assumes synchronous operation and the latter two autonomous operation.

2.3.1 KAUFFMAN NETWORKS

In 1969, Kauffman popularized a synchronous Boolean network description for genetic control circuits, where genes are approximated as Boolean nodes that switch between active ("on") and inactive ("off") and links that describe the interactions of genes via Boolean functions. The nodes change their Boolean states at fixed time steps in synchronous temporal evolution. This is mathematically described with a map, where one time step corresponds to the node processing delay.

In Kauffman's description, N Boolean nodes interact via their Boolean states X_i, according to the Boolean map

$$X_i(t+1) = \Lambda_i(X_{i_1}(t), X_{i_2}(t), ..., X_{i_K}(t)), \quad i = 1, ..., N, \tag{1}$$

where $\Lambda_i(\cdot) : \{0,1\}^K \to \{0,1\}$ are Boolean functions with inputs from K nodes in the network [Kau69, Kau93]. The Boolean functions, which are associated with the genetic interaction, are picked at random because they were (and still are) unknown. Kauffman assumed that the Boolean functions are evaluated simultaneously in discrete time steps t. Under these conditions, this Boolean network model is known as *Kauffman N-K networks* or simply *Kauffman networks*.

Mathematically, such a Boolean map description is a finite-state machine or cellular automaton, with a phase space composed of 2^N states and rules for the transition between states [Neu66, Wol83]. The finite number of states means that every trajectory will at some point reach a previously visited state. From there, since the dynamics is deterministic, the trajectory will fall into a limit cycle.

To characterize the dynamics, distance measures tailored for Boolean systems are needed. This is especially necessary to characterize the complexity of the dynamics, such as the divergence of nearby orbits [Gon12]. A widely used Boolean distance measure is the *Hamming distance* from coding theory, which reads for two network states $\{X_i\}_{i=1}^N$ and $\{Y_i\}_{i=1}^N$ with Boolean states $X_i, Y_i \in \{0,1\}$,

$$h = \sum_{i=1}^{N} |X_i - Y_i| \,. \tag{2}$$

The Hamming distance corresponds to the number of nodes in the network that differ in their Boolean states.

For Kauffman networks, the Hamming distance can under certain conditions increase exponentially over time calculated between two initially close network states, *i.e.* consider a small perturbation of the network dynamics by switching the Boolean states of a few nodes. These networks satisfy, therefore, the sensitivity to initial conditions of chaotic systems [Pom09]. On the other hand, because Kauffman networks are finite-state machines, all orbits are closed and periodic, which violates one condition for deterministic chaos. I discuss deterministic chaos and its requirements in detail in Sec. 4.2. The periods can, however, be as long as $T = 10^{150}$ iterations for N-K networks of $N = 10^3$ nodes and in-degrees of $K = N$ [Kau69].

Kauffman networks can display a dynamical transition to such long trajectories with exponential growth of the Hamming distance. The dynamical instability has implications for biology because Kauffman proposed that different attractors in Boolean networks correspond to different cell types of organisms [Kau69]. Specifically, this dynamical instability in biological systems has been hypothesized to be the cause for some types of cancer.

Furthermore, researchers have proposed that a method of controlling this dynamical instability could be a route towards curing cancer [Pom09].

Kauffman's description of genetic interaction is appealing from a network point of view because it reduces complex interacting systems, especially genetic circuits, to systems involving only network topology and Boolean functions. But, it also neglects several aspects of the physical system that could be important for the dynamics. For example, Boolean descriptions neglect continuous-variable (non-Boolean) interactions that account for the amplitude of the dynamics. Furthermore, Kauffman's description does not include continuous-time interactions and finite transmission delays between nodes. Time delays have been proven to play a crucial role for the dynamics in many systems because they lead to an infinite-dimensional phase space. For example, time delays can dictate the periodicity of oscillations and stabilize and destabilize fixed points and periodic orbits [Sch07, Hoe05, Ern09, Ata10, Jus09, Flu13, Sun13a].

2.3.2 BOOLEAN DELAY EQUATIONS

Ghil and Mullhaupt introduced Boolean delay equations as an autonomous Boolean network model [Ghi85]. The Boolean state of the node X_i evolves according to the Boolean delay equation

$$X_i(t) = \Lambda_i(X_{i_1}(t-\tau_{i,1}), X_{i_2}(t-\tau_{i,2}), ..., X_{i_N}(t-\tau_{i,N})), \quad i = 1, ..., N, \tag{3}$$

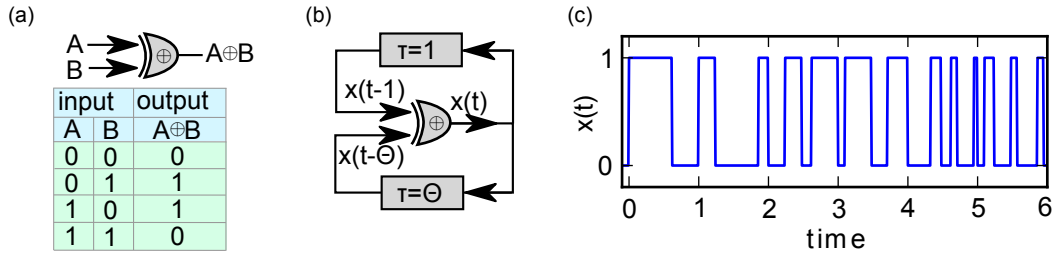
which has a similar structure as Kauffman networks in Eq. (1) with the Boolean function Λ_i on the right hand side. However, the resulting dynamics can be very different from Kauffman networks because it includes continuous-time updating and time delays $\tau_{i,j}$, which correspond to the transmission times along the links. Boolean delay equations can be used to model genetic circuits [Che13].

Ghil and Mullhaupt are especially interested in the dynamics of a particular Boolean network given by the Boolean delay equation

$$X(t) = X(t-\theta_1) \oplus X(t-\theta_2) \oplus ... \oplus X(t-\theta_\delta), \tag{4}$$

which includes $\delta \geq 2$ time delays θ_i with $0 < \theta_\delta < ... < \theta_2 < \theta_1 = 1$ [Ghi85]. The operator $\oplus: \{0,1\} \times \{0,1\} \to \{0,1\}$ denotes the "*exclusive or*" (*XOR*) operation that maps two Boolean inputs that have combined $2^2 = 4$ possible states, namely 00, 01, 10, and 11, to one Boolean output. This mapping is uniquely defined by a *look-up table* that connects all possible Boolean input combinations (here, a total of four) to one Boolean output value. Specifically, Fig. 2(a) shows the look-up table for the XOR logic operation. Equation (4) includes $\delta - 1$ XOR operations with two inputs each, which is equivalent to a single generalized δ-input XOR operation.

The Boolean delay equation (4) is visualized with a circuit diagram in Fig. 2(b) for $\delta = 2$, where I use the standard graphical representation of an



input		output
A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

FIGURE 2: (a) Look-up table and a variation on the ANSI/IEEE Std 91-1984 representation for an XOR logic gate. The look-up table determines the Boolean output of the logic gate for every possible combination of Boolean inputs. (b) Illustration of the circuit that is represented by the Boolean delay equation (4) with $\delta = 2$ delays. The delayed feedback lines are represented by wire connections and rectangles. (c) The solution $x(t)$ of Eq. (4) with delays of $\theta_2 = (\sqrt{5}-1)/2$ and $\theta_1 = 1$; the dynamics are initialized with one transition at time $t = 0$.

XOR logic gate. It can be seen that the XOR logic operator is subject to two delayed feedback lines.

This Boolean delay equation [Eq. (4)] leads to aperiodic dynamics, when the delays $\{\theta_i\}_{i=1}^{\delta}$ are incommensurate, for all initial conditions except $x(t) \equiv 0$ [Ghi85, Ghio8]. On the other hand, the initial condition $x(t) \equiv 0$ ($t \in (-1, 0]$) leads to a stable fixed point, where the output and the input of the Boolean function stays at the low Boolean value.

The resulting dynamics is shown in Fig. 2(c) for an initial function that includes one initial transition at time $t = 0$ and $\delta = 2$ delays of $\theta_2 = (\sqrt{5}-1)/2$ and $\theta_1 = 1$ in Eq. (4). The figure shows that with each time unit (corresponding to the delay $\theta_1 = 1$), the number of transitions increases. In fact, this increase follows a power law in time as reported by Ghil and collaborators [Ghi85, Ghio8]. Because these increasingly fast dynamics result in an unlimited growth of frequency over time, Zhang and collaborators referred to that effect as an inevitable *ultraviolet catastrophe* [Zha09a].

This complex behavior is not practically observed in nature because the information-transmitting wires (or media) and the processing element (the XOR logic operator) have, when physically realized, a maximum operation frequency. Hence, they cannot transmit or generate signals above a certain frequency. For electronics, this effect is known as low-pass filtering. A maximum operation frequency also exists in biological systems, such as biological genes.

2.3.3 PIECEWISE-LINEAR DIFFERENTIAL EQUATIONS

To overcome these problems, Glass and collaborators proposed an autonomous Boolean network model with continuous-time, continuous-state differential equations that include Boolean switching terms [Gla98, Edwoo,

Mes97]. Specifically, Kauffman networks in Eq. (1) are expanded with piecewise-linear differential equations that include a first-order approach to the Boolean levels, according to

$$\frac{dx_i}{dt} = -x_i + \Lambda_i(X_{i_1}(t), X_{i_2}(t), ..., X_{i_K}(t)), \quad i = 1, ..., N, \tag{5}$$

where, similar to Kauffman networks, $\Lambda_i(\cdot) : \{0,1\}^K \to \{0,1\}$ are the Boolean functions and $\{X_i\}_{i=1}^N$ the Boolean states. The equation describes the continuous temporal evolution of continuous states $\{x_i\}_{i=1}^N$, which are used to calculate the Boolean states according to the threshold condition

$$X(t) = \begin{cases} 1, & \text{if } x(t) \geq 0.5, \\ 0, & \text{if } x(t) < 0.5, \end{cases} \tag{6}$$

Equation (5) is more realistic than Eq. (1) to describe physical systems, but it is still a highly simplified model. Glass and collaborators justify this step towards higher complexity with the model's "remarkable mathematical properties that facilitate theoretical analysis" [Gla98, Edw05]. For example, Eq. (5) can be solved analytically with simple exponential functions.

To construct the analytical solution, consider the times $\{t_1, t_2, ..., t_k\}$ of switching events when any of the variables x_i crosses the threshold 0.5 and hence the Boolean functions can change values. The solution of Eq.(5) is then

$$x_i(t) = x_i(t_j)\mathrm{e}^{-(t-t_j)} + \Lambda_i(X_{i_1}(t_j), X_{i_2}(t_j), ..., X_{i_K}(t_j))(1 - \mathrm{e}^{-(t-t_j)}), \tag{7}$$

for $t \in [t_j, t_{j+1}]$ [Gla98].

As an example, I discuss the resulting dynamics for a single variable x in the network with $\Lambda = 0$ for $t < 0$ and $\Lambda = 1$ for $t \geq 0$. Then, the dynamics, shown in Fig. 3, approaches the Boolean level of Λ with a rise time of

$$T_{1/2} = \ln(2). \tag{8}$$

The figure also shows the corresponding Boolean variable X that switches Boolean states when x reaches 0.5. Due to the finite rise time $T_{1/2}$, the Boolean variable X takes on the value of Λ only after $T_{1/2}$, leading to the effective processing delay $T_{1/2}$.

Mestl and collaborators have investigated the dynamics of Eq. (5) for random networks of N nodes with a fixed in-degree of K [Mes96, Mes97]. They have chosen the Boolean functions of the nodes at random by filling the look-up table with 0's and 1's with a probability p. This probability p is known to produce more complex dynamics the closer it is to 0.5 [Shm04]. They also include a slight variation on the Boolean levels for each gate by ± 0.01 to introduce heterogeneity and thus increase the complexity in the

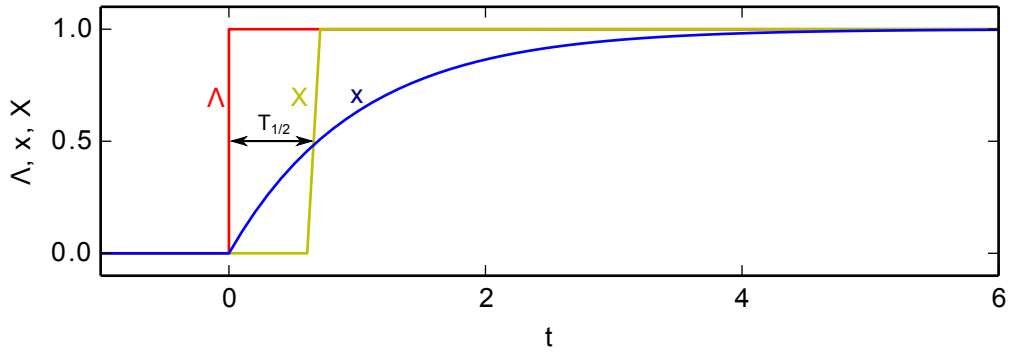


FIGURE 3: Analytic solution Eq. (7) of Eq. (5) for a Boolean driving term Λ that switches from zero to one at $t = 0$. Shown are Λ, x, and X and the rise time $T_{1/2}$.

dynamics. They have shown that, for $N = 64$, $9 \leq K \leq 25$, and $p = 0.5$, chaos is the usual behavior [Mes97, Gla98]. It is assumed that chaos occurs for most network realizations when the network is above a certain size and $p \approx 0.5$ [Gla13]. In these studies, the network topologies exclude self-feedback loops and loops that are composed of only two nodes because they can lead to fast oscillations that dominate the dynamics [Gla98].

The solution of this network of N nodes exists in a phase space of N dimensions. An inclusion of time delays in Eqs. (5), however, will result in a much larger phase space and is hence likely to have a drastic effect on the dynamics. I include such time delays to model the transmission time of the signals between nodes to model experimental dynamics in Sec. 4.4.2.

2.3.4 OVERVIEW OF BOOLEAN NETWORK MODELS

The three standard Boolean network models are summarized in Table 1. The Boolean networks interact via discrete states and, in the piecewise-linear differential equations by Glass and collaborators, an additional continuous variable is used for the temporal evolution of nodes. They evolve either in discrete time steps or in continuous time, which determines their type to be either synchronous or an autonomous, respectively.

	N-K networks	Boolean delay equations	Glass models
states x	discrete	discrete	discrete/continuous
time t	discrete	continuous	continuous
type	synchronous	autonomous	autonomous
math. descrip.	finite-state machine	Boolean delay equation	ordinary differential equation

TABLE 1: Overview of the three discussed models for Boolean networks. I use the abbreviation 'Glass models' for piecewise-linear differential equations by Glass and collaborators [Gla98]

2.4 ELECTRONIC REALIZATIONS OF BOOLEAN NETWORKS

Central processing units (CPUs) are highly specialized, electronically realized synchronous Boolean networks, similar to Kauffman networks. These systems are finite-state machines where set rules determine the transition from one state to the next every clock cycle, where clock speeds can be as high as several gigahertz. CPUs are the method of choice to perform linear operations at a high rate and are included in everyday electronic equipment. However, from a fundamental point of view, the physical network problem becomes more interesting when synchronous clocking is removed from the setup and replaced by continuous-time evolution, *i.e.* the autonomous operation. Especially when the signal transmission times matter, the system's dimensionality increases substantially. Furthermore, the operation frequency increases to the limit of the Boolean nodes. Then, the system can be used for novel network-based computing approaches and other applications.

For fundamental research, unclocked logic circuits can be used to test the validity of autonomous Boolean network models, such as the piecewise-linear differential equations by Glass and collaborators. With this goal, they have built an electronic realization of a Boolean network of five nodes based on unclocked logic gates [Glao5]. In this study, they found qualitative agreement between model and experiment in both periodic and chaotic dynamical states, when parameters used in the simulation are derived from the experimentally measured dynamics. However, they have not shown that the experimental dynamics is indeed deterministic chaos. Furthermore, their dynamics is, with a timescale on the order of tens of milliseconds, rather slow for applications.

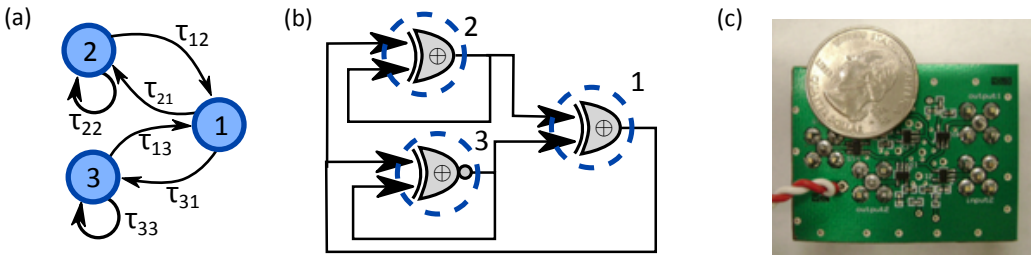


FIGURE 4: (a) Schematic of the network topology considered by Zhang and collaborators in Ref. [Zha09a]. (b) Schematic of the corresponding logic circuit with XOR and XNOR logic gates. The look-up table of an XNOR gate can be obtained by inverting the output row of the look-up table of an XOR gate shown in Fig. 2(a). (c) Experimental implementation with integrated circuits that perform Boolean operations (logic gates, black rectangles) on an electronic circuit board (photo by Seth Cohen).

2.4.1 AUTONOMOUS BOOLEAN NETWORK BY ZHANG AND COLLABORATORS

As an extension, Zhang and collaborators have realized an unclocked logic circuit with a timescale on the order of nanoseconds and have shown that deterministic chaos occurs [Zha09a].

Their Boolean network is composed of three nodes with a topology shown in Fig. 4(a). The system is an electronic circuit that realizes the Boolean nodes with logic gates, specifically two XOR logic gates and one XNOR (inverted XOR) logic gate as shown in Fig. 4(b). For the physical implementation of this logic design, they use several separate electronic integrated circuits that each execute one logic function and connect them with electronic wires on a printed circuit board as shown in Fig. 4(c).

Zhang and collaborators find that, depending on the delays in the circuit, the circuit displays either periodic dynamics or chaotic dynamics. Figure 5(a) shows a time series from chaotic dynamics recorded by Zhang and collaborators. The dynamics fluctuates between the Boolean low and high voltage of 0 and 3 V with an irregular timing of transitions. Narrow pulses and dips in the chart do not reach the Boolean voltages because of finite rise and fall times. This non-ideal behavior is due to low-pass filtering of the electronic logic gates, specifically, capacitances in the micro circuits that constitute a logic gate. In addition, amplitude noise is present as can be seen in the graph when the system is close to the Boolean voltage levels. Figure 5(b) shows the power spectrum of this dynamics. It extends from dc to high frequencies of $\sim 1.3\,\mathrm{GHz}$ at $-10\,\mathrm{dB}$ dropoff. This large bandwidth is a characteristic of chaos, which is reassured by the irregularity of the waveform.

Zhang and collaborators model Boolean chaos with an extension on Ghil's Boolean delay equations that includes non-ideal attributes of the ex-

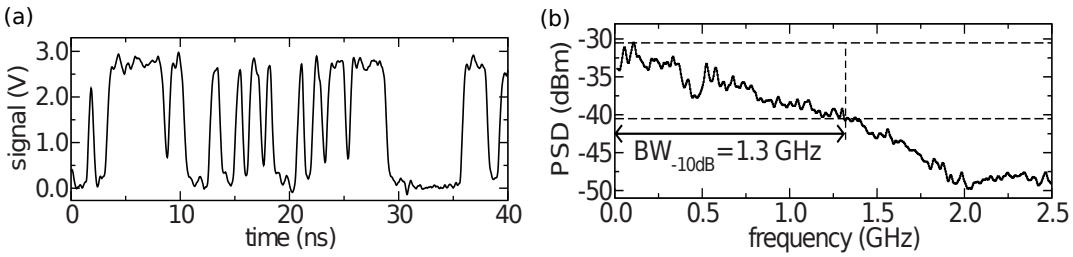


FIGURE 5: (a) Waveform of Boolean chaos generated by the system described in Fig. 4(b). (b) Power spectrum of the dynamics with a bandwidth at $-10\,\mathrm{dB}$ dropoff of $1.3\,\mathrm{GHz}$. The illustrations are taken from Ref. [Zha09a].

periments [Cav10, Zha09a]. Specifically, important effects included in the model are low-pass filtering and a degradation function that includes a rejection of short-pulses and a history-dependent gate delay.

2.4.2 BOOLEAN CHAOS

In Section 2.3.1, I have discussed that the quantification of complexity in Boolean systems requires a distance measure specialized for Boolean systems, such as the Hamming distance. However, the Hamming distance is a measure for synchronous Boolean systems that does not work for small autonomous Boolean systems. As a solution for autonomous Boolean networks, Zhang and collaborators use a distance measure that is sensitive to the timing of Boolean transitions, termed the *Boolean distance* [Ghi85, Zha09a]. It is defined as

$$d\,[x,y]\,(t) = \frac{1}{T}\int_t^{t+T} x(t') \oplus y(t'), \tag{9}$$

where x and y are two Boolean waveforms that are compared, T indicates an integration interval, which should include on average about five transitions, and $\oplus$ indicates the XOR Boolean function of the Boolean scalars $x(t')$ and $y(t')$ [Ghi85, Zha09a]. The result is a contribution to the integral whenever the two waveforms have different Boolean states; specifically, $d\,[x,x] = 0$.

Using the Boolean distance, they calculate the largest Lyapunov exponent Λ of their system, which is a measure for the divergence of close orbits used to quantify chaotic systems, as I introduce in Section 4.2.1. They calculate a Lyapunov exponent of $\Lambda = 0.16\,\mathrm{ns}^{-1}$ from the experimental time series of their Boolean oscillator. The positive sign confirms the divergence of close orbits and is usually considered a proof of deterministic chaos.

They show that the complexity and chaoticity of the dynamics is encoded in irregular timing of transitions. Specifically, with the calculation of the Lyapunov exponent, they show that small perturbations in the timing of

transitions grow exponentially over time, leading to completely different transition times after a long time [Zha09a]. On the other hand, a small perturbation in the voltage from the Boolean level does, in most cases, not affect the dynamics over time.

Zhang and collaborators termed the chaotic dynamics in an autonomous Boolean system *Boolean chaos*. Boolean chaos can possibly be applied to random number generation and chaotic radar (radio detection and ranging) because of the broad power spectrum and the fast time-scale dynamics. Furthermore, Boolean chaos in the experiment also gives fundamental insight into the dynamics generated by autonomous Boolean networks [Zha09a].

2.5 CONCLUSION

In this chapter, I have discussed previous work on Boolean networks. I have destinguished between synchronous and autonomous operation, which results in very different dynamics. In the next chapter, I discuss a new method of realizing experimental autonomous Boolean networks.

3 AUTONOMOUS BOOLEAN NETWORKS ON ELECTRONIC CHIPS

3.1 ABSTRACT

In this chapter, I discuss the experimental implementation of autonomous Boolean networks on electronic chips. Specifically, I describe the setup and non-ideal characteristics of the used microelectronic chips in Sec. 3.2 and the design flow of implementing circuits in Sec. 3.3. With this chapter, I lay the technical foundation for this thesis. As a simple exemplary system, I implement the autonomous Boolean network by Zhang and collaborators [Zha09a] that is introduced in Sec. 2.4.1. Instead of implementing the logic circuit with discrete logic gates on a printed circuit board as in their study, I realize it on a single electronic chip known as a field-programmable gate array (FPGA), which has several advantages. Specifically, the re-configurable chip allows for fast and inexpensive design cycles when compared to printed circuit boards that have to be re-manufactured for each instantiation. In addition, electronic chips allow for a much larger number of network nodes on the order of 100,000. Similar to the design by Zhang and collaborators, the resulting network evolves on a fast timescale, which is indispensable for many applications. The purpose of this chapter is to introduce the experimental platform used in the rest of this thesis.[1]

3.2 FIELD-PROGRAMMABLE GATE ARRAYS

Autonomous Boolean networks can be realized experimentally with electronic logic circuits on various microelectronic chips, which are the foundation of modern computing hardware. In this thesis, I mainly implement the logic circuits with programmable microelectronic chips called *field-programmable gate arrays (FPGA)*; specifically, I use the FPGA Cyclone IV with model number EP4CE115F29C7N. In Sec. 5.4.1, I also use several other FPGAs and a device called a complex programmable logic device (CPLD) to show that some of the autonomous Boolean networks I study generate similar dynamics independent of the specific hardware. CPLDs are based on an older technology than FPGAs with a lower number of programmable logic elements. The autonomous Boolean networks can, in principle, also

1 A part of the content of this chapter is published in Refs. [Ron12, Ros13b].

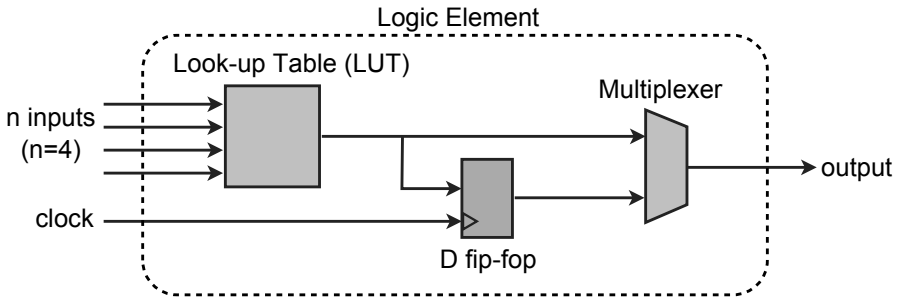


FIGURE 6: Functional description of an FPGA logic element.

be implemented as application-specific integrated circuits (ASIC), which are custom-built microelectronic chips. ASICs allow for faster logic and, in large production sizes, lower costs, but they have much longer and more expensive design cycles.

3.2.1 ARCHITECTURE OF FIELD-PROGRAMMABLE GATE ARRAYS

FPGAs are off-the-shelf devices that include a grid of CMOS-based programmable logic elements and programmable connections. They can be configurated to realize a custom hardware design for a wide range of applications, such as digital signal processing, prototyping, and high-performance computing [Max09]. Here, I discuss both programmable logic elements and programmable connections.

Logic gates are physical implementations of Boolean functions such as the XOR Boolean function. By cascading electronic logic gates, mathematical algorithms can be implemented physically to perform calculations, such as addition. A Boolean function of K inputs is defined with a look-up table of 2^K Boolean entries, leading to 2^{2^K} possible operations.

The programmable logic gates on FPGAs are called logic elements, which include a look-up table block (LUT), a flip-flop, and a multiplexer as shown in Fig. 6. The LUT can implement any of the 2^{2^K} possible Boolean functions, defined by 2^K bits that are saved to random access memory (RAM) at the configuration phase (startup) of the FPGA. Furthermore, each logic element includes a flip-flop to allow for clocked operation. A multiplexer, controlled by another RAM bit, is used to switch between clocked and un-clocked operation. The logic gate has some additional features that are not shown here, such as different outputs routing back to itself, routing to the logic gates close to it within a region called logic array block, and to the routing fabric. Furthermore, logic gates have a carry-bit input and output that connects neighboring logic gates with reduced delay used for the implementation of fast adders [Alt10].

The Altera Cyclone IV FPGA, which I mainly use in this thesis, includes more than 10^5 logic elements. Each element has $K = 4$ inputs and can drive up to 48 other logic elements [Alt10].

The connections between logic gates are achieved via on-chip wires called interconnect. The interconnect is organized as shown in Fig. 7; logic elements are grouped together in logic array blocks (LAB) of 16 logic elements, which are connected via local interconnect. In addition, the local interconnect is connected to adjacent LABs via direct links and to all other LABs via row and column interconnect. The specific connection between logic elements is turned on and off by RAM bits that are loaded onto the chip at startup [Alt10].

In addition to logic elements and interconnect, FPGAs also contain various other elements depending on the complexity of the chip. The Altera Cyclone IV device used in this thesis, for example, includes four configurable phase-locked loops, configurable memory, and several embedded multipliers [Alt10].

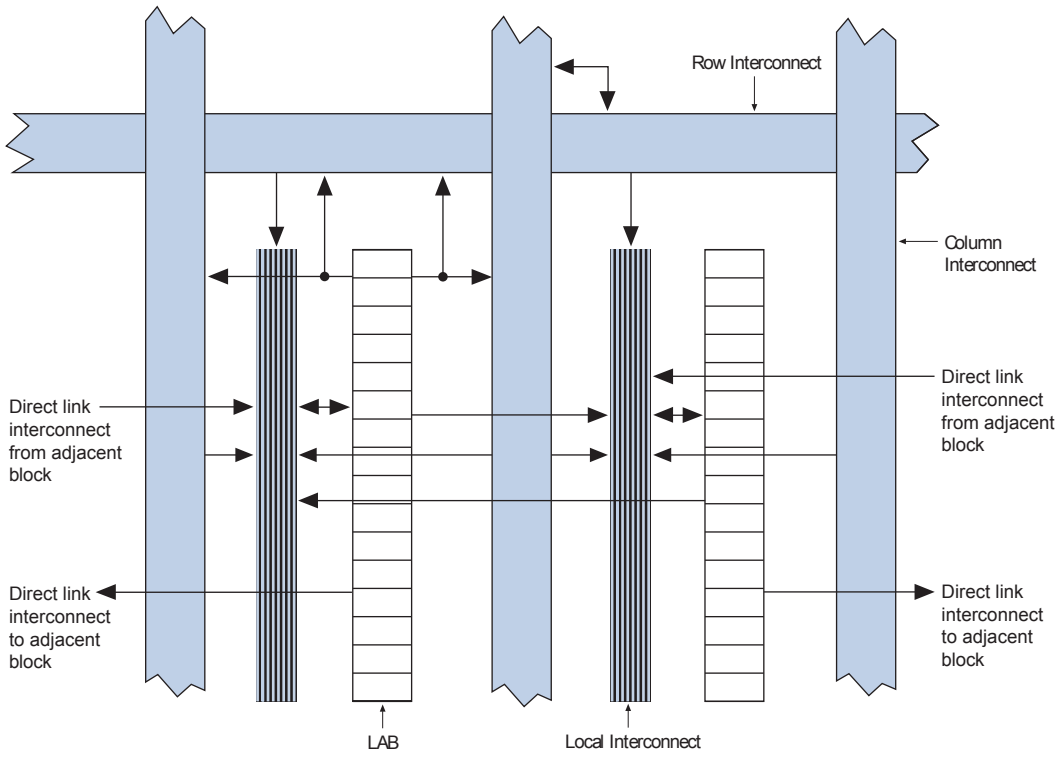


FIGURE 7: Schematic of the interconnect on the Cyclone IV FPGA. The Figure is modified from Ref. [Alt10].

3.2.2 AUTONOMOUS MODE OF OPERATION

Logic circuits can be set to operate in different modes with important implications for the dynamics, similar to differences between synchronous and autonomous Boolean networks (see Sec. 2.2). In this thesis, I deal with two modes of operation of logic elements: synchronous (clocked) and autonomous (un-clocked). Another mode of operation that I do not discuss further is known as asynchronous design, where a self-clocked circuit generates its own clock signals that indicate completion of operations [Wer97].

Synchronous operation is used for most applications of digital designs, such as for processor designs. This mode is achieved by including clocks and flip-flops in the logic circuit. Flip-flops store the state information and update only once every period of the clock. The clock frequency is chosen slow enough to ensure that logic gates have enough time to settle to unambiguous Boolean states between consecutive clock cycles [Bro08]. As a result, a properly clocked system behaves in a digital fashion as a fully predictable finite-state machine, similar to Kauffman networks (see Section 2.3.1).

In the autonomous mode of operation, in contrast, the logic circuit does not include clocks. As a result, the circuit displays a continuous-time dynamical evolution governed by the logic gates' continuous dynamics and propagation delays [Zha09a]. The logic gates, however, still fulfill Boolean threshold conditions and output the Boolean voltages most of the time, so that autonomous logic circuits can be regarded as Boolean networks.

Different from the synchronous operation, autonomous logic circuits can be very sensitive to small changes in the properties of the logic elements, caused, for example, by ambient temperature fluctuations. In synchronous operation, the clocking guarantees that the system will implement the same finite-state machine if the transmission delays stay below the clock period. However, in the autonomous operation, the system is sensitive to non-ideal effects, such as time delays, because they are an important part of the dynamical system. Consequently, two autonomous logic circuits of identical layout that are realized on different regions on the FPGA may display slightly different dynamics because logic gates vary slightly in their non-ideal effects due to production variation.

3.2.3 NON-IDEAL EFFECTS OF AUTONOMOUS LOGIC GATES

Physically-implemented autonomous logic gates are subject to non-ideal effects that deviate from perfect Boolean switching. Figure 8 shows these non-ideal effects within an equivalent circuit for an autonomous logic gate. The logic gate is shown with an equivalent circuit that includes ideal Boolean operation on the Boolean inputs, a sigmoidal gate activation function, a low-pass filter, and a gate propagation delay.

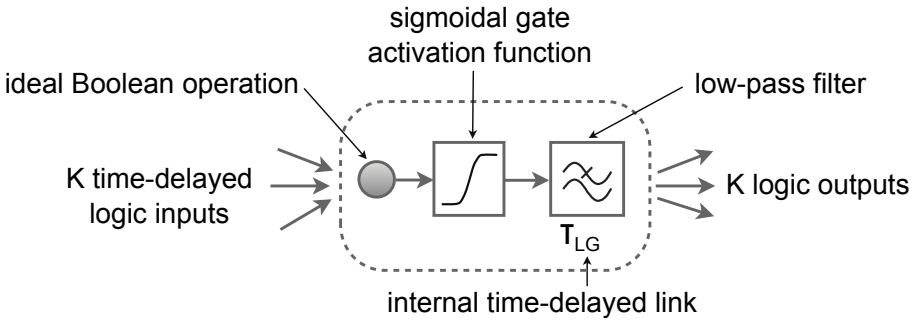


FIGURE 8: Simplified model of a few non-ideal behaviors present in an electronic logic gate.

The low-pass filtering is caused by capacitors inside the logic gate that need a finite time to charge until the output can change value, leading to a maximum frequency that a logic gate is able to respond to. One result of the low-pass filter is short-pulse rejection, where short pulses at the input to a logic gate do not affect the output of a logic gate.

The delay and the low-pass filter properties can be state- and history-dependent, meaning that its parameters, such as the propagation delay or the rise and fall times, depend on the near history of Boolean states and the current Boolean state. One example for state dependency of the low-pass filter is that rise and fall times can be different. Cavalcante and collaborators have identified memory effects as an important dynamical feature of the system to generate chaotic dynamics [Cav10]. They termed the memory effect "degradation" and described it mathematically with a degradation function.

Another non-ideal property of physically-realized Boolean networks is heterogeneity, where copies of the same logic gates differ in their properties, such as the filter properties and the gate propagation times. I quantify the heterogeneity of the propagation delay of logic gates in Appendix A.2. Furthermore, the dynamics are subject to amplitude noise and phase noise.

3.3 DESIGN FLOW OF IMPLEMENTING AUTONOMOUS BOOLEAN NETWORKS ON ELECTRONIC CHIPS

In this section, I explain how I generate configuration binary files, which are loaded onto an FPGA to implement a custom logic circuit, such as an experimental realization of an autonomous Boolean network. This binary file is generated with the help of *computer aided design (CAD) tools*, such as Altera Quartus II, which, among other operations, optimizes the logic circuit for best functionality. However, the optimization algorithms are usually

written for synchronous and not autonomous designs. To define the logic design, one can, for example, draw a logic diagram, known as schematic design. I prefer a text-based approach with a hardware description language because it allows me to generate hardware descriptions of many nodes using for-loops, rather than the graphics-based schematic design, where the logic circuit has to be specified by hand.

3.3.1 HARDWARE DESCRIPTION FOR AUTONOMOUS BOOLEAN NETWORKS

I discuss the hardware description for the autonomous Boolean network by Zhang and collaborators as a general example for an autonomous hardware design on an FPGA [Zha09a]. In their original hardware design they used a printed circuit board, which allowed them to control the delays by varying the supply current of the logic gates. For simplicity, I realize the delays here by separating the logic gates spatially on the electronic chip; I assume that the delay is proportional to the length of on-chip wire connecting the logic gates. This is different from the following chapters, where I use a more efficient way to realize time delays. I extend the original circuit, shown schematically in Fig. 9(a), by adding two buffer gates, so that I can adjust time delays by moving the buffers and the other logic gate with respect to each other on the chip. In addition to the buffers, the circuit includes two XOR gates and one XNOR gate.

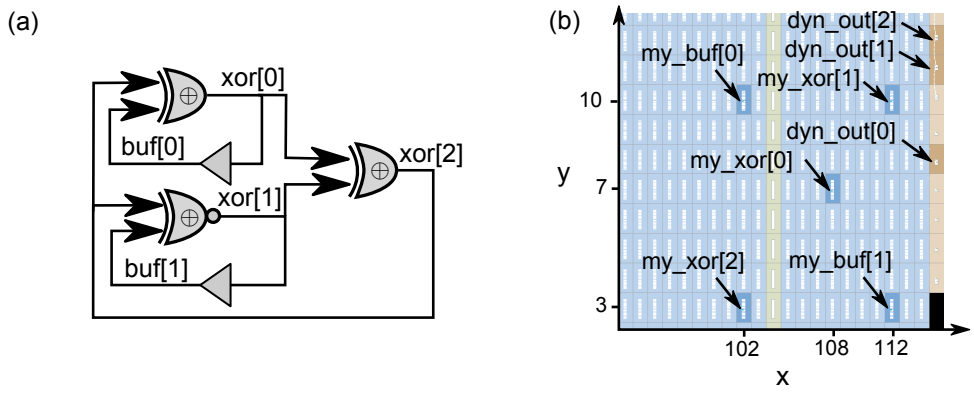


FIGURE 9: (a) Schematic of Zhang and collaborators' logic circuit [Zha09a] extended by two buffer gates. (b) Chip layout that visualizes the placement of the logic circuit on the FPGA Altera Cyclone IV with product number EP4CE115F29C7N (less than 3% of the FPGA real estate is shown). The three XOR gates are routed on locations $(x,y,z) = (108,7,8)$, $(112,10,8)$, $(102,3,8)$ and the buffer gates are at $(102,10,16)$ and $(112,3,8)$ as indicated by arrows. The output pins are also marked with arrows. The blue part of the chip are blocks of programmable logic gates, the brown line are output pins, and the green line are memory elements. The black area has no functionality.

```
1 module zhang_osc(dyn_out);
2  output [2:0] dyn_out;
3  wire [2:0] my_xor /*synthesis keep*/;
4  wire [1:0] my_buf /*synthesis keep*/;
5
6  assign my_xor[0] = my_buf[0] ^ my_xor[2];
7  assign my_xor[1] = ~(my_buf[1] ^ my_xor[2]);
8  assign my_xor[2] = my_xor[0] ^ my_xor[1];
9
10 assign my_buf = my_xor[1:0];
11 assign dyn_out  = my_xor;
12 endmodule
```

FIGURE 10: This Verilog module specifies the logic gates (nodes) and connections (topology), contained between the statements `module` and `endmodule`. The dynamics are routed to three output pins named `dyn_out`, specified as a vector with three components in line 2. The XOR, XNOR, and buffer gates are introduced in lines 3 and 4 and specified in lines 6-11. The directive `/*synthesis keep*/` (for Altera FPGAs) guarantees that the logic gates are implemented by the compiler. Some logic gates such as the buffer gates are redundant in synchronous operation and would hence be removed by the compiler. The implementation of an XOR logic gate is specified with the "^" operator; the XNOR logic gate is specified as a combination of an XOR and an inversion operation, specified with "~". The buffers are specified with a simple equal assignment in line 10. For deeper understanding of the syntax of Verilog, I refer the reader to Ref. [Mcn01].

Figure 10 shows the hardware description language to generate the circuit as an example. The code snippet shows that the logic circuit can be easily defined in a few lines. In the code, the outputs of the XOR, buffer, and output buffer logic gates are named `my_xor`, `my_buf`, and `dyn_out`, respectively. The code also shows statements that force the implementation of logic gates that are seen as redundant by the compiler and would otherwise be removed. These logic gates are indeed redundant in synchronous circuit design, but can be important in autonomous circuit design. Compiling the high-level hardware description leads to a binary programming file that includes the specific Boolean function and the routing of the logic gates on the FPGA. After loading this on the FPGA, I obtain a true physical (not emulated) network of logic gates, which is a physically realized autonomous Boolean network on a chip.

3.3.2 CHIP PLACEMENT OF AUTONOMOUS BOOLEAN NETWORKS

The specific placement of logic gates on the FPGA is usually handled by the compiler. It can, however, be modified using the Altera CAD tool Quartus II Chip Planner, which allows for better control over the circuit implementation. Each possible logic gate has a physical address x, y, z, where in addition to the address of the logic array block (LAB) $x \in \{1, 2, ..., 114\}$, and $y \in \{1, 2, ..., 72\}$, the z dimension $z \in \{0, 2, 4, ..., 30\}$ specifies the index of a logic gate within a LAB (numeric values for the Altera Cyclone IV FPGA with model number EP4CE115F29C7N). By assigning the logic gates to specific coordinates on the chip, I specify the layout shown in Fig. 9(b).

I vary the specific placement of the logic gates on the chip, which changes the time delays along the links, until chaotic dynamics appears. This method is equivalent to the method by Zhang and collaborators of changing the time delays via the supply current of logic gates [Zha09a]. For an estimate of the transmission delays between logic gates, I measure the transmission delay between two logic gates that are spaced as far as the two opposite corners of the chip. Depending on the specific wiring, the delay varied between 2.5 and 5 ns, which is one order of magnitude greater than the characteristic timescale of a logic gate (rise time of $\tau_{LG} = 280 \pm 10$ ps). The large variation of the time delay is a problem of this method of separating logic gates on the chip. This method is also very inefficient because the circuit could fit in a single LAB if the delays were generated more efficiently, as introduced in the next chapter.

Figure 9(b) also shows output gates that are used on the FPGA to route the signals outside the chip. Such gates are also implemented when signals are sent into the chip. These input output (I/O) gates alter the time course of the waveforms because they include the non-ideal effects discussed in Section 3.2.3, such as a sigmoid gate activation function and lowpass-filtering. For example, some characteristics of the waveform could be pruned when the lowpass filtering of the output gate is stronger than the one of the programmable logic gates. The output logic gates, however, cannot be circumvented because they protect the chip from overvoltage and scale the input and output voltages to desired values. Furthermore, the chip manufacturers require these logic gates to protect them against reverse engineering [Max09]. This is also the reason why they do not provide specific information of the properties of the programmable logic gates, such as the SPICE model parameters [Ant93].

3.3.3 RESULTING DYNAMICS

Figure 11 shows the resulting dynamics of the logic design (a detailed discussion of this dynamics is left for the next chapter). The measured dynam-

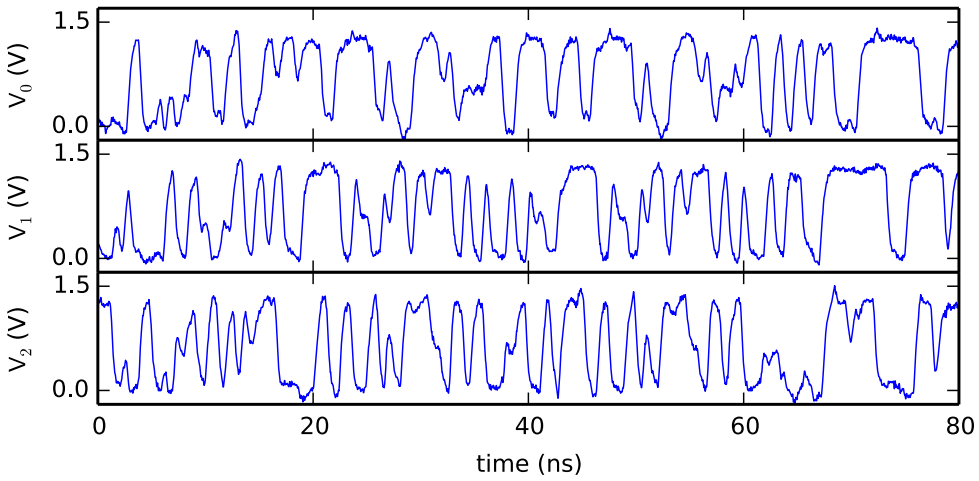


FIGURE 11: Dynamics of the autonomous Boolean network in Fig. 9 realized on the FPGA. V_0, V_1, and V_2 are the outputs of the three XOR logic gates in the circuit. The dynamics oscillate between the Boolean voltages $V_L \approx 0$ and $V_H \approx 1.4\,\mathrm{V}$ (depending on the specific output pin). I use the FPGA Altera Cyclone IV model EP4CE115F29C7N.

ics shown in the figure are similar to the dynamics measured by Zhang and collaborators on a printed circuit board in Fig. 5(a) [Zha09a], which reassures me that the implementation of autonomous Boolean networks on FPGAs is possible and results in dynamics that agree with previous studies.

3.4 CONCLUSION

In this chapter, I have shown that autonomous Boolean networks can be implemented on programmable logic devices. The resulting dynamics agree with previous experimental implementations. By using reprogrammable chips, however, I am able to build much larger networks in shorter time compared with previous studies. As a first dynamics, I have observed Boolean chaos, which is the topic of the next chapter.

4

CHAOTIC DYNAMICS OF AUTONOMOUS BOOLEAN NETWORKS

4.1 ABSTRACT

In this chapter, I apply autonomous Boolean networks to one particularly popular topic of complex systems research: deterministic chaos. I first introduce the concept of deterministic chaos in Sec. 4.2 and then extend in Sec. 4.3 the previous work on Boolean chaos with a simple autonomous Boolean network that has been proposed in a similar form in an early theoretical study by Ghil and Mullhaupt [Ghi85]. I measure and analyze the dynamics of an experimental implementation and develop a time delay piecewise-linear switching model for autonomous Boolean networks in Sec. 4.4. Both experimental and simulated dynamics agree qualitatively in power spectrum and autocorrelation.

The main result of this chapter is the development of a particularly simple autonomous Boolean network for generating Boolean chaos using guidelines from Boolean network models. These guidelines, however, cannot predict with certainty whether a network will display complex dynamics. For example, a prediction of complex dynamics with a Boolean network model breaks down when the network is realized in the experiment, where the network instead shows transient relaxation towards a fixed point.[1]

4.2 INTRODUCTION TO DETERMINISTIC CHAOS

Deterministic chaos is a non-repeating, deterministic behavior of dynamical systems that can be found in meteorology, physics, engineering, economics, and biology [Str94a, Sch84]. It can be defined with three properties [Str94a]. First, chaotic systems display aperiodic long-term behavior, meaning that they do not settle down asymptotically to fixed points or periodic orbits. Second, they are deterministic in that their behavior arises from the system's nonlinearity rather than from noise. Third, chaotic systems are sensitively dependent on initial conditions, which is popularly known as the *Butterfly Effect*.

The paramount example of a system generating deterministic chaos is the *Lorenz system*, which was developed to describe atmospheric convection

1 Results of this chapter are published in Ref. [Ros13b].

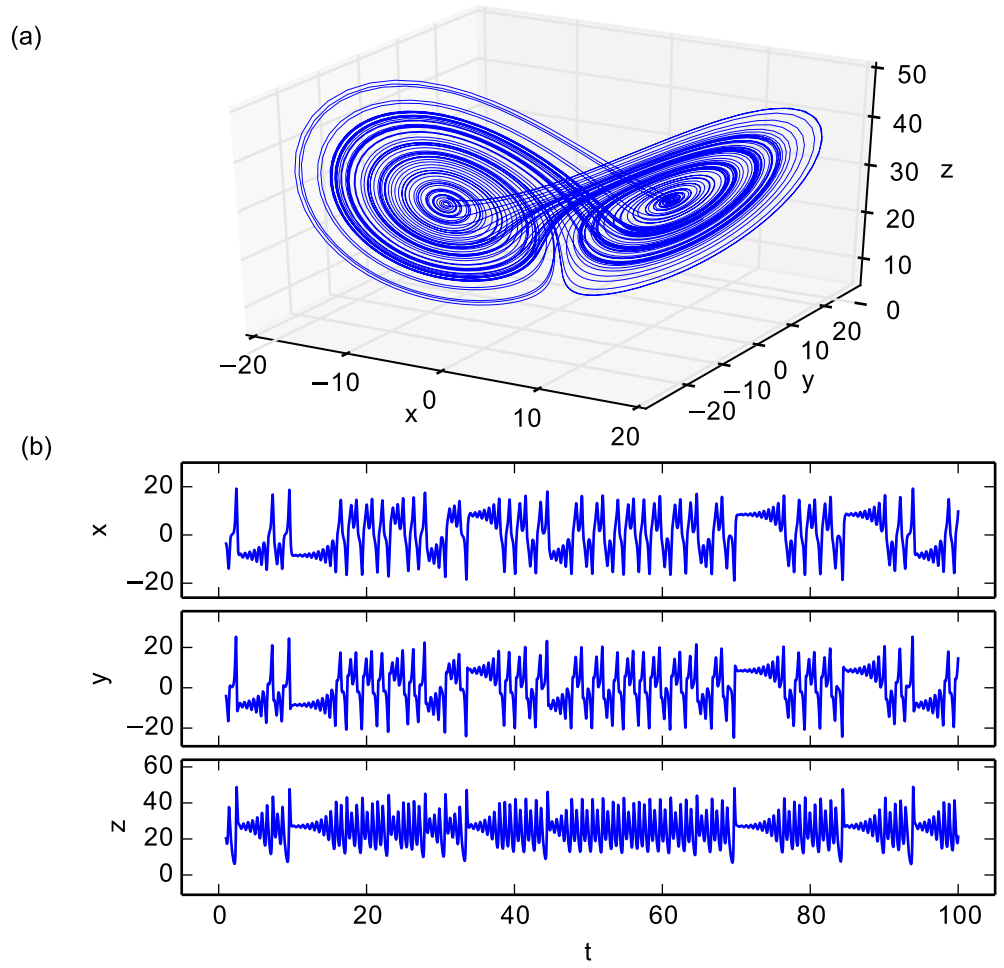


FIGURE 12: (a) Lorenz attractor and (b) waveforms generated by integrating differential equations (10). Parameters are $\sigma = 10$, $\rho = 28$, and $b = 8/3$. The differential equation is integrated with the Euler method with a time step of $dt = 0.01$.

[Lor63]. It is given by a system of the following three ordinary differential equations

$$\dot{x} = \sigma(y - x), \tag{10a}$$
$$\dot{y} = x(\rho - z) - y, \tag{10b}$$
$$\dot{z} = xy - \beta z, \tag{10c}$$

where the dot denotes the time derivative. The Lorenz system generates deterministic chaos for certain parameter values. Figure 12(a) shows its characteristic chaotic or strange attractor known as Lorenz attractor. The attractor is a subset of the phase space that trajectories are attracted to. It is generated by following a trajectory, shown in Fig. 12(b) as waveforms of the three components, in phase space over time.

4.2.1 LYAPUNOV EXPONENT

A characteristic measure for chaotic systems is the *Lyapunov exponent* [Str94a], which measures the rate of separation between close trajectories. Consider two trajectories that start at $\vec{Z}_0$ and are separated at time $t = 0$ by a small amount $\delta\vec{Z}_0$; then, this initial separation will grow or decay over small time periods according to

$$\left|\delta\vec{Z}(t)\right| \approx \exp(\lambda t) \left|\delta\vec{Z}_0\right|, \tag{11}$$

where λ is the local Lyapunov exponent in the direction of $\delta\vec{Z}_0$ at $\vec{Z}_0$. The average of the local Lyapunov exponents along the attractor results in the (global) Lyapunov exponents. When the largest Lyapunov exponent is greater than one, small differences in initial conditions grow over time; otherwise they decay or stay constant. Specifically, a positive largest Lyapunov exponent describes mathematically the necessary condition for deterministic chaos that the system is sensitive to initial conditions, which can be confirmed for the Lorenz system with $\lambda_{\max} \approx 0.9$ [Str94a].

The Lyapunov exponent λ has practical applications for random number generation, which is the topic of Ch. 5. Within a timescale given by $1/\lambda$, a small perturbation of the dynamical state approximately triples in magnitude so that it dominates the system dynamics already after several $1/\lambda$ time periods. In an experimentally realized system, such perturbations are constantly supplied by physical noise, such as thermal and shot noise. Shot noise originates from the discrete nature of electronic charge, which leads to random fluctuations in an electronic current. Thermal noise or Johnson noise originates from statistical deviations from thermodynamic equilibrium of electronic distributions, which leads to random voltage fluctuations [Hor89]. These fluctuations are a weak physical entropy source that is inherently unpredictable. When subjected to a chaotic system, the physical entropy is amplified to a rate given by the Lyapunov exponent [Mik12]. Therefore, physically-realized chaotic systems can be applied to random number generation to achieve a random bit rate that is high compared to direct measurement of the physical noise source. In addition, non-ideal characteristics of the noise source can be removed by the system dynamics, which is known as the mixing property of chaotic systems [Har12].

4.2.2 STRONG AND WEAK CHAOS

A recent important addition to chaos theory in the framework of time-delayed coupled systems is to distinguish between strong and weak chaos [Hei11]. Coupling of oscillators can, as discussed in Sec. 6.2, lead to synchronization with striking implications for biology and important applications to communication. For example, a network of two chaotic oscillators can display synchronization so that one oscillator displays the same or a

shifted chaotic waveform as the other oscillator [Hei01b, Lu04] which can be applied as a private communication scheme [Cu093, Col94, Van98a]. Encryption using synchronization of chaotic systems—so-called Chaos communication—is a popular field of research and received considerable attention when a network of three chaotic optoelectronic oscillators was implemented in the metropolitan fiber network for Athens [Arg05].

For communication applications, time delays are of particular importance because the communicating parties are usually far apart and the timescale of the dynamics is fast to encode information at high rates. Even at the speed of light, the distances translate into time delays that are large compared to the internal timescale.

Two synchronized oscillators can be analyzed by considering the so-called synchronization manifold, where the system equations reduce to the equations of one oscillator as the dynamics of the second oscillator can be deduced from the first. In this description, mutual coupling terms change to feedback terms and time-delayed coupling changes to time-delayed feedback [Koc96, Hei11, Flu09]. The Lyapunov exponent associated with the synchronization manifold encodes the chaoticity of the synchronized system and is hence of great importance for applications like chaos communication.

For coupled systems showing weak chaos, the maximum Lyapunov exponent in the synchronization manifold decreases towards zero when the delay approaches infinity ($\lambda \to 0$ for $\tau \to \infty$). Therefore, when weak chaos is used for chaos communication, the communication channel will allow for decreased privacy or a decreased rate when the communication partners increase their distance. Strong chaos, on the other hand, does not show this effect as the maximum Lyapunov exponent in the synchronization manifold tends towards a constant positive value when the delay approaches infinity ($\lambda \to \lambda_0$ with $\lambda_0 > 0$ for $\tau \to \infty$) [Hei11].

Strong and weak chaos has been measured in optical and electrical systems. In Refs. [Hei11, Hei13], the existence of strong and weak chaos is demonstrated in a single system by monotonically increasing an adjustable system parameter which shifts the system from weak to strong chaos and back to weak chaos in agreement with the system equations. The authors distinguish the system to be in a state of strong or weak chaos by measuring whether the network desynchronizes or synchronizes with another node, respectively. However, to my knowledge, strong and weak chaos has not yet been measured experimentally by observing changes in the Lyapunov exponent when the time delay is changed. Autonomous Boolean systems might be ideally suited for this task as their realization on microelectronic chips allows for an adjustment of time delays over a wide range. For this, two chaotic autonomous Boolean systems have to be synchronized, which has not been realized yet, and could be a possible extension of this thesis.

4.3 DELAYED-FEEDBACK XNOR OSCILLATOR

In this section, I introduce a new chaotic oscillator, which I call a delayed-feedback XNOR oscillator.

4.3.1 MOTIVATION FOR DEVELOPING A NEW CHAOTIC OSCILLATOR

The fundamental motivation for this chapter is to find a simple network showing Boolean chaos that consists of a small number of nodes and links. Particularly, I generalize the specific network topology studied previously by Zhang and collaborators of three interconnected nodes [Zha09a, Cav10] to a network of only one node with three time-delayed feedback lines.

This chapter is also motivated technologically by two problems of the device developed by Zhang and collaborators for applications of chaos, such as random number generation and chaos-based radar. First, their design is restricted to one specific topology and, second, it only shows chaos in certain parameter ranges of the feedback delays [Zha09a, Cav10]. However, for applications of chaos, different topologies and implementations should be explored to find the most efficient one. More importantly, the system should display chaos consistently for a wide range of parameter values.

4.3.2 SEARCH FOR A SIMPLIFIED NETWORK TOPOLOGY

In this section, I search for a simple network topology with a small number of nodes and links that displays Boolean chaos. In the construction of networks, I am guided by known properties of Boolean network models introduced in Ch. 2. These models, however, can only approximately describe the dynamics of experimentally-realized autonomous Boolean networks because they do not include all non-ideal behaviors, discussed in Section 3.2.3. I use these models to identify two guidelines by analyzing previous studies on complexity in various Boolean network models and add another guideline that I derive from an experimental observation. Then, I implement different experimental networks in accordance with the guidelines until a simple network with Boolean chaos is found.

Synchronous Boolean networks, introduced in Sec. 2.3.1, are Boolean maps, where the network state—a vector of N Boolean variables—is modified each iteration according to Boolean functions of the network state. Specifically, synchronous Boolean networks called linear feedback shift registers can generate dynamics with high complexity, which is why they can be used for deterministic random (pseudorandom) number generation (see Sec. 5.2.2). Here, high complexity can be defined via the length of limit cycles, with the maximum length of $2^N - 1$ for synchronous Boolean networks of N nodes. Heavily studied synchronous Boolean networks are Kauffman

networks, where certain assumption are made to apply them to describe biological gene networks [Kau69, Kau93] (see also Sec. 2.3.1). These assumptions are that each Boolean node evaluates a Boolean function of K inputs from the network with synchronous updating and that the Boolean functions are random, but fixed. Under the assumption of Kauffman networks, several known findings about dynamical complexity exist in the literature.

The application of predictions resulting from Kauffman networks to experimentally realized autonomous Boolean networks, however, have to be treated with caution because they are synchronous Boolean network models and hence do not account for the continuous evolution of time and time delays in the experimental system. Nevertheless, the large body of work on Kauffman networks can still provide valuable guidelines for the design of autonomous Boolean networks.

Complexity in Kauffman is assessed with the Lyapunov exponent λ using the Hamming distance, where a larger positive Lyapunov exponent means that the Boolean state vector diverges faster [Pom09] (see also Sec. 2.3.1). However, the synchronous network cannot show chaos because the phase space is restricted to $2^N - 1$ points. The Lyapunov exponent has been shown to follow the relation

$$\lambda = \log E, \tag{12}$$

where E is the average Boolean sensitivity, *i.e.*, the larger E the larger the complexity of the dynamics [Sol96, Shm04, Fri98]. The Boolean sensitivity is defined for a point $v \in \{0,1\}^K$ by the number of neighboring points (points, where only one component of v is changed) for which f differs, according to $|\{v' : f(v') \neq f(v), \mathrm{dist}(v,v') = 1\}|$ (here, the Hamming distance is used, defined in Eq. (2)) [Fri98]. The average Boolean sensitivity E can also be expressed with the bias p of "0"s and "1"s in the look-up table describing Boolean functions. Then the average Boolean sensitivity is

$$E = 2Kp(1-p), \tag{13}$$

where the average is taken from a distribution of look-up tables with bias p and in-degree K (the number of input connections to a node) [Shm04, Pom09]. The Boolean sensitivity is the highest for an XOR or XNOR gate with $E = K$.

To summarize, results obtained from synchronous Boolean networks can only give guidelines for the design of an experimental autonomous Boolean network that displays chaos because the models neglect the continuous nature of time. The synchronous models display the most complex behavior when Boolean functions have high average Boolean sensitivity, which can be achieved with randomly chosen Boolean functions of low bias an high in-degree K or most effectively by using XOR and XNOR Boolean functions.

Guideline (i)—Experimentally realized autonomous Boolean networks should include XOR and XNOR Boolean functions to achieve chaotic dynamics.

More evidence for the preference of XOR and XNOR Boolean functions for generating chaotic dynamics can be found from studies on autonomous Boolean networks modeled with piecewise-linear switching networks by Glass and collaborators [Gla98, Mes97]. They have shown that large autonomous Boolean networks have the highest probability to generate chaos when the look-up table of Boolean functions includes zero bias.

Guideline (i) is also supported by Ghil and Mullhaupt's Boolean delay equations for autonomous Boolean networks [Ghi85]. They identified complex dynamics in a system of a single n-input XNOR Boolean function with n incommensurate time delayed feedback terms (see Sec. 2.3.2). In the framework of Boolean delay equations, complex dynamics corresponds to unordered timing of transitions, where the rate of transitions grows without limit over time, identified as an ultraviolet catastrophe (see also Sec. 2.3.2).

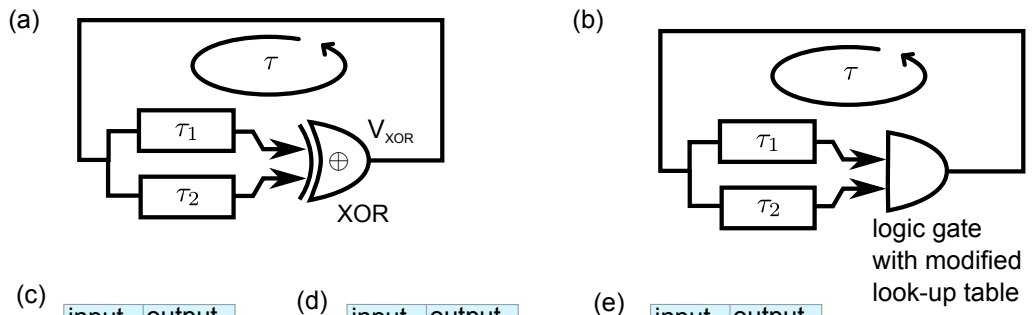
Ghil and Mullhaupt's study gives further important insight that time delays along the links play a crucial role for chaotic dynamics. Specifically, the relation of time delays is important because the dynamics changes from complex to regular when the relation of time delays changes from commensurate to incomensurate [Ghi85].

The importance of time delays is also supported by previous work by Zhang and collaborators [Zha09a], who found Boolean chaos in an experimental circuit of 2 XOR and and one XNOR logic gates with time delays (see also Sec. 2.4.1). When the relation of the time delays is changed, the dynamics can also show a transition between chaos and regular oscillations. Note that Cavalcante and collaborators found that Boolean chaos can originate from history-dependent delay, which they term degradation [Cav10].

Guideline (ii)—Experimentally realized autonomous Boolean networks should include time delays that might need to be adjusted to achieve chaotic dynamics.

My first approach to finding a simple network with a low number of nodes and links is to implement the network by Ghil and Mullhaupt shown in Fig. 13(a) and introduced with Eq. (4) ($\delta = 2$) in Section 2.3.2 [Ghi85]. This network is likely to lead to Boolean chaos because it shows complex dynamics when described with a Boolean delay equation in the framework of autonomous Boolean networks. The topology is simple: it includes only one node—an XOR Boolean function—with two time-delayed feedback links that can be adjusted, so that it conforms with guidelines (i) and (ii). I realize the network with unclocked logic gates.

Figure 14(a) shows the experimental dynamics after initialization with the high Boolean state as explained in the figure caption. After a long transient of about 50 μs, the dynamics relax to the low Boolean state. The transient length varies considerably. Therefore, the dynamics do not show



(c)

input		output
A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

(d)

input		output
A	B	
0	0	1
0	1	1
1	0	0
1	1	0

(e)

input		output
A	B	
0	0	1
0	1	0
1	0	1
1	1	0

FIGURE 13: (a) Two-input XOR gate with two time-delayed feedback lines τ_1 and τ_2 (shown as rectangles), as proposed by Ghil and Mullhaupt [Ghi85]. (b) Same setup as in (a) with a modified logic gate. Specifically, the logic gate should exclude Boolean fixed. (c) Look-up table of the 2-input XOR gate. (d), (e) The two look-up tables that do not lead to fixed point when connected as in (b) and have an equal number of "1" and "0".

stable chaos, which contrasts the theoretical results by Ghil and Mullhaupt, where the fixed point is not reached for all but one initial condition [Ghi85]. This breakdown is due to the non-ideal behaviors in the experimental realization of autonomous Boolean networks, such as finite rise times and jitter (see also Sec. 3.2.3), because it does not show up in the ideal mathematical model of delay differential equations.

Figure 14(b) shows the dynamics within the transient on a nanosecond timescale. The dynamics show a complex non-repeating pattern that is reminiscent of the chaotic dynamics found by Zhang and collaborators [Zha09a]. When the delay of the feedback is changed, the dynamics are affected. They can, for example, show much shorter transients of a few nanoseconds or stable periodic behavior. Because of the fixed point, however, I believe that stable Boolean chaos is not a possible in this specific experimentally-realized system as the dynamics will earlier or later always collapse into the fixed point. I confirmed this behavior of the setup with several delay combinations, where I kept one delay constant, constructed with 8 inverters, and changed the other delay to vary between 2 and 20 inverters (see also Appendix A.1).

The ultimately regular dynamics in this network is caused by a Boolean fixed point that satisfies the Boolean algebra of the network. To prevent the collapse of the dynamics to such a fixed point, I first introduce a guideline for the network topology to remove Boolean fixed points from the network.

A Boolean fixed point in the feedback system corresponds to rows in the lookup table, where all entries have the same value and hence inputs and outputs can be the same. For example, in the XOR Boolean function, the

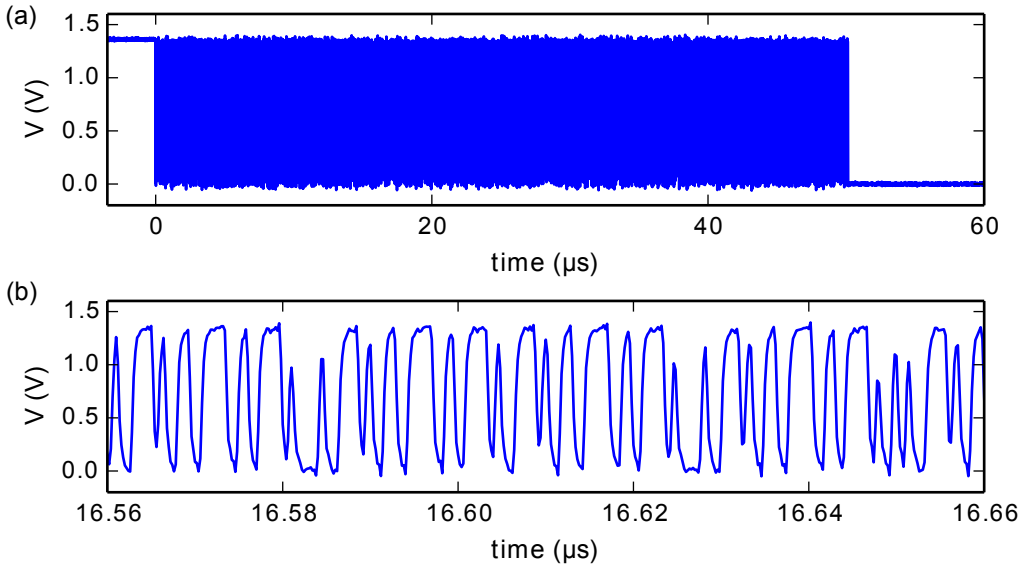


FIGURE 14: (a) Dynamics of an XOR gate with two delayed feedback lines as shown in Fig. 13(a). A clock signal disables the XOR logic gate for about 0.5 s to output a constant Boolean "1;" subsequently, it enables the XOR logic gates also for about 0.5 s. In the figure, the logic gate is enabled at time $t = 0$. (b) Dynamics as in (a) on a short timescale of several nanoseconds. The delay lines are constructed from 2 and 8 inverters as discussed in Appendix A.1.

fixed point corresponds to the first row that is filled exclusively with "0," as marked red in Fig. 13(c). This leads me to the following guideline.

Guideline (iii)—Experimentally realized autonomous Boolean networks should not include a Boolean fixed point, for which all Boolean functions are satisfied simultaneously.

The rows in the look-up table corresponding to the Boolean fixed point in the delayed-feedback oscillator should be changed. For general autonomous Boolean networks, all states have to be tested to make sure that no Boolean fixed point exists, which is also referred to as frustration [Fog84].

I modify the logic gate in the feedback system, as shown in Fig. 13(b). In agreement with guideline (iii), I require that the first and last row in the lookup table is "1" and "0", respectively, and, as a weak form of guideline (i), I require an equal number of "0"s and "1"s, which leads to higher Boolean sensitivity in synchronous Boolean networks. Then, only two Boolean functions fulfill guideline (i), (ii) and the weak form of guideline (iii), whose lookup tables are shown in Fig. 13(d) and (e). These, however, correspond to Boolean functions that are independent of one input, so that they are equivalent to a circuit with a single feedback line, which I discuss in Ch. 6 as a ring oscillator. I find that these autonomous Boolean networks lead to oscillations rather than Boolean chaos.

The remaining 2-input Boolean functions that fulfill guideline (iii) have a large bias, where either "0" or "1" appear three times more in the lookup

(a)

input			output
A	B	C	¬A⊕B⊕C
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

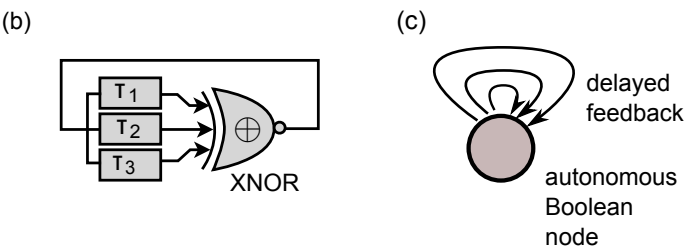


FIGURE 15: (a) Look-up table of a 3-input XNOR Boolean function. (b) Setup of the delayed-feedback XNOR oscillator composed of a 3-input XNOR logic gate with three time-delayed feedback lines. (c) Topology of the autonomous Boolean network underlying the delayed-feedback XNOR oscillator.

table. These are unlikely to show chaos, but they cannot be fully ruled out from theoretical analysis with Boolean network models that do not include all aspect of the experiment.

Because the feedback circuit with two delay lines and a 2-input logic gate is unlikely to show Boolean chaos, I increase the complexity of the system by allowing for 3-input Boolean functions and adding another time-delayed feedback line to the system. I find that the 3-input XNOR Boolean function is the only 3-input Boolean function that fulfills guideline (i) and (iii) as it has no Boolean fixed point in contrast to the 3-input XOR function. The 3-input XNOR Boolean function with a look-up table as in Fig. 15(a) is a generalization of the XOR and XNOR Boolean functions to more than two inputs and corresponds to the least-significant bit of an addition operation and the inversion of it or the parity and inverted parity operation on the Boolean input states, respectively. The complexity of the dynamics of the resulting autonomous Boolean network also depends on guideline (ii) that the delays are important and might need to be adjusted.

4.3.3 SETUP OF THE DELAYED-FEEDBACK XNOR OSCILLATOR

Figure 15(b) shows the schematic setup of the autonomous Boolean network termed *delayed-feedback XNOR oscillator*. Three delayed feedback lines connect the output of the XNOR logic gate to its three inputs. The network evolves autonomously without clocks or external inputs. The network topology is similar to the Boolean network proposed by Ghil and Mullhaupt with Eq. (4) in Sec. 2.3.2 with $\delta = 3$, only differing in inverted entries of the look-up table.

I realize delay lines by cascading n_i inverter gates, where in total the signal gets delayed by n_i times the gate propagation delay of an inverter logic gate $\tau_{\mathrm{LG}} = 0.28 \pm 0.01\,\mathrm{ns}$, corresponding to a total delay of

$$\tau_i = n_i \tau_{\mathrm{LG}}, \ (i = 1, 2, 3). \tag{14}$$

i	n_i	τ_i
1	18	5.04 ± 0.18
2	6	1.68 ± 0.06
3	2	0.56 ± 0.02

TABLE 2: Numerical values used for the three delay lines in the delayed feedback XNOR oscillator using Eq. (14).

This construction of delay lines and the measurement of the gate propagation time is explained in detail in the Appendices A and A.2. This construction of delay lines is also used for other setups in this thesis.

I can vary the delay of the feedback lines with parameters n_i. Here, I use parameters as shown in table 2, which lead to a total number of logic gates included in the oscillator of $n_1 + n_2 + n_3 + 1 = 27$. The delay lines are chosen so that complex dynamics emerge [corresponding to guideline (iii)]; different parameter choices can lead to regular behavior as discussed in the next section. I show the corresponding hardware description in Appendix B.2.

With 27 logic gates, the delayed-feedback XNOR oscillator has a considerably larger logic gate count than the oscillator by Zhang and collaborators that uses three logic gates and six time delays [Zha09a]. The oscillator presented here is hence not preferable for applications, such as random number generation, but it provides a fundamentally simple topology considering that most of the logic gates are used for delay lines. From a fundamental point of view, this network is very simple as it consists of a single autonomous Boolean node with three delayed feedback links as shown in Fig. 15(c).

4.4 DYNAMICS OF THE DELAYED-FEEDBACK XNOR OSCILLATOR

In this section, I investigate the dynamics of the delayed-feedback XNOR oscillator in both experiment and simulation. For the numerical simulation, I develop a model based on piecewise-linear Boolean network models with time delays.

4.4.1 DYNAMICS MEASURED FROM THE EXPERIMENT

When implemented on the FPGA, I observe that the delayed-feedback XNOR oscillator displays complex and non-repeating dynamics as shown in Fig. 16(a). The dynamics alternate irregularly between the Boolean high and low voltage of $V_H = 1.3\,V$ and $V_L = 0\,V$, respectively. Fast but finite-

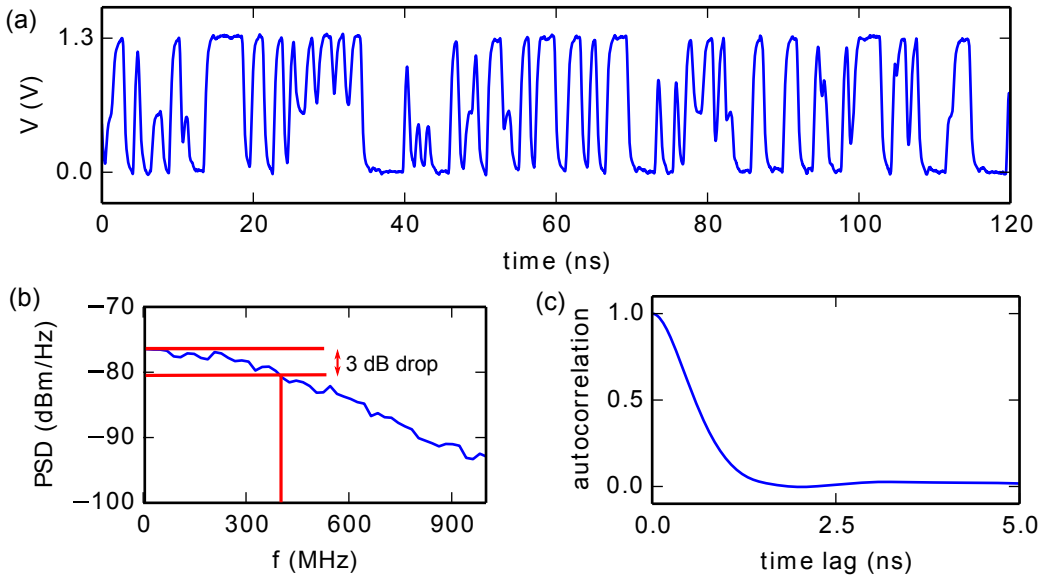


FIGURE 16: (a) Dynamics of the delayed-feedback XNOR oscillator implemented on the FPGA and measured after an input output logic gate (see Sec. 3.3.2). (b), (c) Power spectrum and autocorrelation of (a). Both are calculated with an acquisition of length $T = 13\,\mu\text{s}$ and sampling time $dt = 25\,\text{ps}$; the power spectrum is averaged over 20 MHz to remove noise. The delayed-feedback XNOR oscillator is realized with parameters $n_1 = 18$, $n_2 = 6$, and $n_3 = 2$. Here and in the following, when not stated differently, I realize the experiments on the FPGA Altera Cyclone IV with the model number EP4CE115F29C7N. The dynamics are measured with an oscilloscope of the device family DSO9 with 8 GHz analog bandwidth.

time transitions connect the Boolean levels and also lead to short pulses and dips that can terminate in-between Boolean levels. These effects are caused by the non-ideal effects of the logic gates as discussed in Sec. 3.2.3.

4.4.1.1 *Power Spectrum*

Figure 16(b) shows the power spectrum of the dynamics, which is the power spectral density (PSD) as a function of the frequency, calculated from the temporal evolution. The PSD in dBm measures the power of the waveform in a certain frequency window Δf, typically 1 Hz, according to

$$\text{PSD}(f, f+\Delta f) = 10\log_{10}\left[P(f, f+\Delta f)/(1\,\text{mW})\right], \tag{15}$$

where $P(f, f+\Delta f)$ is the power in the spectral range $[f, f+\Delta f]$ [Opp97]. It can be calculated with the Fourier transform of the waveform, according to

$$P(f, f+\Delta f) = A_f^2 \Delta f / R, \tag{16}$$

where A_f is the Fourier amplitude of a sine wave corresponding to the frequency f and $R = 50\,\Omega$ is the termination of the oscilloscope.

I calculate the Fourier amplitude with the Fast Fourier Transform algorithm ($A_f^2 \widetilde{\Delta f} \mathrel{\widehat{=}} |\mathrm{FFT}\,[V]\,(f)/N|^2$) with a frequency step $\widetilde{\Delta f} = 1/T$, where T is the total acquisition time. To rescale the power to correspond to $\Delta f = 1\,\mathrm{Hz}$, I calculate

$$P(f, f+\Delta f) = \frac{\Delta f}{\widetilde{\Delta f}} \frac{|\mathrm{FFT}\,[V]\,(f)/N|^2}{R}. \tag{17}$$

The important measure that can be read from the resulting power spectrum in Fig. 16(b) is the width of the spectrum characterized by the commonly used 3 dB dropoff, corresponding to half the power. Beyond the 3 dB dropoff, frequency modes have less than 50% of the maximum power and hence contribute significantly less to the dynamics. With this measure, I find that the system has a wide spectrum of about 400 MHz.

The power spectral density $P(f, f+\Delta f)$ has a low amplitude with a maximum value of $-76\,\mathrm{dBm/Hz}$. The total power, however, has the expected numerical value of about $(V_H/2)^2/R \approx 7\,\mathrm{mW}$ with the high Boolean voltage $V_H = 1.3\,\mathrm{V}$, which can be calculated by integrating the graph over a wide range of about 400 MHz with the PSD given in frequency units per Hz. The absolute power is of lower significance for the dynamics because it changes when the output logic gates are configured to different Boolean voltage levels.

4.4.1.2 *Autocorrelation*

The autocorrelation function is defined for a real-valued, infinite long signal f as

$$R_f(\tau) = \frac{1}{A} \int_{-\infty}^{\infty} f(t) f(t-\tau)\,\mathrm{d}t. \tag{18}$$

Here, the constant A is chosen as a normalization, so that $R_f(\tau = 0) = 1$. For a finite-time, discrete signal, such as the one measured with the oscilloscope, the autocorrelation function is expressed as a sum and windowing procedures have to be considered depending on the maximum shift τ. $R_f(\cdot)$ can be calculated efficiently using the Wiener-Khinchin theorem and the Fast Fourier Transform. The autocorrelation is a measure for the similarity of the signal with a shifted copy of itself. In a delayed chaotic system, the autocorrelation is an important tool that can uncover time delay signatures of the system at certain time shifts, which can diminish the applicability of chaos for random number generation, for example [Rono7].

Figure 16(c) shows the autocorrelation of the dynamics in the setup, which falls off to close to zero within a fast de-correlation time of $\sim 1\,\mathrm{ns}$ and then stays close to zero. Therefore, the system does not show time delay signatures, which makes the dynamics potentially useful for applications, such as random number generation. The fast drop of the autocorrelation function is a sign of strong chaos.

4.4.1.3 *Chaos*

Strong evidence exists that the dynamics is chaotic because I generate the signal with a similar electronically realized autonomous Boolean network of simple topology like the network by Zhang and collaborators, who proved the existence of chaos [Zha09a]. Furthermore, the waveform of the dynamics of the XNOR oscillator in Fig. 16(a) is similar to the waveform of the oscillator by Zhang and collaborators in Fig. 5(a) [Zha09a]; both waveforms show an irregular spacing of transitions, for example. Other evidence is given by the fast-decaying and quasi-unstructured autocorrelation function. I tried to measure the Lyapunov exponent with the method of the Boolean distance similar to Zhang and collaborators [Zha09a]using 26 μs of data, corresponding to approximately 15,000 transitions. Even with this amount of data, I was not able to identify Boolean neighbors required for the calculation. The failure to apply this method for the delayed-feedback XNOR oscillator could be due to a higher dimensionality of the dynamics. Future work could explore different approaches to measuring the Lyapunov exponent of an experimental Boolean network or consider applying the Boolean distance with much more data than 26 μs at 20 GSa/s sampling rate.

4.4.2 BOOLEAN NETWORK MODEL FOR THE CHAOTIC DYNAMICS

In this section, I motivate the autonomous Boolean network model by Glass and collaborators [Gla98, Mes96] that is introduced in Sec. 2.3.3; I take an experimental point of view and extend the model with time delays. This derivation is also the foundation for the mathematical models in the following chapters, where I use similar descriptions. The model is simulated numerically to compare the resulting waveforms to experimental observations.

The dynamic behavior of electronic logic gates can in principle be modeled from first principles using SPICE models [Vla94]. Here, however, I use simple piecewise-linear switching models for two reasons. First, a single programmable look-up table block (logic gate) on the FPGA includes at least 30 transistors [Max09], which would each be modeled with several differential equations depending on the model, leading to very slow numerical simulations for larger circuits. Second, the required parameters are not provided by the manufacturer to protect against reverse engineering; hence, the SPICE models are unknown.

I describe the electronic logic gates on the FPGA with an equivalent circuit shown in Fig. 17, which separates its operation into ideal Boolean switching and subsequent frequency filtering. I assume a simple first-order low-pass filter for the logic gate because the actual filter characteristics is un-

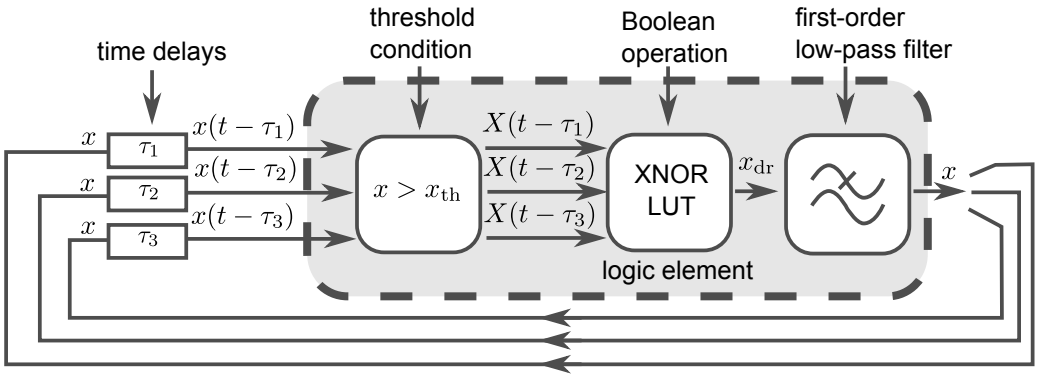


FIGURE 17: Equivalent circuit for the model. The output state x is delayed by three different time delays τ_1, τ_2, and τ_3. Within the equivalent logic gate, the states are subject to a threshold condition, an ideal Boolean operation, and a first-order low-pass filter.

known and cannot be measured. Instead only signals from the output gates can be measured, which may have different filter characteristics. Therefore, a simple model is preferred.

A frequency filter can be expressed mathematically with a transfer function

$$H(\omega) = \frac{\mathcal{F}\{V_{\mathrm{out}}(t)\}(\omega)}{\mathcal{F}\{V_{\mathrm{in}}(t)\}(\omega)} \tag{19}$$

that relates the Fourier spectrum $\mathcal{F}\{\cdot\}(\omega)$ of the output signal V_{out} to the Fourier spectrum of the input signal V_{in} for frequencies ω. Specifically, for a first-order low-pass filter, the transfer function reads

$$H^{LP}(\omega) = \frac{\gamma_0}{1 + i\omega\tau_{LP}}, \tag{20}$$

where γ_0 is the attenuation of the low-pass filter and τ_{LP} is the filter constant given by the inverse of the cutoff frequency

$$\tau_{LP} = \frac{1}{\omega} = \frac{1}{2\pi f_+}. \tag{21}$$

I assume that the filter is loss-less ($\gamma_0 = 1$), leading to

$$(1 + i\omega\tau_{LP})\,\mathcal{F}\{V_{\mathrm{out}}(t)\}(\omega) = \mathcal{F}\{V_{\mathrm{in}}(t)\}(\omega). \tag{22}$$

Applying the property

$$i\omega\mathcal{F}\{f(t)\}(\omega) = \mathcal{F}\left\{\frac{\mathrm{d}}{\mathrm{d}t}f(t)\right\}(\omega) \tag{23}$$

and the linearity of the Fourier transform, I obtain

$$\mathcal{F}\left\{\left(1 + \tau_{LP}\frac{\mathrm{d}}{\mathrm{d}t}\right)V_{\mathrm{out}}\right\}(\omega) = \mathcal{F}\{V_{\mathrm{in}}\}(\omega), \tag{24}$$

which becomes when comparing the arguments of the Fourier transform

$$\left(1 + \tau_{LP}\frac{\mathrm{d}}{\mathrm{d}t}\right) V_{\mathrm{out}} = V_{\mathrm{in}}(t). \tag{25}$$

This is equivalent to the differential equation

$$\tau_{LP}\dot{x}(t) = -x(t) + x_{\mathrm{dr}}(t), \tag{26}$$

where I have replaced the output voltage $V_{\mathrm{out}}(t)$ by a dimensionless variable $x(t)$, and the input voltage $V_{\mathrm{in}}(t)$ by a driving term $x_{\mathrm{dr}}(t)$ and abbreviate the temporal derivative with a dot. The dimensionless variable x can be scaled to correspond to the experimental variable V by multiplying $V_{\mathrm{H}} = 1.3\,\mathrm{V}$, depending on the used output gate.

For the setup of the delayed-feedback XNOR oscillator, the low-pass filter is driven by a 3-input inverted XOR Boolean function XNOR: $\{0,1\} \times \{0,1\} \times \{0,1\} \to \{0,1\}$ as shown in Fig. 17. I model the three delayed feedback lines (τ_1, τ_2, τ_3) with mathematical time delays. The driving signal of the lowpass filter is then

$$x_{\mathrm{dr}}(t) = \mathrm{XNOR}\left[X(t-\tau_1), X(t-\tau_2), X(t-\tau_3)\right], \tag{27}$$

where the Boolean variable $X(t)$ is calculated from $x(t)$ with the threshold condition

$$X(t) = \begin{cases} 1, & \text{if } x(t) > x_{\mathrm{th}}, \\ 0, & \text{if } x(t) \leq x_{\mathrm{th}}, \end{cases} \tag{28}$$

with the low and high Boolean values 0 and 1 and a symmetric threshold $x_{\mathrm{th}} = 0.5$. Combining Eq. (26) and (27), I arrive at the delay differential equation

$$\tau_{LP}\dot{x}(t) = -x(t) + \mathrm{XNOR}\left[X(t-\tau_1), X(t-\tau_2), X(t-\tau_3)\right], \tag{29}$$

which I use in the following to model the dynamics of the delayed-feedback XNOR oscillator. This equation is an extension with time-delayed feedback terms of the piecewise-linear differential equations introduced by Glass and collaborators for autonomous Boolean networks [Mes96, Gla98].

I integrate Eq. (29) numerically with a linear multistep method as discuss in Appendix B.8. I choose the three time delays so that complex dynamics emerge. As the low-pass filter constant, I use $\tau_{\mathrm{LP}} = 0.4\,\mathrm{ns}$, which corresponds to the 3 dB dropoff of $f_+ = 400\,\mathrm{MHz}$ measured from the experiment [see Eq. (21), Sec. 4.4.1.1 and Fig. 16(b)]. In addition, this value also corresponds to the propagation delay of a logic gate $\tau_{\mathrm{LG}} = \tau_{\mathrm{LP}}\ln(2)$ [see Eq. (8)] with $\tau_{\mathrm{LG}} = 0.28 \pm 0.01\,\mathrm{ns}$ measured as discussed in the Appendix A.2.

Figure 18(a) shows the waveform resulting from numerical integration of Eq. (29) for time delays stated in the caption. The waveform displays

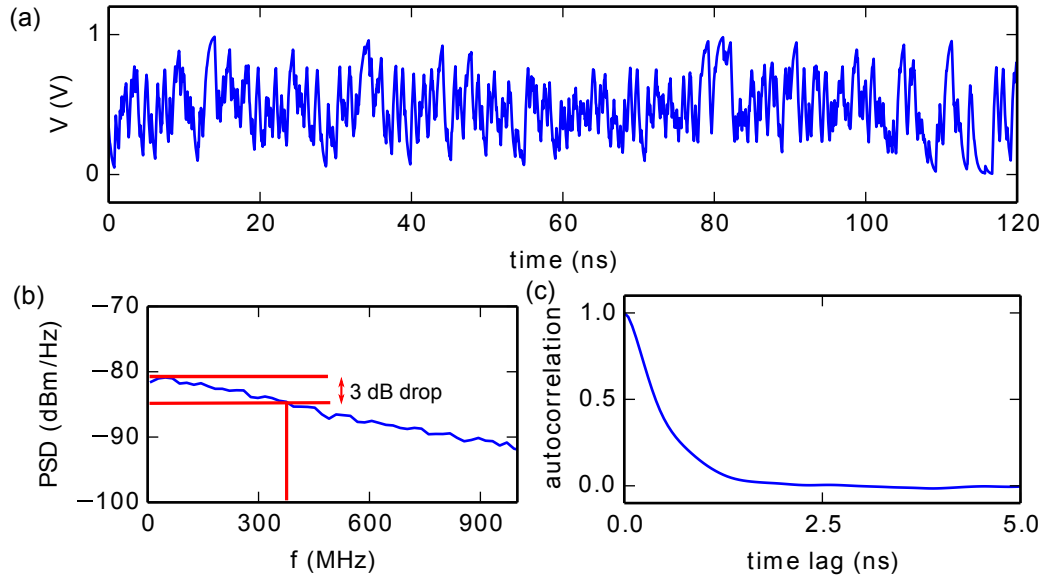


FIGURE 18: (a) Dynamics of the delayed-feedback XNOR oscillator from numerical simulation of Eq. (29). (b), (c) Power spectrum and autocorrelation of (a). The power spectrum is calculated with $T = 26\ \mu s$ and $dt = 50\,\text{ps}$ and averaged over 20 MHz to remove noise. The model parameters are $\tau_{\text{LP}} = 0.4\,\text{ns}$, $\tau_1 = 10.119\,\text{ns}$, $\tau_2 = 1.732\,\text{ns}$, and $\tau_3 = 0.297\,\text{ns}$.

complex behavior and irregular spacing of transitions. The number of transitions per nanosecond, however, is larger than for the experimental waveform in Fig. 16(a). Moreover, the numerical integration leads to chaotic dynamics only for narrow ranges of the feedback delays, whereas the experiment shows chaotic dynamics consistently when the feedback delays τ_1, τ_2, and τ_3 are above a certain value as discussed in the next section.

These differences may be due to several non-ideal behaviors of the experimental system. For example, as discussed in Sec. 3.3.1, the voltages measured from the FPGA are routed through output buffer gates before they are measured with the oscilloscope. These can reduce the number of transitions, especially when they have stronger filtering than the programmable logic gates or a different threshold voltage. The disagreement may also be due to history- and state-dependency of the feedback delays in the experiment, heterogeneity, and noise, which are not captured by the simplified model (see also Sec. 3.2.3). Lastly, the disagreement may also be caused by a frequency characteristic of the logic gates that differs from a first-order low-pass filter characteristic used to derive the model.

I also show the power spectrum calculated from the numerical data in Fig. 18(b). It has a 3 dB dropoff corresponding to the cutoff frequency $f_+ = 400\,\text{MHz}$ ($\tau_{\text{LP}} = 0.4\,\text{ns}$. The power spectrum from numerical integration is similar to the experimental measurement in Fig. 16(b). I also show the autocorrelation of the numerical data in Fig. 18(c). It drops quickly to values

close to zero after a de-correlation time of 1 ns, which is similar to the experimental data in Fig. 16(c).

To summarize the comparison between experiment and simulation, I find that they have similar power spectra and autocorrelations, but differ in the waveforms as they show a different number of transitions per unit time. I find, however, that the model is well suited to simulate regular dynamics, such as oscillations in autonomous Boolean networks as discussed in later chapters.

4.4.3 TRANSITION TO CHAOS

While I have shown in the previous section that the delayed-feedback XNOR oscillator with the setup shown in Fig. 15(b) displays chaos for one particular choice of feedback time delays, I discuss in this section the dynamics of the experimental system for various different feedback delays.

The three time-delayed feedback lines are constructed from chains of n_i inverter gates, which result in time delays according to Eq. (14). The oscillator shows chaos for parameters $n_1 = 18$, $n_2 = 6$, and $n_3 = 2$, as discussed in the previous section. I explore different dynamics of the oscillator by changing n_1 from $n_1 = 0$ to $n_1 = 14$ in steps of 2 while keeping $n_2 = 6$ and $n_3 = 2$ fixed. Figure 19 shows the resulting dynamics as waveforms of the output voltage of the delayed-feedback XNOR oscillator over time for different values of n_1.

The first tested value of the feedback delay of $n_1 = 0$ corresponds to a feedback line that does not include inverter gates, but rather is built only with on-chip interconnect leading to a delay that is negligible compared to the delay of an XNOR logic gate, so that the total delay corresponds approximately the delay of the XNOR logic gate (the delay of a 3-input XNOR gate is $\tau_{\mathrm{XNOR}} = 0.38 \pm 0.02$ ns, as measured in Appendix A.4). The resulting output voltage of the oscillator is at a constant value of $V \approx 0.7$ V, but also shows small voltage fluctuations on the order of 0.01 V due to electronic amplitude noise. This value is close to $V = V_{\mathrm{th}} = (V_{\mathrm{H}} - V_{\mathrm{L}})/2$, which is the threshold voltage of the logic gate. Therefore, I observe that short delayed feedback can stabilize the threshold value. Depending on the placing of this setup on the electronic chip, I also sometimes observe constant Boolean levels of high or low Boolean voltage, which is likely due to a difference between the threshold values of the XNOR logic gate and the output logic gates.

In the waveform in Fig. 19(b), I have increased the delay substantially from the delay of one logic gate to $\tau = 2\tau_{\mathrm{LG}} + \tau_{\mathrm{XNOR}} = 0.94 \pm 0.02$ ns by including $n_1 = 2$ cascaded inverter gates in the feedback line. The waveform shows a periodic evolution of the voltage over time, where the same pattern repeats with a period of $T = 4.3 \pm 0.1$ ns. This period is related to the largest delay in the system that consists of six inverter gates,

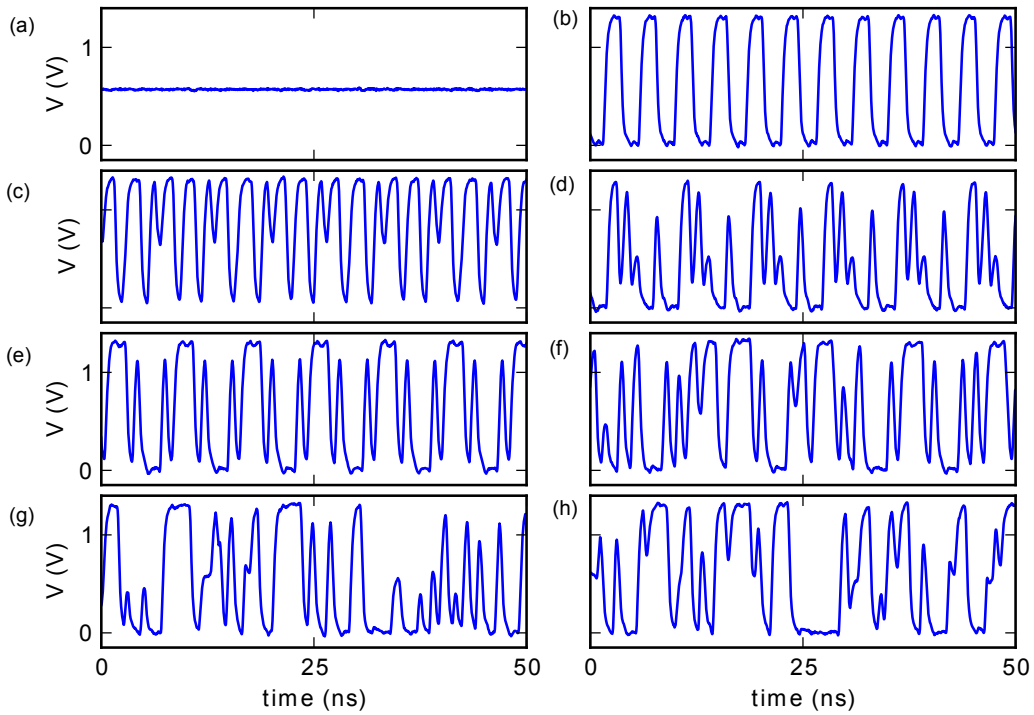


FIGURE 19: Dynamics of the delayed-feedback XNOR oscillator for different values of feedback delays measured from the experiment. From (a) to (h), n_1 is increased from zero to 14 in steps of two leading to different delays according to Eq. (14). $n_2 = 6$, $n_3 = 2$ are fixed corresponding to delay lines of length $\tau_{n_2} = (1.7 \pm 0.1)$ ns, $\tau_{n_3} = (0.56 \pm 0.02)$ ns. For the total feedback delay, the gate delay of the XNOR logic gate ($\tau_{\text{XNOR}} = 0.38$) has to be added. FPGA and oscilloscope as in Fig. 16.

resulting in $\tau_3 = 6\tau_{\text{LG}} + \tau_{\text{XNOR}} = 2.06 \pm 0.08$ ns (using $\tau_{\text{LG}} = 0.28 \pm 0.01$ ns and $\tau_{\text{XNOR}} = 0.38 \pm 0.02$ ns, as measured in Appendix A.4). As explained in Ch. 6, the period in the oscillatory regime for such feedback systems is given by twice the delay, *i.e.*, $T = 2\tau_3$, which corresponds to the measured value.

For values from $n_1 = 4$ to $n_1 = 8$ in Fig. 19(c)-(e), the resulting waveforms show periodic oscillations, but with more complex patterns than in Fig. 19(b). For example, in Fig. 19(c), the voltage shows a pattern of two minima and two maxima and one dip to a level close to the threshold voltage and, in Fig. 19(d), the periodic pattern shows four maxima with different amplitudes and different spacing. An extension of this thesis could be to study the formation of these complex patterns as a function of the time delays.

For values of n_1 above ten in Fig. 19(f)-(h), the voltage shows a complex temporal evolutions without strict periodicity, which I have identified with Boolean chaos in Sec. 4.4.1.3. For $n_1 = 10$ and $n_1 = 12$, the waveform includes recurring patterns within the chaotic dynamics (not shown in the figure), which is not the case for $n_1 = 14$ and for several parameter values

$n_1 \geq 14$ that I have tested. Therefore, I conjecture that the dynamics is chaotic consistently for $n_1 \geq 14$. The dependence of the dynamics on the time delays in the system is in agreement with guideline (ii).

The dynamics can be affected by placing the oscillator on a different area of the chip, because the logic gates are heterogeneous (see Section 3.2.3). This can lead to different wave patterns in the periodic dynamics and a different threshold value n_k at which the network displays Boolean chaos.

I characterize the complexity of the dynamics for different network sizes in the next chapter, which also includes extensive numerical simulation and comparison between model and experiment. For this reason, I have not included numerical simulation in this section.

4.5 CONCLUSION

In this chapter, I have studied simple autonomous Boolean networks that display Boolean chaos. The networks are designed with the help of three guidelines derived from Boolean network models that are likely to increase the complexity in experimentally realized autonomous Boolean networks. These guidelines, however, do not guarantee complex dynamics because of differences between Boolean network models and experimental realizations. Specifically, a network that is predicted to show stable complex dynamics with an autonomous Boolean network model relaxes to a fixed point after a complex transient in the experiment. This breakdown of the theory is one important result of this chapter. I have also identified a network topology with stable complex dynamics in the experiment—termed the delayed-feedback XNOR oscillator—that is a structurally simple network of only one node with three delayed feedback lines. The delayed-feedback XNOR oscillator shows a dynamic transition from regular dynamics to Boolean chaos when the feedback delays are increased.

Boolean chaos has several applications, such as chaos-based radar and physical random number generation [Soboo, Liuo4]. In the next chapter, I explore the latter striking application.

5 ULTRA-FAST PHYSICAL GENERATION OF RANDOM NUMBERS USING HYBRID BOOLEAN NETWORKS

5.1 ABSTRACT

I discuss in this chapter how chaotic dynamics in autonomous Boolean networks can be used for high-speed physical random number generation. I start this chapter in Sec. 5.2 by introducing to random number generation. In Sec. 5.3, I develop a hybrid Boolean network that consists of both autonomous and synchronous Boolean nodes. In Sec. 5.4, the Boolean network is utilized for random number generation. [1]

The main contribution of this chapter are:

- introducing a network-based approach to random number generation, which allows for post-processing schemes that do not reduce the rate or increase the size of the system;
- realizing a physical random number generator based on a chaotic Boolean system with a compact circuit that is inexpensive and can be integrated with other components as a system on a chip (SoC);
- realizing an ultra-high bit rate of 12.8 GHz.

5.2 INTRODUCTION TO RANDOM NUMBER GENERATION

Random numbers are the backbone of cryptographic protocols used for private communications, such as everyday bank transactions and cloud services, and proof-of-work protocols, which are the foundation of cryptocurencies such as Bitcoin [Asm07, Jun99, Nak08]. They are also essential for Monte-Carlo simulations, which are used in various fields, such as climate or biomedical sciences [Bin10, Met49, Moo97].

1 Results of this chapter are published in Ref. [Ros13a].

5.2.1 APPLICATION OF RANDOM NUMBERS TO PRIVATE COMMUNICATION

A common situation in cryptography is when two parties, called Alice and Bob, communicate, while an eavesdropper, called Eve, is able to read all exchanged messages. Eavesdropping is easily possible for wireless communication and communication via the internet. Therefore, private communication requires message encryption, which, in its most reliable form, is achieved with symmetric-key encryption as visualized in Fig. 20(a). Alice and Bob share a secret key via a secure line, for example by meeting each other or via quantum key distribution [Ved06]. This key is then used by Alice to encrypt the message, where it is converted to a ciphertext via a complex invertible algorithm in combination with the key, and used by Bob to decrypt the message, where the ciphertext is converted back to the message via the inverted algorithm in combination with the key. Eve can read the ciphertext, but deciphering the text is computationally prohibitive. However, the key has to be changed on a regular basis so that Eve has limited time for finding it. A historical example of a simple invertible encryption algorithm supposedly used by Julius Caesar is to shift the letters in the alphabet by a certain number given by the secret key, which is easy to encrypt and decrypt when the key is known but harder to decipher if the key is unknown. Today, encryption algorithms are, of course, much more complex [Sin11].

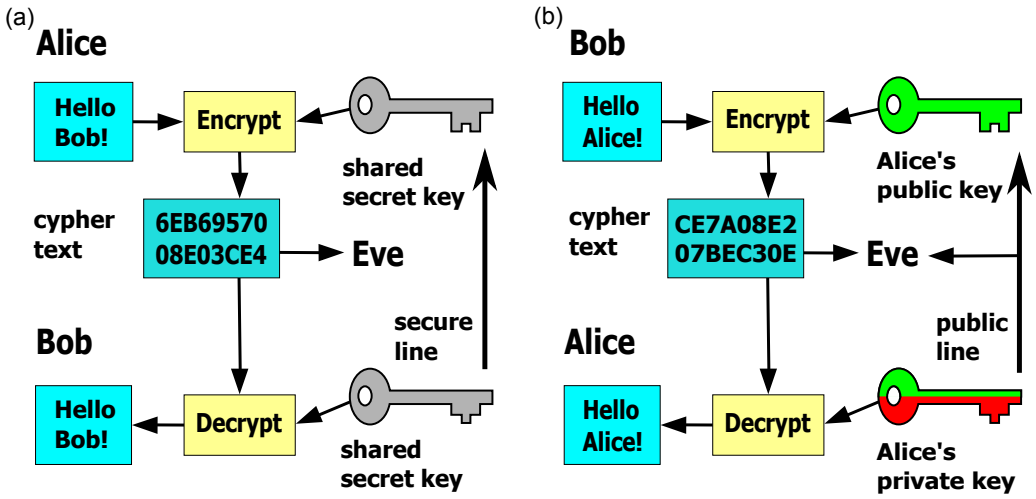


FIGURE 20: (a) Schematic of symmetric-key encryption with a shared key via a secure line. (b) Schematic of public key exchange via a public line. The picture is a modified version of the public domain content http://commons.wikimedia.org/wiki/File:Public_key_encryption.svg.

In modern private communication, the key is exchanged publicly, as visualized in Fig. 20(b). Alice randomly generates a private key and infers

from it a public key, where the private key is hard to infer from the public key (the inverse operation). Bob uses an encryption mechanism to encrypt a message with the public key, such that Eve cannot decipher the message in reasonable time without the private key. Alice then decrypts Bob's message using a combination of the private and public key. The message includes a new secret key that is used to communicate via symmetric-key encryption. The common algorithm for public key exchange, known as RSA[2] encryption, relies on multiplying two prime numbers, which is fundamentally hard to invert (known as prime number factorization) scaling in computation time exponentially when the number of digits increases [Kato7, Robo3, Ade83].

Both the symmetric-key encryption and RSA encryption rely on the unpredictability of encryption keys. On the other hand, predictable keys or unsecure storage of keys can put these encryption schemes at jeopardy. For example, a recent major security breach in RSA encryption was discovered on April 2014 with the Heartbleed Bug, which allowed hackers for two years to spy out stored private keys from the server and hence decrypt the communication, but it is unknown if hackers were aware of this possibility for longer than a few hours after this possibility was officially disclosed [Per14]. The impact of this security breach could have been lower if keys would be generated just before they are used instead of saving them in advance on the server. Encryption keys are generated from random numbers, which are required to be as random as possible for highest cryptographic security. Two ways of generating random numbers are pseudorandom number generated using mathematical algorithms and physical random number generation using a physical entropy source.

5.2.2 PSEUDORANDOM AND PHYSICAL RANDOM NUMBER GENERATION

Pseudorandom number generators are mathematical algorithms that can be used to calculate a sequence of numbers with properties of true random numbers. The algorithm starts the calculation from an initial value called a seed, which can originate from a physical entropy source, such as user input. Because the sequence of random numbers can be reproduced when the seed and algorithm is known, pseudorandom numbers are not truly random, so that the entropy of pseudorandom numbers is restricted to the entropy of the seed [Jun99, Ruko1]. This means that poorly chosen seeds can lead to failure of pseudorandom-based protocols. For example, a prominent security breach existed in the web browser Netscape between 1994 and 1995, where the seeds were chosen as a combination of the current time and process IDs [Gol96].

2 The letters RSA are the initials of the inventors Ron Rivest, Adi Shamir, and Leonard Adleman.

An example for a mathematical algorithm that generates numbers with properties of randomness is a linear feedback shift register (LFSR), as shown in Fig. 21(a) with the example of a Fibonacci LFSR. In the illustration, the shift register stores 16 values, where, in every iteration, the leftmost value is replaced by the XOR Boolean function of certain bits in the shift register and the rightmost bit is discarded. The leftmost bit in the shift register can be used as the generated random bit. This algorithm can be initialized by defining the bits in the shift register, which is the seed of the pseudorandom number generator. Interestingly, the Fibonacci LFSR is a synchronous Boolean network similar to Kauffman networks introduced in Section 2.3.1. As such, it has a finite number of states and will eventually enter a repeating cycle, which can, however, be very long with a maximum of $2^{16} - 1 = 65535$ states for a 16-bit LFSR. Figure 21(b) shows a similar setup called Galois LFSR, which can be implemented more efficiently in software than a Fibonacci LFSR. However, both LFSRs can be implemented very efficiently in hardware to generate random numbers at a high rate.

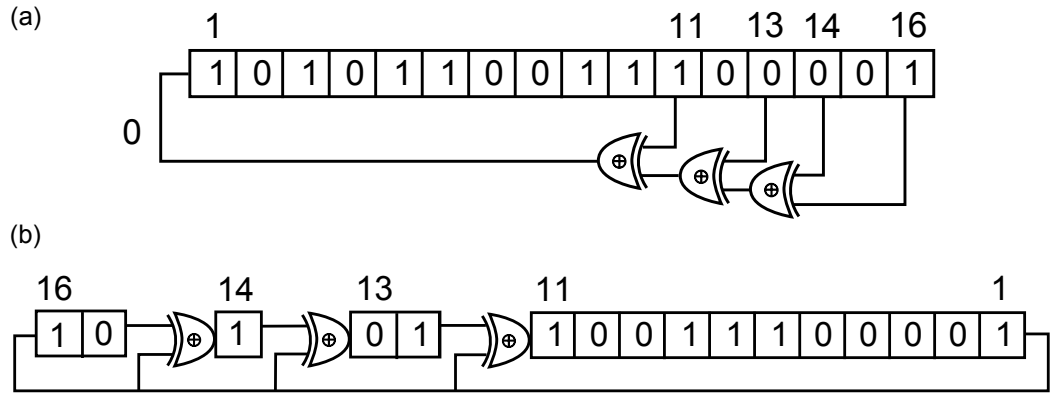


FIGURE 21: (a) A 16-bit Fibonacci linear feedback shift register (LFSR). The tab numbers above the shift register characterize the specific Fibonacci LFSR as the bit positions of which the XOR operation is executed. (b) A 16-bit Galois LFSR. The pictures are modified versions of the public domain content http://en.wikipedia.org/wiki/File:LFSR-F16.gif and http://en.wikipedia.org/wiki/File:LFSR-G16.gif.

In physical random number generation, in contrast, a measurement from a physical entropy source provides the random numbers. For example, the voltage of electronic noise over time can be measured to generate physical random numbers, such as implemented in the Intel random number generator [Jun99], or quantum effects can be exploited [Way10]. Other examples are to throw dice or take random user input via a pointing device, such as a touch screen.

While physical random numbers are regarded secure, their generation is usually slow and expensive. An attractive secure alternative to pure physical random number generators is to use them to repeatedly generate seeds

for pseudorandom number generators. One similar scheme, as already discussed in Sec. 4.2.1, is to use chaotic systems that include a small-amplitude entropy source, *i.e.*, deterministic chaos under constant perturbation of microscopic noise. The microscopic noise provides the physical randomness and the deterministic chaos provides amplification of the physical randomness similar to a pseudorandom number generator because deterministic chaos follows mathematical equations [Har12]. The chaotic dynamics amplifies the entropy to a rate given by the Lyapunov exponent and can also make its statistical properties more desirable, known as the mixing property of chaos [Har12, Rei09].

An important early study on chaos-based random number generation is described in Ref. [Sto01], where a CMOS circuit realizes a one-dimensional chaotic, piecewise linear map leading to a rate of 1 Mbit/s. But this circuit is an exact physical implementation of a mathematical map and is hence not sensitive to electronic noise and implements a pseudorandom number generator. High real-time random bitrates of up to 300 Gbit/s of physical randomness could been realized with continuous-time chaotic photonic devices because of their large bandwidth [Uch08, Rei09, Kan10, Li11, Har11]. Theoretical studies of the Lang-Kobayashi equations, which is a standard model for laser dynamics, could prove that the randomness in these photonic systems originates from physical entropy [Har12]. In Ref [Ros13a], my collaborators and I have shown that very high data rates can also be achieved with lower-speed chaotic electronic devices by using parallelization, which is the work discussed in this chapter. Such parallelization is especially easy to implement using electronic microchips. Since my publication, another group has also reported on a high-speed random number generator based on an electronic tunnel diodes in Ref. [Li13], claiming high bitrates also through parallelization. In contrast to my work, however, they require computationally-intensive post-processing.

5.2.3 DESIRED STATISTICAL PROPERTIES OF RANDOM NUMBERS AND POST-PROCESSING

The applications mentioned in the beginning of this chapter require certain statistical properties for the random numbers. Good statistical properties are, for example, high entropy and low bias, where entropy measures the unpredictability of the random numbers and bias measures its deviation from a uniform distribution [Cov91]. These statistical properties can be assessed with statistical randomness test suites that include multiple tests, such as the statistical test suite by the National Institute of Standards and Technology (NIST) for random number generators for cryptographic protocols [Ruk01].

Statistical tests determine if a sequence of numbers fails certain criteria for true randomness using a statistical approach of a null hypothesis H_0.

For example, H_0 for testing a sequence of random numbers could be that the random numbers are free of bias or free of correlations. A null hypothesis test can have two outcomes: First, it can result in statistical evidence for the rejection of H_0 with a confidence of, for example, $3\sigma = 99.7\%$; second, it can result in no rejection of the null hypothesis. The second case, however, does not imply $3\sigma = 99.7\%$ confidence that the null hypothesis holds, just as it is wrong to assume that no evidence of bias implies evidence of no bias [Cov91, Tal10]. A setting where the null hypothesis is not rejected, but the random number generator has flaws, corresponds to a type-II error, which statistical randomness tests try to minimize [Ruk01]. Type-II errors result from a finite sample size used for the testing and from a finite number of tests; for example, the NIST test searches for repetition of patterns only up to a certain pattern length. Because of type-II errors, a flawless random number generator—a so-called 'true random number generator,'—is impossible to identify.

Good statistical properties are usually hard to achieve for physical random number generators because of intrinsic bias and correlations originating from the physical source. Bias and correlations and other statistical flaws can be reduced using post-processing, such as by combining multiple bitstreams from identical uncoupled systems [Uch08] or hashing random bits from the bitstream produced by a single device [Jun99]. These approaches either reduce (divide) the generation rate of random numbers or increase (multiply) the system size by including multiple random number generators.

A frequently used post-processing algorithm is based on the XOR Boolean function. For example, a generalized n-input XOR logic gate can hash n bitstreams resulting in one bitstream with reduced bias. The resulting bias is $\tilde{b} = 2^{n-1}b^n$, where b is the bias of the uncorrelated input bitstreams (see Appendix C.1). Specifically, a bias of $b = 1\%$ can be reduced to a theoretical value of $\tilde{b} = 8 \cdot 10^{-8}$ using a 4-input XOR gate, which corresponds to an undetectable level of bias when analyzing a gigabit of data (typically used in standard statistical randomness tests, such as NIST tests [Ruk01]).

Post-processing can limit the bitrate of physical random number generators substantially, such as when a fast-timescale optical system is hashed with a much slower electronic post-processing unit. For this reason, many studies on ultra-fast random number generation do not include the low rates of post-processing in the quoted bitrate of the random number generator. A case in point is a study by Kanter and collaborators who implement a photonics-based random number generator with a raw data rate of 300 Gbit/s. In post processing, they use a complicated algorithm that takes the 15th derivative of the data, which is not accounted for in the random bitrate [Kan10]. On a 4-core, 2 GHz processor, the rate decreases consequently to about 600 MHz including post-processing, assuming that the calculation of the 15th derivative requires 15 processor operations.

To my knowledge, all approaches to fast random number generation by the nonlinear dynamics community are limited by their acquisition method with high-speed oscilloscopes. These devices can acquire digital data at rates as high as 400 Gbit/s (assuming a 10 bit quantization with a digital bandwidth of 40 GHz), but they can maintain this rate only for a short time, until its fast random access memory is filled, which usually takes only about 10 μs. It is, however, not clear if this rate could also be maintained for constant streaming into a communication interface by altering the device. The use of high-speed oscilloscopes also increases the price of random number generators to as much as 100,000 USD, where the actual physical device can typically be manufactured for less than 10,000 USD. Though the high cost, the high-speed analog-to-digital converters in such oscilloscopes are very useful for random number generation, which has been showcased by Reidler and collaborators, who have realized a photonic-based device with a rate of 12.5 GHz, where the post-processing simply discards several of the most significant bits of the 8-bit quantized acquisition [Rei09].

Uchida and collaborators have developed a physical random number generator based on two copies of a photonic device that are monolithically-integrated. This study is a first step towards adding electronic post-processing to the chip by hashing the two bitstreams with an XOR logic gate. This would allow them to claim a real-time (online) bitrate of 2.08 GHz including post-processing on a compact device [Har11].

5.2.4 UTILIZATION OF AUTONOMOUS BOOLEAN NETWORKS FOR RANDOM NUMBER GENERATION

Autonomous Boolean networks with chaotic dynamics are technological favorable for random number generation because:

- their dynamics evolves on a sub-nanosecond timescale, which is fast compared to other electronic circuits allowing for high random bit rates;
- when realized on microelectronic chips, they can be integrated together with other logic-circuit-based electronic devices such as processors, resulting in a system on chip (SoC);
- they are inexpensive, especially when comparing the number of required logic gates to the requirements for a central processing units (CPUs) (modern CPUs include on the order of a billion transistors, whereas a single 2-input logic gate can be implemented with less than ten transistors [Hor89]).

Autonomous Boolean networks (even if not identified as such) have been used before to generate physical random numbers. For example, methods

have been developed to generate physical random numbers from jitter in periodic autonomous Boolean networks [Sun07, Wol09]. In one approach, random numbers are generated using multiple unidirectional ring oscillators, which are oscillatory systems that I describe in Ch. 6, built from $\sim$ 1300 autonomous inverter gates together with multiple clocked XOR logic gates to remove bias. However, this standard approach requires a large number ($\sim$170) of logic gates to generate random numbers at a rate of 100 Mbit/s and, in addition, evidence exists that this approach has flaws [Dic07]. Dichtl and Golić have developed a chaos-based approach to random number generators with autonomous Boolean networks [Dic07]. The topology of their autonomous Boolean networks are based on Fibonacci and Galois linear feedback shift registers as introduced in Sec. 5.2.2, but the networks are operated without clocks. Multiple copies of such autonomous Boolean networks can be combined with a clocked XOR operation to reduce bias and increase entropy. These setups have been reported to generate uncorrelated random numbers at 20 Mbit/s, but neither the chaotic dynamics nor the quality of the resulting random bit sequence have been characterized [Dic07].

In this chapter, I generate random numbers with an autonomous Boolean network described in a US patent [Bae08] with a ring topology and 3-input XOR and XNOR logic functions as discussed in the next section. This network, which I call an *XOR ring network*, is similar to the delayed-feedback XNOR oscillator in Ch. 4, but has three substantial advantages:

1. The XOR ring network uses of many 4-input logic gates on the FPGA as a 3-input logic gates, leading to a high logic-gate utilization, which might increase the complexity per logic gate (see Sec. 3.2.1);

2. The XOR ring network includes multiple autonomous nodes that can be sampled to generate multiple bitstreams per system;

3. The randomness produced by the XOR ring network, when tested with the NIST test suite, is found to have good statistical properties, such as low bias and correlations.

The characterization of the dynamics of the XOR ring network has not been documented prior to my publication [Ros13a] and the resulting random numbers have not been confirmed to be of high quality. I also integrate the network with a different sampling and processing strategy than that proposed in the patent because I find that the sampling strategy proposed in the patent leads to biased random numbers.

5.3 HYBRID BOOLEAN NETWORK APPROACH

The random number generator discussed in this chapter is illustrated in Fig. 22(a)-(d). It is a hybrid Boolean network composed of autonomous (unclocked) Boolean nodes and synchronous (clocked) Boolean nodes. The autonomous Boolean nodes form the XOR ring oscillator, which is an autonomous Boolean network. It includes N autonomous nodes that are assembled in a ring topology with bidirectional nearest neighbor coupling and feedback. Each node has an in-degree of $K = 3$, with two inputs connections from the output of each of its two nearest neighbors and one input from its own output, where $N - 1$ nodes execute the XOR and one node the XNOR Boolean function, as shown in Fig. 22(e)-(i) for different N. This autonomous Boolean network has four outputs tapped from four distant nodes, specifically, every fourth node in a ring of $N = 16$ nodes. In the hybrid Boolean network, the synchronous node has an in-degree of $K = 4$ and an out-degree of 1 and executes the 4-input XOR Boolean function, where the inputs are the outputs of the autonomous Boolean network. The autonomous Boolean signals are converted with four flip-flops into synchronous Boolean signals—streams of well-defined "0" and "1" that can change Boolean values at every period given by the clock. The hardware description of this network can be found in Appendix B.3.1.

The autonomous Boolean network, the XOR ring oscillator, can be scaled in the number of nodes N. The network topology for $N = 1$, shown in Fig. 22(e), is identical to the delayed-feedback XNOR studied in the previous Ch. 4 and introduced in Sec. 4.3, so that the XOR ring network is its natural extension.

5.3.1 DYNAMICS OF THE XOR RING NETWORK

In this section, I describe the dynamical behavior of the XOR ring network, which is the autonomous part of the hybrid Boolean network. The network shows a similar dynamical transition with the size of the network as discussed for the delayed-feedback XNOR oscillator in Sec. 4.4.3.

For a small number of nodes $N < 4$ the network displays a steady state, which involves constant node output voltages at the low or high Boolean voltage. For $N = 2$, as an example, the network shows dynamics where one node is in the high Boolean state and one node is in the low Boolean state, as shown in Fig. 23.

The reason for steady-state dynamics is that complex dynamics is prevented by the low-pass filter characteristics of the system, as described in Sec. 3.2.3. For example, more complex dynamics that can appear in such a network are oscillations with a frequency $f = 1/(2\tau)$, where τ is the total propagation delay through the network, as I explain in Ch. 6. The delay $\tau = \tau_{\mathrm{LG}} N$ is given by the number of logic gates in the network N and the

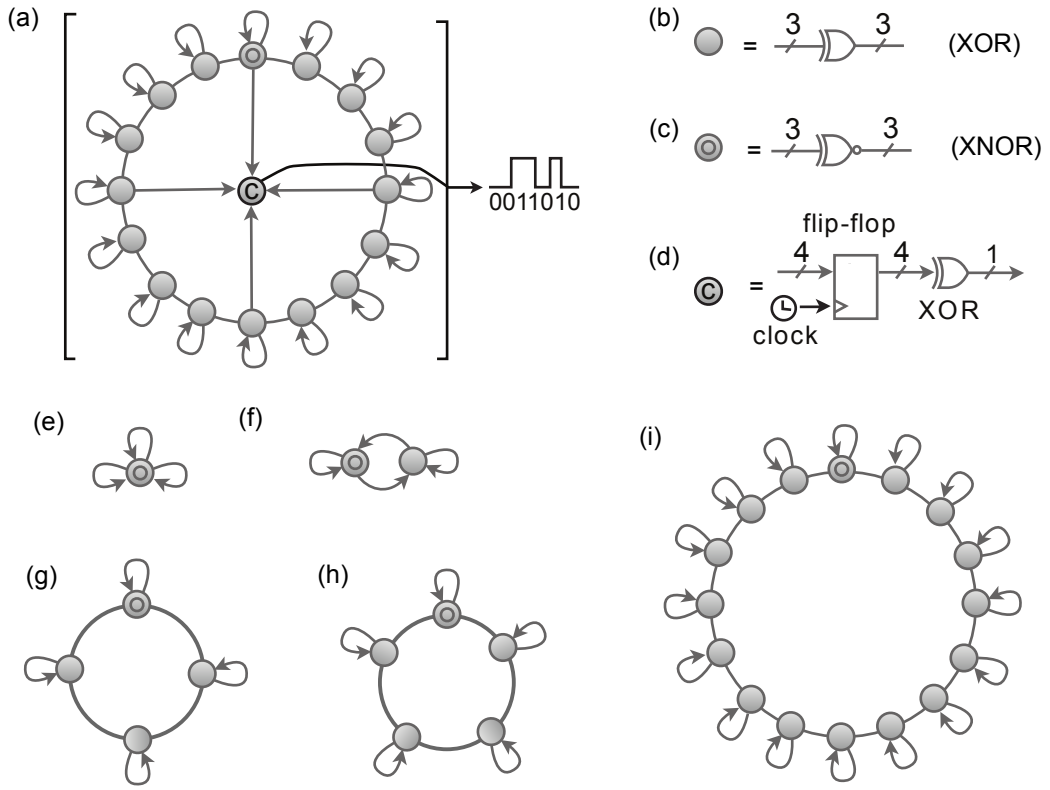


FIGURE 22: (a) Hybrid Boolean network composed of $N = 16$ autonomous Boolean nodes and one synchronous Boolean node. Links with (without) arrows are directional (bidirectional). The output of the hybrid Boolean network is a synchronous bitstream. (b), (c) Notion of an XOR and XNOR node. (d) Flip-flops are used to realize synchronous nodes. (e)-(i) XOR ring network for five different network sizes $N = 1$, $N = 2$, $N = 4$, $N = 5$, and $N = 16$, respectively.

propagation delay of a single node τ_{LG}. For small network sizes N, this frequency can be larger than the cutoff frequency of the lowpass filter, so that the system attenuates oscillations. For the autonomous Boolean network to bifurcate to non-steady dynamics, the propagation time must be increased, which can be achieved by increasing N.

For a network sizes between $N = 3$ and $N = 5$ nodes, depending on the specific realization of the network on the FPGA, the network starts to show oscillations. For $N = 4$, the first and the third node display a constant Boolean voltage, where nodes two and four display small-amplitude oscillations, as shown in Fig. 24. The two oscillatory nodes both receive constant Boolean inputs from their two neighbors and oscillatory inputs from themselves [see also the network topology in Fig. 22(g)]. Note that the network implementation includes output buffers as discussed in Sec. 3.3.2 that can influence the measurement of the dynamics. For example, if an output buffer has a slightly different threshold than the nodes in the network,

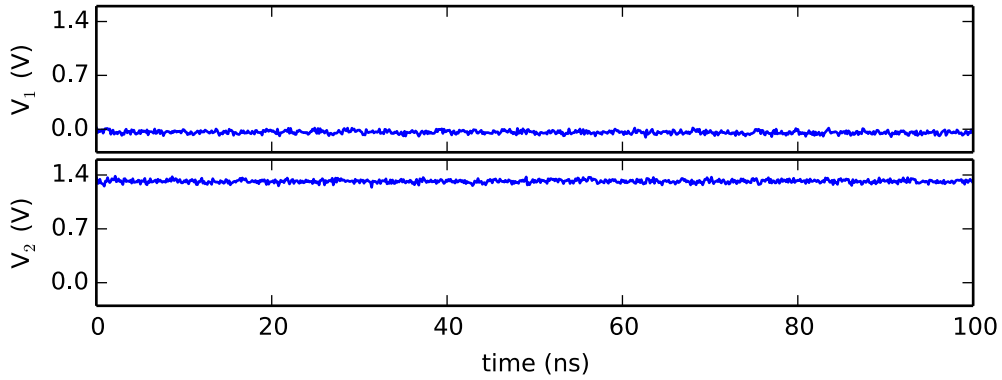


FIGURE 23: Temporal evolution of a network of four nodes with the topology shown in Fig. 22(f). The Boolean network is realized on the FPGA Altera Cyclone IV EP4CE115F29C7N. Data and Boolean transitions are sampled at 40 GSa/s with 12-bit quantization and 8 GHz analog bandwidth using oscilloscope DSO 9.

small-amplitude oscillations about the threshold might be observed as a constant Boolean state.

In this scenario, the propagation delay for a signal to travel through the network is increased, so that the frequency of the oscillations is decreased below the cutoff frequency of the lowpass filter, allowing for a single mode to oscillate with a low amplitude.

The network dynamics in a network of $N = 5$ nodes is shown in Fig. 25, where only four nodes are measured because the oscilloscope has only four channels. Three nodes display regular oscillations with large amplitudes that reach the high and low Boolean levels and one node shows oscillations with decreased amplitude of about 0.3 V with an oscillation pattern that is composed of pulses of different heights. This pattern is a result of the periodic inputs from the neighboring nodes.

The oscillations have a frequency $f \approx 1/(2N\tau_{\mathrm{LG}})$ as discussed above, originating from the low-pass filter characteristics of the logic gates. The first mode of oscillation with frequency $f = 1/(2N\tau_{\mathrm{LG}})$ is amplified; the next mode with frequency $f = 1/(N\tau_{\mathrm{LG}})$ and higher modes are still attenuated. Hence, the autonomous Boolean network has a size corresponding to a propagation delay that allows for exactly one Boolean transition to propagate.

I find that the network dynamics are chaotic for $N > 5$, which is shown in Fig. 26 for $N = 16$. The four recorded nodes display an irregular fluctuations between the two Boolean voltage levels similar to the measurement of Boolean chaos in Ch. 4.

The occurrence of chaos is not astonishing because the autonomous Boolean network fulfills the three guidelines for Boolean chaos identified in Ch. 4: It includes exclusively XOR and XNOR gates, it does not include a Boolean fixed point, and various time delays exist for signals to travel from

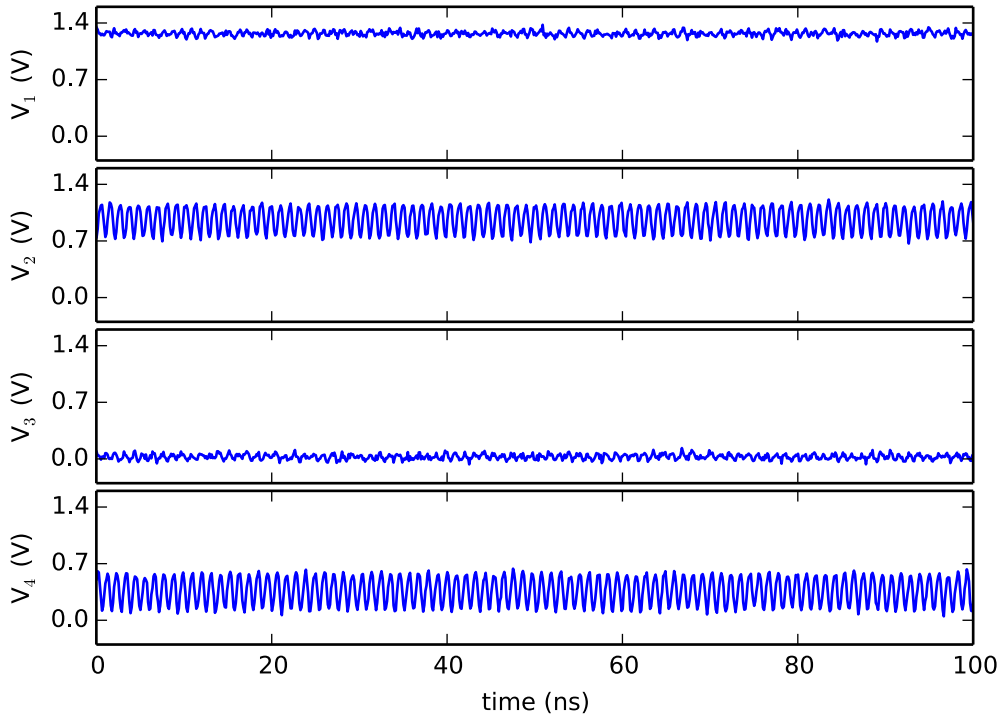


FIGURE 24: Temporal evolution of a network of four nodes with the topology shown in Fig. 22(g). Realization as in Fig. 23.

node to node. In accordance with the third guideline, Boolean chaos only appears when the accumulated time delay in the ring is long enough.

5.3.2 CHARACTERIZATION OF BOOLEAN COMPLEXITY

The complexity of the dynamics can be characterized by the distribution of time intervals between two consecutive Boolean transitions, as shown in Fig. 27 for a single node for different network sizes N. For the periodic oscillations for $N = 5$, shown in Fig. 27(a) for a single node, Fig. 27(b) shows the corresponding distribution of time intervals between Boolean transitions. The time intervals are distributed with a Gaussian shape with a mean at the period $T = 2.7\,\text{ns}$ and a width of $\Delta T = 17\,\text{ps}$. The mean corresponds to the average oscillation period of the waveform in Fig. 27(a) and the width corresponds to the timing jitter of the oscillations, which is a random fluctuation of the phase due to thermal, shot, and flicker noise from each logic gate [Haj98]. Jitter is the inherent entropy in the system, which can be directly harvested to generate random numbers using large ring oscillators, as described in Sec. 5.2.4. Direct harvesting of the jitter, however, results in low random bit rates, which can be increased by using chaotic dynamics.

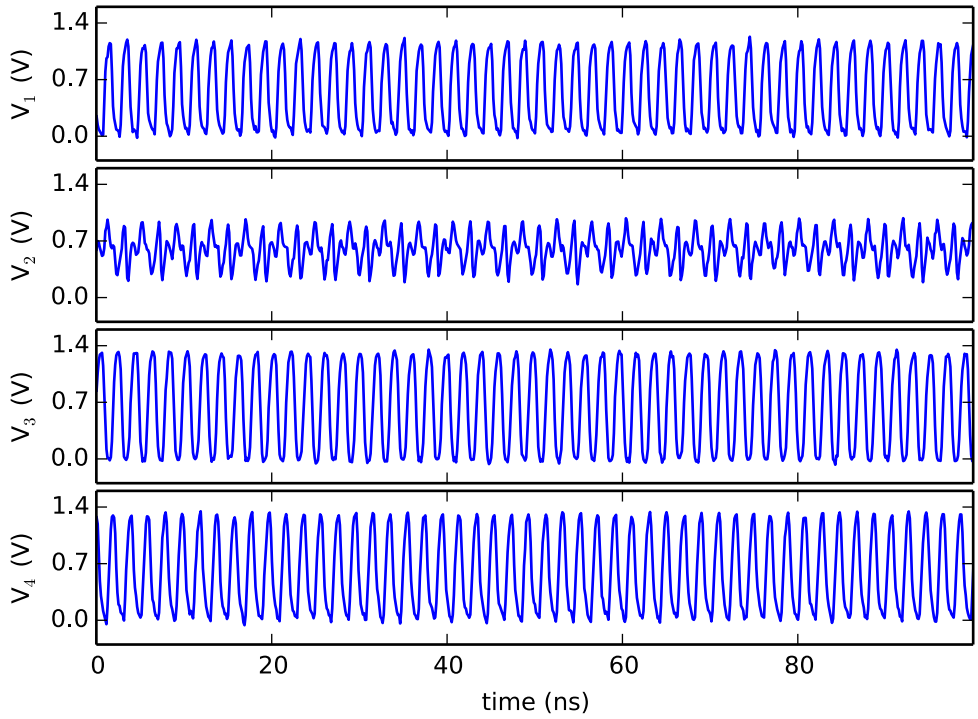


FIGURE 25: Temporal evolution of a network of five nodes with the topology shown in Fig. 22(h). Realization as in Fig. 23.

Waveforms of chaotic dynamics for one node are shown in Fig. 27(c) and (e) for network sizes of $N = 6$ and $N = 16$, respectively. These dynamics have the distribution of intervals between transitions shown in Fig. 27(d) and (f). These figures show a ∼200 times broader distribution than the jitter distribution in the periodic regime. The distribution of transitions for $N = 6$ presents significant fluctuations [Fig. 27(d)] that can introduce undesired statistical properties of the random numbers, when compared to those provided by a network with $N = 16$ nodes [Fig. 27(f)]. Because of the broader distribution, more randomness can be extracted from the dynamics in the chaotic regime than from jitter in the periodic regime.

The unpredictability of the information content of a message, such as a sequence of zeros and ones, can be assessed with the Shannon entropy $H(\cdot)$, defined as

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i), \tag{30}$$

where $X = \{x_i\}_{i=1}^{n}$ are n distinct outcomes of a random variable and $P(x_i)$ is the probability of the outcome x_i [Iha93]. For example, the entropy of a binary process with a probability of measuring "1" of $P(1) = p$ and probability of measuring "0" of $P(0) = 1 - p$ is $H(X) = -p \log_2 p - (1 - p) \log_2 (1 - p)$. For a bias-less random number stream, the binary entropy

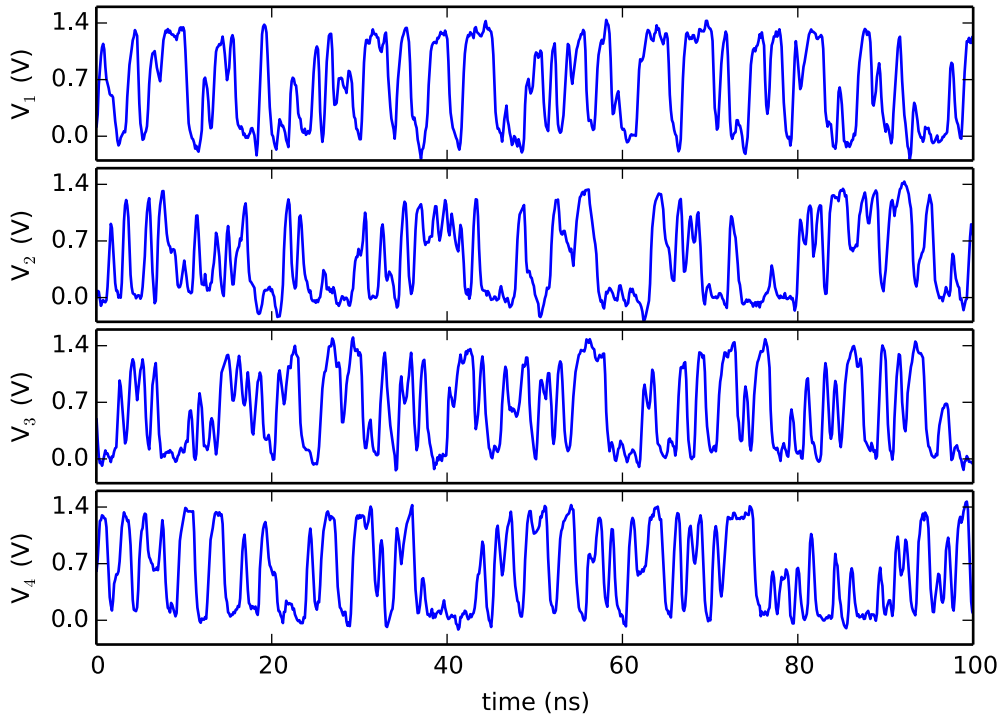


FIGURE 26: Temporal evolution of a network of 16 nodes with the topology shown in Fig. 22(i). The waveforms of four consecutive nodes including the node that executes the XNOR are shown. Realization as in Fig. 23.

results in $H(X) = 1$, which is the maximum value. Therefore, the binary entropy of a random bitstream is simply a measure of bias, which is at a maximum when the bias is zero. A bias-less stream of alternating bits, however, should lead to a low entropy as it is completely predictable. This is achieved by defining different outcomes x_i depending on sequences of binary values up to a certain length.

Here, the entropy is calculated similar to the approximate entropy test algorithm in the NIST statistical test suite [Ruko1] by counting the occurrences for all 2^m binary sequences x_i of length m in the tested bitstream and computing the probabilities p_i of pattern x_i for $i = 1, 2, ...2^m$. Specifically, the waveform acquired with the oscilloscope is sampled periodically with period $T_s = 10\,\text{ns}$ and 1-bit quantization using a threshold condition with symmetric threshold. The resulting bitstream of $n = 6550$ bits is tested for patterns of length $m = 7$ (parameters as suggested by the NIST statistical test suite [Ruko1]), resulting in probabilities p_i, which are used to calculate an estimate of the entropy with Eq. (30).

Figure 28 shows the entropy measured from one autonomous node as a function of N. For $N < 5$, the entropy is $H = 0\,\text{bit/sample}$. For oscillatory dynamics with $N = 5$, it is $H \approx (0.30 \pm 0.01)\,\text{bit/sample}$, but a larger

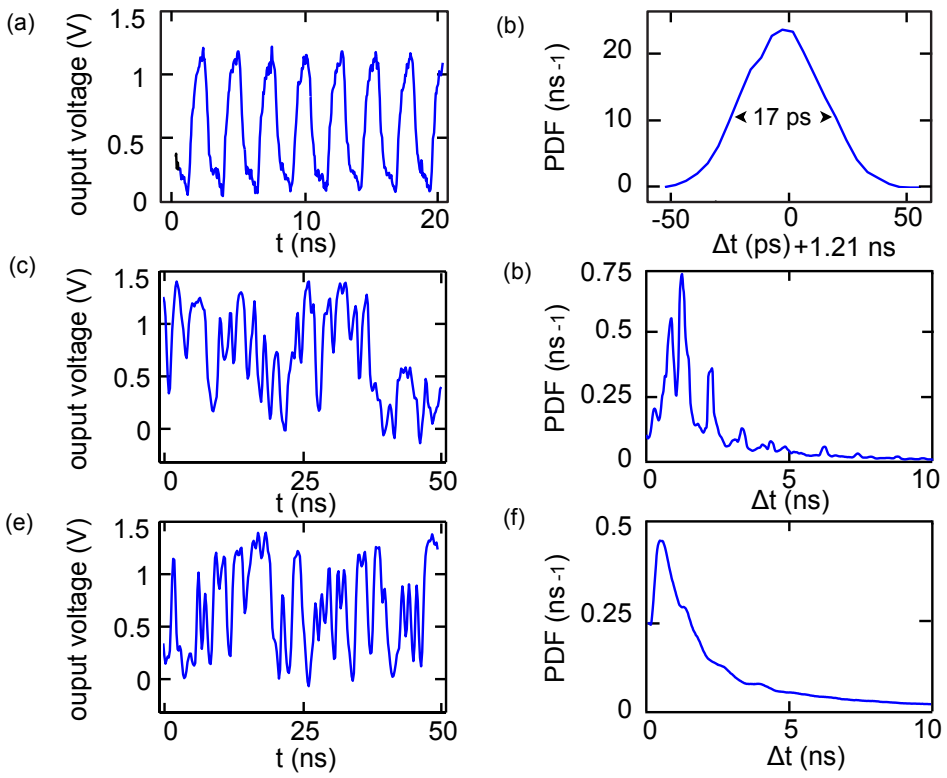


FIGURE 27: (a) Temporal evolution of the network for $N = 5$ measured from one autonomous node, displaying periodic dynamics. (b) Distribution of the time differences Δt between two consecutive Boolean transitions for $N = 5$, quantified by the probability density function (PDF) centered at zero. (c) Temporal evolution and (d) PDF for Δt for $N = 6$ when the autonomous Boolean network displays chaos. (e) Temporal evolution and (f) PDF for Δt for $N = 16$ when the autonomous Boolean network also displays chaos. Realization as in Fig. 23.

sequence length m and especially smaller sampling period T_s for the calculation of H could potentially lead to an entropy close to zero because this periodic pattern is predictable except for small jitter. In the chaotic regime with $N = 6$, the entropy increases to $H \approx (0.82 \pm 0.01)\,\text{bit/sample}$, and for $N > 7$ it is $H \approx (0.96 \pm 0.01)\,\text{bit/sample}$ close to the maximum achievable value of $H = 1\,\text{bit/sample}$ when sampled with T_s [Cov91]. The overall increase in entropy describes a dynamical transition with N and is a consequence of the increase of complexity in the dynamics with N as discussed in Sec. 5.3.1.

5.3.3 MODELING RESULTS FOR THE DYNAMICS OF THE XOR RING NETWORK

The dynamics of the XOR ring network, the coupled autonomous nodes in the hybrid Boolean network, can be modeled with dynamic equations

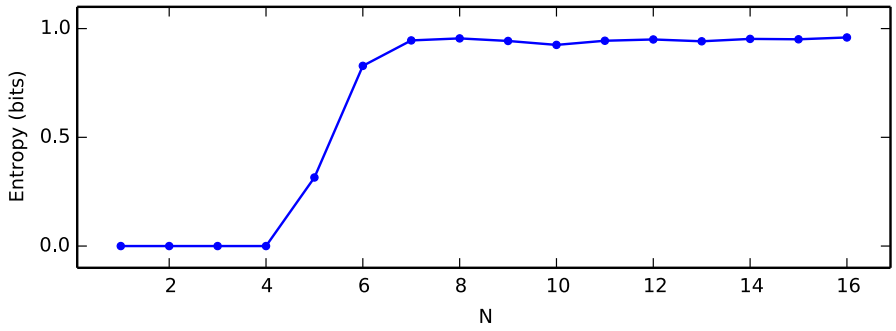


FIGURE 28: Entropy of the analog waveform generated of an autonomous node in the hybrid Boolean network as a function of N as defined in the text.

similar to the framework developed in Sec. 4.4.2, which is an extension of piecewise-linear switching networks by Glass and collaborators [Gla98, Mes96]. The dynamic equation reads

$$\tau_{LP}\dot{x}_i = -x_i + x_{\mathrm{dr}}, \tag{31}$$

where $\tau_{LP} = T_{\mathrm{rise}}/\ln(2) = 0.4\,\mathrm{ns}$ is the characteristic timescale of the dynamics (see Sec. 4.4.2), x_i is the continuous state of node $i \in \{1,...,N\}$, and x_{dr} is the Boolean driving term. From the setup of the XOR ring network in Fig. 22(e)-(i), the Boolean driving term x_{dr} is a 3-input XOR or XNOR Boolean function according to

$$x_{\mathrm{dr}}(t) = \mathrm{XOR}\left[X_{i-1}(t-\tau_{i,1}), X_i(t-\tau_{i,2}), x_{i+1}(t-\tau_{i,3})\right] \text{ (cyclic)}, \tag{32}$$

for $i \in \{2,...,N\}$; for $i = 1$, the XOR function is replaced with an XNOR function. Here,

$$X_i(t) = \begin{cases} 1 & \text{if } x_i(t) > x_{th} \\ 0 & \text{if } x_i(t) \leq x_{th}, \end{cases} \tag{33}$$

is the Boolean state of node i. The quantity $x_{th} = 0.5$ is the Boolean threshold, which is assumed to be symmetric between the two Boolean levels of 0 and 1. The time delays $\tau_{i,j}$ are fixed random values sampled from a uniform distribution, so that

$$\tau_{i,j} = \tau_0 + r\tau_1, \tag{34}$$

with $r \in [-1,1]$ a uniformly distributed random variable, $\tau_0 = 0.2\,\mathrm{ns}$, and $\tau_1 = 0.02\,\mathrm{ns}$. This distribution is adjusted to achieve qualitative agreement with the experimental results, but the quantitative values also agree physically for the larger time delays between logic array blocks [Khe94, Max09] (see also Sec. 3.2.1). With this distribution, I introduce heterogeneity in the time delays of the network. The system is simulated with an Adam-Bashforth-Moulton method as explained in Appendix B.8.

The results of the simulations are shown in Fig. 29. The simulation of a small network of $N = 2$ nodes results in periodic oscillations [Fig. 29(a)]. The oscillations show a peak in the distribution of transition times at half the period $T/2 = 0.34\,\mathrm{ns}$ of zero width given by the measurement quantization [Fig. 29(b)]. This indicates that the period of oscillation is constant, caused by the description of the dynamics with a deterministic model without noise. For a large network of $N = 16$ nodes, the dynamics show irregular chaos-like dynamics [Fig. 29(c)]. The distribution of transition times is continuous, but still shows a large maximum at $t = 0.34\,\mathrm{ns}$ and a smaller maximum at twice this values.

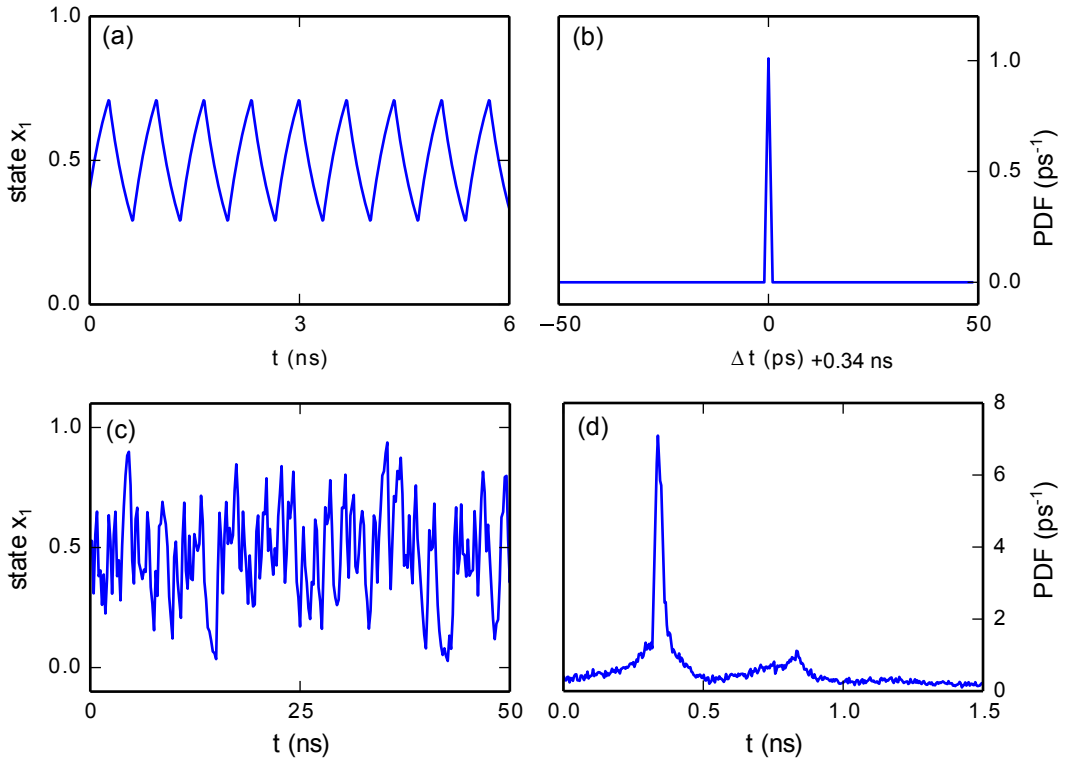


FIGURE 29: Dynamics from the numerical simulation of Eq. (31)-(34). Temporal evolution of one node and corresponding distribution of transition times in a network of (a), (b) $N = 2$, and (c), (d) $N = 16$ nodes, respectively. (b) and (d) show peaks at $t = 0.34\,\mathrm{ns}$. I simulate the system with a timestamp $\mathrm{d}t = 0.001\,\mathrm{ns}$ and the timescale parameter $\tau_{\mathrm{LP}} = 0.4\,\mathrm{ns}$.

When compared to the experimental data in Fig. 27, I observe similar waveforms in the simulations, but the distribution of transition times is broader in the experiment. The latter difference could be due to a different time delay distribution and jitter in the experiment, which is not yet included in the model.

5.3.4 MODELING RESULTS FOR BOOLEAN COMPLEXITY

The dynamical transition to high complexity with the network size N observed in the experiment also occurs in the model. To find the transition, I sample the simulated dynamics every 10 ns with a threshold condition, leading to a bitstream of 10^8 bit/s. Figure 30 shows the entropy of the dynamics for different node sizes N. The entropy increases towards 1 for $N \geq 6$. For $N < 6$, the entropy is below 0.5, corresponding to regular oscillations.

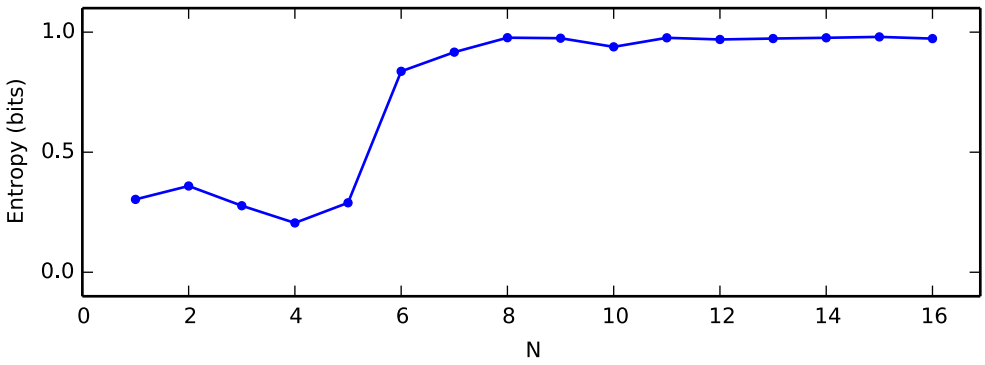


FIGURE 30: Analysis of the numerical simulation data similar to Fig. 28 showing the entropy of the dynamics for different network sizes N. I simulate the system with a time step of $\mathrm{d}t = 0.01\,\mathrm{ns}$.

Compared to the experimental data in Fig. 28, the entropy of the simulated dynamics does not decrease to zero for small network sizes because the simulated dynamics oscillates for small N about the switching threshold in the simulation and are not constant. In fact, it could be that the experiment also shows oscillations for small node number, but that this dynamics is measured as a constant Boolean voltage due to output gates in the experiment that are not included in the simulation (see also Sec. 3.2.1).

5.3.5 THE SYNCHRONOUS PART OF THE HYBRID BOOLEAN NETWORK

The autonomous Boolean network can be used for physical random number generation by sampling the Boolean output from one node. However, I observe a bias of $\approx 1\%$, which prevents the system to pass standard randomness tests, so that the autonomous Boolean network alone cannot be used for high-quality random number generation. As a solution, I can—similar to most other studies on physical random number generation—include a bias-reduction technique.

Bias removal usually results in a decrease in the bitrate or an increase in the system size as discussed in Sec. 5.2.2 for previous setups. This is not the

case when the network character of multiple nodes is exploited, where multiple bitsteams from the autonomous Boolean network are probed in parallel. Specifically, four bitstreams are routed from the autonomous Boolean nodes to the synchronous Boolean node constituting a clocked 4-input XOR logic gate. The synchronous node is a part of the hybrid Boolean network operating in real time.

The synchronous node realizes a 1-bit quantization operation, which is adequate for Boolean chaos whose complexity originates from the timing of Boolean transition rather than the voltage amplitude [Zha09a]. The clocked node performs an XOR operation, which is known to reduce bias if the incoming bitstreams are sufficiently uncorrelated [Uch08] (see Sec. 5.2.3 and Appendix C.1). In the setup shown in Fig. 22(a), autonomous nodes with inputs to the synchronous nodes have a distance of four. The dynamics between these nodes is sufficiently correlated to result in low enough bias to pass standard randomness tests. Specifically, the correlation between these nodes is less than 7.5%, measured with the normalized cross-correlation function. The clocked node includes a flip-flop that samples the autonomous nodes with a clock frequency of 100 MHz, as shown in Fig. 22(d). This sampling rate corresponds to the maximum transfer rate to the memory elements on the FPGA and leads to a bitrate of 100 Mbit/s. Higher bit rates are allowed by the network because of a short correlation time ($\approx$ 590 ps) of the chaotic dynamics generated by an autonomous node. However, the memory transfer speed of the inexpensive FPGA platform precludes me from extracting this accessible entropy rate.

5.4 UTILIZATION AS A PHYSICAL RANDOM NUMBER GENERATOR

In this section, I test the hybrid Boolean network as a random number generator and increase its rate through parallelization.

5.4.1 TESTING THE PHYSICAL RANDOM NUMBER GENERATOR

The hybrid Boolean network-based random number generator is implemented on various FPGA platforms, such as the Altera Cyclone IV, Altera Stratix IV, Xilinx Virtex VI, and the CPLD Altera MAX II with a tranfer protocol to the computer as detailed in Appendix B.3.2. For all of the hardware chips, I assess the quality of the random numbers with 1000 bitstreams of 1 Mbit of data (total of 1 Gbit) using the NIST test suite [Ruk01] and observe consistently successful passes at a rate of 100 Mbit/s with specific test re-

sults discussed in the next section. This shows that my approach is robust to changes in technology.

The resulting random numbers generated by the clocked node of the hybrid Boolean network are non-deterministic because of the mixing property of chaos, where the electrical noise that induces jitter is mixed into the chaotic dynamics of the autonomous nodes (see also Sec. 5.2.2).

5.4.2 PARALLELIZATION TO INCREASE THE BITRATE

The achievable random bit rate of a single hybrid Boolean network remains one order of magnitude below that of photonic systems. However, the small fraction of required logic gates is less than 0.02% of the logic gate on an Altera Cyclone IV FPGA, allowing me to parallelize thousands of random number generators, thus increasing the overall bit rate.

The parallelization scheme is shown in Fig. 31 with a hardware description discussed in Appendix B.3.3. I implement 128 uncoupled hybrid Boolean networks in parallel that have independent temporal evolution. Together, the networks generate 128 random bits per clock cycle that are saved to on-chip memory in parallel. As I keep the sampling rate at 100 MHz, I achieve a cumulative bit rate of 12.8 Gbit/s. A slower clock is then used to convert the parallel data to a single bitstream and to send it to a computer. In principle, a better communication architecture will be able to also stream the data at 12.8 Gbit/s to a computer. In fact, this rate is only limited by the memory capacity of the FPGA.

The cumulatively generated random numbers pass successfully the NIST tests using 1 Gbit of data with results shown in Table 3. Each of the 15 tests is run on all of the 1000 bitstreams, resulting in 1000 test results. The proportion of passed repetitions of a test is one deciding factor of the quality of the random numbers. Not all tests are passed because of the statistical nature of the test, where a certain fraction of tests are expected to fail. The P-value gives the probability to achieve test results as extreme as the measured ones. According to Ref. [Ruk01], a sequence can be considered to be non-random with confidence of 99.9% for a P-value < 0.001. On the other hand, a P-value > 0.001 does not allow to refute the random number generator as non-random with that confidence.

The random numbers are also visualized as a pixel matrix, where a ”1“ is shown as a black dot and a ”0“ is shown as a white dot. Certain patterns can be spotted by the human eye, but, in the figure, no patterns are visible, which gives further confirmation that he random numbers are of high quality.

In addition to using the statistical test suites, I compute the 3σ confidence intervals for the bias b and serial-correlation coefficient ρ under the statistical null hypothesis H_0 of a perfectly uncorrelated, unbiased random number generator using $n = 3 \times 10^9$ bits. They are given respectively

TABLE 3: Results of the 800-22 NIST test suite [Ruk01] using 1 Gbit of data (1000 sequences of 1 Mbit) generated by the 128 hybrid Boolean network-based random number generators implemented in parallel on an Altera Cyclone IV EP4CE115 FPGA. All tests are passed successfully because the P-value is larger than 10^{-4} and the proportion is greater than 0.980.

Statistical Tests	P-value	Proportion	Result
Frequency	0.0856	0.991	Success
Block frequency	0.7887	0.993	Success
Cumulative sums	0.3191	0.988	Success
Runs	0.2954	0.989	Success
Long runs	0.0081	0.992	Success
Ranks	0.1147	0.995	Success
Fast Fourier transform	0.4750	0.991	Success
Nonoverlapping templates	0.1445	0.983	Success
Overlapping templates	0.6621	0.987	Success
Universal	0.0288	0.990	Success
Approximate entropy	0.5728	0.989	Success
Random excursion	0.3694	0.982	Success
Random excursion var	0.3917	0.982	Success
Serial	0.5544	0.987	Success
Linear Complexity	0.4944	0.992	Success

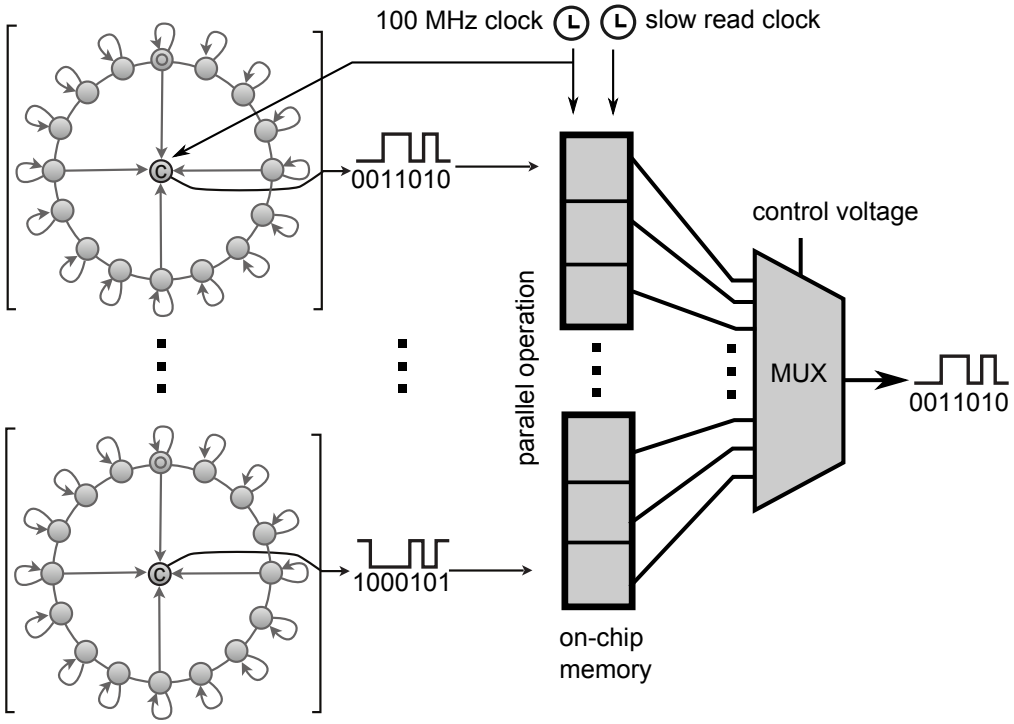


FIGURE 31: Schematic of the parallelization on the electronic chip. 128 hybrid Boolean networks are implemented on the FPGA and operate in parallel of which two are shown. The output bitstream of 100 Mbit/s is written in parallel to the on-chip memory. After 1 Mbit of data is written, a serializer converts the saved parallel data into a single bitstream via a slow read clock and a multiplexer. The serial data is sent to the computer for analysis.

by $\hat{b} \pm 3\hat{\sigma}/\sqrt{n} = [-2.826, 2.651] \times 10^{-5}$ and $\hat{\rho} \pm 3\sqrt{(1-\hat{\rho}^2)/(n+1)} = [-4.177, 6.777] \times 10^{-5}$ [Whi57] with $\hat{b}$, $\hat{\sigma}$, and $\hat{\rho}$ being the test statistics for the bias, standard deviation, and serial-correlation coefficient, respectively. Because these two intervals contain $b = 0$ and $\rho = 0$, H_0 is not rejected.

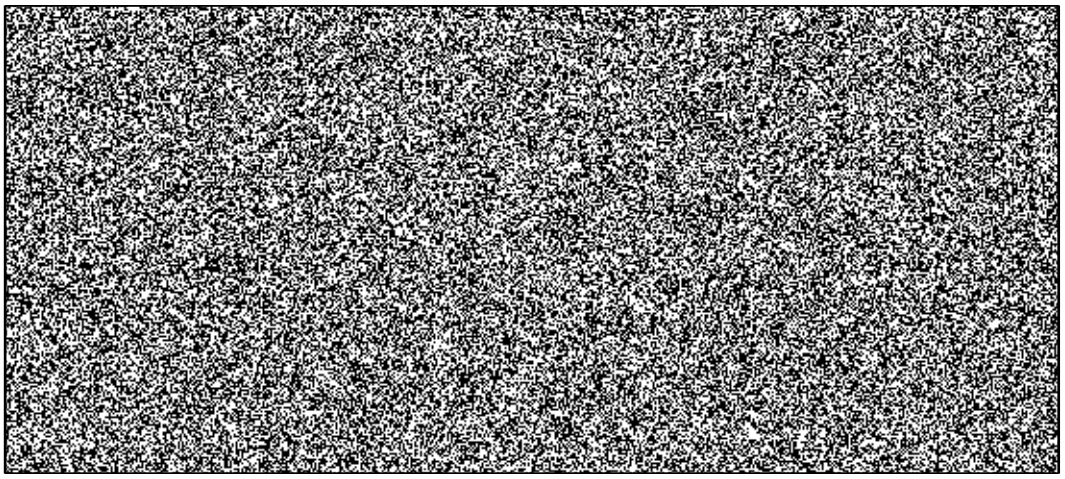


FIGURE 32: Visualization of 1 million bits of random data.

5.5 CONCLUSION

In this chapter, I have discussed the generation of random numbers with autonomous Boolean networks. Specifically, I have studied experimentally and numerically a hybrid Boolean network comprising autonomous and synchronous nodes. The autonomous nodes form an autonomous Boolean network, which shows a transition to complexity similar to the delayed-feedback XNOR oscillator studied in Ch. 4.

The hybrid Boolean network generates high-quality physical random numbers; it can be realized with a small number of logic gates on an FPGA, allowing for parallel implementation. Both a parallel implementation and a single hybrid Boolean network generate high-quality random numbers that pass the tests in the NIST statistical test suite for randomness. I achieve a maximum bitrate of 12.8 Gbit/s without further processing. With current chip technology, I conjecture that tens of thousands of hybrid Boolean networks can be implemented in parallel because the implementation of 128 random number generators exploits less than 1% of the available logic gates on typical FPGAs. The approach using hybrid Boolean networks potentially opens the path towards Tbit/s physical random number generation on a single electronic device.

To my knowledge, the proposed random number generator is being considered for use at NIST as an inexpensive replacement for commercial random number generators that cost about 1,000 USD, whereas my random number generator design can be implemented on any spare FPGA. When bought new, programmable chips are also inexpensive; for example, MAX II CPLDs that I have also tested to pass the standard randomness test cost only 6 USD.

I have mentioned above that the autonomous Boolean network is already patented [Bae08]. My contribution is that I have reported to the community that the random number generator with my specific way of post-processing produces reliable random numbers. On the other hand, when I implement the post-processing technique proposed in the patent, it does not generate high-quality random numbers. Further contributions are that I have increased the bitrate in this system through parallelization, leading to a new record bitrate for an inclusion of the XOR operation on the random number generator and that I have described the dynamics of the system with a piecewise-linear switching model.

An autonomous Boolean network with chaotic dynamics can be implemented as a network node in a meta-network. When multiple such nodes with chaotic dynamics are coupled together, chaos synchronization is in-principle possible as demonstrated with electronic circuits on printed circuit boards [Gao09]. On FPGAs, however, I am not able to synchronize two chaotic systems because of heterogeneities of logic gates, e.g., in their propagation delays, low-pass filter characteristics, and electronic noise. These non-ideal effects result in significant parameter mismatch when implementing multiple copies of chaotic oscillators on the same FPGA.

The coupling of meta-networks of autonomous Boolean networks is possible on the FPGA for autonomous Boolean networks in the periodic regime. This is the topic of the next chapter.

6

PERIODIC DYNAMICS IN AUTONOMOUS BOOLEAN NETWORKS

6.1 ABSTRACT

This chapter focuses on periodic dynamics in autonomous Boolean networks. The goal is to design and characterize a periodic Boolean oscillator that can be coupled into networks to study the resulting network dynamics. The main challenge is to implement a coupling mechanism that is adjustable and can be tuned weak. In Sec. 6.2, I first introduce the existing theoretical and experimental work on periodic dynamical systems and their synchronization. Then, in Sec. 6.3, I show how periodic autonomous Boolean networks can be coupled in an "on"-"off" fashion in network motifs consisting of two periodic oscillators. In Sec. 6.4, I introduce and characterize a more advanced periodic Boolean oscillator that allows for weak coupling with adjustable coupling strength. I term this experimental periodic oscillator a Boolean phase oscillator because it bears resemblance to the Kuramoto model. Its coupling mechanism is characterized within network motifs.[1]

The main contributions of this chapter are:

- studying the dynamics of a simple periodic Boolean oscillator and small network motifs of this dynamical system;
- designing, characterizing, and coupling of a periodic Boolean oscillator that allows for weak coupling;
- developing models for the dynaical systems in this chapter.

6.2 INTRODUCTION TO PERIODIC DYNAMICAL SYSTEMS

In this section, I introduce previous work on dynamical systems with periodic dynamics.

1 Results of this chapter are published in Refs. [Ros13b] and [Ros14].

6.2.1 HISTORICAL PERSPECTIVE ON SYNCHRONIZATION OF PERIODIC OSCILLATORS

In the seventeenth century, the Dutch researcher Christiaan Huygens invented the pendulum clock, studies of which lead him to the discovery of synchronization phenomena, such as mutual synchronization [Huy86]. Following Huygens' discovery, Lord Rayleigh used acoustical systems to discover more complex synchronization phenomena, such as oscillation quenching—also known as amplitude death—in 1870 [Ray96]. Appleton and van der Pol used electronic components to find that the frequency of an oscillator can be synchronized to a driving signal of a slightly different frequency, which they also described theoretically as discussed in Sec. 6.2.4.1 [App22, Pol20]. Early studies on synchronization were not limited to man-made systems, but were also conducted on biological system as early as in 1729 [Pik01].

6.2.2 MOTIVATION FOR THE STUDY OF SYNCHRONIZATION OF PERIODIC OSCILLATORS

Synchronization of periodic oscillations is ubiquitous in biological systems, such as circadian clocks [Ued05, Har01], oscillations of the central pattern generator controlling rhythmic body movements [Mar01, Ste99, Sel10] (see also Sec. 9.2), or fireflies that flash in unison [Buc76, Str93]. In neural systems, synchronization can have both positive and negative effects as detailed in Sec. 9.2.1. Synchronization of electronic clocks is important in electrical engineering for operation of electronic circuits, specifically synchronous logic circuits, such as central processing units (CPU), and communication systems, such as data transfer protocols. The synchronization of signals limits, for example, both the data transfer rate and the size and speed of CPUs [Bro08]. To allow for synchronization, engineers use electrical circuits called phase-locked loops (PLLs), which are discussed in Sec. 6.2.5.3. For many examples, the dynamics of simple coupling topologies are already important such as one oscillator synchronizing to a periodic signal and much current interest exists is the study of oscillators networks, such as chemical and biological oscillators discussed in Ch. 7.

In this chapter, I study small coupling topologies of two oscillators and develop experimental dynamical systems for the study large networks.

6.2.3 NOTATION OF A PERIODIC OSCILLATOR, PHASE, AND SYNCHRONIZATION

A periodic oscillator is a dynamical system that produces a periodic oscillatory output even when it is isolated from its environment [Pik01]. The dynamical system of a periodic oscillator has a closed attractive orbit in

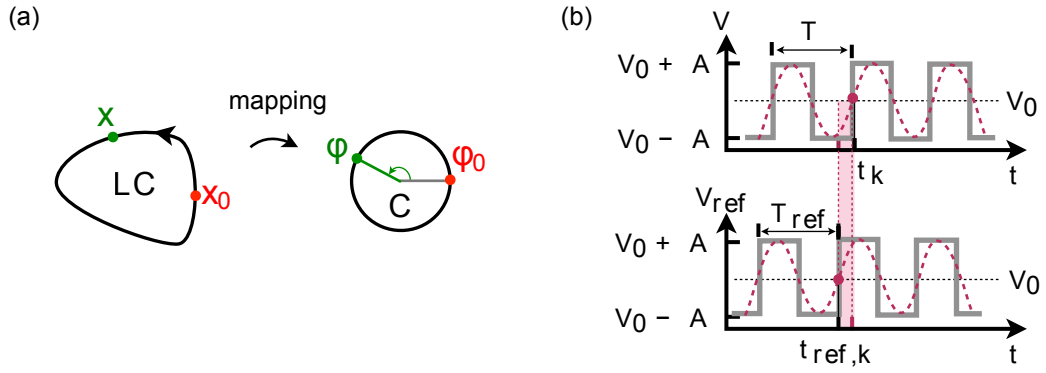


FIGURE 33: (a) Illustration of the mapping between limit cycle (LC) and unit circle (C). (b) Measurement of the phase difference between two square waves. The first Fourier mode of the square wave is shown with red dashed lines. The shaded region highlights the phase difference between the two square waves.

phase space, known as a limit cycle. When the dynamical system is on this limit cycle, its dynamics can be expressed in form of the phase description with frequency f_0 and phase ϕ

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = 2\pi f_0, \tag{35}$$

by scaling the system variables of a single oscillator in time as illustrated in Fig. 33(a) [Pik01]. In the figure, a limit cycle is mapped onto the unit circle, where the frequency f_0 is constant. In this description, the phase increases constantly over time with slope given by the frequency. Figure 33(b) shows that the phase can also be extracted from a square wave, which is discussed in detail in Sec. 6.2.5.2.

Synchronization of multiple oscillators can be defined as an "adjustment of rhythms of oscillating objects due to their weak interaction" [Pik01]. Two oscillators are synchronized when their frequencies f_1 and f_2 take on the same value due to small coupling adjustments; hence, they are frequency synchronized, according to

$$f_1 = f_2. \tag{36}$$

This is also often expressed as phase locking, according to

$$\phi_2(t) - \phi_1(t) = \text{constant}, \tag{37}$$

where the phases of two periodic oscillators have a constant phase shift [Pik01].

The synchronization of weakly coupled oscillators appears only in a finite range of parameter mismatch, such as mismatch of free-running frequencies. The synchronization region is a function of the coupling strength, which is illustrated with the Arnold tongues for an oscillator with periodic output signal of frequency f_s that is externally driven by a periodic signal

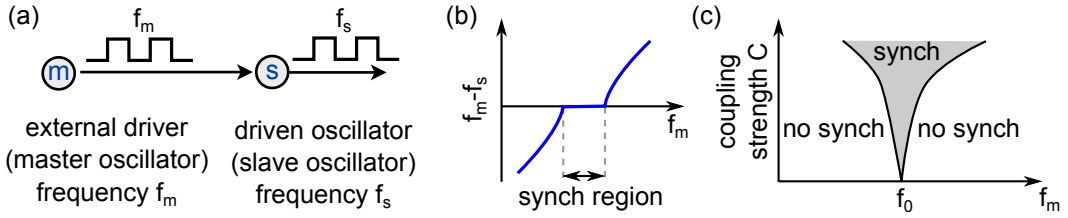


FIGURE 34: (a) Illustration of the measurement of the synchronization region with a master and slave oscillator. (b) Typical measurement results, where $f_m - f_s = 0$ denotes the synchronization region. (c) The Arnold tongue, showing the synchronization region as a function of the coupling strength. f_0 denotes the free-running frequency of the slave oscillator.

of frequency f_m (m, s stand for master and slave oscillator) as shown in Fig. 34(a).

The synchronization region can be measured by scanning f_m and recording the region where $f_s = f_m$, corresponding to Eq. (36), leading to the graph in Fig. 34(b). Also fractional synchronization regions are possible, where $l \cdot f_m = k \cdot f_s$ with two integers l and k. If such synchronization regions appear, the graph is also known as the devil's stair case, as discussed in Sec. 6.4.2.

Arnold tongues denote the synchronization region as a function of the coupling strength. Typically, the synchronization region grows with the coupling strength as shown in Fig. 34(c). For driving frequencies f_m within the gray area (the Arnold tongue), the slave oscillator frequency f_s locks; otherwise, it runs at a different frequency than the input frequency. When fractional synchronization is considered, each integer pair k and l (using $l \cdot f_m = k \cdot f_s$) can lead to a synchronization region, so that mutiple Arnold tongues are possible [Pik01, Rei99].

6.2.4 DYNAMICAL SYSTEMS WITH PERIODIC DYNAMICS

In this section, I introduce two examples of periodic dynamical systems that were historically important to increase the understanding of coupled oscillators.

6.2.4.1 *Van der Pol Oscillator*

In 1920, Appleton and Van der Pol studied several experimental realization of a nonlinear electronic oscillator, shown in Fig. 35(a) [App22, Pol20]. In the figure, the oscillator includes a neon lamp to realize the cubic nonlinearity, but it can also be realized with triodes and tunnel diodes. They model the circuit with the following nonlinear differential equation for the normalized voltage measured across the capacitor x

$$\ddot{x} - \epsilon(1 - x^2)\dot{x} + x = 0, \tag{38}$$

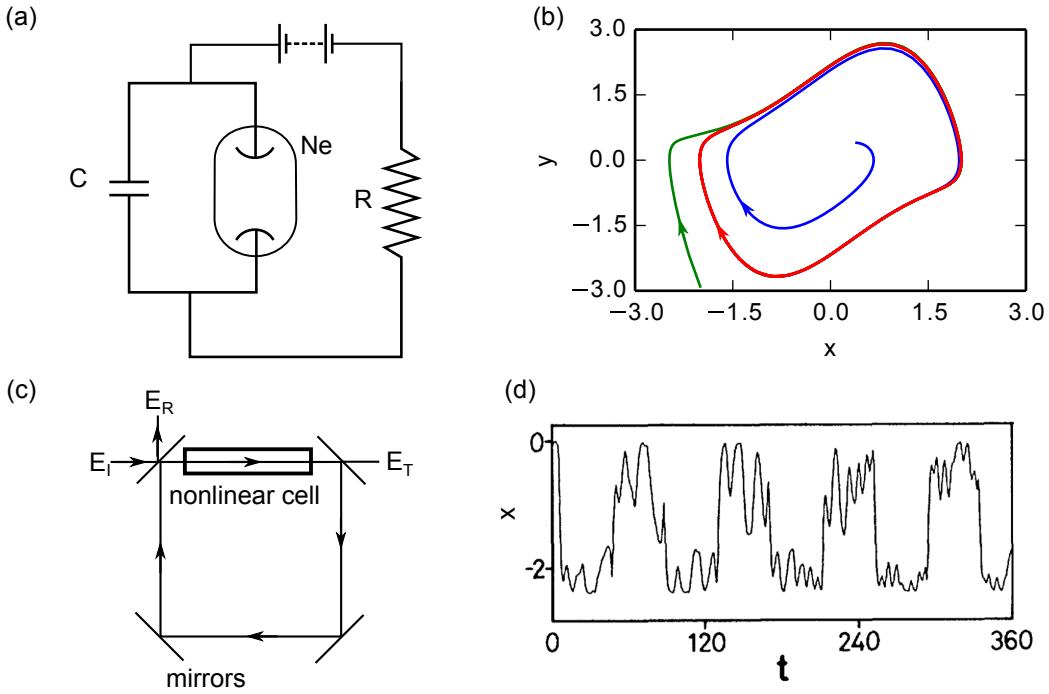


FIGURE 35: (a) Experimental realization of a Van der Pol oscillator with a resistor (R), capacitor (C), and a neon lamp to realize the nonlinearity. Setup proposed in Ref. [Pol28] is a simplified version of the original setup that uses a triode as nonlinearity. (b) Simulation results of Eq. (39) with $\epsilon = 1$ for two different initial conditions outside (green) and inside (blue) the limit cycle (red). (c) Setup of the Ikeda system that uses an absorption cell as a nonlinear medium (denoted nonlinear cell) in a ring cavity. (d) Example dynamics of the Ikeda system, simulated with Eq. (41), showing square-wave-like oscillations with period 2τ. (c) and (d) are modified from Refs. [Ike79, Ike82].

which includes nonlinear damping $\gamma = \epsilon(1 - x^2)$. When $x > 1$, the damping is positive and slows down the oscillations, when, on the other hand, $x < 1$, the damping is negative, leading to an acceleration. The result is a limit cycle.

Using the transformation $y = x - x^3/3 - \dot{x}/\epsilon$, Eq. (38) can be rewritten as

$$\dot{x} = \epsilon(x - \frac{1}{3}x^3 - y) \tag{39}$$

$$\dot{y} = \frac{x}{\epsilon}. \tag{40}$$

This is a special case of the FitzHugh-Nagumo model discussed in Sec. 8.2.4 for neural systems. The resulting dynamics are shown in Fig. 35(b), where the limit cycle is shown in red and trajectories in phase space are attracted to it, thus leading to stable periodic oscillations.

Using the electronic implementation of the system as shown in Fig. 35(a), Appleton and Van der Pol discovered frequency locking, *i.e.* frequency synchronization, with different fractions of the driving and oscillator frequency

and also documented signs of deterministic chaos, but did not identify them as such [App22, Pol20, Pol27].

As a predecessor of the FitzHugh-Nagumo model, the Van der Pol oscillator has contributed substantially to the study of neural systems (see references in Sec. 8.2.4). It also serves in nonlinear dynamics as a model system [Ran80]. The study of synchornization by Van der Pol and Appleton were also of great practical importance for the development of radio communication systems [Pik01].

6.2.4.2 *Ikeda System*

Ikeda studied in 1979 an optical feedback system that includes a nonlinearity and a time delay in the feedback due to finite transmission times [Ike79]. The system includes a nonlinear absorption cell and several mirrors as shown in Fig. 35(c). The mirrors implement a ring cavity through which light propagates in a circular fashion, leading to a delay given by the roundtrip time. The absorption cell is a nonlinear medium that the light propagates through in every roundtrip. The dynamics for the phase lag imposed by the nonlinear medium x are described by the delay differential equation

$$\dot{x} = -x(t) + \gamma f\left[x(t-\tau)\right], \tag{41}$$

with a specific nonlinear equation $f[\cdot]$, feedback gain γ, and time delay τ [Ike82].

For low feedback gain γ, the system can show periodic oscillations at frequency

$$f = \frac{1}{2\tau}, \tag{42}$$

where the factor of two appears for negative feedback, where one roundtrip results in a polarity-flipped signal and two roundtrips lead to the original signal as shown in Fig. 35(d). The square-wave oscillations are modulated with unordered deterministic fluctuations [Ike82]. For higher feedback gain, a similar system has been shown to display breathing, where long periodic oscillations are modulated by unordered, deterministic fluctuations, and deterministic chaos [Cal10, Lar10, Pei09]. The system has also been shown to display features of excitability [Ros11a]. The dynamics of Eq. (41) can also be implemented with an optoelectronic oscillator, which includes both optical and electronic components [Pei09, Arg05, Kou05].

Recent studies on highspeed optoelectronic oscillators include bandpass-filtered electronic devices that allow for increased speed. The bandpass introduces two timescales: the high and low cutoff frequency, another timescale is provided by the delay. These three timescales are often found to determine features of the dynamics [Pei09]. Today, optoelectronic oscillators are used in modern telecommunication interfaces to realize ultra-stable periodic oscillations at high frequencies [Yao96].

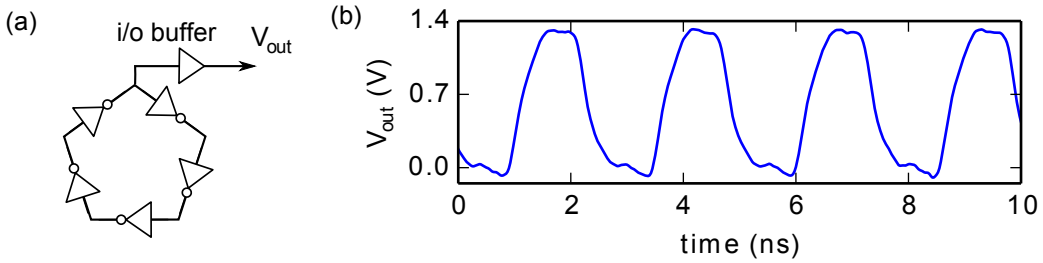


FIGURE 36: Ring oscillator of size $n = 5$. (a) Setup realized with autonomous logic gates. (b) Experimental time series with frequency $f = 390 \pm 4\,\mathrm{MHz}$.

6.2.5 PREVIOUS WORK ON PERIODIC AUTONOMOUS BOOLEAN NETWORKS

Synchronization of periodic electronic oscillators is an important topic in the engineering community because of its relevance for various applications, such as highspeed communication protocols. Periodic oscillators designed to lock onto an external signal are called phase-locked loops (PLLs). Great interest exists in the implementation of PLLs with all-digital circuitry because it allows to integrate them on logic chips [Bes03]. The foundation for all-digital PLLs are ring oscillators, which are dynamical systems with a time delay and negative feedback.

6.2.5.1 *Ring Oscillators*

Ring oscillators are unclocked electronic circuits illustrated in Fig. 36(a). They consist of an odd number of cascaded inverter gates assembled in a unidirectional ring topology [Kat98]. When realized on an FPGA, the setup also includes a buffer output gate as discussed in Sec. 3.3.1. They display periodic oscillations with approximately the shape of a square wave with high and low voltage given by the high and low Boolean voltage V_H and V_L, respectively [see Fig. 36(b)]. The period of the oscillations is given by twice the total delay in the ring, leading to the frequency [Sun07, Wol09]

$$f = \frac{1}{2\tau}, \tag{43}$$

which is typical for delayed feedback systems with negative feedback (see Sec. 6.2.4.2). Specifically, the roundtrip delay is given by $\tau = n\tau_{\mathrm{LG}}$ with $\tau_{\mathrm{LG}} = 0.28 \pm 0.1\,\mathrm{ns}$ for ring oscillators realized with n inverter logic gates.

The system is an autonomous Boolean network without a Boolean fixed point, for which all Boolean functions are satisfied simultaneously as defined in Sec. 4.3, hence leading to non-steady-state dynamics. To understand the origin of oscillations in the system, consider a finite-state machine with the topology of a ring oscillator in Fig. 37. The circuit is initiated with

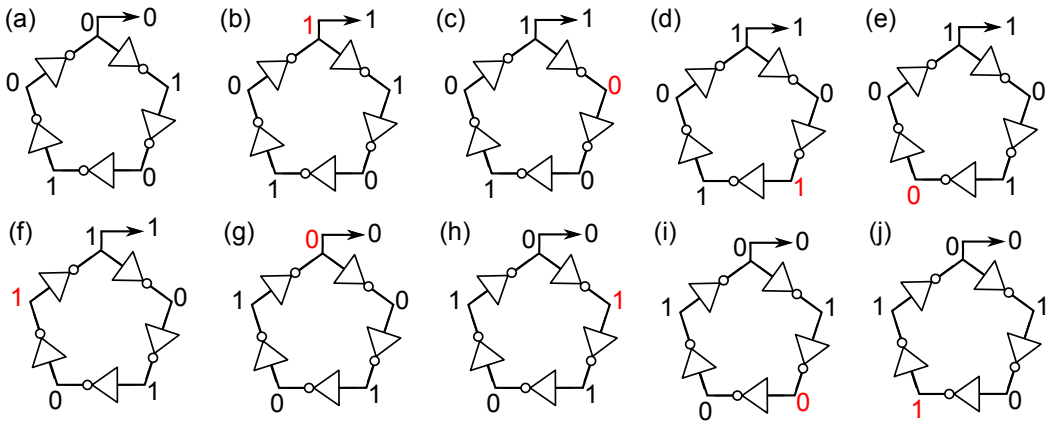


FIGURE 37: Illustration of the operation of a ring oscillator of five inverter gates with a measurement indicated by an arrow, the Boolean state of each node is indicated. (a)-(j) indicate ten timesteps in the synchronous operation.

Boolean values as shown in Fig. 37(a). In each of ten time steps, the Boolean states of each inverter logic gate are updated as the inverse of the input logic state. The state of one logic gate is measured and considered the output state of the synchronous ring oscillator as indicated by the arrow. After ten time steps, the state in Fig. 37(d) is reached, which leads, in an eleventh time step, to the initial state Fig. 37(a). The output state of the network indicates that the period in the synchronous operation is ten iterations. Within these ten timesteps, a change of Boolean state—a Boolean transition—has time to travel around the network twice.

In ring oscillators in the autonomous operation, similar dynamics are observed, where the period of the oscillation is twice the roundtrip time of a transition in the network given by the propagation time delay, according to Eq. (43). However, different from the discussion above, the experimental waveform in Fig. 36(b) includes a finite rise time, fluctuations on the Boolean states and jitter. In addition, when the ring oscillator consists of only one logic gate, the total delay is too short for the system to sustain stable periodic oscillations, similar to the discussion in Sec. 4.4.3.

6.2.5.2 *Phase of a Ring Oscillator*

The phase of ring oscillators can be calculated at the rising and falling transitions of the resulting periodic square wave [see Fig. Fig. 33(b)]. With the threshold value V_0 at mid-amplitude and an external reference waveform $V_{ref}(t)$ with the same period as the oscillator $T = T_{ref}$, the threshold-crossing times t_k and $t_{ref,k}$ can be calculated. With these, the relative phase $\phi = 2\pi \left(t_k/T - t_{ref,k}/T_{ref} \right)$ mod 2π reference to a reference waveform can be measured. Alternatively, the phase can also accessed continuously by representing the waveform with a series of sine waves via Fourier decomposition.

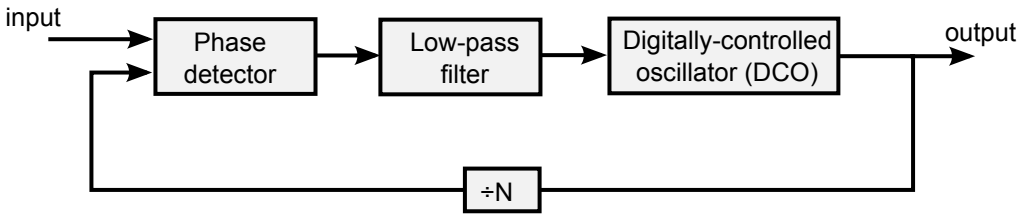


FIGURE 38: Illustration of functional blocks comprising a phase-locked loop (PLL).

6.2.5.3 *Phase-Locked Loops (PLLs)*

More sophisticated Boolean oscillators are all-digital PLLs. They are widely used in digital communication systems for frequency multiplication and clock synchronization, for example [Bes03].

PLLs comprise three functional blocks: (i) a phase-detector block, (ii) a filter block, and (iii) a controlled-oscillator block. These three blocks are assembled as shown in Fig. 38. In addition, an optional divided-by-N block is shown that can be used to scale the output frequency to multiples of the input frequency. The phase detector generates an error signal that is proportional to the phase difference between the output signal of the PLL and a reference signal. The error signal is filtered before being applied to the controlled-oscillator block, which adjusts its phase and frequency [Bes03].

Recent advances in digital electronic systems have spurred the development of all-digital PLLs, where all three blocks shown in Fig. 38 are realized with electronic logic circuits [Al-06]. The phase detector in all-digital PLLs typically generates N-bit numbers representing the phase shift between the input and output signals. The filter in all-digital PLLs is usually implemented with a digital up-down counter that integrates the signal.

While there has been much effort in improving the locking performance of PLLs, my goal in the Sec. 6.4 is to develop a very simple, resource-efficient PLL design that allows me to create large networks.

In the following section, I discuss how ring oscillators can be coupled and synchronized.

6.3 COUPLING OF MODIFIED RING OSCILLATORS

In this section, I couple multiple ring oscillators (considered dynamical nodes) to form a meta-network of autonomous Boolean networks. From now on, I refer to ring oscillators as nodes and meta-networks connecting them simply as networks. The nodes are almost identical logic circuits

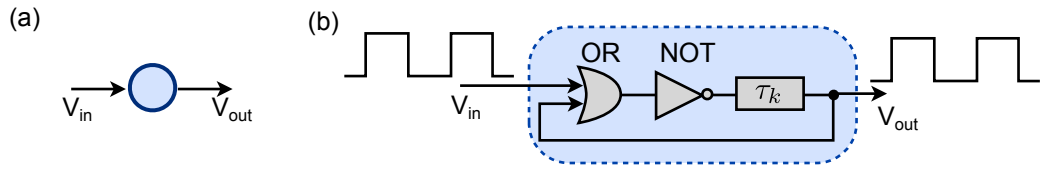


FIGURE 39: Setup of Boolean oscillators constructed from an OR gate, an inverter gate and a delay line that is separately constructed from k inverter gates. Each oscillator has a single input and an output.

schematically shown as a circle in Fig. 39(a). The small differences between nodes originate from heterogeneity in the parameters of logic gates, such as the propagation delay, as discussed in Sec. 3.2.3.

To couple ring oscillators, I first consider the addition of a 2-input OR logic gate that allows me to apply a Boolean input signal to the ring oscillator, as shown in Fig. 39(b) (see Appendix B.4 for the hardware description). In the figure, the delay line is realized with a chain of k inverter logic gates. The total delay in the loop is given by the delay of all components including the OR gate and the inverter gate.

The free-running frequency of each oscillator is given by Eq. (43) with feedback delay $\tau = (k+2)\tau_{\mathrm{LG}}$, where k is the (even) number of inverter gates in the delay line and the addition of 2 accounts for the additional inverter gate and the OR logic gate. I assume that the propagation delay is approximately the same for inverters and OR logic gates $\tau_{\mathrm{LG}} = 0.28 \pm 0.1\,\mathrm{ns}$ (see also Appendix A.4).

6.3.1 UNIDIRECTIONAL COUPLING OF MODIFIED RING OSCILLATORS

In this section, I couple unidirectionally two modified ring oscillators, denoted by (m) and (s) for master and slave oscillator, respectively, as shown in Fig. 40(a) and (b). Both modified ring oscillators are realized with an identical number of inverter logic gates $n_m + 1 = n_s + 1 = 21$; with frequencies $f_m = (92.1 \pm 0.9)\,\mathrm{MHz}$ and $f_s = (87.5 \pm 1.2)\,\mathrm{MHz}$ measured for the uncoupled oscillators, respectively. This difference in the free-running frequencies is due to the additional OR gate in (s) included to realize the coupling and also due to heterogeneity in the gate delay.

When the coupling is turned on, phase- and frequency-locking is achieved with frequency $f_m = f_s = (92.2 \pm 0.1)\,\mathrm{MHz}$, as illustrated in Fig. 40(c). Further confirmation of phase synchronization is given in Fig. 40(d), where the phase portrait $(V_m(t), V_s(t-\tau^*))$ shows a straight line with slope of approximately one. I measure the quality of synchronization by computing the cross-correlation coefficient between V_m and V_s, which is $\rho_{V_m V_s} \approx 0.995$. A skew time $\tau^* \approx 225\,\mathrm{ps}$ is used to compensate for the additional propagation time of the OR gate, the difference in propagation

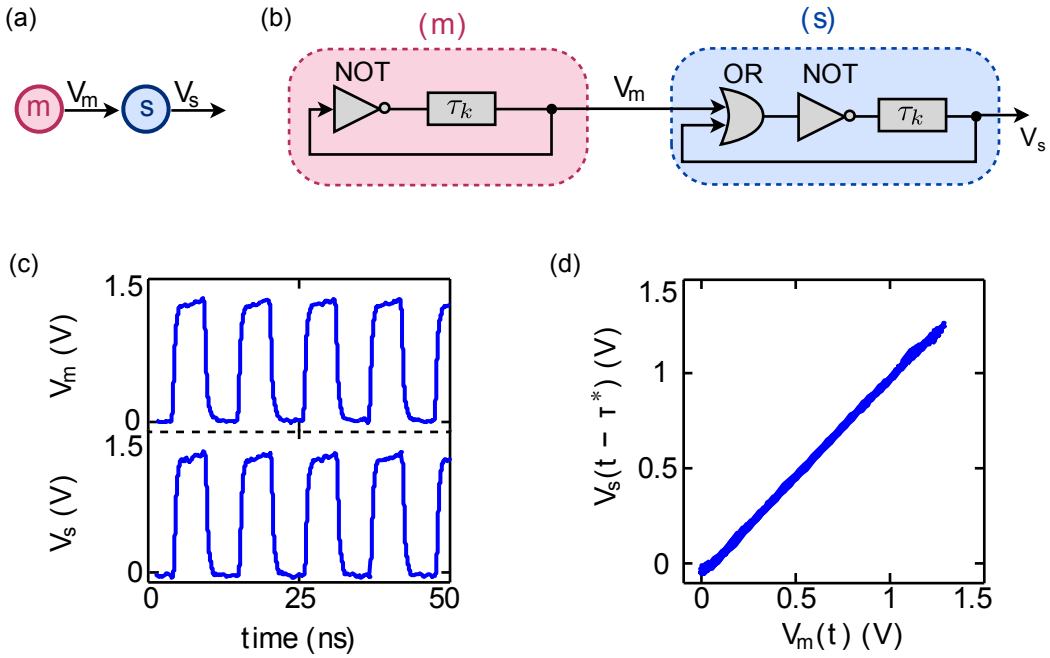


FIGURE 40: Experimental demonstration of unidirectional synchronization of modified ring oscillators. (a) Illustration and circuit diagram (b) of the setup with master (m) and slave (s) oscillators. (c) Temporal evolutions of the oscillators (m) and (s) showing in-phase square-wave oscillations with period $T_{m,s} = 10.9 \pm 0.4\,\text{ns}$. (d) Evolution in phase plane $(V_m(t), V_s(t-\tau^*))$. The time series are acquired with a high-speed oscilloscope (DSO80804A) with 8 GHz bandwidth and 40 GSa/s sampling rate.

time of the two signals to the output port of the FPGA to the oscilloscope, and for a small propagation delay in the coupling.

The stable phase-locked dynamics corresponds to one Boolean transition propagating in each oscillator with constant relative phase shift. The OR gate used in (s) leads to the creation of a Boolean transition in (s) whenever (s) is in the V_L state and (m) generates a Boolean transition ($V_L \to V_H$ or $V_H \to V_L$ with the high and low Boolean voltages V_H and V_L, respectively). This implies that multiple transitions can potentially propagate in (s) if (m) and (s) are not phase locked. However, the lowpass filter transfer function of logic gates in (s) likely has a higher gain at lower frequencies, so that an oscillatory state with lower frequency is preferred [Sas82]. In addition, the choice of an OR gate for the coupling of ring oscillators prevents an accumulation of Boolean transitions in (s) because, if one of the two inputs is in V_H, then a Boolean transition in the other input has no influence on the output of the OR logic gate.

6.3.2 MUTUAL COUPLING OF MODIFIED RING OSCILLATORS

With the modified ring architecture, I can also couple two ring oscillators bidirectionally, as illustrated in Fig. 41(a) and (b), with a flexible choice of

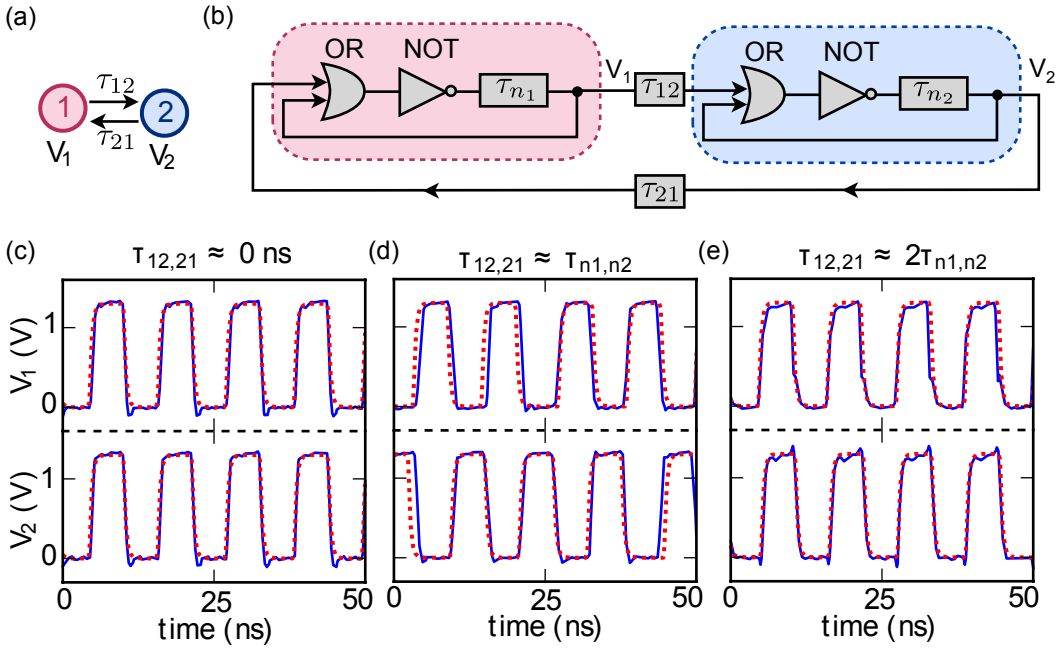


FIGURE 41: Illustration of bidirectional coupling of two modified ring oscillators. (a) Topology and (b) circuit diagram of two coupled Boolean oscillators labeled (1) and (2) built with $n_1 = n_2 = 21$ inverter gates and coupled by two links with time delays $\tau_{12} \approx \tau_{21}$. (c) Temporal evolution of both Boolean oscillators showing in-phase square-wave oscillations with identical period $T_1 = T_2 = 10.7 \pm 0.4\,\text{ns}$ for $\tau_{ij} = 0\,\text{ns}$. The time delay is due to the on-chip wires connecting the two oscillators and is small ($\tau_{12} \approx \tau_{21} \approx 0$). (d), (e) Temporal evolution for the oscillators with $\tau_{12} \approx \tau_{21} \approx \tau_{n_1} \approx \tau_{n_2}$ ($\tau_{12} = 6.2\,\text{ns}$, $\tau_{21} = 6.5\,\text{ns}$) and $\tau_{12} \approx \tau_{21} \approx 2\tau_{n_1} \approx 2\tau_{n_2}$ ($\tau_{12} = 11.7\,\text{ns}$, $\tau_{21} = 11.05\,\text{ns}$), respectively. The blue solid lines show the experimental time series. The red dotted lines show the dynamics of x_{buf1} and x_{buf2} from numerical simulation of Eqs. (44)-(46) with $\tau_1 = \tau_2 = 5.4\,\text{ns}$ and mutual time delays τ_{12}, τ_{21} as stated above. The dimensionless quantities x_{buf1} and x_{buf2} are scaled in amplitude and time ($V_{1,2} \to x_{buf1,2} V_H$ and $t \to t T_{\text{rise}} / \ln(2)$, with $V_H = 1.3\,\text{V}$ and $T_{\text{rise}} = 0.26\,\text{ns}$).

the coupling time delays τ_{12} and τ_{21} and identical constructions of the two oscillators (hardware description shown in Appendix B.5).

When the coupling time delays are negligible $\tau_{12} \approx \tau_{21} \approx 0\,\text{ns}$, the two oscillators are synchronized in phase, as shown in Fig. 41(c) with the frequency of each oscillator being slightly pulled from their respective free-running frequencies $f_1 = (81.9 \pm 0.7)\,\text{MHz}$ and $f_2 = (87.54 \pm 0.7)\,\text{MHz}$ to a common frequency $f = (87.7 \pm 0.7)\,\text{MHz}$.

The synchronization patterns change when time delays along the links are included. The two ring oscillators display either in-phase or anti-phase synchronization depending on the coupling time delays τ_{12} and τ_{21} with respect to the period of the oscillators $T_1 \approx 2\tau_{n_1}$ and $T_2 \approx 2\tau_{n_2}$. Experimentally, it is found that a relation of coupling delays $\tau_{12} \approx \tau_{21} \approx p\tau_{n_1} \approx p\tau_{n_2}$ with $p \in \mathbb{N}$ even (odd) lead to a synchronization state, where the two oscillators are in-(anti-)phase synchronized. To illustrate this, the tempo-

ral evolution of each oscillator is shown for $\tau_{12} \approx \tau_{21} \approx \tau_{n_1} \approx \tau_{n_2}$ and $\tau_{12} \approx \tau_{21} \approx 2\tau_{n_1} \approx 2\tau_{n_2}$ in Fig. 41(d) and (e), respectively.

The experimental result on mutual synchronization reminds of phase synchronization states predicted theoretically for two coupled Kuramoto oscillators with time-delay feedback loops and links [Dhuo8]. In this reference, however, the periodic oscillator can oscillate without the presence of time-delayed feedback, which is not the case for the modified ring oscillator—without the time-delayed feedback, the modified ring oscillator reduces to an OR and a NOT gate with a fixed Boolean state. Similar behavior has been observed numerically for two delay-coupled FitzHugh-Nagumo systems, each of which is in the excitable regime, *i.e.*, does not exhibit self-sustained oscillations in the uncoupled case as discussed in Sec. 8.2.4.

6.3.3 MODEL FOR MODIFIED RING OSCILLATORS

The autonomous Boolean network of the ring oscillator modified with an OR gate for the coupling can be modeled in various ways. Here, I use a similar modeling framework as used for chaotic dynamics in Sec. 4.4.2 and introduced in general in Sec. 2.3.3. Specifically, I describe the system with a piecewise-linear switching model developed by Glass and collaborators and extended with a feedback delay [Gla98, Mes96]. I compare the dynamics measured from the experiment with the simulation results.

The differential equations for the network in Fig. 41(b) with continuous and Boolean states x_i and X_i (see Secs. 2.3.3 and 4.4.2) are

$$\dot{x}_1 = -x_1 + \mathrm{NOR}\left[X_1(t-\tau_1), X_2(t-\tau_{2,1}-\tau_2)\right], \tag{44}$$

$$\dot{x}_2 = -x_2 + \mathrm{NOR}\left[X_2(t-\tau_2), X_1(t-\tau_{1,2}-\tau_1)\right], \tag{45}$$

$$\dot{x}_{buf1,2} = -x_{buf1,2} + X_{1,2}(t), \tag{46}$$

with two inverted OR (NOR) Boolean functions NOR: $\{0,1\} \times \{0,1\} \to \{0,1\}$ and delayed feedback according to the two OR gates with consecutive inverter gates and two logic gate-based delay lines. The time delays originate from chains of consecutive inverter gates in the setup (τ_1, τ_2, τ_{12}, τ_{21}). The third equation describes the temporal evolution of two buffer logic gates x_{buf1} and x_{buf2} that perform the Boolean identity operation on $X_1(t)$ and $X_2(t)$; the buffer gates correspond to output gates on the FPGA. Different from my consideration in Sec. 4.4.2, I have modeled the system with dimensionless equations and have included dimensions afterwards in the graphical illustration 41.

In Fig. 41(c)-(e), the dotted red line denotes the solutions obtained from the model for x_{buf1} and x_{buf2} by evolving the analytical solution of the piecewise linear differential equations between the switching of the NOR Boolean function, similar to Ref. [Gla98]. Apart from a low level of amplitude noise and jitter in the experiment, which are both not included in

the model, the dynamics generated by the model agrees well with the experiment. Both display similar waveforms, rise times, and periodicity of the oscillations. The discrepancy between model and experiment can be quantified via differences in timing of transitions, which is a common measure in autonomous Boolean systems [Zhao9a], and amounts to average values of 0.20 ns, 0.94 ns, and 0.49 ns for the waveforms in Fig. 41(c)-(e), respectively. The error is small in comparison to the oscillation period of $T = 10.7 \pm 0.4$ ns.

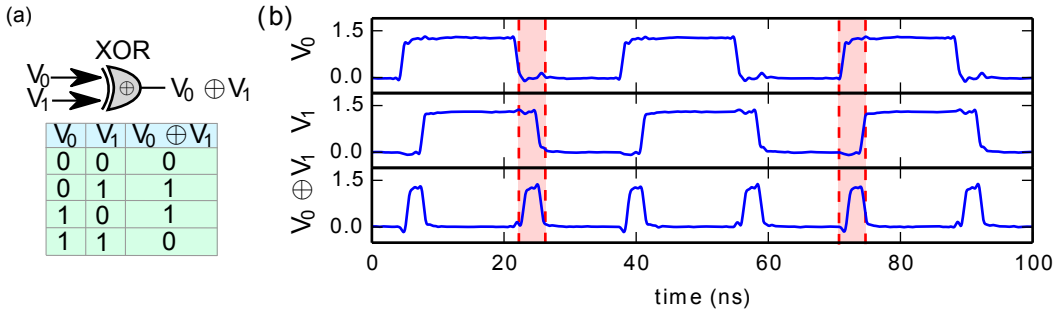
6.3.4 DISCUSSION

I have shown that the framework of experimentally-realized autonomous Boolean networks can be used to realize coupled dynamical systems with periodic dynamics. Specifically, I have realized periodic oscillators, which are autonomous Boolean networks themselves, in simple network motifs of two coupled oscillators and have observed phase synchronization.

A limitation of this approach, however, is the realization of adjustable coupling. The coupling in the design of ring oscillators with OR logic gates is all-or-nothing because an incoming transition determines the dynamics of the oscillator when the second input to the OR logic gate is in the low Boolean state. A manifestation of such a strong coupling is synchronization that is achieved for wide ranges of driving frequencies, as discussed in Sec. 6.2.3. On the other hand, a weak coupling with an adjustable strength is required to see other interesting network dynamics such as chimera states [Kuro2a, Abro4]. This motivates the next section, where I develop an autonomous logic circuit with adjustable coupling strength even when exchanging only Boolean signals.

6.4 BOOLEAN OSCILLATORS WITH VARIABLE COUPLING STRENGTH

In this section, I propose and study a periodic Boolean oscillator that can be coupled with an adjustable coupling strength. The periodic oscillator is termed Boolean phase oscillator to highlight its similarity to Kuramoto phase oscillators introduced in Sec. 7.2.1 due to similar dynamical equations, as derived in Sec. 7.3.2. The design is based on a modification of all-digital PLLs introduced in Sec. 6.2.5.3 and relies on state-dependent delay. The delay is realized with cascaded copier gates instead of cascaded inverter gates, which has the effect of a lowpass filter required in typical PLL designs (see Sec. 6.2.5.3). Without a lowpass filter, high frequency modes can be excited [Sas82]. The reason for the lowpass filter effect of buffers is large pulse growth in cascaded buffer gates compared to cascaded inverter gates, as shown in Appendix A.3.



V_0	V_1	$V_0 \oplus V_1$
0	0	0
0	1	1
1	0	1
1	1	0

FIGURE 42: (a) Symbol of an XOR logic gate that is used as a one-bit phase detector and its associated look-up table. The low (V_L) and high (V_H) Boolean voltage are denoted by symbols "0" and "1" respectively. (b) The error signal $V_0 \oplus V_1$ resulting from the phase detector with two phase shifted input signals V_0 and V_1 of frequency $f_0 = f_1 = 30.0 \pm 0.1$ (MHz). The phase difference is highlighted by shaded regions at a falling and a rising Boolean transition. The waveform is obtained from an experimental implementation on a field-programmable gate array (FPGA), specifically the model Altera Cyclone IV EP4CE115F29C7, which is used for all experimental implementations in this paper.

6.4.1 BOOLEAN PHASE OSCILLATOR

The design of the Boolean phase oscillator follows the structure of three functional blocks: Phase detector, low-pass filter, and controlled oscillator as shown in Fig. 38. For the phase detection block, I use a two-input XOR logic gate that is a single-bit digital phase detector [see Fig. 42(a)] [Bes03]. The waveform of the resulting error signal V_c is shown in Fig. 42(b) for two Boolean input waveforms of different phase and equal frequency. The error signal satisfies $V_c = V_H$ (Boolean signal "1") when the two input signals have different Boolean values and $V_c = V_L$ (Boolean signal "0") otherwise. This results in a pulse-shaped waveform at the output of the phase detector with a pulse width proportional to the phase difference between the two inputs as highlighted in the figure. Note that the XOR function does not give the sign of the phase difference.

The second block, the filter, is not explicitly implemented in my design. Nevertheless, each logic gate low-pass filters intrinsically the voltage generated by the phase-detection block with a cutoff frequency related to the gate propagation delay $\tau_{LG} = 0.28 \pm 0.01$ ns.

Third, the controlled-oscillator block is based on a simplified design proposed in Ref. [Hsu99] and consists of an inverter gate with a state-dependent feedback delay. Specifically, it is realized using one inverter gate, two series of $n \in \mathbb{N}$ and $n - k \in \mathbb{N}$ cascaded buffer gates, and a Boolean switch as shown in Fig. 43(a) and (b). In the resulting delayed feedback system with negative gain, the dynamics has a periodicity equal to twice the feed-

back delay because it requires two inversions to recover the initial Boolean state [Kat98]. Therefore, the frequency of oscillation is

$$f(V_c) = \frac{1}{2\tau(V_c)}, \tag{47}$$

where $\tau(V_c)$ is the state-dependent feedback delay that depends on the control signal V_c. The cascaded buffer gates implement two feedback delays $\tau_{n-k} = (n-k)\tau_{\mathrm{LG}}$ and $\tau_k = k\tau_{\mathrm{LG}}$. The feedback lines can also be realized with cascaded inverter gates, but the choice of buffer gates results in enhanced low-pass filtering (see Section III.B for a detailed explanation). The switch, implemented as a three-input logic gate with the look-up table of a multiplexer, selects between the two feedback delays τ_{n-k} and $\tau_n = \tau_{n-k} + \tau_k$ depending on the control signal V_c, which results in a state-dependent feedback delay of

$$\tau(V_c) = \begin{cases} \tau_n = n\tau_{\mathrm{LG}} & \text{if } V_c \le V_{th}, \\ \tau_{n-k} = (n-k)\tau_{\mathrm{LG}} & \text{otherwise.} \end{cases} \tag{48}$$

Here, the integer $k \approx (\tau_n - \tau_{n-k})/\tau_{\mathrm{LG}}$ is proportional to the difference of the two possible values of the feedback delay. The system will oscillate at a free-running frequency of $f_n = 1/2\tau_n$ or $f_{n-k} = 1/2\tau_{n-k}$ when a constant control voltage of $V_c = V_L$ or $V_c = V_H$ is applied to the controlled oscillator, respectively.

The dynamics of the controlled-oscillator block is first characterized by sending an external periodic signal of frequency $f_c = 60.0 \pm 0.1\,\mathrm{MHz}$ to the V_c port. Figure. 43(c) shows the external waveform, the resulting output

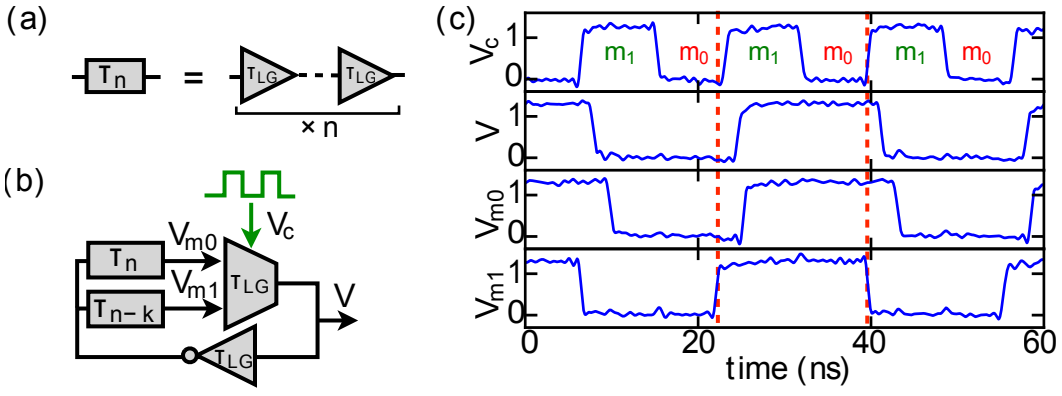


FIGURE 43: (a) Construction of delay line τ_n with a series of n cascaded buffers with individual propagation time τ_{LG}. (b) Schematic of the controlled-oscillator block of the Boolean phase oscillator. (c) Locking of the controlled-oscillator block to an externally-generated control signal V_c with frequency $f = 60.0 \pm 0.1\,\mathrm{MHz}$. Also shown are the output waveform V and the signals from the delay lines that are input to the switch V_{m0} and V_{m1}. The parameters of the controlled-oscillator block are $n = 65$ and $k = 10$. The gate propagation delay as measured for buffer gates is $\tau_{LG} = 0.275 \pm 0.010\,\mathrm{ns}$

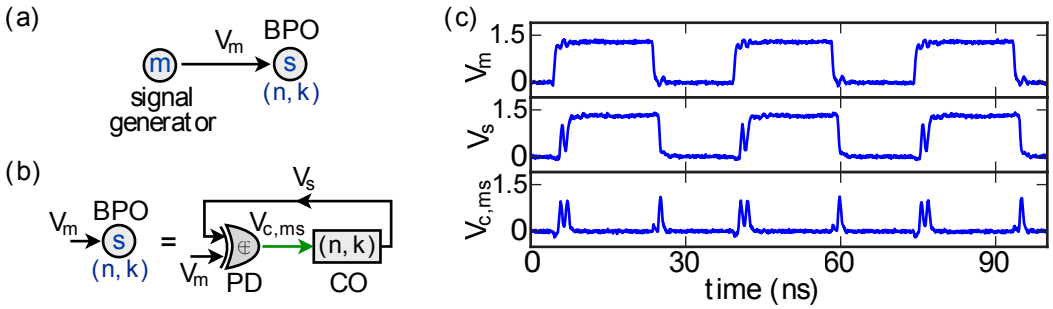


FIGURE 44: (a) Experimental setup of the master-slave coupling scheme for two Boolean phase oscillators. (b) Construction of the slave Boolean phase oscillator comprising the controlled-oscillator block labeled (n,k) [see Fig. 43] and an XOR-based phase detector (PD) block. (c) Waveforms of the master oscillator (generated externally with waveform generator Tektronix AFG3251) V_m of frequency $f_m = 28.6 \pm 0.1\,\mathrm{MHz}$, the output waveform of the slave oscillator V_s, and the error signal $V_{c,ms} = V_m \oplus V_s$. The parameters of the controlled-oscillator block in the Boolean phase oscillator are the same as the ones in Fig. 43.

voltage of the controlled-oscillator block $V(t)$, and the output voltages of the two delay lines $V_{m0}(t)$ and $V_{m1}(t)$. These two signals are time shifted by the delay τ_k so that $V_{m1}(t) = V_{m0}(t + \tau_k)$. In the figure, the output voltage is $V(t) = V_{\mathrm{m0}}(t)$ $[V(t) = V_{\mathrm{m1}}(t)]$ when $V_c = V_{\mathrm{L}}$ $[V_c = V_{\mathrm{H}}]$, which is due to the functionality of the switch in the setup. As a result, a rising-edge Boolean transition in V_c [highlighted with dashed lines in Fig. 43(c)] leads to a positive phase shift, which triggers rising and falling edges in the output voltage $V(t)$. Therefore, switching between the two feedback delays leads to frequency locking. For more details, see Ref. [Hsu99].

6.4.2 UNIDIRECTIONAL SYNCHRONIZATION OF BOOLEAN PHASE OSCILLATORS AND WEAK COUPLING ANALOGY

By combining the phase detector from Fig. 42(a) with the controlled-oscillator block from Fig. 43(b), I obtain the Boolean phase oscillator with hardware description in Appendix B.6.1. I use the Boolean phase oscillator in the rest of this chapter to study synchronization phenomena

I first test the locking capabilities of the Boolean phase oscillator with an external driving signal of frequencies f_m. This setup, shown in Fig. 44(a) and (b), can be interpreted as a master-slave configuration, where two Boolean phase oscillators are coupled unidirectionally. Here, the equivalent master Boolean phase oscillator is an external function generator, which provides finer frequency control of f_m than can be obtained by adding or removing buffers in the ring of the master Boolean phase oscillator.

In Fig. 44(c), I show the dynamics resulting from the master-slave configuration. Specifically, I show the phase-locked waveforms of the master and slave oscillator with frequencies $f_m = f_s = 28.6 \pm 0.1$ MHz, and the error signal V_c. Here, the dynamics of the Boolean phase oscillator leads to a constant phase shift between master and slave oscillator that results in a pulsed signal V_c [similar to Fig. 42(b)]. The pulses in V_c provide the phase correction that allows synchronization. The waveforms of V_s and V_c display fast oscillations at the Boolean transitions, which are due to unfiltered feedback between the phase detector and the control port of the controlled-oscillator block.

I repeat the coupling experiment by tuning the frequency of the master oscillator f_m from 20 to 105 MHz while keeping the parameters of the slave Boolean phase oscillator unchanged. I measure f_s and calculate the ratio $f_\mathrm{m}/f_\mathrm{s}$ as a function of f_m. This measurement leads to the so-called *devil's staircase* shown in Fig. 45(a) [Rei99, Pik01] (see also Sec. 6.2.3). In the graph, synchronization regions with constant ratios $f_\mathrm{m}/f_\mathrm{s}$ are represented by horizontal plateaus—the *stairs*. The most prominent synchronization regions are associated with integer ratios $f_\mathrm{m}{:}f_\mathrm{s} = p{:}q$ with $q = 1$ and $p = 1, 2, 3$. In the synchronization region 2:1, $f_\mathrm{m}/f_\mathrm{s} \neq 2$ for a narrow range of f_m. This imperfection appears also in numerical simulations as discussed below. Moreover, many narrow fractional synchronization regions $p{:}q$ exist.

The widths of the synchronization regions—the stairs—are known as the locking range U [Hsu99]. The maximum theoretical locking range U_{max} can be determined, so that $U \subseteq U_{max}$ with $U_{max} = [f_n, f_{n-k}]$ for the first integer synchronization region 1:1 with free-running frequencies of the Boolean phase oscillator f_n and f_{n-k}. Then, the locking range is given by $f_m \in U_1 \subseteq [f_n, f_{n-k}]$. For integer synchronization regions of higher order p:1, the frequency locking of master and slave is given by $f_m = pf_s$. This leads to larger synchronization regions of $U_p \subseteq [pf_n, pf_{n-k}]$. In Fig. 45(a), I display the theoretical boundaries $[pf_n, pf_{n-k}]$ with dashed lines. The figure shows that the right theoretical locking boundaries are not reached in the experiment.

The non-saturation of the right theoretical boundaries in the p:1 synchronization region originates from the generation of the control signal $V_{c,ms}$ with little frequency filtering. Specifically, the right boundary corresponds to large detuning between the frequency of the master oscillator f_m and and the free-running frequency of the slave oscillator f_n. Then, the control signal $V_{c,ms}$ displays the Boolean voltage V_H for an increased time to allow for greater frequency adjustment. However, this also results in increased high-frequency oscillations due to unfiltered feedback in the oscillator as discussed above and visible in Fig. 44(c). These high-frequency oscillations decrease the locking abilities when the frequency detuning is large.

To complete the analysis of the locking range, I also measure them as a function of k. In Fig. 45(b), I map out the integer synchronization regions

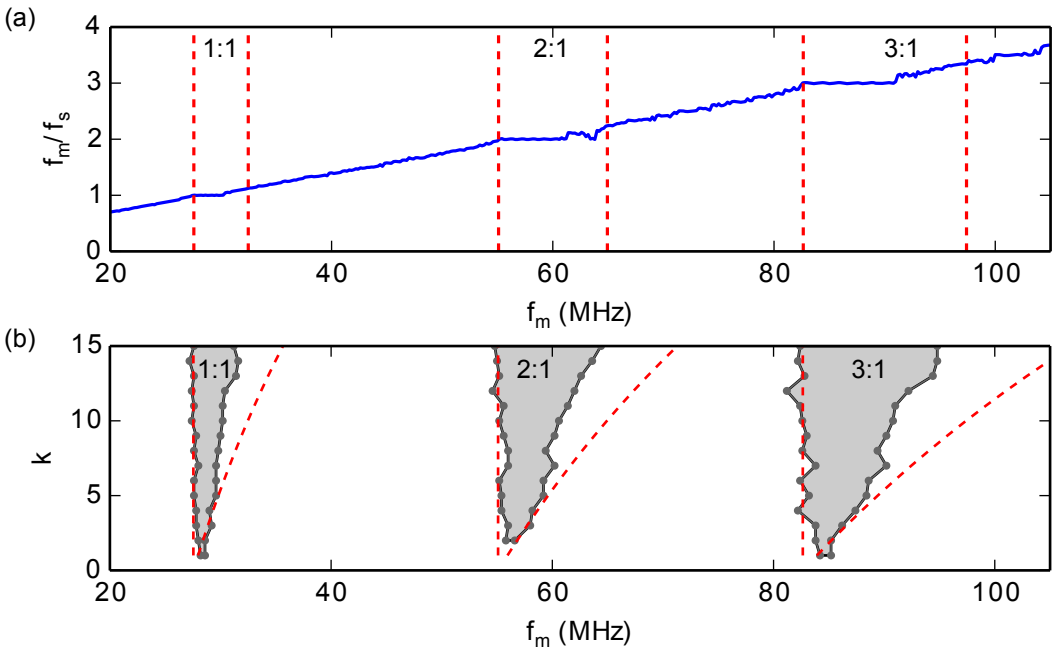


FIGURE 45: (a) Experimental Devil's staircase for a Boolean phase oscillator with $k = 10$ showing synchronization regions as horizontal lines. For the three integer synchronization regions f_m:f_s = 1:1, 2:1, and 3:1, the theoretical limits are shown with red dashed lines. (b) Experimental Arnold tongues for the three integer synchronization regions as a function of k and the theoretical limits of the regions marked with red dashed lines. Other Arnold tongues are not shown. Setup and experimental parameters as in Fig. 43.

and their analytic boundaries as a function of k and f_m. Here, the system parameter k is proportional to the range of values of the state-dependent delay $\tau_k = k\tau_{LG}$ in the controlled-oscillator block. The resulting synchronization regions, also known as *Arnold tongues* [Rei99], have a triangular shape that opens with increasing k (see also Sec. 6.2.3). The analytic boundaries corresponding to pf_n are shown with dashed lines. The Arnold tongues are subject to experimental variation of $\pm 3.5\%$ due to variations of gate propagation delays τ_{LG} that appear in the experiment when implementing the oscillators on different location on the FPGA for different values of k.

The increase of synchronization regions U_p with k allows me to draw an analogy of k to the coupling strength of phase oscillators because they display shapes of the synchronization regions that increase with the coupling strength [Pik01]. Therefore, I have shown that adjustable weak coupling is possible in the Boolean phase oscillator. Note that the Arnold tongues are asymmetric, which is unlike other models, such as the Kuramoto model [Kur84].

The mechanism to generate the coupling is via state-dependent delay as described in Section II.C. Specifically, the phase detector generates a high Boolean voltage $V_c(t) = V_H$ with a duration proportional to the phase dif-

ference between the input periodic signal—generated either by another oscillator or a signal generator—and the output signal of the Boolean phase oscillator. The signal $V_c(t)$ induces an increase in frequency of the Boolean phase oscillator because the feedback delay switches to a shorter value. The magnitude of the resulting frequency adjustment is proportional to k. Therefore, larger values of k lead to a larger frequency adjustment, which reduces the phase difference between the state of the Boolean phase oscillator and the external periodic signal. This is analogous to the phase-correction mechanism encountered in theoretical models of phase oscillators [Pik01].

6.4.3 MODEL FOR BOOLEAN PHASE OSCILLATOR

Similar to the model of ring oscillator in this chapter, I model Boolean phase oscillators with a piecewise-linear switching model developed by Glass and collaborators and extended with a feedback delay [Gla98, Mes96] that is introduced in Secs. 2.3.3 and 4.4.2. I also include 1) a state-dependent delay in the model that accounts for the switching between two delay lines in the controlled oscillator and 2) the filtering effect due to the use of buffer-based delay lines in the model (see Appendix A.3).

To model the master-slave setup, I describe the master oscillator simply by a periodic continuous square wave $y_m \in [-1, 1]$ with low and high Boolean values -1 and 1, respectively. These Boolean values can be scaled to correspond to the experimental Boolean voltages of $V_L = 0\,\mathrm{V}$ and $V_H = 1.3\,\mathrm{V}$. The delay differential equations for the slave oscillator are

$$\tau_{x_s}\dot{x}_s = -x_s + [\neg X_s(t - \tau_s(t))], \tag{49a}$$
$$\tau_{y_s}\dot{y}_s = -y_s + [\neg X_s(t - \tau_s(t))], \tag{49b}$$

where (x_s, X_s) are continuous and Boolean internal variables of the slave Boolean phase oscillator (see Sec. 2.3.3), (y_s, Y_s) are the variable associated with the output of the slave Boolean phase oscillator, $\neg X = -X$ denotes the Boolean inversion (NOT) operation, τ_{x_s}, τ_{y_s} are the characteristic timescales associated with first-order low-pass filtering, and $\tau_s(t)$ is a state-dependent delay that I discuss in detail below.

Equation (49a) represents the low-pass filtering effect that results from the construction of the two delay lines with cascaded buffer gates. The characteristic timescale τ_{x_s} associated with the filtering is different for rising and falling transitions when measured for the CMOS-based logic gates [Wes00]. To account for this behavior, I include the following state-dependent switching condition depending on rise and fall

$$\tau_{x_s} = \begin{cases} \tau_{\mathrm{LG}}/\ln 2 & \text{if } x_s(t - \tau_s(t)) > 0, \\ (\tau_{\mathrm{LG}}/\ln 2)\left(1 + n\Delta\tau_{rf}/\tau_{\mathrm{LG}}\right) & \text{if } x_s(t - \tau_s(t)) \leq 0, \end{cases} \tag{50}$$

with $\Delta\tau_{rf} = 24 \pm 2\,\mathrm{ps}$ the time difference between rising and falling transitions of a single logic gate. The numeric value for $\Delta\tau_{rf}$ is measured by

propagating a periodic square pulse through n cascaded buffer gates that constitute a delay line. The measured output signal is changed according to Eq. (50) resulting in a alteration of time difference between falling and rising transitions by $\tau_{\mathrm{LG}}/(\ln 2)(n\Delta\tau_{rf}/\tau_{\mathrm{LG}})$. The accumulation of $\Delta\tau_{rf}$ produced by each logic gate results in pulse growth and enhanced low-pass filtering as described in appendix A.3.

The dynamics of the switch in the controlled oscillator is modeled with state-dependent delay according to the following condition

$$\tau_s(t) = \begin{cases} \tau_n & \text{if } y_{c,ms}(t) \leq 0, \\ \tau_{n-k} & \text{otherwise,} \end{cases} \tag{51}$$

where $y_{c,ms}(t) = y_s(t - \tau_{c,ms}) \oplus y_m(t - \tau_{c,ms})$. The additional delay $\tau_{c,ms} = 2\tau_{\mathrm{LG}}$ accounts for the time that it takes for V to propagate through the phase detector and reach the control port of the controlled-oscillator block [see also Fig. 44(b)]. As a result, the delay switches between two values depending on the error signal $y_{c,ms}$ generated by the XOR-based phase detector [see Fig. 42(a) for the look-up table of the XOR operation].

In contrast to Eq. (49a), Eq. (49b) has no rising-falling asymmetry associated with the filtering as $\tau_{y_s} = \tau_{\mathrm{LG}}/\ln 2$ is constant. This is because its associated experimental voltage V is measured after the switch and not after the delay lines that mainly contribute to the asymmetry.

Figure 46(a) and (b) illustrate graphically the model of the Boolean phase oscillator in master-slave configuration. It is shown that the state-dependent delay of the slave oscillator is driven via the error signal $y_{c,ms}$, which is implicitly included in Eq. (51) as an XOR operation of the signal y_s and the output of the master oscillator y_m. The resulting variable x_s is filtered by two different low-pass filters with and without asymmetric rise and fall times. Negative feedback results from the inverter gate.

The waveforms generated by numerical simulation of the model are shown in Fig. 46(c). The control signal $y_{c,ms}$ is obtained by low-pass filtering, similar to Eq. (49b), the Boolean expression $Y_s(t - \tau_{c,ms}) \oplus Y_m(t - \tau_{c,ms})$ where $Y_{m,s}$ are the Boolean variables associated to continuous variables $y_{m,s}$. The model reproduces qualitatively the experimental measurements shown in Fig. 44(b). For example, it exhibits similar fast oscillations at the rising edge of output voltage of the slave Boolean phase oscillator. With the model, I can also generate the devil's staircase and Arnold tongues as shown in Fig. 45(a) and (b). I notice that the simulations and experiments agree quantitatively [compare to Fig. 45(a) and (b)]. The widths of the synchronization region p:1 as a function of k differ only by 14% on average. Note that the model does not include fitted parameters, but all parameters are measured using independent experiments.

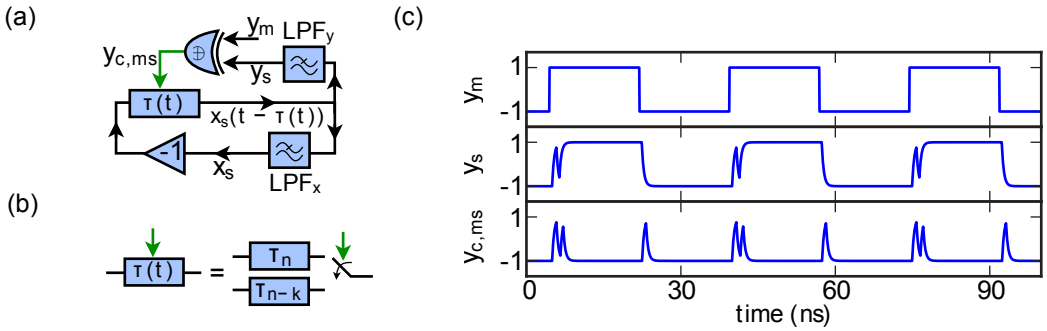


FIGURE 46: (a) Block representation of the mathematical model for the master-slave setup. LPF$_{x,y}$ are low-pass filters with time constants τ_x and time constant τ_y, respectively. (b) Illustration of the state-dependent delay. (c) Numerical simulation with parameters as in Fig. 43. The variables y_m, y_s, and $y_{c,ms}$ are the model representations of the experimental voltages V_{m}, V_{s}, and $V_{\mathrm{c,ms}}$ in Fig. 44(b). The parameters are the same as in the experiment, see Fig. 43.

6.4.4 SYNCHRONIZATION IN A BIDIRECTIONAL COUPLING CONFIGURATION

To analyze the synchronization properties of Boolean phase oscillators further, I now consider bidirectional coupling. Figure 48(a) and (b) shows the experimental setup schematically. The oscillators are coupled symmetrically, *i.e.*, the coupling strength k is identical in both oscillators. The free-running frequency between the oscillators, on the other hand, is detuned by choosing parameters $n_1 = 65$ and $n_2 \in \{50, \ldots, 80\}$, where n_1 and n_2 are proportional to the feedback delay in the two oscillators. A difference of n_1 and n_2 of $\Delta n = n_2 - n_1$ results in detuning of the free-running frequency of the two oscillators of $\Delta f = \Delta n/(2\tau_{LG} n_1 n_2)$. Besides varying the frequency detuning, I also change the coupling strength from $k = 0$ to $k = 15$.

Following the theoretical framework proposed in the previous section, I model the bidirectional coupling setup by a four-dimensional system of coupled differential equations, according to

$$\tau_{x_{1,2}} \dot{x}_{1,2}(t) = -x_{1,2}(t) + [\neg X_{1,2}(t - \tau_{1,2}(t))], \tag{52a}$$

$$\tau_{y_{1,2}} \dot{y}_{1,2}(t) = -y_{1,2}(t) + [\neg X_{1,2}(t - \tau_{1,2}(t))], \tag{52b}$$

where $(x_{1,2}, X_{1,2})$ and $(y_{1,2}, Y_{1,2})$ are the pairs of continuous and Boolean variables describing the two coupled Boolean phase oscillators. Parameters $\tau_{x_{1,2}}$, $\tau_{y_{1,2}}$, $\tau_{1,2}(t)$ are analogous to τ_{x_s}, τ_{y_s}, $\tau_s(t)$ in the previous section.

The two Boolean phase oscillators are considered to be synchronized when the absolute value of the normalized beat frequency

$$f_b = |f_1 - f_2| \,/\, \sqrt{f_1^2 + f_2^2} \tag{53}$$

of the two Boolean phase oscillators is below 0.25%. Coupled oscillators are generally expected to synchronize under large coupling k and small

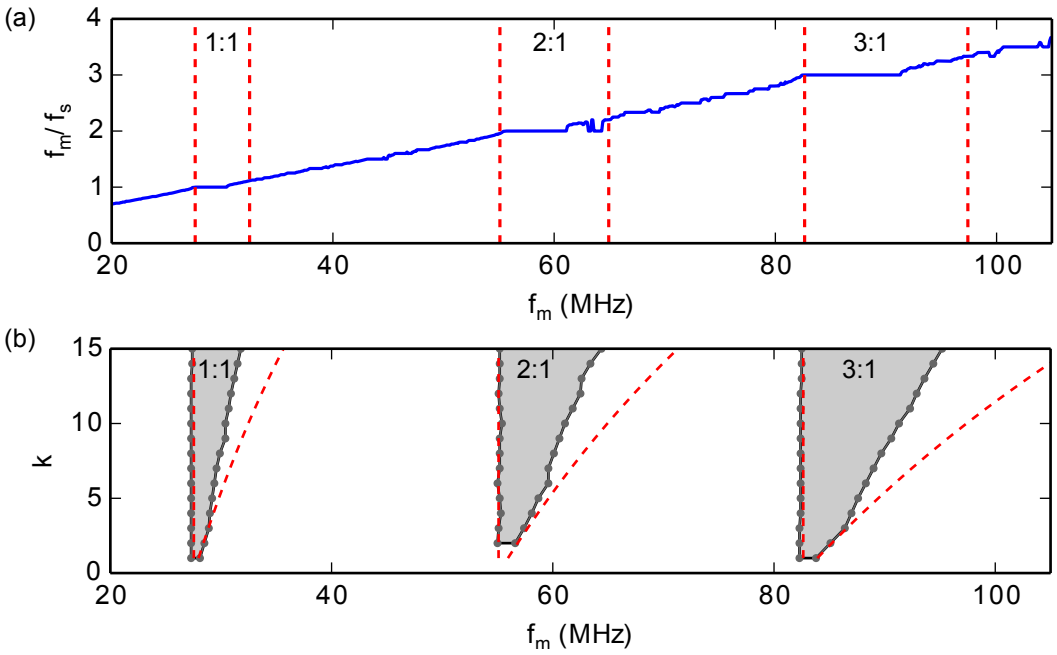


FIGURE 47: (a) Devil's stair case and (b) Arnold tongues from numerical simulations, similar to Fig. 45. The parameters are the same as in Fig. 43.

frequency detuning Δf [Pik01]. Here, I show that this general property also holds for Boolean phase oscillators.

I analyze the synchronization properties in parameter space $(\Delta n = n_2 - n_1, k)$ of frequency detuning and coupling strength. The experimental and numerical results are shown in Fig. 48(c) and (d). In both cases, the synchronization region (white area) is V-shaped; it is maximally extended for small values of Δn and large values of coupling strength k as expected for coupled oscillators. Experiment and simulation agree quantitatively. I find that the slope associated with the linear regression for the boundaries of the synchronization region differ approximately by 8%. Furthermore, outside of the synchronization region, simulation and experiment differ by less than 5%.

The border of the synchronization region can be approximated with a necessary condition on the coupling strength for synchronization. Specifically, the coupling strength has to exceed the detuning of the two Boolean phase oscillators, *i.e.*, $k \geq |n_2 - n_1|$. The resulting maximal border of synchronization $k = |n_2 - n_1|$ is shown in Fig. 48(c) and (d) with dashed lines. However, this condition is not sufficient to guarantee synchronization, similar to my considerations for unidirectional coupling. Therefore, the triangle delimited by the dashed lines is not entirely filled by the locking region. Interestingly, the analytic synchronization region with bidirectional coupling has a symmetric triangular shape different from the locking region for unidirectional coupling [see Fig. 45(b)].

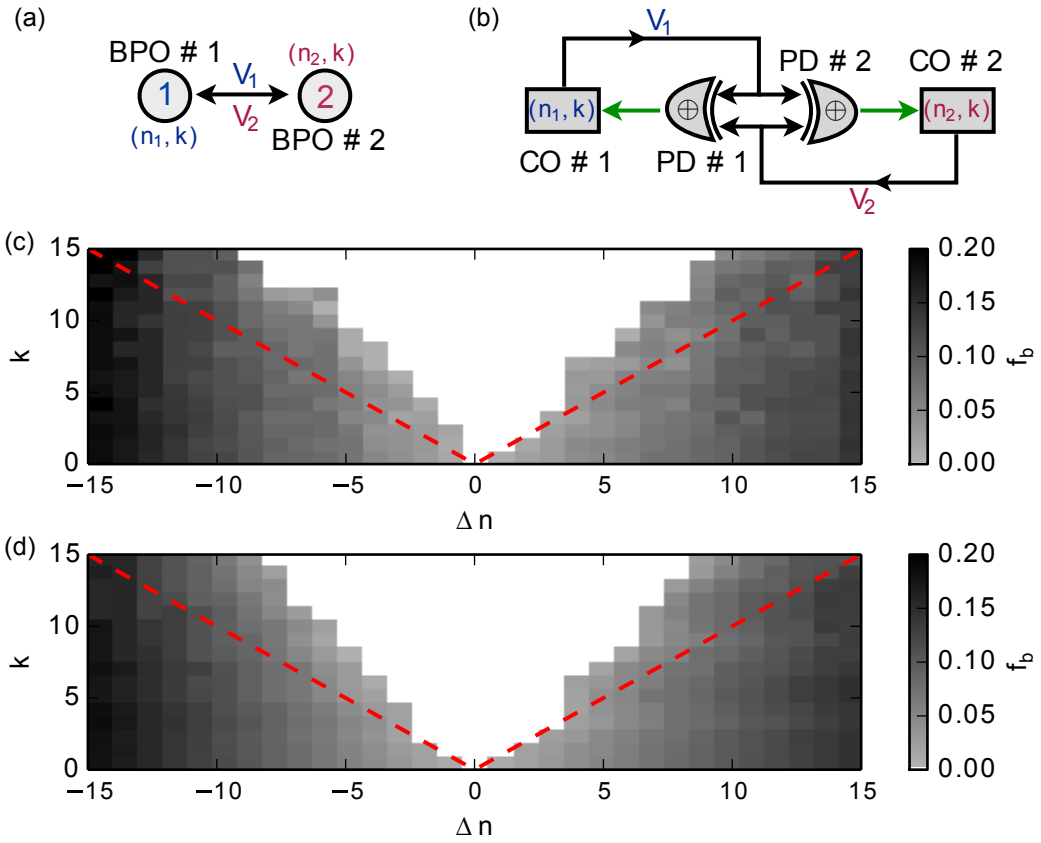


FIGURE 48: (a), (b) Topology and block diagram of the setup of the two mutually coupled oscillators. The coupling is realized with two directed links of equal strength k. (c) Experimental and (d) numerical synchronization domain in parameter space $(\Delta n, k)$ measuring the absolute value of the normalized beat frequency f_b for $\Delta n \in \{-15, \ldots, 15\}$, $k \in \{0, \ldots, 15\}$, and $n_1 = 65$ ($n_2 = n_1 + \Delta n$). The frequency is color-coded; white color indicates that the normalized beat frequency $f_b < 0.0025$. The red dashed lines indicate the analytically predicted synchronization boundary $k = |n_2 - n_1|$. The parameters are same as in Fig. 43.

6.4.5 DISCUSSION

I have proposed and studied a new dynamical system called Boolean phase oscillator that includes state-dependent delays and is a simplified version of previous designs of all-digital phase-locked loops (PLLs). The Boolean phase oscillator allows me to study experimentally various coupling topologies with variable coupling strength based on Boolean signals. I have observed and analyzed rich synchronization phenomena similar to those encountered in coupled phase oscillators, such as asymmetric and symmetric V-shaped synchronization regions in uni- and bidirectional coupling schemes of two coupled periodic oscillators, respectively. The Boolean phase oscillator also contributes to the study of state-dependent delay, which is a topic of great current interest [Walo3].

A limitation of the Boolean phase oscillator is, however, that its in-degree is restricted to one (to a single input), so that it can only be coupled to small network topologies. To study the very interesting network synchronization states discovered recently, such as chimera states [Kur02a, Abr04], the possibility of large in-degrees are needed. This necessary extension of the Boolean phase oscillator and the study of chimera states is the topic of the next chapter.

6.5 CONCLUSION

In this chapter, I have studied periodic dynamics in autonomous Boolean networks. After an introduction to previous work on periodic dynamical systems, I have designed and studied periodic oscillators based on autonomous Boolean networks called ring oscillators that are coupled via the inclusion of an OR logic gate. These systems allow only for strong coupling and the coupling strength cannot be adjusted. As a solution, I have developed and studied Boolean phase oscillators that includes weak, adjustable coupling, which is demonstrated by mapping out the synchronization regimes in different coupling topologies.

In the next chapter, I generalize the design of Boolean phase oscillators to allow for a high in-degree so that large, strongly-connected networks of Boolean phase oscillators can be built and used to study chimera states.

7

CHIMERA DYNAMICS IN NETWORKS OF BOOLEAN PHASE OSCILLATORS

7.1 ABSTRACT

In this chapter, I study Boolean phase oscillators in a large, heavily connected network and observe a network dynamics called the chimera state, where two dynamical groups of nodes coexist in the network, one synchronized and the other unsynchronized. Surprisingly, chimera states appear even when all oscillators are identical, hence representing a form of symmetry breaking. In Sec. 7.2, I first introduce previous work on chimera states in theoretical and experimental studies. Then, I extend the setup of Boolean phase oscillators in Sec. 7.3 to allow for large in-degrees, so that suitable network topologies can be realized, and show that the oscillators can be modeled similar to Kuramoto phase oscillators. In Sec. 7.4, I document complex dynamics that includes chimera states. I find in Sec. 7.5 that the complex dynamics is only a transient state similar to recent theoretical predictions in finite-size networks. I find that the complex transient includes chimera states for about 14% of the time for networks of $N = 128$ nodes and ends in a nearly synchronized state. I find that the transient time follows a Poisson process with an average transient time increasing exponentially with the network size, which is a result of the synchronization rate that follows a power law of the phase space volume. These findings are supported by numerical simulations as shown in Sec. 7.6.[1]

The main contributions of this chapter are:

- implementing large experimental networks of Boolean phase oscillators with non-local coupling;
- observing chimera states in experimental networks with all-physical coupling;
- identifying an exponential scaling of the transient time according to a Poisson process and identifying a power law relation between the average transient time and the phase space volume.

1 Results of this chapter are submitted for publication; the preprint can be found in Ref. [Ros14a].

7.2 INTRODUCTION TO CHIMERA STATES IN THEORY AND EXPERIMENT

The term chimera originates from Greek mythology, where it is a fire-breathing monster composed of three animals: lion, snake, and goat. Today, a chimera indicates fantastical things that are composed of different parts. The dynamical chimera state, so named by Abrams and Strogatz in 2004, was an unexpected discovery of complex dynamics in networks of identical oscillators by Kuramoto and Battogtokh in 2002 [Kuro2a, Abro4]. Prior to this work, it was believed that oscillator networks can only show interesting dynamics when the oscillator frequencies are heterogeneous [Pan14].

Chimera states were found first in numerical simulations of the Kuramoto model, which is a common mathematical model for large networks of coupled oscillators. In this section, I introduce the Kuramoto model, discuss theoretical studies of chimera states, their transient behavior in finite-size networks, and, finally, their realizations in laboratory experiments.

7.2.1 KURAMOTO MODEL

The Kuramoto model is motivated by the phenomenon of collective synchronization of coupled oscillators, which is ubiquitous in nature and technology, as discussed in Sec. 6.2.2 [Stroo]. It is based on the phase description of Eq. (35) in Sec. 6.2.3 with coupling depending on the difference of coupled oscillators [Kur84], according to

$$\dot{\phi}_i = \omega_i + \sum_{j=1}^{N} G_{ij} \sin(\phi_j - \phi_i). \tag{54}$$

Here, $i \in \{1, 2, ..., N\}$ is the oscillator index with the total number of oscillators N, the phase ϕ_i and free-running frequency ω_i of oscillator i, and the coupling strength G_{ij} that oscillator j imposes on i.

The Kuramoto model is a special case of Eq. (54), where the coupling strength is homogeneous for all links ($G_{ij} = K/N$), according to

$$\dot{\phi}_i = \omega_i + \frac{K}{N} \sum_{j=1}^{N} \sin(\phi_j - \phi_i). \tag{55}$$

I also assume here homogeneous frequencies $\omega_i = \omega_0$, which is a common assumption in the framework of chimera states. Then, I can redefine the phases $\phi_i \to \phi_i + \omega_0 t$, which is the rotating frame at frequency ω_0, resulting in

$$\dot{\phi}_i = \frac{K}{N} \sum_{j=1}^{N} \sin(\phi_j - \phi_i). \tag{56}$$

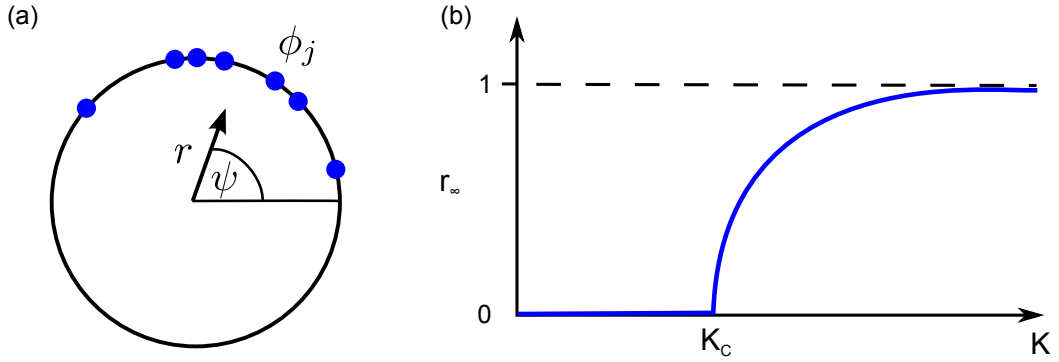


FIGURE 49: (a) Phases ϕ_j of 7 oscillators are visualized on the unit circle similar to the description in Fig. 33 and Sec. 6.2.3. The Kuramoto order parameter r and the parameter ψ represent the average angle and the average radius in the complex plane, respectively. (b) Order parameter r after a steady state is reached for a Kuramoto network with frequency heterogeneity as a function of the coupling strength K. Figure modified from Ref. [Str00].

The Kuramoto model, even with heterogeneous frequencies (under certain assumptions on the distribution), can be solved analytically using the transformation

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\phi_j} \tag{57}$$

with order parameters r and ψ [Stroo]. As illustrated in Fig. 49(a), the parameter ψ is the mean phase and the Kuramoto order parameter r indicates the phase coherence. For $r = 0$, the phases are incoherent (ϕ_i uniformly distributed over the unit circle), while, for $r = 1$, the phases are coherent ($\phi_i = \phi_j$ for all i and j, indicating phase synchronization).

Multiplying Eq. (57) with $e^{-i\phi_i}$ and taking the imaginary part, leads to

$$r\sin(\psi - \phi_i) = \frac{1}{N}\sum_{j=1}^{N} \sin(\phi_j - \phi_i). \tag{58}$$

This can be used to obtain the mean field description of Eq. (56)

$$\dot{\phi}_i = Kr\sin(\psi - \phi_i), \tag{59}$$

where each oscillator is driven by the mean field ψ. From this equation, the dynamics are always driven to the fully synchronized state $\phi_i = \psi = \text{const}$. On the other hand, when the frequencies are heterogeneous, the dynamics show a transition towards synchrony as a function of the coupling strength K. As shown in Fig. 49(b), the onset of synchronization appears at a critical coupling strength

$$K_c = \frac{2}{\pi g(0)}, \tag{60}$$

where $g(\cdot)$ is a symmetric distribution of frequencies ω_i shifted so that it peaks at $g(0)$ [Stroo]. Due to frequency heterogeneity, oscillations stay

incoherent for $K < K_c$, start to synchronize for $K > K_c$, and coherent oscillations, *i.e.*, complete synchronization, are only approached asymptotically for $K \gg K_c$.

From Eq. (59), the Kuramoto model with equal free-running frequencies (identical oscillators) has a globally stable synchronized state for $K > 0$. But, when the coupling in the Kuramoto model is changed from all-to-all coupling to non-local coupling and a phase lag parameter is introduced, chimera state can emerge even when all oscillators are identical as discussed in the next section. However, chimera states do not necessarily require identical oscillators [Pan14].

7.2.2 CHIMERA STATES IN THEORETICAL MODELS

In this section, I describe the conditions to observe chimera states in theoretical models. I also discuss the characteristics of chimera states and the appearance in various models.

7.2.2.1 *Non-Local Coupling*

A certain type of coupling topology has been hypothesized to be necessary for chimera states to appear. This coupling topology is called non-local coupling, where oscillators, assembled in a one- or two-dimensional geometry receive stronger coupling from geometrically close oscillators than from distant ones. However, the requirement of non-local coupling has been questioned in recent studies [Pan14].

Several modifications of the global coupling in the Kuramoto model, Eq. (55), have been proposed to achieve non-local coupling. For example, in the initial article on chimera states, Kuramoto and Battogtokh used the coupling kernel

$$g_{ij} = \frac{\kappa}{2} \exp\left[-\kappa\,|x|\right] \tag{61}$$

with $x = 2\pi(i-j)/N - \pi$, periodic boundary conditions on $[-\pi, \pi]$, and parameter κ. This function describes coupling that decreases exponentially with the distance $|i-j|$ between oscillators [Kuro2a]. Strogatz and Abrams used the coupling function

$$g_{ij} = \frac{1}{2\pi}\left[1 + A\cos\left(x\right)\right] \tag{62}$$

with parameter A, which allowed them to show analytically the existence and stability of chimera states in the limit $N \to \infty$ [Abro4]. This function is similar to Eq. (61) because $x \in [-\pi, \pi]$. Similar coupling functions have also been realized with two-dimensional topologies [Pan14]. In this chapter, I use a form of non-local coupling in a ring topology with a coupling range [Wol11], which is particularly easy to implement in Boolean systems.

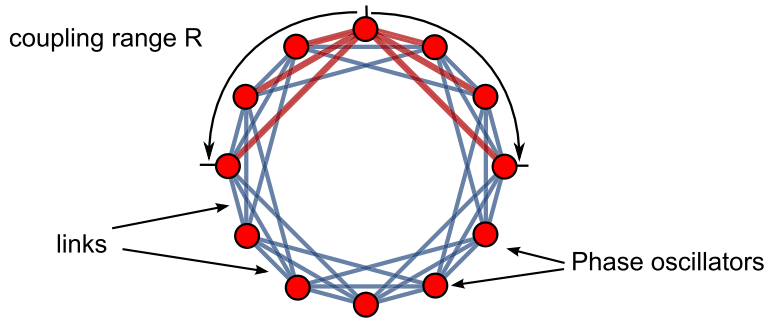


FIGURE 50: Illustration of a ring network with coupling range R. In the figure $N = 12$, $R = 3$.

Figure 50 illustrates the topology resulting from a coupling range R. Oscillators are coupled to their nearest neighbors, next nearest neighbors, and so on until the neighbors with distance R. This ensures that not only local oscillators are coupled (non-local) and that not all oscillators are coupled as long as $1 < R < N/2$.

7.2.2.2 *Phase-Lag Parameter*

Another hypothesized requirement to observe chimera states is the inclusion of a phase-lag parameter α in the coupling, according to

$$\dot{\phi}_i = \omega_i + \frac{1}{2R} \sum_{j=k-R}^{k+R} \sin(\phi_j - \phi_i + \alpha), \tag{63}$$

where the coupling topology is realized with a coupling range R. The phase-lag parameter can be interpreted as an approximation of a time delay along the links or as a means to balancing the coupling between phase attraction and phase repulsion [Pan14].

7.2.2.3 *Characteristics of Chimera States*

The numerical simulation of Eq. (63) with asymmetric initial conditions (see caption) leads to a phase and frequency picture shown Fig. 51. Asymmetric initial conditions are required because chimera states coexist with a stable synchronized state.

The main characteristics of chimera state is the appearance of synchronized and desynchronized domains in the phase and frequency pictures. Figure 51(a) shows a snapshot of the phase of the oscillators in a chimera state. The oscillators outside (inside) the dotted lines, marked as region I (region II), have equal (different) phases and hence are phase synchronized (desynchronized). Therefore, the oscillators in region I stay synchronized and the oscillators in region II drift apart because they have different frequencies. These frequencies are shown in Fig. 51(b). Similar

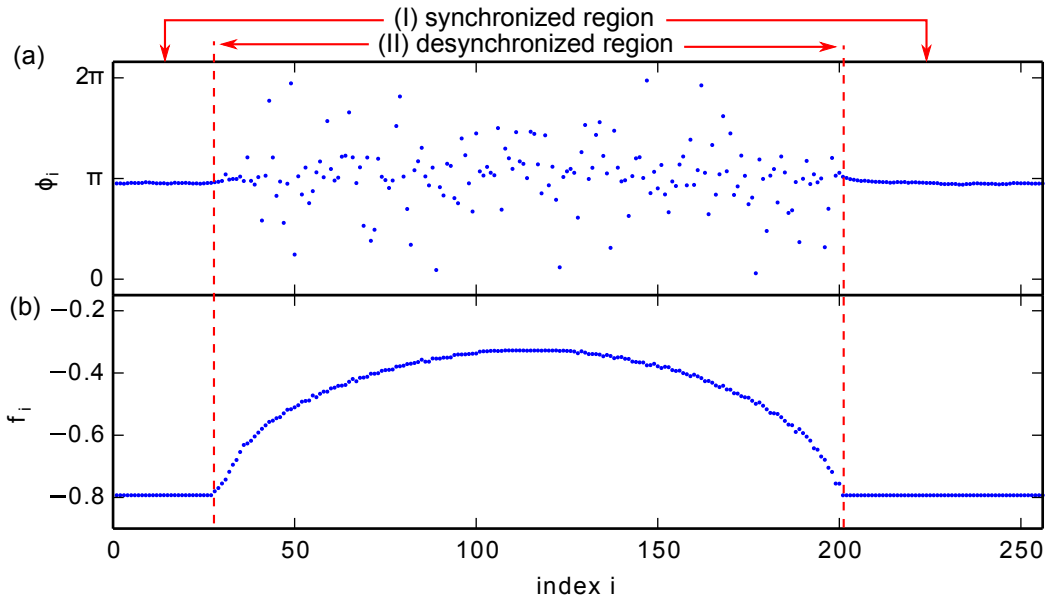


FIGURE 51: (a) Phases ϕ_i and (b) frequencies $f_i = \langle \dot{\phi}_i \rangle / (2\pi)$ of the network at $t = 300$. Dynamics are obtained from numerical simulation of Eq. (63) with $N = 256$, $R = 90$, $\omega_0 = 0$, $\alpha = 1.46$. Dynamics are initialized as in Ref. [Abr04] with $\phi_i = 6p \exp\left(-0.76x^2\right)$, where p is a uniform random variable on $[-0.5, 0.5]$ and $x = 2\pi i/N - \pi$. The parameters are the same as in Ref. [Wol11].

to the phase picture, the oscillators in region I (region II) are frequency synchronized (desynchronized) as they have equal (different) frequencies within the digitalization precision of $\pm 0.2\,\mathrm{MHz}$. The oscillators in region II show a regular arch shape, which is another characteristics of chimera states [Kuro2a, Abro4]. Chimera state in other dynamical systems, however, also appear without the arch [Hag12]. Another recent extension are amplitude chimeras, where a arch-shaped shift appears for the oscillation amplitude similar to Fig. 51(b) for the mean frequency and oscillators are phase synchronized [Zak14].

7.2.2.4 *Chimera States in Various Theoretical Models*

Since their discovery in 2002 [Kuro2a], chimera states have been found in various models with various different characteristics [Pan14]. In addition to ring topologies, they have been found in two coupled oscillator populations [Abro8], in two-dimensional shapes, such as planes, spheres and tori with the possibility of a spiral chimera [Mar10, Pan13, Ome12a, Pan14], and in various models, such as chaotic maps [Ome11, Ome12, Hag12].

Chimera states also appear in coupled neural FitzHugh-Nagumo systems, where multiple synchronized and desynchronized domains can exist [Ome13] (see Sec. 8.2.4.2 for an introduction to the FitzHugh-Nagumo system).

Recently, a connection between chimera states and oscillator death has been made and was termed chimera death [Zak14]. Oscillator death is a symmetry-breaking phenomenon, where coupling of oscillatory systems can suppress oscillations and lead to newly created inhomogeneous stable steady states. Chimera death denotes a state, where domains of coherent and incoherent inhomogeneous steady states coexist.

7.2.3 TRANSIENT BEHAVIOR OF CHIMERA STATES IN FINITE-SIZE NETWORKS

In the thermodynamic limit ($N \to \infty$), chimera states have been shown to be neutrally stable for certain phase-oscillator networks, meaning that they are neither stable nor unstable [Otto8, Piko8, Wol11, Wol11a]. On the other hand, for finite-size oscillatory networks, chimera states have been found numerically to be unstable with long transients towards the synchronized state [Wol11, Wol11a]. Here, I summarize the finite-size effects found in Ref. [Wol11] for the model in Eq. (63).

Wolfrum and Omel'chenko simulated Eq. (63) for a finite-size network with $N = 40$ and $R = 14$ and found several finite size effects shown in Fig. 52(a). The figure shows the frequency plot similar to Fig. 51(b) as a color plot over time. Specifically, high frequencies are denoted with yellow color, while low frequencies are denoted by purple color. The domain of high frequencies (the unsynchronized domain) and the domain of low frequencies (the synchronized domain) move over time incoherently [Ome10]. After a time denoted T_{40}, the dynamics synchronize (the index denotes the size of the network $N = 40$). This collapse time, however, varies strongly with the initial conditions.

In Fig. 52(b), a histogram of collapse times T_{40} is shown. The probability ρ to observe a collapse times T_{40} follows the exponential distribution

$$\rho(T_N) = \lambda e^{-\lambda T_N} \tag{64}$$

with a collapse rate λ that is given by the average lifetime

$$T_c = \langle T_N \rangle = \lambda^{-1}, \tag{65}$$

which normalizes the exponential distribution.

The average lifetime and hence also the collapse rate λ scale exponentially with the number of oscillators N in the network

$$T_c \propto e^{\kappa N}. \tag{66}$$

Because of this exponential growth of the average lifetime, the transient times become very long for $N > 60$, so that the numerical study of the average lifetime for $N > 60$ is not amenable to numerical simulation [Wol11].

Wolfrum and Omel'chenko identified chimera states as chaotic transients because, with Eqs. (64) and (66), chimera states show the same scaling properties as super-transients [Wol11, Wol11a, Wac95b].

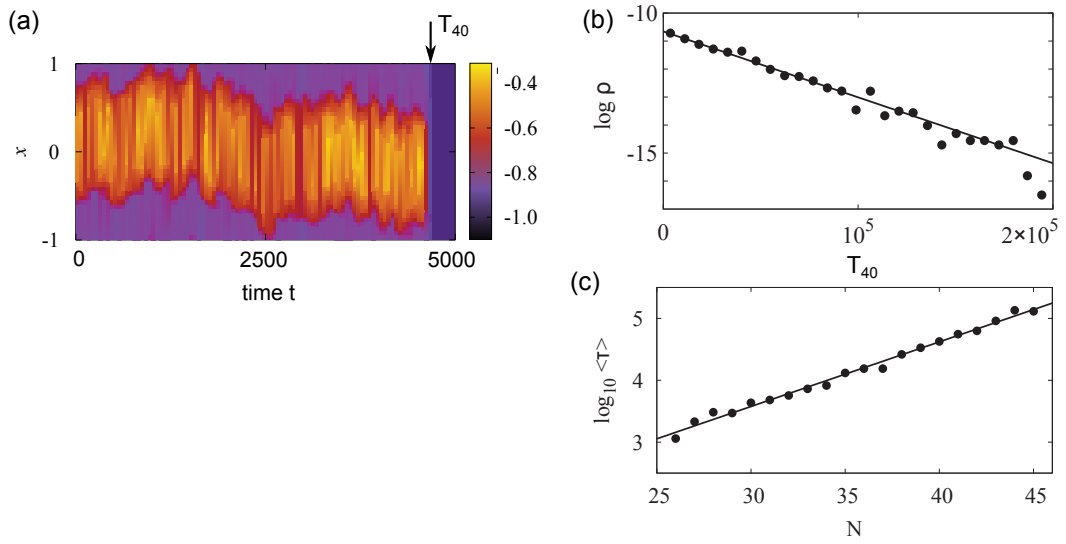


FIGURE 52: (a) Frequency of oscillators over time with a color code. The y-axis denotes the oscillator index normalized to an interval $[-1, 1]$. (b) Histogram of transient times T_{40} measured from 2000 numerical simulations (circles) and predicted distribution function according to Eq. (64) (solid line). (c) Average transient time $\langle T_N \rangle$ as a function of network size N (circles), fitted with Eq. (66) (solid line) with $\kappa = 0.23$. Simulation parameters as in Fig. 51. The figure is taken from Ref. [Wol11].

7.2.4 CHIMERA STATES IN EXPERIMENTS

Recent experiments have proven that chimera states are robust enough to be observable. There has, however, not yet been a conclusive observation of chimera states in a natural, non-engineered system.

7.2.4.1 *Previous Chimera States in Experiments*

The first experimental realizations of chimera states were published simultaneously in the same volume of Nature Physics, realized with an optical system called a liquid crystal spatial light modulator realizing a chaotic map [Hag12] and chemical systems with a Belousov-Zhabotinsky reaction [Tin12]. These realizations use computers to mediate the coupling between the network nodes. Therefore, the part of the network dynamics involving the coupling is calculated numerically using mathematical equations similar to the Kuramoto model, which has been criticized [Sma12]. This has stimulated further experiments with physical (not computer mediated) coupling that have also shown chimera states. One of these experiments is a network of mechanical metronomes coupled via swings and springs [Mar13]. Two other experimental realizations of chimera states are based on electrochemical reactions [Sch14a, Wic13]. Chimera-like states can also be realized in electronic and optoelectronic oscillators with delayed feed-

back using an elegant mapping by associating time intervals $[0, \tau]$ with a continuous spatial variable to identify virtual oscillators [Lar13].

7.2.4.2 *Limitations of Previous Experiments*

All previous experiments, besides the experiments by Larger [Lar13] on virtual chimera states, evolve on a slow timescale on the order of seconds. For that reason, the transient scaling of chimera states discussed in Sec. 7.2.3 has not yet been confirmed in an experiment. Only for electrochemical oscillators in Ref. [Wic13], a transient towards synchrony has been observed but not quantified, which would require the acquisition of several hundred transients. Also, virtual chimera states cannot be used to show the scaling because the virtual oscillator networks used in this study include an infinite number of nodes and are hence stable.

7.3 BOOLEAN PHASE OSCILLATOR NETWORKS

In this section, I discuss the extension of Boolean phase oscillators to large in-degrees, derive a phase model description, and detail the electronic implementation of large networks.

7.3.1 SETUP OF BOOLEAN PHASE OSCILLATORS WITH LARGE IN-DEGREE

I study networks of N coupled Boolean phase oscillators in a non-locally coupled ring topology with a coupling range R, as introduced in Sec. 7.2.2.1 and also illustrated in Fig. 50. In this topology, nodes have an in-degree of $2R$, which requires to extend the previous setup of Boolean phase oscillators in Sec. 6.4.1, so that in-degrees greater than one are possible. In the setup with one input, the coupling is realized with a feedback delay τ_i that switches between two values depending on the phase difference with the single coupled oscillator. Here, I extend this method by configuring the feedback delay so that it can switch between $2^K = 2^{2R}$ values depending on K phase differences between the local oscillator and the coupled oscillators as shown in Fig. 53.

The state-dependent delay τ_i of an oscillator is realized with a constant part $\tilde{\tau}_{0,i}$, and a variable part realized with XOR logic gates, Boolean switches, and short delay lines of value σ_i. The XOR logic gate is a simple phase detector that compares the phase of the oscillator x_i with the phase of a neighboring oscillator x_j. Its signal $x^c_{j,i}$, which corresponds to the phase difference between oscillators i and j, activates one of two paths in the Boolean switch: one path leads to an additional delay σ_i and the other leaves the delay unchanged, where the shorter delay is selected when

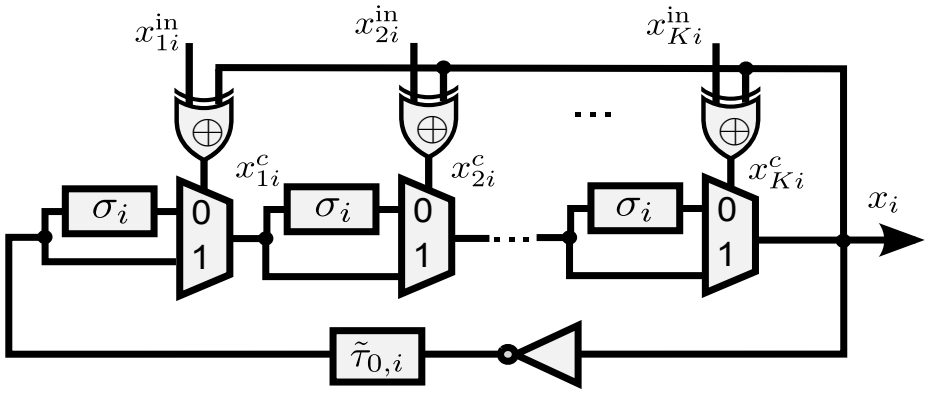


FIGURE 53: Circuit diagram of the Boolean phase oscillator showing the coupling mechanism. The rectangles, trapezoids, triangles, and shapes with $\oplus$-signs denote delay lines, Boolean switches (also known as multiplexers), inverter, and XOR gates, respectively. The K coupling inputs are realized with K pairs of delay lines, multiplexers, and XOR gates, of which three are shown. The dynamical output variable is x_i.

a phase difference is detected. The hardware description of this system is discussed in Appendix B.6.2. The coupling mechanism is detailed and quantified in the next section.

7.3.2 DERIVATION OF A PHASE MODEL FOR BOOLEAN PHASE OSCILLATORS

In this section, I derive the phase model for a network of N Boolean phase oscillators shown in the setup in Fig. 53. The N oscillators are described by output variables $\{x_i\}_{i=1}^N$, which are normalized output voltages $\{V_i\}_{i=1}^N$ of oscillators so that $x_i = 1$ ($x_i = -1$) corresponds to the high (low) Boolean voltage; specifically $x_i = 2V_i/V_H - 1$ with high Boolean voltage V_H [Ros14]. The coupling of oscillators is realized with state-dependent delay that depends on the phase differences between the oscillator output x_i and K oscillatory input signals $\left\{x_{ji}^{\text{in}}\right\}_j$. Delay lines in the setup are based on chains of buffer gates that each add a gate propagation delay to the total delay as detailed in Appendix A. In the setup in Fig. 53, I include several delay lines of value $\sigma = 0.328 \pm 0.012\,\text{ns}$, corresponding to a single buffer gate and a long delay line of value $\tau = 8.4 \pm 0.3\,\text{ns}$, corresponding to 30 cascaded buffer gates. Several XOR logic gates are included as simple Boolean phase detectors [Bes03] to generate a control signal $x_{ji}^c \in \{0, 1\}$. The control signal is given by a comparison of the output Boolean state of the oscillator x_i and an input Boolean state x_{ji}^{in}, according to

$$x_{ji}^c = x_{ji}^{\text{in}} \oplus x_i, \tag{67}$$

where $\oplus : \{0,1\} \times \{0,1\} \to \{0,1\}$ denotes the XOR Boolean function. Several multiplexers, which are 3-input logic gates, are included as Boolean switches to modify the feedback delay of the system by a delay σ depending on the control signals $\left\{x_{ji}^c\right\}_j$. This leads to the total feedback delay of the i-th oscillator according to

$$\tau_i = \tau_{0,i} - \sigma \sum_{j=0}^{K} x_{ji}^c, \tag{68}$$

where $\tau_{0,i} = \tilde{\tau}_{0,i} + K\sigma$ denotes the maximum delay in the loop. This is a state-dependent delay because the $\left\{x_{ji}^c\right\}_j$ depend on the state of the local oscillator [Wal03].

I also include a single inverter logic gate in the feedback loop in Fig. 53, which leads to oscillations with frequency

$$f_i = \frac{1}{2\tau_i}. \tag{69}$$

for the i-th oscillator (see Sec. 6.2.5.1).

I assume that Eq. (69) is valid for state-dependent delay when the frequency f_i is replaced with the instantaneous frequency $\dot{\phi}_i$. Equations (67)-(69) and a Taylor approximation, with a maximum error of $\approx 3\,\mathrm{MHz}$, leads to

$$\dot{\phi}_i = \omega_{0,i} + \tilde{\sigma}_i \sum_{j=j_0}^{j_K} x_j(\phi_j) \oplus x_i(\phi_i), \tag{70}$$

with angular frequency $\omega_{0,i} = 2\pi/(2\tau_{0,i}) = 2\pi \cdot 9.3\,\mathrm{MHz}$ (for in-degree $K = 60$) and coupling strength $\tilde{\sigma}_i = 2\sigma\omega_{0,i}^2 = 0.089\,\mathrm{MHz}$ (using a value $\sigma = 0.515\,\mathrm{ns}$ to adjust for the error by the Taylor approximation). The summation from j_0 to j_K is over K coupling inputs in the network topology with a coupling range R. I replace $\left\{x_j\right\}_j$ with $\left\{\phi_j\right\}_j$ using the following definition of the phase ϕ_j from Ref. [Ome11]

$$\sin\left[\phi_j(t)\right] = \frac{2x_j(t) - \max\left[x_j\right] - \min\left[x_j\right]}{\max\left[x_j\right] - \min\left[x_j\right]} = x_j(t), \tag{71}$$

where x_j oscillates between the Boolean values -1 and 1. I express the XOR Boolean function with a difference of Heaviside functions and add a phase-lag parameter α_{ij}, leading to the following phase oscillator model

$$\dot{\phi}_i = \omega_{0,i} + \tilde{\sigma}_i \sum_{j=j_0}^{j_K} \left|\Theta\left[\sin(\phi_j)\right] - \Theta\left[\sin(\phi_i + \alpha_{ij})\right]\right|. \tag{72}$$

The phase-lag parameter α_{ij} is vital for observing chimera states [Kuro2a, Abro4] and is likely caused experimentally by delays along wire connections (see also Sec. 7.2.2.2).

I couple the Boolean oscillators in a ring network with non-local coupling realized with a coupling radius R, corresponding to a rectangular coupling kernel. Specifically, the resulting dynamical equation for the network of N nodes ($i \in \{1, 2, ..., N\}$) with periodic boundary conditions is

$$\dot{\phi}_i = \omega_{0,i} + \tilde{\sigma}_i \sum_{j=i-R}^{i+R} \left| \Theta\left[\sin(\phi_j)\right] - \Theta\left[\sin(\phi_i + \alpha_{ij})\right] \right| . \tag{73}$$

Equation (73) has a similar form as the modified Kuramoto model used to discover chimera states (see Sec. 7.2.1 and compare to Eq. (63)) [Kuro2a]. The nonlinear coupling function in Eq. (73) differs from the coupling in Eq. (63) that includes a single sine function of the phase differences. Because of the similarity between the two models, the network of Boolean phase oscillators is ideally suited to study chimera states experimentally.

The free-running frequency of an oscillator $\omega_{0,i}$ as a function of R results from the construction of the oscillator with logic gates as shown in Fig 53. It is given by

$$\omega_{0,i} = 1/[2(30\tau_{\text{buf}} + 2R\tau_{\text{mux}} + 2R\tau_{\text{buf}})] \tag{74}$$

with $n_0 = 30$ the number of buffer gates used to build the delay line denoted τ, and the propagation delays of multiplexer and buffer are $\tau_{\text{mux}} = 0.404 \pm 0.014\,\text{ns}$, $\tau_{\text{buf}} = 0.328 \pm 0.012\,\text{ns}$ (for the specific wiring), respectively. In the model, I assume identical oscillators ($\omega_{0,i} = \omega_0$ and $\tilde{\sigma}_i = \tilde{\sigma}$) and homogeneous coupling ($\alpha_{ij} = \alpha$).

For numerical simulation, it is advantageous to replace the Heaviside functions and the absolute value in Eq. (73) with a smooth sigmoidal function (tanh here), leading to

$$\dot{\phi}_i = \omega_{0,i} + \tilde{\sigma}_i \sum_{j=i-R}^{i+R} \left\{ \tanh\left[-c \sin(\phi_j) \sin(\phi_i + \alpha)\right] + 1 \right\} / 2 \tag{75}$$

with slope $c = 4$ used in the numerical simulation. For $c \to \infty$, Eq. (75) and (73) are identical.

7.3.3 ELECTRONIC IMPLEMENTATION OF BOOLEAN PHASE OSCILLATORS

I implement the network with electronic logic circuits on an FPGA with a hardware description disccused in Appendix B.6.3. A Boolean phase oscillator is realized with up to 218 logic elements, requiring 14 logic array blocks (see also Sec. 3.2.1 for more details). The layout of the oscillators on the chip is shown in Fig. 54. Allocated logic array blocks are marked

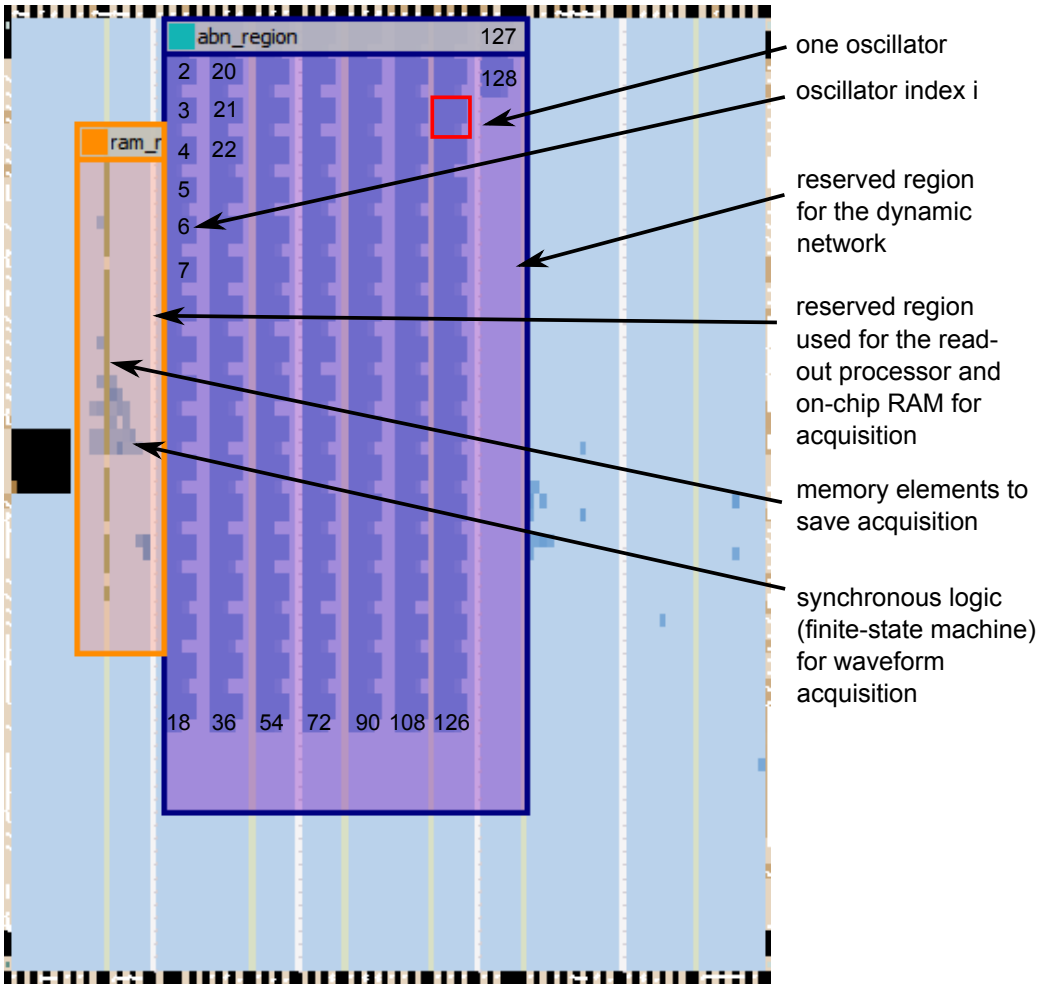


FIGURE 54: Picture showing the arrangement of logic gates on the FPGA for the implementation of the network with $N = 128$ nodes. The picture was generated with the Altera Chip Planner.

in darker blue while unused logic array blocks are marked in light blue. The blue frame marks a region reserved for the dynamical network. The orange frame marks a region reserved for the readout and the acquisition of the network dynamics. The physical implementation of $N = 128$ nodes with a non-local coupling range $R = 30$ requires 27,000 logic gates and 7,552 wires, which requires about 25% of the resources on the FPGA, and is implemented together with a custom-built processor for data acquisition.

This implementation does not account for the ring structure of the network, so that the delay between some neighboring oscillators, *e.g.*, 18 and 19 and especially 1 and 128, is long compared to the delay between most neighboring oscillators, such as 1 and 2, even though these are neighbors in the ring network topology. I estimate the maximum transmission delay between two oscillators to be 2 ns corresponding to the delay between two

gates separated across the chip, which is small compared to the period of about 100 ns.

I measure and reduce the heterogeneity in the free-running frequency of the N oscillators by adjusting the constant part of the feedback delays $\tau_{0,i}$, resulting in a heterogeneity of $\sigma_f^2/\bar{f} = 0.3\%$ (average frequency $\bar{f} = 9.14\,\mathrm{MHz}$ with standard deviation $\sigma_f^2 = 0.03\,\mathrm{MHz}$) for $N = 128$. Implementing large networks requires me to sort the input connections to the oscillators, giving rise to heterogeneity in the coupling time delays whose implications are discussed in Sec. 7.6.2. When I do not sort the input connections, the computer-aided design tool used to assign logic gates and interconnect lines generates an error message, restricting the network to 90 nodes. This could be due to a depletion of the interconnect lines and crossings on the electronic chip.

7.4 COMPLEX DYNAMICS IN BOOLEAN PHASE OSCILLATOR NETWORKS

In this section, I discuss the dynamics of networks of non-locally coupled Boolean phase oscillators. I find that the dynamical system shows both chimera states and unsynchronized dynamics, where the chimera can disappear in favor of unsynchronized dynamics and reappear.

7.4.1 CHIMERA DYNAMICS IN BOOLEAN PHASE OSCILLATOR NETWORKS

I first describe the dynamics of the network when it is in a chimera state, as shown in Fig. 55(a) with a snapshot of the phases of oscillators. The oscillators outside (inside) the dotted lines, marked region I (region II), have equal (different) phases within the digitalization precision of $\Delta\phi = \pm 0.25\,\mathrm{rad}$ and hence are considered phase synchronized (desynchronized). Therefore, the oscillators in region I stay synchronized and the oscillators in region II drift apart because they have different frequencies. These frequencies are shown in Fig. 55(b) and are measured over a time period of 6 μs, which represents approximately 60 oscillations. Similar to the phase picture, the oscillators in region I (region II) are frequency synchronized (desynchronized) as they have equal (different) frequencies within the digitalization precision of $\pm 0.2\,\mathrm{MHz}$. The oscillators in region II show a regular arch shape, which is another characteristics of chimera states [Kuro2a, Abro4] (see also Sec. 7.2.2.3).

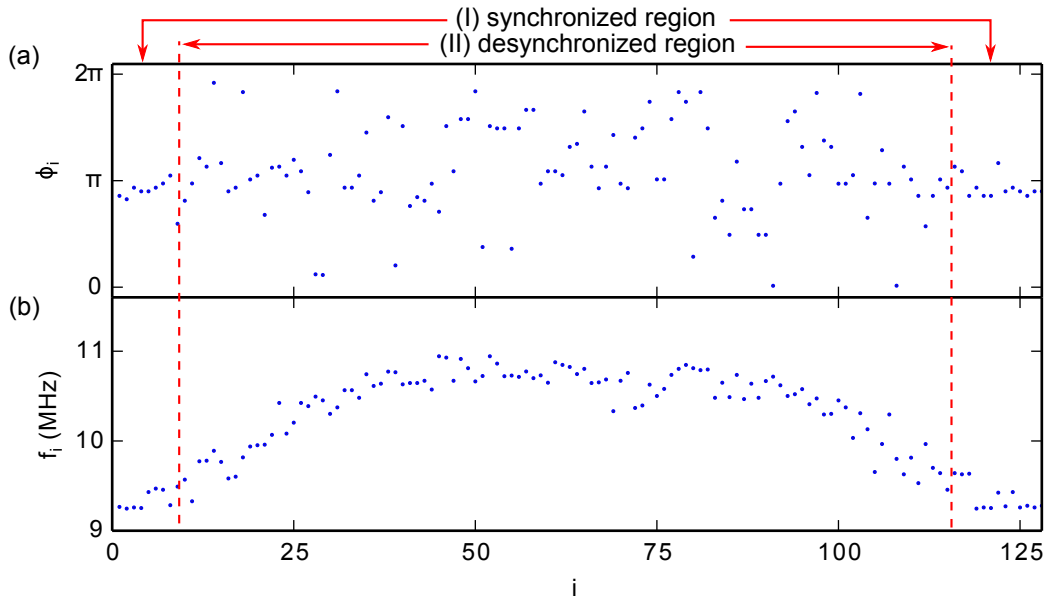


FIGURE 55: Dynamics measured from coupled Boolean phase oscillators with $N = 128$, $R = 30$, $\omega_0 = 2\pi(9.3 \pm 0.03)\,\mathrm{MHz}$, $\tilde{\sigma} = 2\pi(0.089 \pm 0.003)\,\mathrm{MHz}$. (a) Snapshot at $t \approx 304\,\mathrm{s}$, (b) frequency profile $f_i = \langle \dot{\phi}_i \rangle / (2\pi)$. I initialize the network by first deactivating the coupling, which results in a randomization of the phases because of small frequency heterogeneity, and then activating the coupling. The oscillator indices are shifted by a constant integer, so that the location of the unsynchronized region is centered.

7.4.2 RESURGENCE OF CHIMERA STATES

The temporal evolution of the frequency is visualized using a gray-scale image in Fig. 56(a) for a period over 7 min, corresponding to more than 4 billion oscillations. The figure shows that, for this specific realization, complex dynamics exist from time $t = 0$ until $t = 6\,\mathrm{min}$ (marked III) because the frequency varies both from node to node and in time. At time $t = 6\,\mathrm{min}$, the dynamics collapses to a nearly synchronized state (dark gray region, marked IV), which I discuss in Sec. 7.4.4. The time until the nearly synchronized state is reached varies considerably for different repetitions of the experiment and is studied in detail in Sec. 7.5. In the following, I discuss the dynamics of the oscillator network on a microsecond timescale, at times marked in the figure.

Figure 56(b) shows the frequency of the Boolean phase oscillators for about 60 periods after 304 s, corresponding to a millionth of the total transient time. The network shows high frequencies (dark gray) for oscillator indices from $i \cong 20$ to $i \cong 100$ and low frequencies (light gray) for the remaining oscillators. This picture corresponds to the dynamics in Fig. 55(a), which I identified as a chimera state. The unsynchronized domain of the chimera (high frequency, dark gray) moves irregularly in the network,

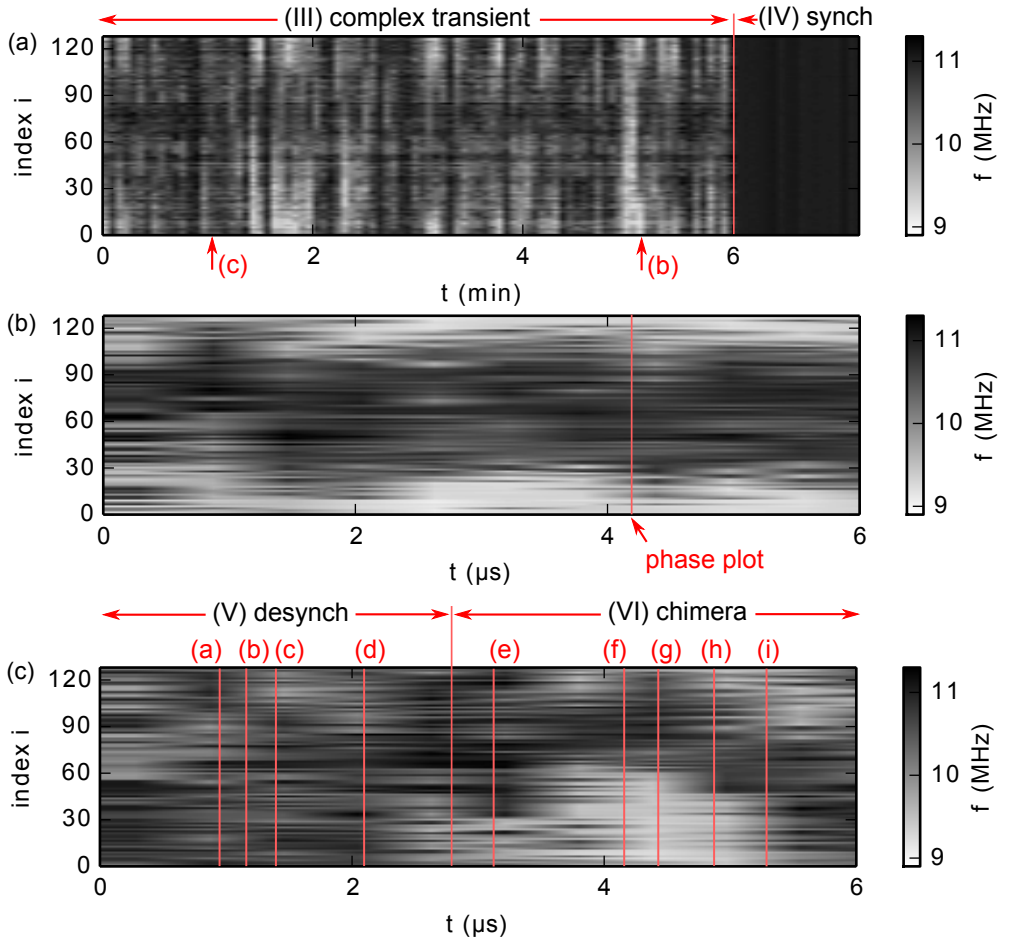


FIGURE 56: (a) Frequency evolution over a time period of 7 min; averaged over 6 μs windows (60 oscillations) every 4 s. (b), (c) Frequency evolutions shown over a time period of 5 μs; averaged over 500 ns windows (5 oscillations). The arrows in (b) and (c) indicates the phase measurements in Fig. 55(a) and Fig. 57, respectively. Parameters of the experiment as in Fig. 55.

which is an effect of the finite size of the network [Ome10a, Nko13] (see also Sec. 7.2.3), and indicates that the chimera dynamics is not pinned to heterogeneity.

At an earlier time in the transient shown in Fig. 56(c), the dynamics alternates between complete desynchronization and chimera states. For times $0 < t < 2.8$ μs (marked V), the figure shows large variations in the frequencies of neighboring nodes without obvious chimera domains, corresponding to a dynamics with time intervals of both desynchronization and chimera-like states. In the remaining part (marked VI), two domains of high and low frequencies can be identified, which correspond to a chimera state lasting over at least 30 oscillations. Again, the unsynchronized domain moves in the network.

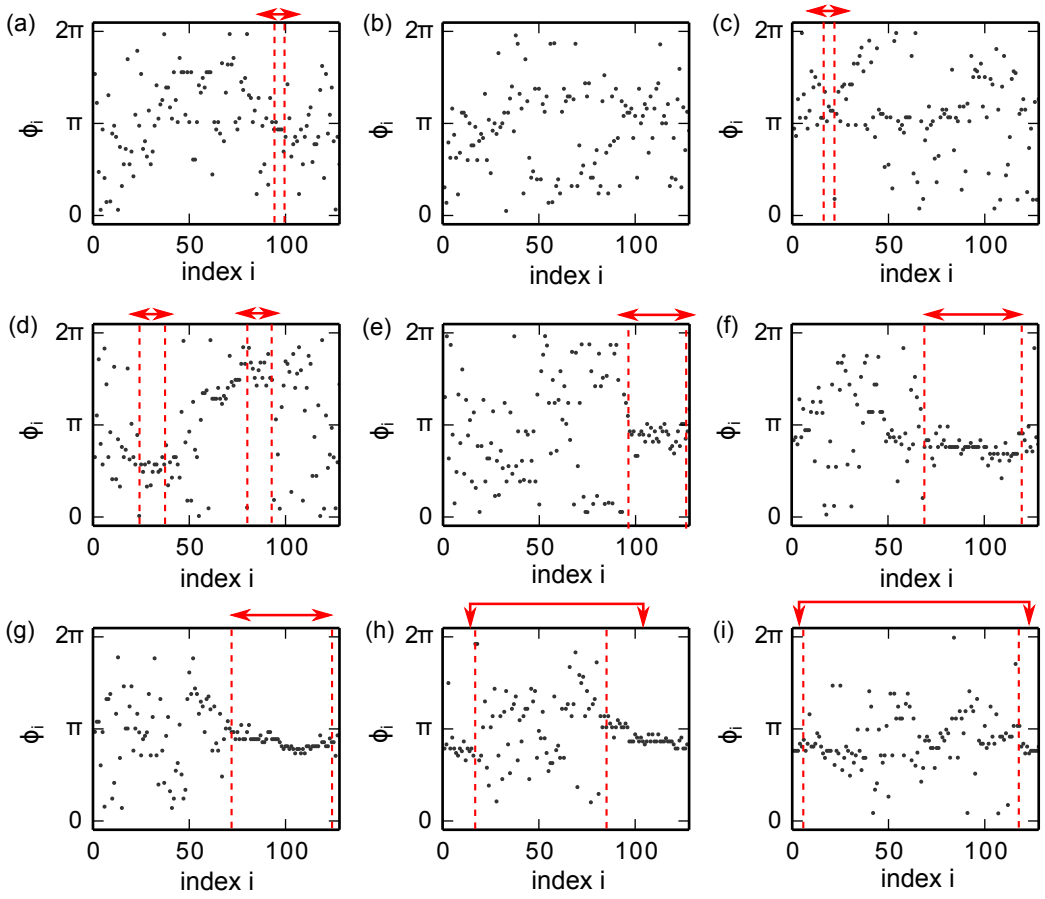


FIGURE 57: (a)-(i) Phase pictures at time indicated in Fig. 56(c) at times 0.94 μs, 1.14 μs, 1.37 μs, 2.05 μs, 4.10 μs, 4.42 μs, 4.84 μs, and 5.26 μs, respectively. Experimental parameters as in Fig. 55.

I discuss the dynamics in Fig. 56(c) at points in time indicated by (a)-(i) using the phase snapshots in Fig. 57(a)-(i), respectively. In the snapshot in Fig. 57(a), there exists a region, where the oscillators have the same phase to within the experimental phase resolution of $\Delta\phi = 0.25$. Outside the dotted lines, the oscillators are not phase-synchronized. Therefore, this state corresponds to both synchronization and desynchronization, which is a chimera-like state. This state exists for six periods, where the synchronized region drifts considerably in the network. In Fig. 57(b), I find an intermediate state, where the phases of the oscillators are distributed randomly so that I cannot identify a phase-synchronized region; hence, it is an unsynchronized state. In Fig. 57(c), a chimera-like state is reached again, with similar properties to the phase snapshot in Fig. 57(a). Therefore, the dynamics change on a timescale similar to the period of oscillation from chimera-like states to unsynchronized states and back.

Figure 57(d) shows the phases of the oscillators in the network at a time when two regions (marked with dotted lines) show the same phases within

the measurement error. Outside the two regions, the oscillators show large differences in the phase and are hence unsynchronized. This state corresponds to a multi-chimera state of two synchronized and two desynchronized regions, which has been reported before with numerical simulations [Ome12a, Nko13, Ujj13, Zak14]. Similar to the findings in these references, the two synchronized cores are phase shifted by π.

Figures 57(e)-(i) show the phases of oscillators in the network in the region VI in Fig. 56(c). The synchronized region of the chimera state moves from the range $i \cong 90$ to $i \cong 127$ to the range $i \cong 115$ to $i \cong 20$ (periodic). In Fig. 57(h), one oscillator in the synchronized region does not synchronize, which is non-ideal.

This phenomenon of disappearance and reappearance of chimera states has not been reported previously. I call it resurgence of chimera states.

7.4.3 DETERMINING THE FRACTION OF CHIMERA STATES IN THE TRANSIENT

The phenomenon of resurgence of chimera states implies that chimera states do not exist through the entire transient. In this section, I determine a lower boundary for the fraction of time that chimera states exist in the transient. For that, I use a measure similar to Ref. [Gop14].

The chimera states in Boolean phase oscillator networks have a typical arch shape, where one domain of adjacent oscillators are synchronized at low frequency, and another domain is unsynchronized with higher frequencies as shown in Fig. 55(b). Such a shape can be detected by measuring the standard deviation of frequencies

$$\sigma = \sqrt{\langle (f_i - \langle f_i \rangle)^2 \rangle}, \tag{76}$$

and the average frequency difference between adjacent oscillators

$$\Delta = \langle |f_i - f_{i-1}| \rangle . \tag{77}$$

An arch-shaped function has a large σ but a small Δ, so that a large

$$\psi = \sigma / \Delta, \tag{78}$$

is a measure to detect chimera states.

I calculate this function for the partly synchronized temporal evolution in Fig. 58(a). Figure 58(b) shows ψ for this temporal evolution, exhibiting a peak above a value of $\psi = 1.5$ approximately when I detect chimera states. Therefore, I use a threshold of $\psi > 1.5$ to distinguish chimera states. With this numeric value, I find that chimera states appear for about 14% of the transient, measured from 100 measurements of length 6 μs on a network of $N = 128$ oscillators within the transient shown in Fig. 56(a).

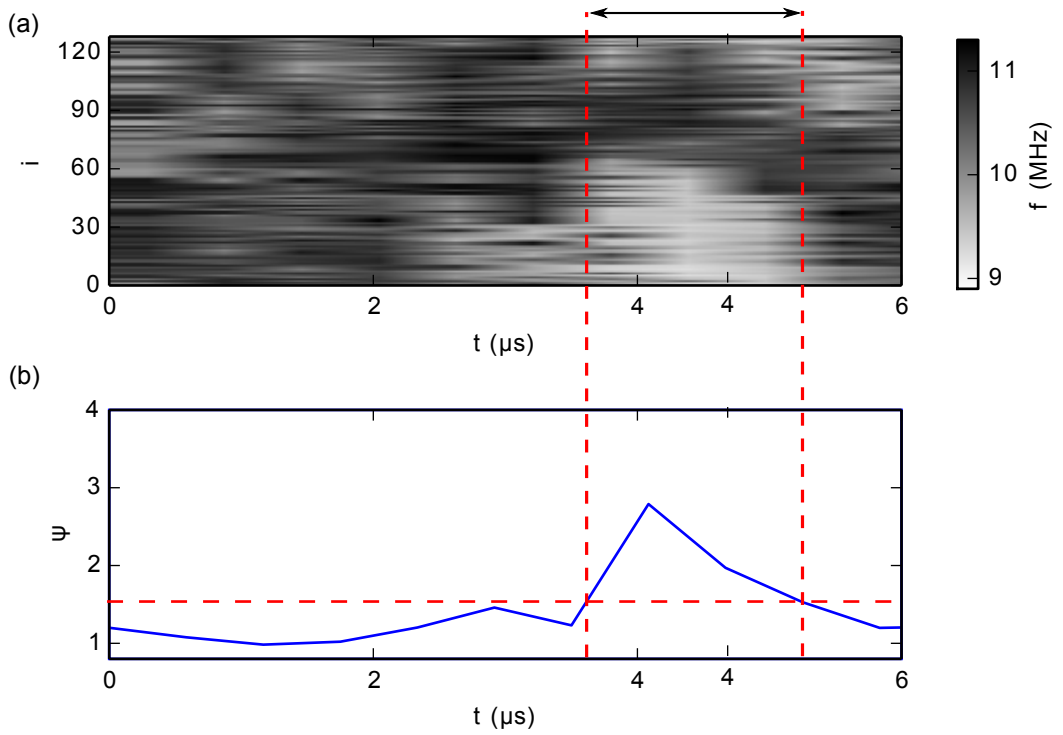


FIGURE 58: (a) Copy of Fig. 56(c). (b) Corresponding function $\psi = \sqrt{\langle (f_i - \langle f_i \rangle)^2 \rangle} / \langle |f_i - f_{i-1}| \rangle$ to detect a chimera state. Specifically, a chimera is detected from time 3.6 μs until 5.3 μs when $\psi > 1.5$ (shown with dashed lines).

7.4.4 NEARLY SYNCHRONIZED STATE

As discussed above, the network displays complex dynamics that collapses to a nearly synchronized state, corresponding to the uniform dark gray area at $t > 6$ min in Fig. 56(a). Figure 59(a) shows the phases of oscillators in this state. The phases of oscillators have different values over the whole range of 2π and are hence not synchronized. Figure 59(b) shows the frequency of oscillators, which are apart from two nodes frequency synchronized with $f = 11.085 \pm 0.002$ MHz (compare to $f = 9.14 \pm 0.03$ MHz for uncoupled oscillators). The two unsynchronized nodes, marked in the figure with arrows, have a $\sim 1\%$ larger frequency. In the model, on the other hand, both frequency and phase synchronize [Wol11]. The appearance of the nearly synchronized state may be due to heterogeneity in the phase lag parameter α_{ij}.

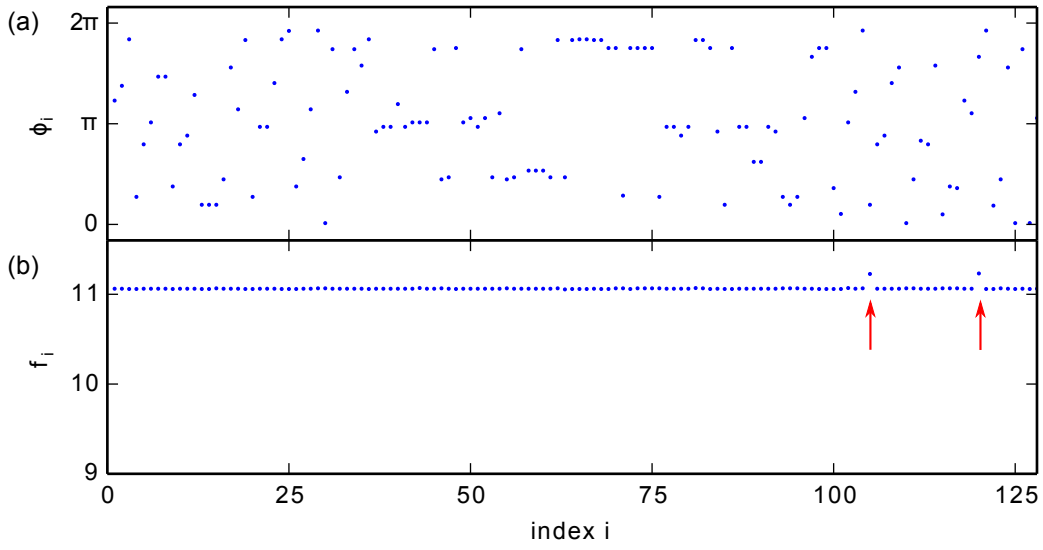


FIGURE 59: (a), (b) Phase and frequency pictures similar to Fig. 55. Phase snapshot at $t \approx 480\ s$ in Fig. 56(a). Experimental parameters as in Fig. 55.

7.5 TRANSIENT SCALING OF BOOLEAN OSCILLATOR NETWORKS

In this section, I discuss the scaling of the transient time and the scaling of the average transient time in networks of non-locally coupled Boolean phase oscillators. I record similar scaling functions as found by Wolfrum and Omel'chenko for chimera states in networks of phase oscillators, as discussed in Sec. 7.2.3 [Wol11]. I find that the event of synchronization can be modeled with a Poisson process and that the synchronization rate follows a power law of the phase-space volume.

7.5.1 SCALING OF THE TRANSIENT TIME

As discussed above and shown in Fig. 56(a), the complex dynamics collapses to the nearly synchronized state after a transient time T_N. I find that T_N for $N = 128$ varies between extreme values of $T_{128} = 1\,\mathrm{s}$ and $T_{128} = 32\,\mathrm{min}$ for 1,000 measurements, where complex dynamics are obtained for every acquisition from random initializations. This is different from Ref. [Wic13], where chimeras appear only for a fraction of the initializations. Figure 60(a) shows the distribution of transient times, where each dot corresponds to the normalized number of transients of T_N within a certain interval. I find that T_N follows an exponential distribution (shown as a solid line) according to

$$\rho_N(T_N) = \langle T_N \rangle^{-1} \exp(-T_N / \langle T_N \rangle), \tag{79}$$

with the average transient time $\langle T_{128} \rangle = 5.4\,\text{min}$ as indicated in the figure. The exponential distribution follows analytically by considering the collapse to synchronization as a Poisson process, which occurs continuously in time at a constant average synchronization rate λ with

$$\langle T_N \rangle = 1/\lambda. \tag{80}$$

The exponential distribution has been found to describe the transient times for chimera states in the Kuramoto model under the assumption of identical oscillators with symmetric coupling [Wol11]. The appearance of the same scaling in my studies is surprising because the experiment has heterogeneity and shows chimeras only for a fraction of the time, which are aspects not included in previous models.

7.5.2 SCALING OF THE AVERAGE TRANSIENT TIME

I measure the average transient time $\langle T_N \rangle$ for networks of different size N, *i.e.*, the number of nodes changes but the network topology is the same. Figure 60(b) shows $\langle T_N \rangle$ for six different network sizes from $N = 105$ to $N = 128$. The average transient time $\langle T_N \rangle$ follows approximately an exponential scaling over three orders of magnitude according to

$$\langle T_N \rangle \propto \exp(\kappa N), \tag{81}$$

with $\kappa = 0.28 \pm 0.10$ shown with a solid line. Using Eqs. (80) and (81), the synchronization rate

$$\lambda \propto \exp(-\kappa N) \propto V^{-\kappa} \tag{82}$$

follows a power law of the state space volume $V = (2\pi)^N$, which is plausible [assuming for a single oscillator's phase space volume $V = 2\pi$ in accordance with Eq. (73), but the power law can also be derived when each system is assumed to be of dimension $d < \infty$]. This super-transient scaling holds for many spatially-extended systems [Tel08] and also for networks, such as neural networks [Zil09] and Kuramoto oscillators [Wol11]. Therefore, I conjecture that Eq. (82) may be a general law for networks under certain conditions, such as that nodes are nearly identical and that a stable synchronized state exists.

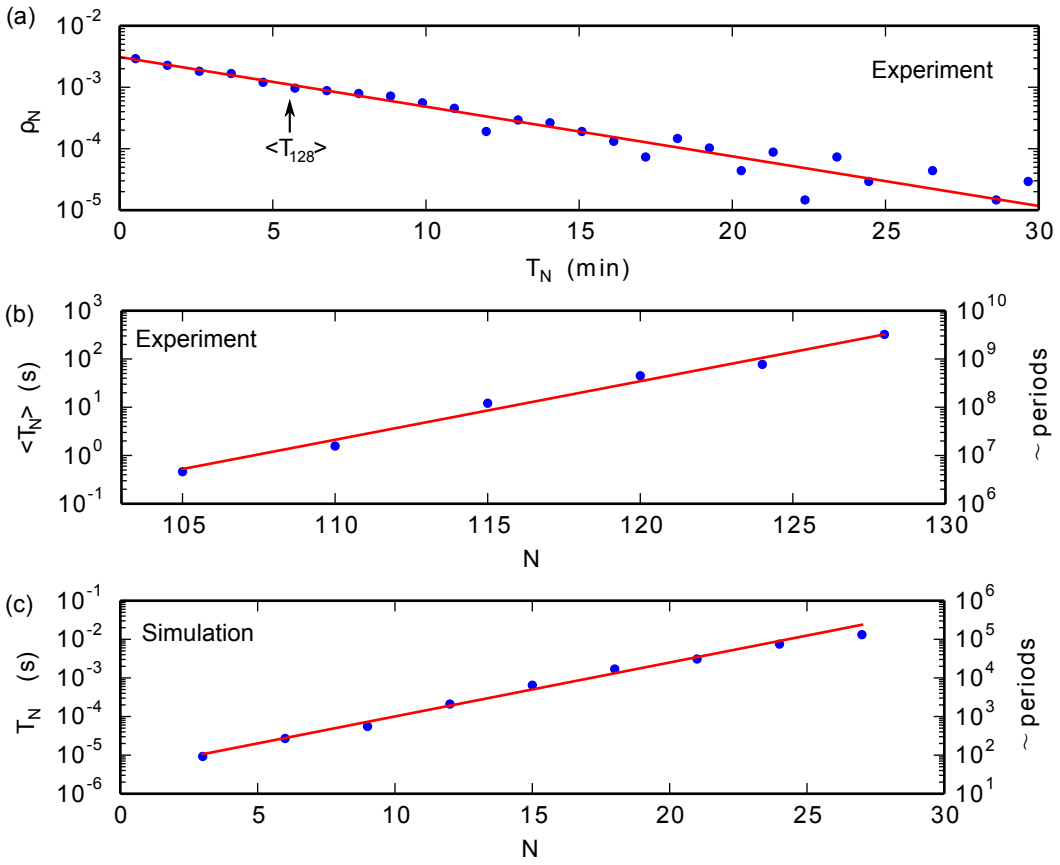


FIGURE 60: (a) Histogram of transient times T_{128} measured from 1000 experimental acquisitions (circles) and distribution function Eq. (79) (solid line). (b), (c) Average transient time $\langle T_N \rangle$ as a function of N measured from (b) 1000 experimental transients each, (c) 200 simulated transients each (circles). Both are fitted with Eq. (81) (solid line) with (b) $\kappa = 0.28 \pm 0.10$, (c) $\kappa = 0.30 \pm 0.08$. The right axis shows the approximate number of periods per transient. Experimental parameters $R/N \approx 0.24$, $\omega_0 \approx 1000/[19.7 + 2.9 \cdot R]$ (see Eq. (74)), $\tilde{\sigma} = \sigma\omega_0^2/\pi$ with $\sigma = 0.515 \pm 0.018\,\mathrm{ns}$, initial conditions as in Fig. 55. Numerical parameters are $R/N = 1/3$, $\tilde{\sigma} = 0.089\,\mathrm{MHz} \cdot 40/R$ (to adjust for the change in coupling range), $\alpha = 0.1$ and initial conditions as in Fig. 61. N in (c) is limited by available computation time.

7.6 NUMERICAL SIMULATION OF NETWORKS OF BOOLEAN OSCILLATORS

In this section, I simulate Eq. (73) numerically and compare the results with the experiments.

7.6.1 CHIMERA STATES IN THE MODEL OF BOOLEAN PHASE OSCILLATORS

In order to see chimera states in the model Eq. (73), the phase-lag parameter has to be chosen in the vicinity of $\alpha_{ij} = 0.1$, which is different from the common value of $\alpha = \pi/2 - 0.18$ used by Abrams and Strogatz to achieve chimera states [Abr04]. This may be due to the differences in the coupling function of the two models, cf. [Ome12a]. Furthermore, I also have to use a coupling range $R = 42$ ($N = 128$, $R/N \approx 1/3$) that is larger than in the experiment ($R = 30$), but agrees with values used in other models [Ome12a]. Note also that the experiment does not include self-coupling, but it is common to assume self-coupling in previous numerical studies of Kuramoto networks.

Figure 61 shows the simulated dynamics in phase and frequency representations analogously to Fig. 55. The figure shows a chimera state with co-existence of a synchronized and desynchronized domains with a typical arch in the frequency picture (see also Sec. 7.2.2.3). The dynamics is similar to chimeras in phase oscillator models that also show an arch in the frequency picture (compare to Fig. 51).

Figure 62 shows the temporal evolution of the frequency of the nodes. Similar to Fig. 56(b), the system shows a chimera state that moves in the network. After a certain time that depends heavily on the initial conditions, the chimera ends in favor of a synchronized state, which happens after 4.2 ms in Fig. 62.

The model also reproduces the characteristic scaling of the transient of Eq. (81), as shown in Fig. 60(c) with $\kappa = 0.30 \pm 0.08$, which is analogous to Fig. 60(b).

These results suggest that the model is well suited to describe the experiment qualitatively. The model is, however, only a first step towards a complete theoretical description of the experimental dynamics because of several differences in the generated dynamics.

7.6.2 DIFFERENCES BETWEEN MODELED AND EXPERIMENTAL DYNAMICS

First, different from the experiment, the simulation shows chimera states for the entire transient (see Fig. 62). A second difference is that the simulation (the experiment) collapses to a synchronized state (nearly synchronized state), where nodes are phase and frequency synchronized (nearly frequency synchronized but not phase synchronized) as discussed in Sec. 7.4.4. Third, chimera states appear in different parameter regions in model and experiment as discussed above. Finally, the initial conditions to see chimera states have to be prepared carefully in the model but can be chosen randomized in the experiment.

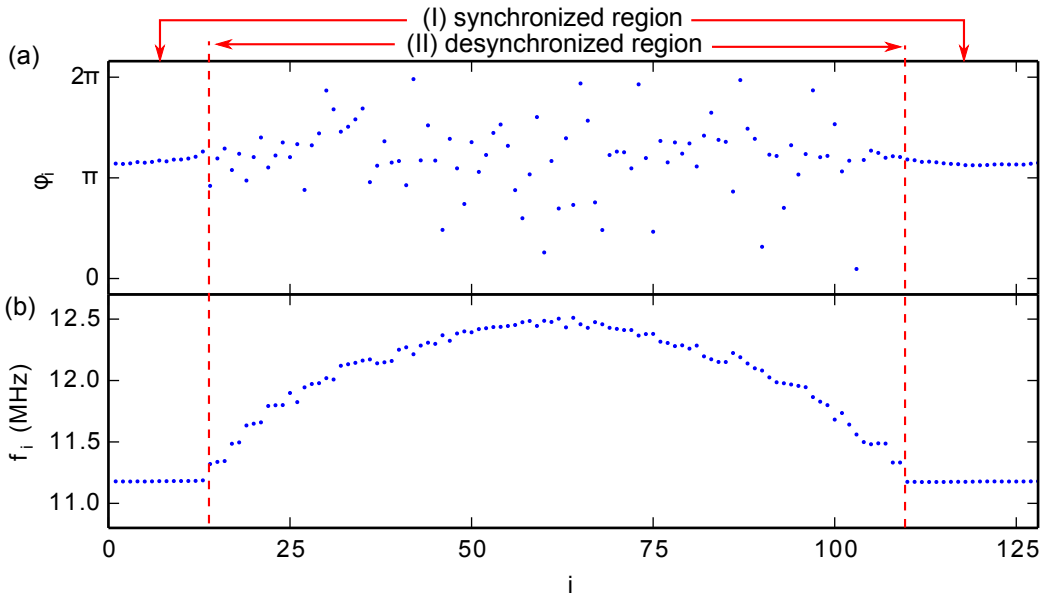


FIGURE 61: (a) Phases ϕ_i and (b) frequencies $f_i = \langle \dot{\phi}_i \rangle / (2\pi)$ of the network at $t = 50\ \mu$s. Dynamics are obtained from numerical simulation of Eq. (75) with $N = 128$, $R = 42$, $\omega_0 = 2\pi \cdot 9.3\,\text{MHz}$, $\tilde{\sigma} = 2\pi \cdot 0.089\,\text{MHz}$, $\alpha = 0.1$. Dynamics are initialized as in Ref. [Abr04] with $\phi_i = 6p \exp\left(-0.76x^2\right)$, where p is a uniform random variable on $[-0.5, 0.5]$ and $x = 2\pi i/N - \pi$. For simplicity, I do not assume frequency heterogeneity and noise in the model.

These differences may be caused by heterogeneity in the experiment $\alpha_{ij} \neq$ const, while $\alpha_{ij} =$ const is assumed in the model. Specifically, the experiment implements heterogeneous wiring, where the j-th input x_{ji}^{in} of node i [Fig. 53(c)] is given by the j-th element of the sorted version of the list $\{(i-R) \bmod N, (i-R+1) \bmod N, ..., (i+R) \bmod N\}$ (see also Sec. 7.3.3). Furthermore, differences may be caused by noise and frequency heterogeneity of 0.3%, and transmission delays along the links ($< 5\,$ns) in the experiment.

Evidence for the importance of heterogeneity in the phase-lag parameter α_{ij} is given in Appendix C.2, where I show that randomized initial conditions also lead to chimera in the simulation when α_{ij} is chosen heterogeneous. However, I was not able to obtain the resurgence of the chimera in the model by including heterogeneity.

Theoretical models do not show the observed alternations between chimera states and unsynchronized states—the so-called resurgence of the chimera. Future work has to fill this gap to uncover the underlying mechanisms.

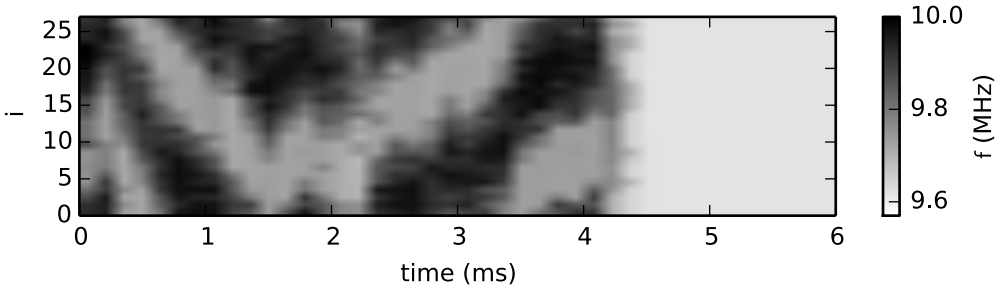


FIGURE 62: Frequency evolution from numerical simulation of Fig. 75 with parameters $N = 27$, others as in Fig. 52(c). Initial condition as Fig. 61.

7.7 CONCLUSION

In conclusion, I have proposed and studied a network of Boolean phase oscillators that approximately follows equations similar to the Kuramoto model [Kuro2a]. Large experimental networks of up to 128 non-locally coupled Boolean phase oscillators show complex dynamics, where chimera states of synchronized and desynchronized regions disappear and reappear, which is not yet theoretically understood. The dynamics collapses to a synchronized state after a long transient time, which follows an exponential scaling with the network size as previously predicted in Kuramoto oscillators [Wol11], neural networks [Zil09], and reaction-diffusion systems [Wac95b].

The measurement of the scaling of chimera states in a large network of 128 nodes is only possible because of the fast timescale of $\sim$100 ns and it is even computationally prohibitive to study the transient behavior for that many oscillators in a simulation.

8

EXCITABLE DYNAMICS IN AUTONOMOUS BOOLEAN NETWORKS

8.1 ABSTRACT

In this chapter, I use autonomous Boolean networks to realize experimental excitable systems that I refer to as Boolean neurons. I couple Boolean neurons into meta-networks that I call Boolean neural networks. After an introduction to excitability in Sec 8.2, I design and test the Boolean neuron in Sec 8.3 and couple two Boolean neurons in a small Boolean neural network in Sec 8.4.[1]

The main contribution of this chapter are:

- designing an autonomous Boolean network with excitable dynamics, which constitutes an accelerated-time artificial neuron termed Boolean neuron;
- modeling of the Boolean neuron;
- confirming experimentally theoretical results for the dynamics of neural network motifs.

8.2 INTRODUCTION TO EXCITABILITY

Excitable systems have three states: the rest state, the excited or firing state, and a refractory or recovery state. The system stays in the rest state if unperturbed or stays close to it if perturbed below a threshold. For above-threshold perturbations, the system goes through the excited state, the refractory state, and back to the rest state. This trajectory describes a nonlinear excursion through phase space, corresponding to a spike while in the excited state. While in the refractory state, the system cannot be excited until it reaches the rest state [Lin04, Izh07]. Excitability is found throughout nature, with prominent examples of the heart and biological neurons [Ade81, Plo00, Wij95]. For example, the study of arrhythmias in the heart is one major application of the theory of excitable media in biomedical science [Nas04, Wij95, Dav92a]. Excitable media are studied in model systems

1 Results of this chapter are published in Ref. [Ros12]; I have published previous work on this subject for neuronal and optoelectronic oscillators in Refs. [Pan12, Ros11a] that helped me with the analysis of the results in this chapter.

such as the Belousov-Zhabotinsky reaction, which is a chemical reaction that shows rich wave patterns [Bel59, Pet93, Ste93a]. An early study of neural excitability was conducted by Hodgkin and Huxley on the giant squid axon, which they describe as an excitable medium to explain the conduction of an action potential along the neuron [Hod52]. Excitability has also been formulated for point-like systems. For example, the one-dimensional excitable medium along the extension of a neuron can be approximated as a point-like excitable dynamical system. This approximation is frequently used for neurons that are connected to neural networks to reduce the complexity of the problem [Izh07, Day03]. Then, the finite transmission times between neurons due to the finite transmission velocities can be approximated with a delay description [Ern09]. A very descriptive network of excitable systems are fans at a sports event that can pass along a Mexican wave [Far02].

8.2.1 NEURONS

The human brain includes about 100 billion interconnected neurons that each have more than 10,000 inputs from other neurons, thereby building a huge neural network that is the core component of the nervous system. Neurons process and transmit information within this network [Izh07]. As biological cells, they can be studied on various levels, such as cell biophysics and molecular biology, but also on higher levels, such as small neural circuits, the whole brain, or the behavior of an entire organism [Izh07]. I introduce here the problem at the level of individual neurons.

Neurons are usually divided into three parts: the soma, the dendrites, and the axon, where the soma is the cell body, the dendrites receive signals and axons conduct them over large distances up to 1 m in the human body [Lev01], as shown in Fig. 63. Information is conducted in the form of spikes—the so-called action potential—which is a pulse-shaped temporal change in the electrical membrane potential of the neuron. These spikes are initiated via sufficiently strong stimuli at the dendrites and then travel away from the soma along the axon. At the axon terminals, the spikes can induce a new spike in another connected neuron via a synapse [Lev01].

Hodgkin and Huxley measured the initiation and conduction of signals along the axon experimentally in 1952 on the squid giant axon and derived a mathematical model called the Hodgkin-Huxley model [Hod52]. The model describes the membrane of axons with an equivalent electrical circuit model with resistors, capacitors, and voltage sources, which arise from ion channels in the biological membrane. The ion channels are activated according to nonlinear differential equations. Systems described with the Hodgkin-Huxley model are excitable and generate a large voltage spike when an applied stimulus is greater than a threshold. A threshold condition for firing, also known as the all-or-none principle, is a key component

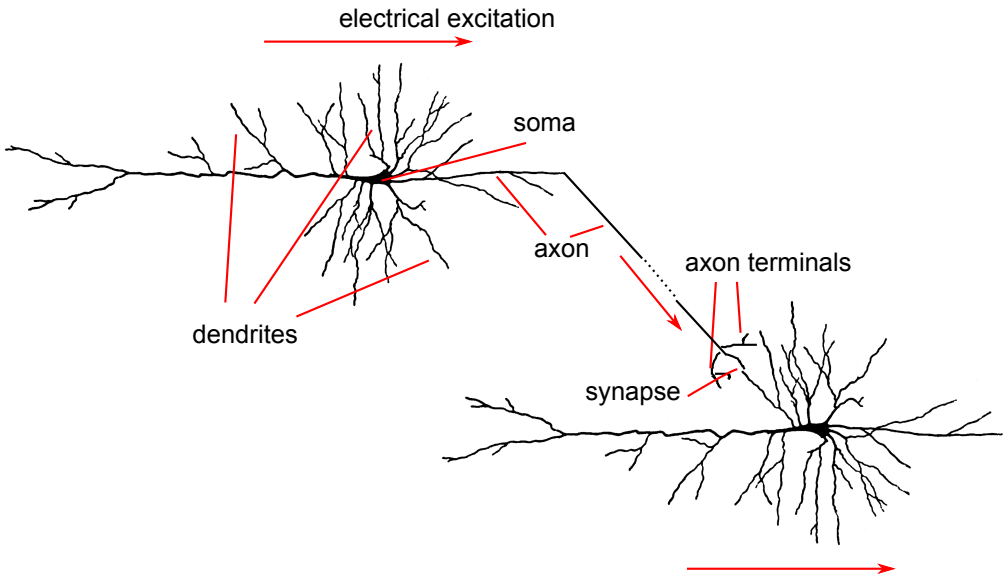


FIGURE 63: Anatomic drawing of two connected phramidal cells. The Figure is a modified version of the open source content http://en.wikipedia.org/wiki/File:Sobo_1906_33.png.

in neurological systems, but the level of the threshold often depends on the shape of the input signal and can even depend on resonance conditions of input spikes, such as a preference of certain periods for input pulse trains, which is the case in the Hodgkin-Huxley model [Izh07].

8.2.2 HODGKIN CLASSIFICATION

Hodgkin classified neurons according to their response to constant input currents of different strength [Hod48, Izh07]. Neurons usually respond to constant input excitations with a train of repeating output pulses as shown in Fig. 64(a). When the firing frequency of pulses f increases with the excitation current I, the neuron is of type I [Fig. 64(b)] and, when f is independent of I, the neuron is of type II [Fig. 64(c)]. But, in both classes, the input current has to be greater than the threshold. Type-III neural excitability corresponds to neurons that generate at most one or two action potentials in response to a constant current instead of a train of pulses. An example for type-III neural excitability is the squid giant axon, which is surprising because the Hodgkin Huxley model is of type II even though it is derived from the same system under different conditions [Cla98, Cla08].

The framework of excitability types is generalized by characterizing excitable systems by their bifurcation mechanism [Izh07, Rin98]. Excitability can be generated with the following bifurcations: subcritical Andronov-Hopf, supercritical Andronov-Hopf, saddle-node of limit cycles, saddle-node infinite-period (SNIPER), where the latter results in type-I excitability and the other bifurcations in type-II excitability [Kuz98, Izh07].

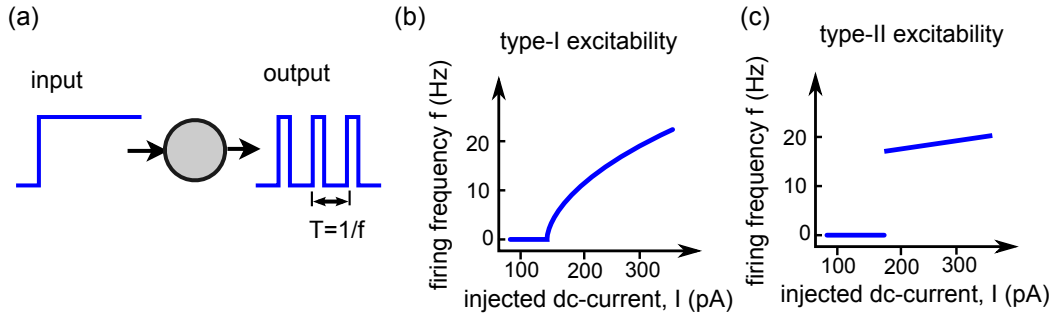


FIGURE 64: Illustration of type-I and type-II neural excitability. (a) For the measurement of the excitability type, a neuron (or a dynamical model for a neuron) is exposed to a constant stimuli with current I and responds with a train of pulses of a certain frequency f. The resulting f-I-curves for (b) type-I and (c) type-II excitability are shown schematically.

8.2.3 KEY COMPONENTS OF EXCITABILITY

Typical excitable systems have a stable fixed point in which they rest until a perturbation above a threshold excites them. In response to the stimulus, they generate large excursions in phase space that results in output spikes [Izh07]. After generating a spike, excitable systems usually recover for a certain time, the so-called refractory time. During the refractory time, excitable systems have a higher excitation threshold or cannot be excited at all, corresponding to the absolute and relative refractory period, respectively [Lev01]. Therefore, three states can be identified for excitable systems: the resting, excited, and refractory state.

These three states of excitable systems are used in the Greenberg-Hastings Cellular Automaton to model excitability, with a synchronous temporal evolution. The cellular automaton evolves for time steps $t = 1, 2, 3, ...$ according to the following rules [Gre78]: The excitable system

(i) changes from the resting state to the excited state if at least one input connection is in the excited state.

(ii) changes from the excited state to the refractory state.

(iii) changes from the refractory state to the resting state.

When evaluated on a grid, this simple model can lead to rich wave patterns [Fis93]. Similar rules to the ones used in the Greenberg-Hastings cellular automaton are used in this chapter to build a continuous-time excitable dynamical system with autonomous Boolean networks.

8.2.4 DYNAMICS OF TWO DELAY-COUPLED NEURONS

To be able to better interpret my experimental results in this chapter, I first present in this section simulation results with a standard neuron model. I

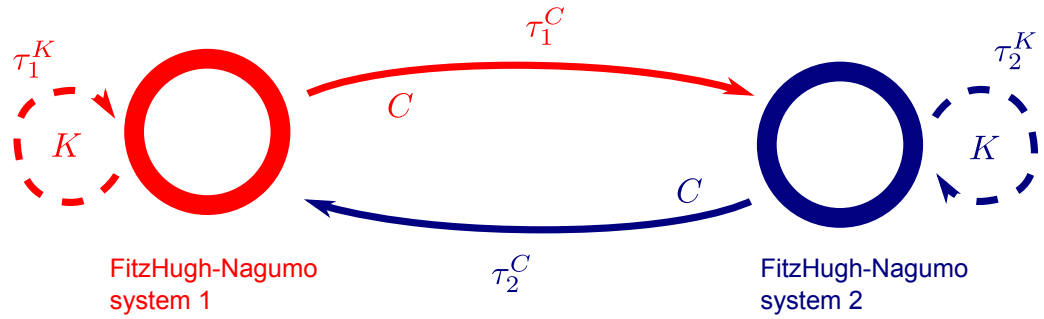


FIGURE 65: Schematic of two coupled FitzHugh-Nagumo systems that include feedback and mutual coupling.

use the same network topology studied in later sections experimentally. I start by motivating the network topology of only two neurons.[2]

8.2.4.1 *Motivation for Studying Network Motifs*

A structure of a few coupled neurons constitutes a network motif, which is a simple recurring substructure present in large networks, such as the brain [Milo02]. The intrinsic dynamics of the network motif may contribute significantly to the dynamics of the larger network and could hence be important for the overall network dynamics. For this reason, dynamics of network motifs is heavily studied [Wei14, Hau06, Cho07, Dhu08, Hoe09, Fie09, Flu09, Bra09, Dhu11, Hic11, Kyr11, Adh11].

The network motif in Fig. 65 of two coupled neurons includes time delays along the links to account for the finite propagation time of spikes along the neuron axons [Wei14, Hoe09, Pan12, Hoe10b, Dah08c, Sch08, Ros12]. This configuration can also be understood as two effective populations of neurons with internal and mutual connections [Vic08]. Synchronization in larger networks of type-I [Kea12] and type-II [Leh11] excitable elements was also studied.

8.2.4.2 *Dynamics Generated by a Standard Neuron Model*

The neurons are described by the FitzHugh-Nagumo model, which simplifies the Hodgking-Huxley model to include the essential mathematical features of excitability and has a similar structure as the Van-der-Pol oscillator (see Sec. 6.2.4.1) [Fit55, Fit61, Nag62]. It is, however, not motivated by physiological processes [Izh06]. The dynamic equations of the FitzHugh-Nagumo model are paradigmatic for neural systems with type-II excitability (see Sec. 8.2.2), where oscillations result from a Hopf bifurcation.

2 The content of this section is published in Ref. [Pan12].

The two coupled FitzHugh-Nagumo systems are described by the delay differential equations

$$\begin{aligned}
\varepsilon_1 \dot{x}_1 &= x_1 - \frac{x_1^3}{3} - y_1 + C\left[x_2\left(t-\tau_2^C\right) - x_1(t)\right] \\
&\quad + K\left[x_1\left(t-\tau_1^K\right) - x_1(t)\right], && \text{(83a)} \\
\dot{y}_1 &= x_1 + a, && \text{(83b)} \\
\varepsilon_2 \dot{x}_2 &= x_2 - \frac{x_2^3}{3} - y_2 + C\left[x_1\left(t-\tau_1^C\right) - x_2(t)\right] \\
&\quad + K\left[x_2\left(t-\tau_2^K\right) - x_2(t)\right], && \text{(83c)} \\
\dot{y}_2 &= x_2 + a, && \text{(83d)}
\end{aligned}$$

where $\varepsilon_i \ll 1$ denotes a timescale ratio between a slow inhibitor variable y_i and a fast activator variable x_i ($i = 1, 2$). The activator variables x_i correspond to the neural membrane potential and encodes the action potential as spikes. The inhibitor variables y_i can inhibit the system from emitting a pulse if y_i is large, leading to a refractory mechanism. The parameter a is known as threshold parameter. For $|a| > 1$, the uncoupled system operates in the excitable regime. The parameters K, C, τ_i^K, and τ_i^C are coupling strength and coupling delays of mutual coupling and feedback according to Fig. 65.

Figure 66 shows typical dynamics of two coupled neurons modeled with Eqs. (83a)-(83d). The arrows indicate excitations due to the different delayed coupling links. The temporal evolution of the activator x_i and inhibitor y_i is shown for two different parameter regimes. The activator x_i shows periodic pulses, forming a pulse train. The inhibitor is also periodic and peaks when a pulse in the activator ends. In Fig. 66(a) the outputs of system 1 and 2 are synchronized in phase, while in Fig. 66(b) the outputs have a phase shift of π.

Coherent spiking of this system, such as shown in Fig. 66(a) and (b), appears for identical self-coupling delays $\tau_1^K = \tau_2^K = \tau_K$ when the following condition for the time delays is fulfilled

$$N^K \tau^K = N^C 2\tau^C, \tag{84}$$

with integer numbers N^K and N^C. Coherent spiking is expected even if the coupling delays differ from Eq. (84) by an amount on the order of the pulse width of the neurons [Pan12]. The period (inter-spike interval) of the resulting self-sustained coherent is

$$T = \tau_K / N_C = 2\tau_C / N_K. \tag{85}$$

Specifically, the two systems pulse in phase (in antiphase) when N^K is even (odd) as shown in Fig. 66(a) and (b), respectively [Pan12].

For the general case of $\tau_1^K \neq \tau_2^K$, the system can display complex dynamics, such as neural bursting [Pan12].

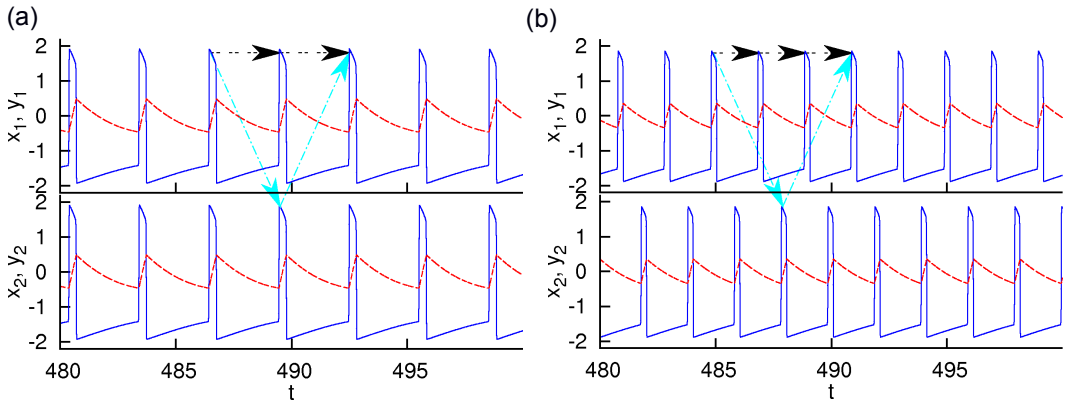


FIGURE 66: Time series of Eqs. (2) for identical self-feedback delays $\tau_1^K = \tau_2^K = \tau_K$. The activator x_i and inhibitor y_i are shown with blue (solid) and red (dashed) lines, respectively. The parameters are chosen as (a) $K = 0.5$, $\tau_K = 3$, (b) $K = 0.5$, $\tau_K = 2$. The black and light blue arrows indicate the excitations due to self-feedback and mutual feedback, respectively. Other parameters are $\epsilon = 0.01$, $a = 1.3$, $\tau_C = 3$, and $C = 0.5$. Time and activators and inhibitors are dimensionless.

8.2.5 ARTIFICIAL NEURAL NETWORKS

Artifical networks of excitable systems engineered with, for example, analog electronic circuits are called artificial neural networks and, when realized on microelectronic chips, the excitable systems are called silicon neurons [Ind11]. They are developed 1) for high-speed modeling of the biological equivalent neural network as fundamental studies and 2) to develop important applications, such as neuro-inspired computers, real-time behaving systems, and bidirectinal brain-machine interfaces [Ind11].

Silicon neurons can be realized in very large scale integration (VLSI), with an example shown in Fig. 67, where 1,024 excitable systems are implemented on a custom board including an analog-electronic chip to implement the neural dynamics and a digital microchip similar to an FPGA to manage the connections [Ind11, Arto6]. Off-the-shelf FPGAs are used because they are much cheaper than custom made chips [Ind11, Arto6].

This configuration, however, presents major hindrances due to speed limitations, the cost and long design cycle time of the custom analog-electronic chip, and its connections to the digital reconfigurable chip. Of particular concern is the fact that the analog signal is digitized in time and voltage, leading to discretization errors in the coupling.

As a solution, I develop an excitable autonomous Boolean network that can be implemented on an FPGA together with the connections, so that a single chip can be used to include both, the silicon neurons and the connections. Specifically, they can be implemented on inexpensive off-the-shelf VLSI (very large scale integration) chips, such as FPGAs resulting in an inexpensive design. The resulting Boolean neuron evolves on a fast timescale on the order of nanoseconds, which is much faster than the

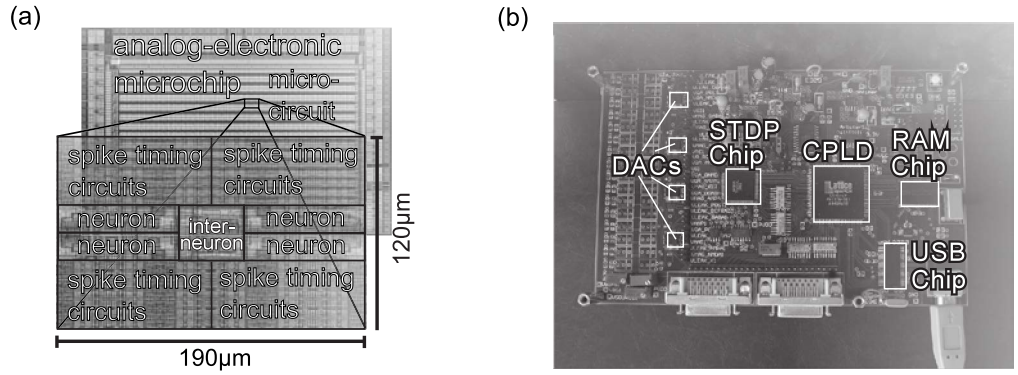


FIGURE 67: Typical implementation a network of silicon neurons as a predecessor of the neurogrid, which is designed for the simulation of biological brains with one million neurons and six billion synapses, reproduced from Ref. [Art06, Boa06]. (a) Shown is a custom-build analog-electronic chip that includes a 16-by-16 array of microcircuits of four neurons each, together with 21 timing circuits to digitize and adjust the time of a pulse. (b) Shown is a circuit board including the analog microchip in (a) called STDP (spike timing-dependent plasticity) chip, four digital-to-analog converters (DACs) are used to adjust parameters of the neurons, one CPLD (complex programmable logic device, similar to an FPGA) is used to realize the coupling topology, and a RAM and USB chip are used to acquire and transfer data.

dynamics of common silicon neurons and biological neurons; hence, it is an accelerated-time neuron. The fast timescale is advantageous for ultra-fast neuro-inspired data processing [Scho8k] such as reservoir computing [Jae04, Maa02, App11].

8.3 DESIGN OF A BOOLEAN NEURON

In this chapter, I implement Boolean neurons, which are electronic excitable systems based on autonomous Boolean networks realized with autonomous logic circuits.

8.3.1 SETUP OF BOOLEAN NEURONS

Embedded in the design of the Boolean neuron are three properties that are important properties of excitability [Izh07]:

(i) the all-or-none principle, where the system responds only if an input is above a threshold and stays quiescent otherwise;

(ii) pulse dynamics, where output pulses have fixed width independent of the input pulse shape; and

(iii) a refractory phase, where the excitable system remains unresponsive during a refractory period after generating an output pulse.

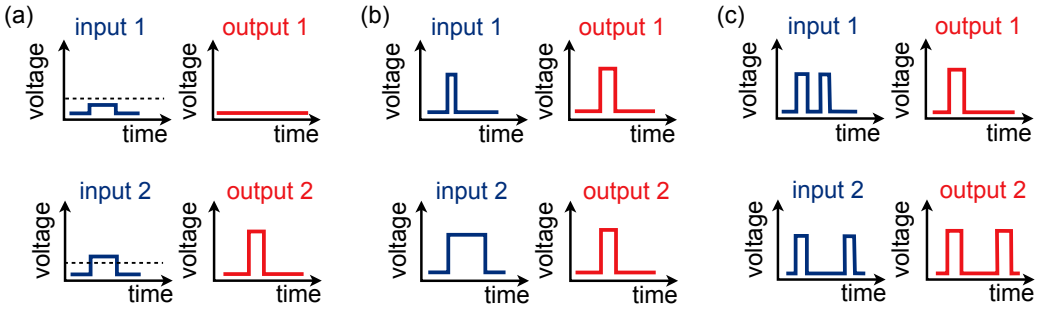


FIGURE 68: Illustration of three characteristics of excitability with typical responses to different input waveforms, where blue-colored graphs are input waveforms and the red-colored graphs are the resulting output waveforms. Two input and output waveforms each are used to demonstrate (a) threshold behavior, (b) characteristic responses (output pulses), and (c) the refractory time.

These properties are continuous-time rules for the behavior of an excitable system corresponding to the discrete-time rules of the Greenberg-Hastings cellular automaton introduced in Sec. 8.2.3. The properties are visualized schematically in Fig. 68 showing the desired response of an excitable system in the fixed-point state to external perturbations. Specifically, the all-or-none principle (i) is illustrated in Fig 68(a), where a perturbation below threshold leads to no or only a small response of the excitable system and a perturbation above threshold leads to the generation of a pulse. This pulse is independent of the length of the external perturbation according to principle (ii). This is shown in Fig. 68(b), where a narrow and a wide above-threshold signal lead to the same output pulse. Principle (iii), the refractory time, is visualized in Fig. 68(c), where two perturbations lead to only one pulse when the perturbations are close together, but they lead to two pulses when they are spaced far enough apart.

Autonomous logic gates are well suited to fulfill these properties. For example, the all-or-none principle (i) is intrinsic to logic gates [Mcc43]. Their output voltages V transition between $V = V_{\text{high}}$ (the *high* level) and $V = V_{\text{low}}$ (the *low* level) as their inputs cross the threshold voltage V_{th}. Pulse dynamics (ii) and the refractory phase (iii) can be realized through pulse generators, which exploit the intrinsic propagation delays of logic gates τ_{gate}. Non-ideal behaviors of the neuron, however, affect the dynamics so that an input pulse that is too short is rejected instead of triggering an output pulse (see also Sec. 3.2.3).

As shown in Fig. 69(a), the pulse generator is implemented using a D-type flip-flop, an inverter-based delay line of delay $\tau_n = n \cdot 0.28\,\text{ns}$ as introduced in Appendix A, and an autonomous logic gate executing the XOR operation [Jeo04]. Its dynamics consists of the generation of a single pulse of width $T_n = \tau_n$ in response to a positive edge (low-to-high transition). Specifically, in response to a positive edge at its clock (clk) input, the flip-flop, with connection from output (Q) to inverted input (D), generates a

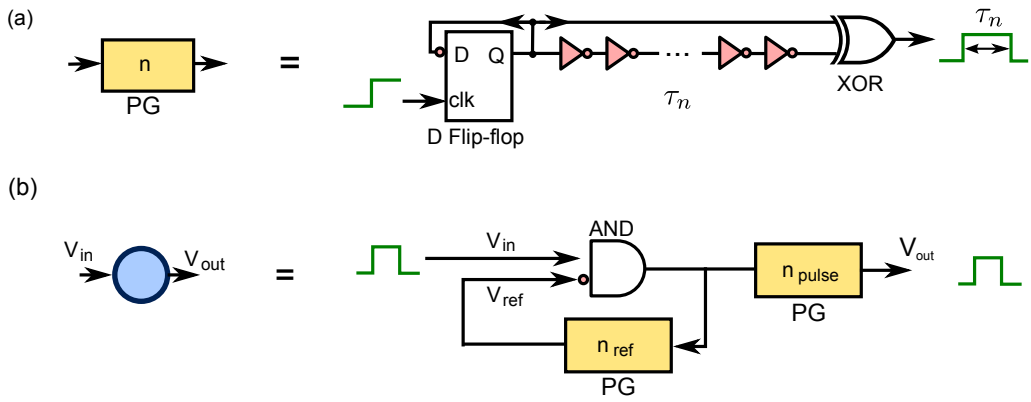


FIGURE 69: (a) Setup of the pulse generator characterized by an integer n, which represents the number of pairs of inverters in the delay line and thus its time delay $\tau_n \sim 2n\tau_{\text{gate}}$. The pairs of inverters act as time delays that do not change the Boolean state. The input of a rising edge leads to the output of a pulse. (b) Design of a Boolean neuron; it combines one pulse generator labeled n_{pulse} to realize the pulse dynamics with another one labeled n_{ref} to realize the refractory phase. The refractory period of the Boolean neuron T_{ref} and its pulse width T_{pulse} are determined by the integers n_{ref} and n_{pulse}, respectively. The voltages V_{in} and V_{ref} are inputs to an AND gate, where the second input is inverted, as indicated by a circle. The input of a pulse leads to the output of a pulse under certain conditions.

Boolean transition at its output (Q). This signal reaches the XOR gate inputs with a time-delay difference $\sim\tau_n$, due to the presence of the delay line. As a consequence, the XOR gate has different input logic values during the time delay and hence generates a high voltage V_{high} of width $T_n = \tau_n \propto n$.

I combine two pulse generators with an AND gate, as illustrated in Fig. 69(b) with a hardware description discussed in Appendix B.7.1 and B.7.2, so that the system exhibits excitable dynamics in its output voltage V_{out} in response to an above-threshold input voltage V_{in}. The pulse generator labeled n_{pulse} (n_{ref}) produces a voltage pulse V_{pulse} (V_{ref}) of width T_{pulse} (T_{ref}). The voltage V_{ref} indicates whether or not the system is in its refractory phase (V_{ref} high or low, respectively); its interplay with V_{in} governs the dynamics of the Boolean neuron.

When V_{ref} is low and V_{in} has a positive edge, the AND gate also generates a positive edge so that each pulse generator produces a pulse. V_{pulse} is sent to the output of the Boolean neuron and the high value of V_{ref} now blocks inputs V_{in} to the AND gate for the refractory period T_{ref} and hence prevents pulse generation during that time. When the refractory phase ends, *i.e.*, V_{ref} is back to a low voltage, the system becomes responsive to input excitations V_{in} again.

The design of the Boolean neuron is motivated by the dynamics of integrate-and-fire neurons [Buro6], where the membrane potential evolves as a function of its synaptic input. When inputs are present, the membrane po-

tential increases (integration) until it reaches a threshold, the condition for generating a pulse (firing). In my approach, in contrast, the Boolean neuron compares its input voltage directly to a threshold without an electronic analog of a membrane potential. Consequently, when increasing V_{in} above the switching threshold of a logic gate, oscillations start with a constant (finite) period, so that the Boolean neuron exhibits a behavior analogous to type-II excitability [Izh07] (see also Sec. 8.2.2). After generating a pulse, the membrane potential of integrate-and-fire neurons returns to a resting value and its dynamics is deactivated for a finite duration [Bur06], which is the same mechanism used in the Boolean neuron to realize a refractory period.

8.3.2 MODEL FOR BOOLEAN NEURONS

In this section, I derive a Boolean map to describe the Boolean neuron theoretically. In contrast to the experimental implementation, this model allows only for Boolean states, *i.e.*, $V \in \{V_{\text{high}}, V_{\text{low}}\}$, the low and high voltage of logic gates.

I model the three components of the setup [Fig. 69(c)], namely the AND gate and the two pulse generators, separately. First, I describe the AND gate with output signal $V_{\text{AND}}^{(j)}(t)$, where the superscript j denotes the nodes in the network. It is modeled by the map where $\wedge$ and $\neg$ denote the Boolean AND and NOT operations, respectively, and Δ is the time step of the map. The NOT operation accounts for an inverted input to the AND gate, as shown in the setup. Second, the pulse generators denoted by n_{pulse} and n_{ref} in the setup are modeled by taking the flip-flop and the delay line combined with the XOR gate into account. The flip-flop creates events after a positive edge in $V_{\text{AND}}^{(j)}$. The delay line, together with the XOR gate, results in output pulses of the two pulse generators, *i.e.*, a high voltage for the time intervals $[s, s + T_{\text{pulse}}]$ and $[s, s + T_{\text{ref}}]$, respectively, after a positive edge in $V_{\text{AND}}^{(j)}$ at time s (denoted in the following as $V_{\text{AND}}^{(j)}(s) = \text{PE}$). The combination of flip-flop, delay line and XOR gate is described mathematically by

$$V_{\text{out/ref}}^{(j)}(t) = \begin{cases} V_{\text{high}} & \text{if } \exists s \in \left(t - T_{\text{pulse/ref}}, t\right] : \\ & V_{\text{AND}}^{(j)}(s) = \text{PE} \\ V_{\text{low}} & \text{otherwise,} \end{cases} \tag{86}$$

for the two pulse generators denoted in the setup as n_{pulse} and n_{ref}, respectively. This description does not account for the processing time h of the flip-flop, so that significant discrepancies between model and experiment can be appear in certain extreme coupling cases, such as sustained excitation, and when coupling delays or pulse lengths are short compared to h, as discussed later.

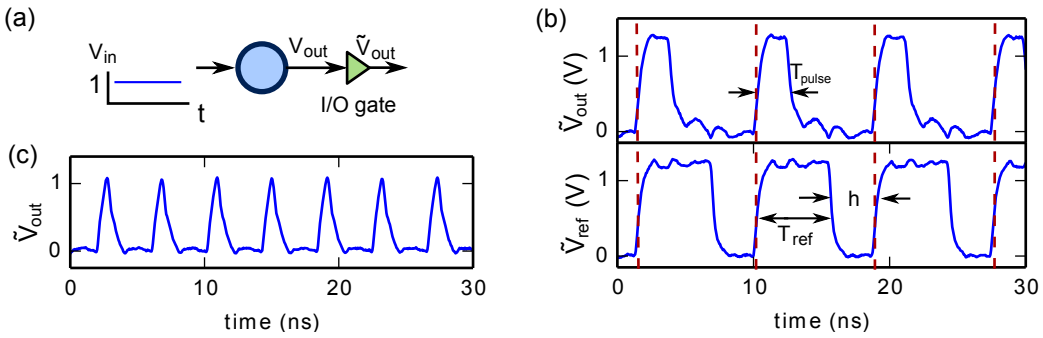


FIGURE 70: (a) A Boolean neuron on the FPGA is subject to a constant above-threshold input voltage V_{in}. The output voltage V_{out} and V_{ref} pass input-output (I/O) gates with outputs labeled $\widetilde{V}_{\text{out}}$ and $\widetilde{V}_{\text{ref}}$, respectively. (b) Both voltages are recorded for $T_{\text{ref}} = (5.40 \pm 0.05)\,\text{ns}$ ($n_{\text{ref}} = 20$), $T_{\text{pulse}} = (2.34 \pm 0.05)\,\text{ns}$ ($n_{\text{pulse}} = 8$). (c) Output $\widetilde{V}_{\text{out}}$ of a minimal implementation of the Boolean neuron with $T_{\text{ref}} = (0.68 \pm 0.04)\,\text{ns}$ ($n_{\text{ref}} = 2$), $T_{\text{pulse}} = (0.80 \pm 0.04)\,\text{ns}$ ($n_{\text{pulse}} = 2$).

8.4 DYNAMICS OF NETWORK MOTIFS OF BOOLEAN NEURONS

In the following, I study the dynamics of small networks of one and two Boolean neurons.

8.4.1 DYNAMICS OF ONE BOOLEAN NEURON

In this section, I conduct experiments on a single Boolean neuron driven first by a constant input and second by self-feedback, constituting a simple network. The experiments are realized on an Altera Cyclone IV FPGA (EP4CE115F29C7N) as introduced in Sec. 3.2. Signals generated within the FPGA pass through an additional input-output logic gate (hardwired to the output pins of the FPGA) before being acquired by a high-speed oscilloscope (DSO80804A) with 8 GHz bandwidth and 40 GSa/s sampling rate.

When a constant, above-threshold input voltage V_{in} is applied to the Boolean neuron, it generates periodic pulses with a width of a few nanoseconds [see Fig. 70(a) and 70(b)]. In this regime, the system generates periodic oscillations, similar to biological neurons with constant stimulus [Izh07]. To understand the dynamics, I analyze the output voltage V_{out} and voltage V_{ref} that indicates the refractory phase.

The pulses in V_{out} and V_{ref} are generated almost simultaneously at times indicated by vertical dashed lines in Fig. 70(b). The pulse in V_{ref} indicates the refractory phase and its pulse width equals the refractory period. Therefore, the refractory phase starts at the red dashed lines and ends when V_{ref} is low. Then, the system generates a new pulse, which is induced by a

negative edge transition in V_{ref}, since $V_{\text{in}} > V_{\text{th}}$. It requires an additional processing time h to generate the output pulse, which is due to the flip-flops in the pulse generators and is measured to be $h = (3.2 \pm 0.4)$ ns. This processing time, together with the refractory period T_{ref}, constitutes the period of the pulses in this experiment ($T = T_{\text{ref}} + h$). The pulse width in V_{out} is given by T_{pulse}. In the theory, the processing time h is not included, leading to a wrong prediction of the period.

In this setup of constant high input, the resulting pulse train generated by the Boolean neuron is determined in its period by the refractory period T_{ref} and in the output pulse width by T_{pulse}. These two quantities are determined by parameters n_{ref} and n_{pulse} according to $T_{\text{ref}} \approx n_{\text{ref}}\tau_{\text{gate}}$ and $T_{\text{pulse}} \approx n_{\text{pulse}}\tau_{\text{gate}}$, as follows from the construction of pulse generators shown in Fig. 69(a) and the delay of cascaded logic gates discussed in Appendix A. This can be seen experimentally when conducting the same experiment with different parameters n_{pulse} and n_{ref}. For example, with $n_{\text{pulse}} = n_{\text{ref}} = 2$, which constitutes a minimal number of seven logic gates, the pulse widths T_{pulse} are on a sub-nanosecond scale and the period T is dominated by the flip-flop processing delay h [Fig. 70(c)].

Experimental fluctuations in T_{ref} and T_{pulse} exist and are characterized in the two following ways. First, when comparing different measurements of T_{ref} and T_{pulse} on a single implementation, I observe temporal fluctuations of $\pm 1\%$ that originate from the experimental non-ideal effects as explained in the next paragraph. Second, when comparing the measurements of T_{ref} and T_{pulse} on several different copies of the same Boolean neuron on the same chip, I obtain a significantly larger error of $\pm 3.5\%$. The origin of this error is heterogeneity in the propagation delays from logic element to logic element.

The dynamics of the Boolean neuron is analog-like and fluctuates in pulse shape and timing. These non-ideal experimental behaviors originate from lowpass filtering, jitter, and history- and state-dependency of the propagation time delays within logic gates [Cav10, Beloob] (see also Sec. 3.2.3). The autonomous logic operation of the Boolean neurons is limited in speed only by the high-frequency cutoff of the logic gates and is much faster than common neural processors that are built in the synchronous operation.

I also investigate the behavior of a simple network consisting of a Boolean neuron with self-feedback that includes a time delay τ [Fig. 71(a)], *i.e.*, $V_{\text{in}}(t) = V_{\text{out}}(t - \tau)$ [Fos96]. The time delay accounts for non-instantaneous transmission times along network links (for example, the propagation time along axons connecting different areas of the brain [Kee09]).

The delayed feedback link is realized as shown in Fig. 71(b) with $n_\tau = 80$ cascaded pairs of inverter gates, each imposing its propagation delay to the path, leading to a total time delay of $\tau = (21.3 \pm 0.5)$ ns, as characterized in Fig. 69(b). The delayed feedback signal and an initial stimulus are both applied to the Boolean neuron. As Boolean neurons have only one input, an

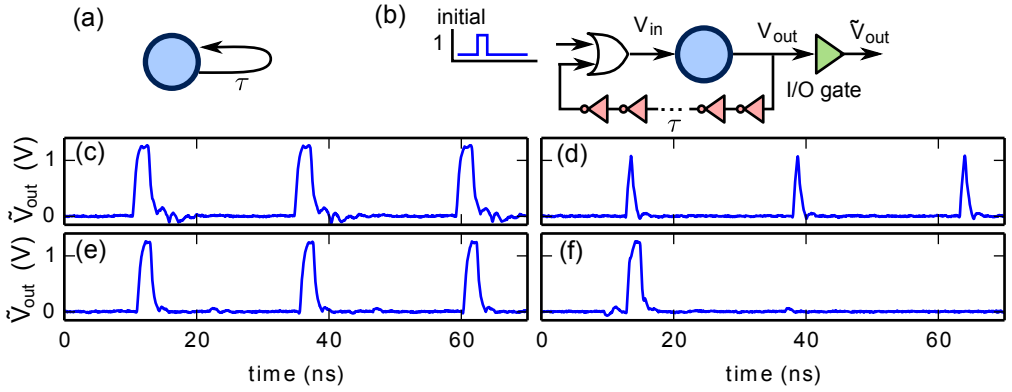


FIGURE 71: (a) Illustration of one Boolean neuron with delayed feedback where the delayed feedback link is represented as an arrow. (b) Representation of (a) with logic elements used for the feedback. The pink triangles with circles represent 80 inverter gates that are incorporated to implement a time delay of $\tau = (21.3 \pm 0.5)\,\mathrm{ns}$ ($n_\tau = 80$). (c),(d) Resulting dynamics with parameters as in Fig. 70(b) and (c), respectively. (e) Same with $T_{\mathrm{ref}} = (10.75 \pm 0.05)\,\mathrm{ns}$ ($n_{\mathrm{ref}} = 40$), $T_{\mathrm{pulse}} = (1.95 \pm 0.03)\,\mathrm{ns}$ ($n_{\mathrm{pulse}} = 8$). (f) Same with $T_{\mathrm{ref}} = (24.04 \pm 0.05)\,\mathrm{ns}$ ($n_{\mathrm{ref}} = 90$), $T_{\mathrm{pulse}} = (2.05 \pm 0.05)\,\mathrm{ns}$ ($n_{\mathrm{pulse}} = 8$).

OR gate is used to combine the two signals. The OR operation allows both signals to excite the node, but also other logic gate operations to combine inputs are possible, as discussed later. Here, the system is operated in its excitable regime.

When no initial stimulus is applied, the feedback system rests in a stable quiescent state. When a pulse of width $(1.6 \pm 0.1)\,\mathrm{ns}$ is injected once, the system generates a periodic pulse train as shown in Fig. 71(c). Initializations with multiple pulses result in similar pulse trains with shorter periods.

The dynamics arise from the delayed feedback. When a pulse is generated by the system, it travels through the delay line during τ. Then, it is input to the node to generate another output pulse after the processing time (system response time) h. Therefore, the period of the pulses is $T = \tau + h$ for this coupling scheme, which is confirmed by the experiment.

This behavior is reproduced for systems with parameters $n_{\mathrm{ref}} = n_{\mathrm{pulse}} = 2$ that have a shorter refractory period and small pulse widths [Fig. 71(d)] and for systems with parameters $n_{\mathrm{ref}} = 40$ and $n_{\mathrm{pulse}} = 8$ that have a longer refractory period [Fig. 71(e)]. However, when the refractory period is increased further to $n_{\mathrm{ref}} = 90 > n_\tau = 80$, i.e., $T_{\mathrm{ref}} > \tau$, self-sustained pulsing dynamics are no longer stable solutions of the system [Fig. 71(f)]. Instead, the system responds only to the initial stimulus and then returns to the quiescent state. The reason for this behavior is that the delayed feedback of the first response fall in the, for these parameters long, refractory period of the Boolean neuron, which cannot generate pulses during this time.

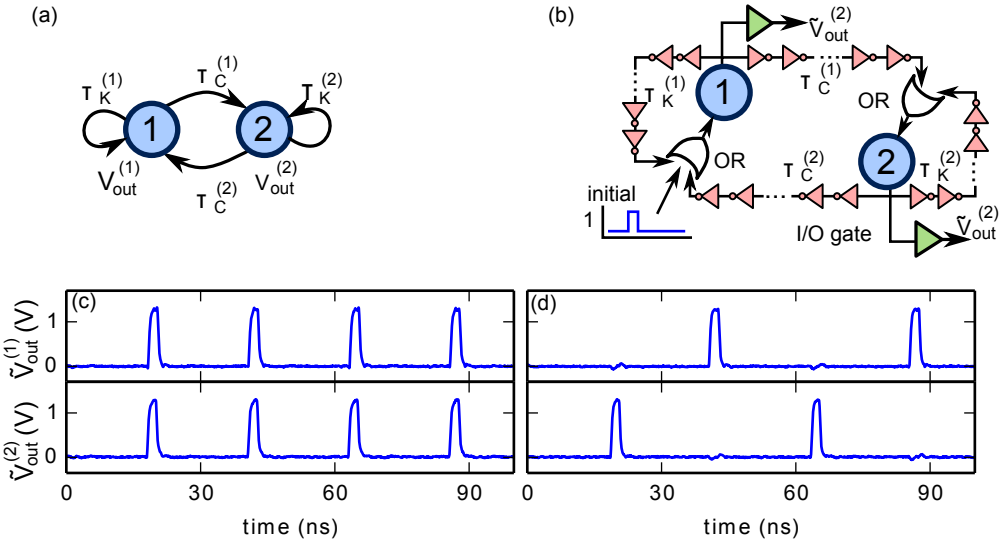


FIGURE 72: (a) Illustration of two Boolean neurons with delayed mutual coupling and delayed feedback where delay links are represented as arrows. b) Representation with logic elements used for the coupling. The pink triangles with circles represent inverter gates that are incorporated in numbers $n_{\tau,C}$ and $n_{\tau,K}$ to adjust the delay times indicated on the links. Node parameters are $n_{\text{pulse}} = 8$ ($T_{\text{pulse}} = (2.1 \pm 0.2)\,\text{ns}$) and $n_{\text{ref}} = 20$ ($T_{\text{ref}} = (5.3 \pm 0.2)\,\text{ns}$). (c) Stable output of both nodes with coupling delays realized with $n_{\tau,C} = n_{\tau,K} = 80$ pairs of inverters leading to link delays $\tau_C^{(1)} = (21.6 \pm 0.2)\,\text{ns}$, $\tau_C^{(2)} = (21.7 \pm 0.2)\,\text{ns}$, $\tau_K^{(1)} = (21.6 \pm 0.2)\,\text{ns}$, $\tau_K^{(2)} = (21.4 \pm 0.2)\,\text{ns}$. (d) Same as (c) with $n_{\tau,C} = 80$, $\tau_C^{(1)} = (22.1 \pm 0.2)\,\text{ns}$, $\tau_C^{(2)} = (21.2 \pm 0.2)\,\text{ns}$, $n_{\tau,K} = 160$, $\tau_K^{(1)} = (43.0 \pm 0.4)\,\text{ns}$, and $\tau_K^{(2)} = (43.5 \pm 0.4)\,\text{ns}$. Electrical cross talk between the node outputs is visible as small oscillations near the noise floor. Additional logic gates are used to measure the link delays of this specific implementation.

8.4.2 DYNAMICS OF TWO DELAY-COUPLED BOOLEAN NEURONS

In this section, I study a network of two delay-coupled Boolean neurons with delayed feedback, as shown schematically in Fig. 72(a) with a hardware description discussed in Appendix B.7.3. This structure is motivated in Sec. 8.2.4.1. The coupling and feedback delays are denoted by $\tau_C^{(1,2)}$ and $\tau_K^{(1,2)}$, respectively.

Here, I study the parameter regime of synchronization between the two nodes, as modeled with the FitzHugh-Nagumo system in Sec. 8.2.4. Specifically, I test, whether the network of Boolean neurons leads to the same spiking pattern for the same relation of delays.

To test the simulation results experimentally, I realize this coupling topology with a setup shown in Fig. 72(b). I use $n_{\tau,C}$ and $n_{\tau,K}$ pairs of inverter gates to create the delay lines for coupling and feedback satisfying

$\tau_C^{(1)} \approx \tau_C^{(2)}$ and $\tau_K^{(1)} \approx \tau_K^{(2)}$, respectively; thus, the same number of inverter gates are employed in both cases. However, these delays are not exactly equal because of heterogeneity in the logic gates' propagation delays.

Furthermore, I use two- and three-input logic gates to combine the two delay lines and connections for external stimuli at the input of nodes. For that purpose, I use OR gates, so that any pulse at the input of this logic gate will be passed to the Boolean neuron. However, for larger networks, an N-input logic gate that combines N inputs to a Boolean neuron can be defined as desired using a 2^N-entry look-up table. This so-called synapse (when the Boolean neuron is considered the soma of a silicon neuron, see also Sec. 8.2.1) [Ind11] allows for implementing inhibitory and excitatory connections and also to vary the coupling strength. The coupling strength is understood as the number of high inputs required for the "synapse" to pass on a pulse to the "soma" (Boolean neuron).

This setup with delay lines and synapses as described above shows coherent spiking in Figs. 72(c) and (d), when perturbed with a single pulse out of the quiescent state. The numeric values for the delays satisfy $\tau_C \approx \tau_K \approx 22\,\text{ns}$ ($N_C = 1$, $N_K = 2$) and $\tau_K \approx 2\tau_C \approx 44\,\text{ns}$ ($N_C = 1$, $N_K = 1$) for Fig. 72(c) and (d), respectively. With these two numerical values, I expect from Eq. (85) oscillations with period T of 22 ns and 44 ns, respectively. This behavior is found approximately in the experiment, where in-phase and anti-phase oscillations are seen with periods of $T = (23.0 \pm 0.2)\,\text{ns}$ and $T = (44.8 \pm 0.2)\,\text{ns}$, respectively. For both sets of parameters, I observe small mismatch ($< 5\%$) between experiment and theory, likely due to the large processing time h.

8.4.3 SIMULATION OF THE DYNAMICS OF NETWORK MOTIFS OF BOOLEAN NEURONS

I integrate numerically the dynamics of the three experiments of delay-coupled Boolean neurons with the model of the Boolean neuron introduced in Sec 8.3.2. For the first experiment of one Boolean neuron with constant input, I set $V_{\text{in}} = V_{\text{high}}$ in the model, which results in dynamics shown in Fig. 73(a). The second experiment with delayed feedback of a single node is modeled with $V_{\text{in}}(t) = V_{\text{out}}(t - \tau)$ [Fig. 73(b)]. Due to the delayed feedback, the theoretical description is a Boolean delay equation [Ghi85], which requires an initial history function for initialization. Here, I initialize $V_{\text{in}}(t)$ with a pulse on the interval $[-\tau, 0]$. Finally, the third experiment is modeled in Fig. 73(c) with $V_{\text{in}}^{(1)}(t) = (V_{\text{out}}^{(1)}(t - \tau_{\text{K}}) \vee V_{\text{out}}^{(2)}(t - \tau_{\text{C}}))$ and $V_{\text{in}}^{(2)}(t) = (V_{\text{out}}^{(2)}(t - \tau_{\text{K}}) \vee V_{\text{out}}^{(1)}(t - \tau_{\text{C}}))$, where $\vee$ indicates the OR operation. As above, I initialize the system with a pulse input to one Boolean neuron.

The dynamics generated by the map is similar to the experiment in the overall picture, but in detail the waveforms differ, as the experiment shows

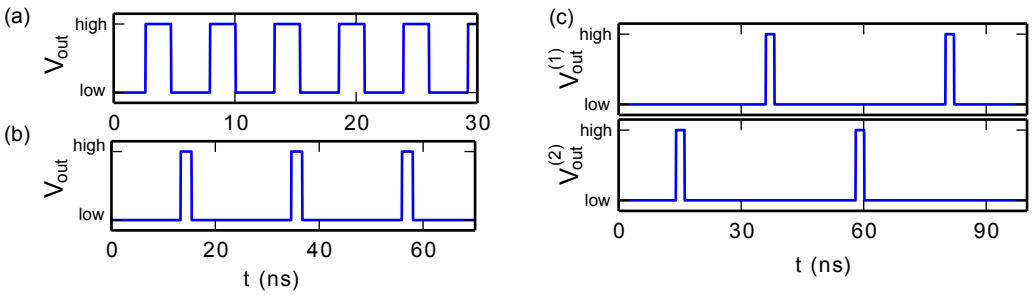


FIGURE 73: Simulation of Boolean map for $T_{\text{pulse}} = 2.1\,\text{ns}$, $T_{\text{ref}} = 5.3\,\text{ns}$. (a) Output of one node with constant stimulus, corresponding to Fig. 70(b). (b) One node with delayed feedback, corresponding to Fig. 71(c) with $\tau = 21.3\,\text{ns}$. (c) Two delay-coupled nodes with delayed feedback, corresponding to Fig. 72(d) with delays $\tau_C = 22\,\text{ns}$, $\tau_K = 44\,\text{ns}$. The model is integrated with a time step $\Delta = 0.01\,\text{ns}$.

imperfections, such as amplitude and timing noise and low-pass filtering effects. To capture these effects, I could use a piecewise-linear switching model as proposed by Glass and collaborators [Mes97, Gla98, Edw00], introduced in Sec. 2.3.3, and extended in Sec. 4.4.2; this model could be further extended with stochastic driving terms to capture most of the non-ideal behaviors of the system.

8.5 CONCLUSION

In this chapter, I have proposed and built Boolean neurons based on autonomous Boolean networks and implemented with electronic logic circuits. The Boolean neurons can be controlled in pulse width and refractory period. A single Boolean neuron responds to a one-time stimulus with a single pulse and, when connected to a time-delayed network, the Boolean neurons show self-sustained oscillations with phases and periods that are controlled by the coupling delays.

The Boolean neurons display basic type-II excitable dynamics that is not as rich as the dynamics from silicon neurons built with custom analog electronic circuits, which can display dynamics almost identical to biological neurons [Ind11]. However, my approach has the advantage that Boolean neurons can be implemented with logic circuits and do not rely on custom analog components.

The logic gates and, hence, also the Boolean neurons operate at timescales on the order of nanoseconds, which is six to nine orders of magnitude faster than common silicon neurons that operate on a timescale of seconds, and a thousand times faster than the fastest accelerated-time silicon neurons [Ind11].

Not only because of their speed, the Boolean neuron may become invaluable for neuro-inspired computing, such bio-inspired data-processing

and machine learning. The Boolean neurons may be especially suitable for network-based data-processing called reservoir computing [Jae04, Sch08k] because of its fast nanosecond time-scale, the possibility to implement large networks, and the possibility to combine them with a conventional processor as a system on a Chip (SoC). A processor is needed for a linear operation at the output of the reservoir.

While I have coupled Boolean neurons in this chapter into very small networks of only two elements, I explore in the next chapter the dynamics resulting from larger networks and compare the resulting dynamics to theoretical predictions.

9 CLUSTER SYNCHRONIZATION IN BOOLEAN NEURAL NETWORKS

9.1 ABSTRACT

This chapter focuses on the dynamics of spiking neural networks built with the Boolean neurons introduced in Ch. 8. I first introduce preceding work on the dynamics of spiking neural networks with realistic neuron models in Sec. 9.2 and discuss the master stability function and the tool of the greatest common divisor (GCD) in Sec. **??**. Then, I present experimental results of the dynamics of networks of Boolean neurons in Sec. **??**-9.8. Specifically, I show the occurrence of cluster synchronization, which is a network dynamics where the network can be separated into groups of synchronized dynamics, where nodes from different groups are not synchronized. This state is achieved in interconnected ring networks of Boolean neurons (Sec. **??**), breaks down under certain scalings of internal timescales (Secs. 9.5 and 9.6), and can be controlled using a small number of nodes in the network (Sec. 9.7). These results are also reproduced with a model (Sec. 9.8).[1]

The main contribution of this chapter are:

- realizing experimentally networks of 32 Boolean neurons showing cluster synchronization on a logic chip;
- pointing out limitations—specifically a breakdown—of a common network theory for cluster synchronization;
- discovering of a control mechanism of cluster network dynamics.

9.2 DYNAMICS OF SPIKING NEURAL NETWORKS

In spiking neural networks, information between neurons is transferred in forms of spikes, generated, for example, by the Hodgkin-Huxley model or the FitzHugh-Nagumo model or by the previously introduced Boolean neuron. The temporal location of spikes is not limited to discrete time steps but can appear at any continuous time. One fundamental question studied

1 Results of this chapter are published in Ref. [Ros13].

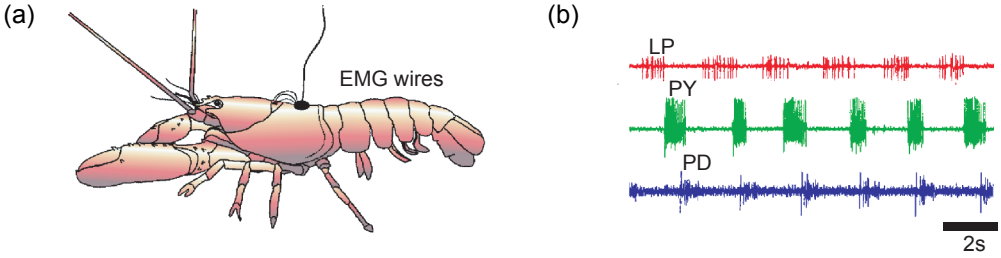


FIGURE 74: (a) Measurement to acquire motor patterns from the lobster stomach with electromyographic (EMG) recording in the behaving animal. (b) Resulting motor patterns that resemble those recorded in vivo from the lobster stomach. The patterns have been measured from three pyloric neurons denoted LP, PY, and PD. Figure from Ref. [Mar01].

with spiking neural networks is to determine whether neurons communicate by a temporal code [Maa01]. One answer to this questions has been given by studies on the central pattern generator.

The central pattern generator is a biological neural network that generates complex spiking patterns (temporal codes) to control (in other words, communicate) rhythmic motor behaviors, such as walking, breathing, flying, and swimming [Mar01, Ste99, Sel10]. For example, Fig. 74 shows the motor patterns that control the stomach muscles of a lobster. Similar patterns are also generated by a neural circuit in vitro [Mar01]. The generated patterns have been found to depend on the network refractory time [Har12]. The complex synchronized neural spiking patterns in the central pattern generator are not yet fully understood.

9.2.1 ZERO-LAG CLUSTER SYNCHRONIZATION

In neural networks, time delays result from the time it takes for neural pulses to propagate along the axons introducing several tens of milliseconds of latency, which is significantly larger than the duration of the action potential ($\lesssim 1\,\mathrm{ms}$) [Rin94]. Time delays have been found to influence the dynamics considerably and to increase the complexity of numerical simulations and analytical studies of the systems to a large extent. Consequently, effects due to time delays have attracted great attention in the studies of neural networks [Ros05, Hau07a, Mas08, Fri09a, Leh11, Kan11a]. Astonishingly, even between distant parts of the brain that involve large signal transmission delays, synchronization of neural activity without a time lag has been observed [Roe97, Rod99, Fri97a, Sch06i, Var01]. This striking dynamical phenomenon is known as zero-lag synchronization in time-delay networks [Roe97, Vic08, Fis06, Dah12, Ros05, Hau07a, Mas08, Mas09a, Sen09a, Leh11, Pop11]. Synchronization of neural activity is important because, on one hand, it has been shown to lead to pathological states, such as Parkinson's disease or epilepsy; on the other hand, it has also been shown to

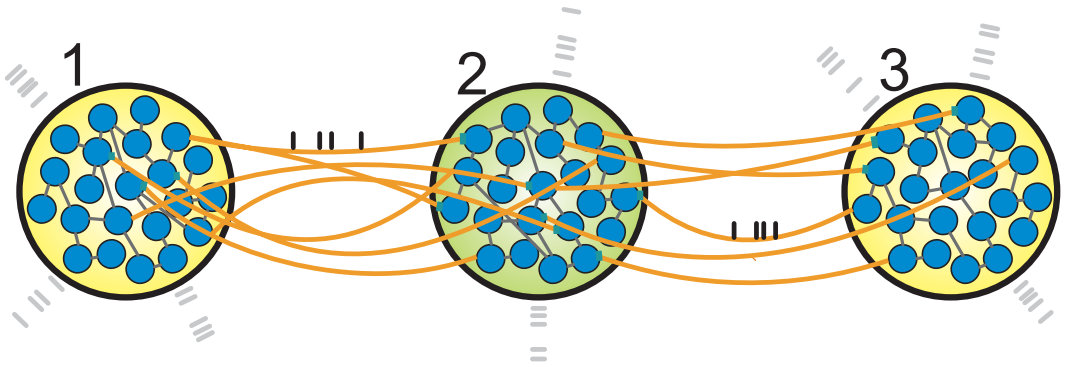


FIGURE 75: Visualization of a network of three neural populations, modified from Ref. [Vic08]. Each population includes multiple neurons, shown as circles with multiple connections within a population and a few long-range connections to other populations.

be beneficial for recognition, learning, or neural information processing [Sch11e, Uhl06].

An extension of zero-lag synchronization is zero-lag cluster synchronization, where the network separates into groups of neurons, where all neurons in a group are synchronized with each other, but neurons of different groups are not synchronized. This is illustrated in Fig. 75 with a network of three neural populations, where the spiking dynamics are separated into two independent synchronized clusters. Specifically, the neurons in population 1 and 3 are synchronized with each other and neurons in population 2 are synchronized, but neurons in population 2 are not synchronized with neurons from population 1 and 3. With this network, Vicente and collaborators have shown dynamical relaying, which is a special case of cluster synchronization, where population 2 relays the synchronization of populations 1 and 3 without being synchronized with them [Vic08]. Group synchronization is a generalization of cluster synchronization where nodes of different dynamics show synchronization in groups [Dah12].

9.2.2 NEURAL TOPOLOGIES OF CONNECTED RING NETWORKS

Studies on neurological networks, such as in C. *elegans*, found recurring topological structures of nodes assembled in loops (or rings) with directed connections [Mil02]. Kanter and collaborators have shown that such networks of connected ring structures can show cluster synchronization for a wide variety of node dynamics, such as excitable, periodic, and chaotic dynamics [Kan11a, Kan11, Nix12, Var12]. A typical network of connected loops is shown in Fig. 76, which includes 32 nodes that are assembled in four directed loops of 8, 10, 12, and 16 nodes. The loops are interconnected because several nodes belong to multiple loops.

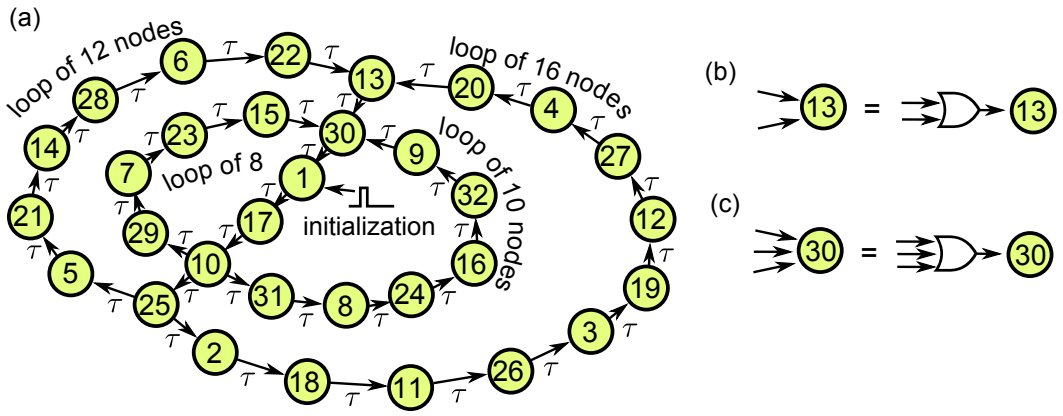


FIGURE 76: (a) A network of several connected loops with unidirectional, time-delay links indicated by arrows. The circles represent network nodes, which are labeled to identify them in the following figures. The node that receives the initial pulse is also indicated. (b), (c) In the electronic realizations, multiple input connections are combined by 2- and 3-input OR gates before sending them to the Boolean neuron that has only a single input.

9.3 THEORETICAL TOOLS TO DETERMINE CLUSTER SYNCHRONIZATION IN NETWORKS

The large interest in these problems has motivated researchers to develop analytical tools for network synchronization. Two of these tools are the master stability function and the theory of the greatest common divisor (GCD) for cluster synchronization in connected ring networks. While the first tool is very general and widely known, the second approach is the tool of choice in this chapter because of its simplicity and because it is specific to cluster synchronization in networks with time-delay links.

9.3.1 MASTER STABILITY FUNCTION FOR NETWORK SYNCHRONIZATION

In 1998, Pecora and Carroll developed the master stability function, which is a mathematical tool that separates the dynamics of identical nodes from the network topology [Pec98, Pec90]. The master stability can be solved to determine which topologies have a stable synchronized state. The master stability approach has been generalized to study systems with constant delay along the links [Kin09, Cho09, Flu10, Hei13] and to assess the stability of group and cluster synchronization [Sor07, Dah12].[2]

2 This section is written in accordance with Ref. [Flu11].

9.3.1.1 *Linear Stability Analysis of Networks*

The master stability function is a tool to determine the stability of the synchronous solution of networks of N identical nodes with time delay τ [Flu11, Pec98]

$$\frac{\mathrm{d}}{\mathrm{d}t}x_i(t) = f\left[x_i(t)\right] + \sum_{j=1}^{N} g_{ij} h\left[x_j(t-\tau)\right], \tag{87}$$

where the $x_i \in \mathbb{R}^n$ with the dimension n of individual nodes represent the state of $i = 1, 2, ..., N$ nodes, $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ represents the local dynamics of each node, $g_{ij} \in \mathbb{R}$ is the coupling matrix that determines the topology and strength of the links, and $h(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ is the coupling function that encodes which components of x_i are coupled. A synchronized solution $x_1 = x_2 = ... = x_N = x_s$ can only exist, if all nodes evolve according to the same differential equations in this state, which requires

$$\sum_{j=0}^{N} g_{ij} = \sigma \tag{88}$$

for all i, referred to as equal row sum with a constant σ. The synchronized solution evolves then according to

$$\frac{\mathrm{d}}{\mathrm{d}t}x_s(t) = f\left[x_s(t)\right] + \sigma h\left[x_s(t-\tau)\right], \tag{89}$$

which is called the synchronization manifold (see also Sec. 4.2.2).

I calculate the stability of the synchronized solution by considering the temporal evolution of small perturbations $\xi_i(t)$ on the synchronous solution for all individual systems

$$x_i(t) = x_s(t) + \xi_i(t). \tag{90}$$

When all $\xi_i(t)$ transverse to the synchronization manifold decay over time, the synchronous solution is stable under small perturbations. To find the dynamics of $\xi_i(t)$, I insert Eq. (90) into Eq. (87) and linearize, leading to

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi_i(t) = Df\left[x_s(t)\right]\xi_i(t) + \sum_{j=1}^{N} g_{ij} Dh\left[x_s(t-\tau)\right]\xi_i(t-\tau), \tag{91}$$

where Df and Dh are the Jacobians. In the following, it is convenient to rewrite Eq. (91) in a vector format as a vector of vectors, using

$$\Xi(t) = \left(\xi_1(t), \xi_2(t), ..., \xi_N(t)\right), \tag{92}$$

leading to

$$\frac{\mathrm{d}}{\mathrm{d}t}\Xi(t) = I_N \otimes Df\left[x_s(t)\right]\Xi(t) + g \otimes Dh\left[x_s(t-\tau)\right]\Xi(t-\tau), \tag{93}$$

where $\otimes$ is the direct product, I_N is the N-dimensional identity matrix, $g = (g_{ij})$. The coupling matrix g can be diagonalized when bidirectional links are assumed, leading to a symmetric g. On the other hand, network topologies with unidirectional links do not necessary allow this step. Using a unitary transformation U, g can be diagonalized with its eigenvalues on the diagonal

$$UgU^{-1} = \mathrm{diag}(\sigma, \gamma_1, \gamma_2, ..., \gamma_{N-1}). \tag{94}$$

The first eigenvalue of g is the row sum σ corresponding to the eigenvector $(1, 1, ..., 1)$ that describes the longitudinal dynamics in direction of the synchronization manifold, also known as the Goldstone mode. The remaining eigenvalues γ_k are called the transverse eigenvalues of g. The unitary matrix U can be used to block-diagonalize Eq. (93), where each of the N eigenvalues corresponds to an $n \times n$ block in a block-diagonalization of the $nN \times nN$ matrix $g \otimes Dh$. The block-diagonalization only affects the last term in Eq. (93) because the other two terms are already block diagonal. This leads to the following N equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi(t) = Df\left[x_s(t)\right]\xi(t) + \sigma Dh\left[x_s(t-\tau)\right]\xi(t-\tau), \tag{95}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi(t) = Df\left[x_s(t)\right]\xi(t) + \gamma_k Dh\left[x_s(t-\tau)\right]\xi(t-\tau), \tag{96}$$

with $k = 1, ..., N-1$, where ξ is the dynamics of the perturbations in the direction corresponding to the eigenvalues σ and γ_k given by the network topology g. Specifically, ξ in Eq. (95) corresponds to the growth of perturbations longitudinal to the synchronization manifold and hence does not determine stability of the synchronized network state, but determines the temporal complexity of the synchronized state.

The $N-1$ Eqs. (96), on the other hand, determine the stability of the synchronized state transverse to the synchronization manifold. The synchronized state is stable if ξ decays for all k, so that the maximum Lyapunov exponent calculated from the variational Eqs. (96) is negative for all eigenvalues γ_k (see also Sec. 4.2.1 for the notion of the Lyapunov exponent). The Lyapunov exponents are calculated numerically for the different γ_k of the network topology.

9.3.1.2 *Master Stability Function*

The breakthrough idea of Pecora and Carroll was to determine the maximum Lyapunov exponent of Eqs. (96) for a regime of eigenvalues $\gamma_k = \alpha + i\beta$ independent of the network topology according to the variational equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi(t) = Df\left[x_s(t)\right]\xi(t) + (\alpha + i\beta)Dh\left[x_s(t-\tau)\right]\xi(t-\tau), \tag{97}$$

resulting in the function $\Lambda(\alpha + i\beta)$, which is calculated numerically for a large and finely sampled domain in $\mathbb{C}$. Then, the eigenvalue spectrum γ_k of any topology g can be used to test $\Lambda(\gamma_k) < 0$ for all k. If this is the case, the synchronized state is stable for this specific topology. Therefore, for any topology, the determination of stability is simple once the master stability function is calculated.

9.3.1.3 *Assumptions and Limitations of the Master Stability Function*

The approach with the master stability function makes the following assumptions:

- The system is of the form of Eq. (87), which, for example, does not allow for nonlinear interaction of coupling and system dynamics or history- or state-dependent delays;
- All oscillators, link time delays, and coupling functions $h(\cdot)$ are identical;
- the coupling topology g has equal row sum and is diagonalizable;
- the perturbations ξ are small so that the linear approximation holds.

These assumptions limit its applicability to real-world networks, which are often not homogeneous [Pec98]. In my experiments, for example, some of the assumptions above are not fulfilled. First, the experiment includes heterogeneity in the time delays and node dynamics. Second, the topologies in my study are not diagonalizable. Consider, for example, the topology in Fig. 76, where nodes with label 25, 29, 31 are all driven by node 10. Therefore, there will be three rows in g that are identical, which limits the rank g; hence, g is not diagonalizable. Third, the system equation cannot be converted in the form of Eq. (87) because the model of the Boolean neurons includes a switching condition similar to a Boolean delay equation (see Sec. 8.3.2).

Because of these limitations, I use in this chapter a different approach with the greatest common divisor, which is tailored for cluster synchronization states in time-delay connected ring networks as shown in Fig. 76.

9.3.2 GREATEST COMMON DIVISOR FOR CLUSTER SYNCHRONIZATION

Kanter and collaborators showed that connected ring networks as discussed in Sec. 9.2.2 have a stable cluster synchronized state when the dynamics of the nodes is excitable and the links have time delays τ that are all equal [Kan11a, Kan11, Nix12, Var12]. Under this assumption of homogeneous link time delays and homogenous node dynamics, the network relaxes to a

cluster synchronization state when initialized with a single stimulus to one node (a pulse). The number of clusters n_C is given by the greatest common divisor (GCD) of the number of nodes in each loop l_i ($i = 1, ..., n_L$ with the number of loops n_L)

$$n_c = \mathrm{GCD}(l_1, l_2, ..., l_{n_L}), \tag{98}$$

and the resulting period of the spiking oscillations is given by

$$T = \tau n_c. \tag{99}$$

Therefore, the theory of the GCD allows to predict the dynamics of the network from the network topology alone and is a non-local criterion [Kan11a].

The theory of the GCD originates from the distribution of one initial pulse in the network via identical, unidirectional time-delayed links. Nodes with multiple inputs combine signals from multiple loops in the network that each have different associated time delays. For example, in Fig. 76, node 13 has two inputs originating from loops of 12 and 16 nodes, associated with time delays 12τ and 16τ with the time delay of a single link τ, respectively. When the link time delays τ are identical, the delays in a ring are integer related according to the number of nodes, so that pulses have a fixed spacing given by the GCD. This rule can be derived by following the generation and combination of pulses in such ring networks [Kan11a].

The theory of the GCD, however, does not account for heterogeneity in the link time delay. Numerical simulations confirm that it is robust under small amounts of heterogeneity below the characteristic timescale of the node dynamics [Kan11a]. In addition, local variations of the coupling strength and noise can affect the synchronization patterns of the network, leading to the possibility of a breakdown of the non-local criterion of the GCD for the description of the synchronization state [Kop12]. The theory of the GCD has also been confirmed experimentally with biological neurons obtained from newborn rats that are coupled synthetically with ideal computer-mediated connections [Var12].

The theoretical approaches with the master stability function and the GCD have helped to uncover and understand the diverse collective behaviors in networks of excitable systems, such as bursting, cluster synchronization, and phase transitions [Dah12, Sch08, Leh11, Kan11a, Vic08]. These findings have also been confirmed by simulations with paradigmatic models for excitability, such as the one proposed by FitzHugh and Nagumo [Fit61, Nag62]. Many models and theoretical approaches, however, do not fully integrate experimental imperfections and heterogeneities like noise and system parameter variation, which may have significant impact on the dynamics.

9.4 OBSERVATION OF CLUSTER SYNCHRONIZATION IN BOOLEAN NEURAL NETWORKS

I realize networks of Boolean neurons with a setup intoduced in Sec 8.3.1 and a topology of interconnected rings discussed in Sec 9.2.2 with a hardware description discussed in Appendix B.7.4. Boolean neurons are connected with directed time-delayed links, where link time delays $\tau = n_\tau \tau_{\mathrm{LG}}$ are realized with n_τ cascaded inverter gates as discussed in Appendix A. When nodes have multiple input connections, these signals are combined with electronic equivalents of neurological synapses [Ind11]. I implement the electronic equivalent of an excitatory synapse using an OR gate so that any of the inputs can excite the node as shown in Fig. 76(b) and (c).

I study cluster synchronization in the network topology shown in Fig. 76 with $N = 32$ Boolean neurons assembled in four directed loops of 8, 10, 12, and 16 elements. First, I consider the case of a separation of timescales with node refractory times of $T_{\mathrm{ref}} = (5.6 \pm 0.2)$ ns. The ith loop has a propagation time T_i, given by

$$T_i = L_i(\tau + \delta) + \Delta_i, \tag{100}$$

where L_i is the number of nodes in the loop, $\tau = (16.7 \pm 0.6)$ ns is the delay of a single link, δ is the processing delay of one node, and Δ_i is the average time delay heterogeneity in loop i. I measure the maximum heterogeneity in the network to be $\Delta = \max_{i,j}(|\Delta_i - \Delta_j|) = (2.8 \pm 0.1)$ ns as explained in Appendix A.2.

With a timescale separation satisfying $\tau > T_{\mathrm{ref}} > \Delta$, the experimental network displays two near zero-lag synchronized clusters as shown in Fig. 77(a). The waveforms of four nodes, two out of each cluster, show coherent spiking with period $T_{\mathrm{cluster}} \approx \mathrm{GCD} \cdot \tau = 2\tau$. This behavior is also predicted by the GCD theory from the number of elements in each loop, as [Kan11a]

$$\mathrm{GCD}(8, 10, 12, 16) = 2. \tag{101}$$

The spiking dynamics of the entire network is shown in the raster diagram in Fig. 77(b), where each dot represents a spiking event, subject to a discretization error of ± 1 ns. The first (last) 16 elements, as also shown in the inset, are in near zero-lag synchronization and belong to a cluster. The variation (± 4 ns) in spike generation time between nodes is due to differences in the link time delays and measurement error that originates from signal propagation delays on the FPGA. The dynamics is considered near zero-lag synchronization because the variation in spike times is small compared to the oscillation period $T_{\mathrm{cluster}} \approx 33.4$ ns

To my knowledge, at the time of the initial publication of this work in Ref. [Ros13], this network was the largest experimentally implemented complex network showing cluster synchronization that operates without com-

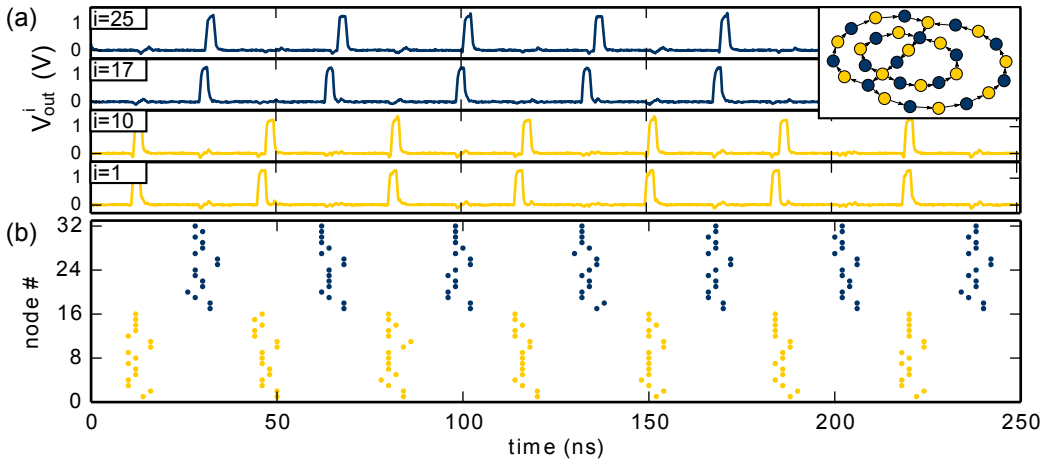


FIGURE 77: Network dynamics of a 2-cluster state with node parameter $n_{\text{pulse}} = 8$ $[T_{\text{pulse}} = (2.2 \pm 0.1)\,\text{ns}]$, $n_{\text{ref}} = 20$ $[T_{\text{ref}} = (5.6 \pm 0.2)\,\text{ns}]$ and link delay times $n_\tau = 60$ $[\tau = (16.7 \pm 0.6)\,\text{ns}]$ after initial stimulation of one node with one pulse of width $w = (1.6 \pm 0.1)\,\text{ns}$. The inset is a replica of the topology of the network, where nodes are colored by cluster. (a) The output waveform of four nodes in the network. Input-output gates are used for the readout with the oscilloscope (8 GHz analog bandwidth, 40 GSa/s sampling rate). (b) Raster diagram of the network, where each point represents the temporal occurrence of a spike with a 2 ns resolution. The network is realized using an Altera Cyclone IV FPGA (EP4CE115F29C7N).

puter assistance, which is commonly used to manage the network coupling in experiments [Ind11, Var12, Hag12]. This illustrates that the setup is well-suited to build large networks compared to other experimental approaches [Nix12]. Since then, multiple realizations of large-scale experimental networks have been realized using networks of optoelectronic, chemical, and mechanical oscillators as discussed in Sec. 7.2.4. In the corresponding Ch. 7, it is also discussed that my approach has various advantages over these experimental networks, as it offers flexibility in the network topology and the coupling function and does not require the mediation of a computer.

9.5 BREAKDOWN OF CLUSTER SYNCHRONIZATION IN BOOLEAN NEURAL NETWORKS

Network dynamics not predicted by the non-local theory of the GCD appear in the network when the separation of timescales is given by $\tau > T_{\text{ref}} > \Delta$. Timescales are re-ordered by adjusting the value of the refractory time of the Boolean neurons. In this section, I study short refractory times T_{ref} on the order of the heterogeneities Δ of the link time delays.

When I decrease the refractory time to a value of $T_{\text{ref}} = (2.8 \pm 0.1)\,\text{ns}$, the network dynamics change. Instead of cluster synchronization with oscillations on the order of τ, the network displays fast, incoherent spiking

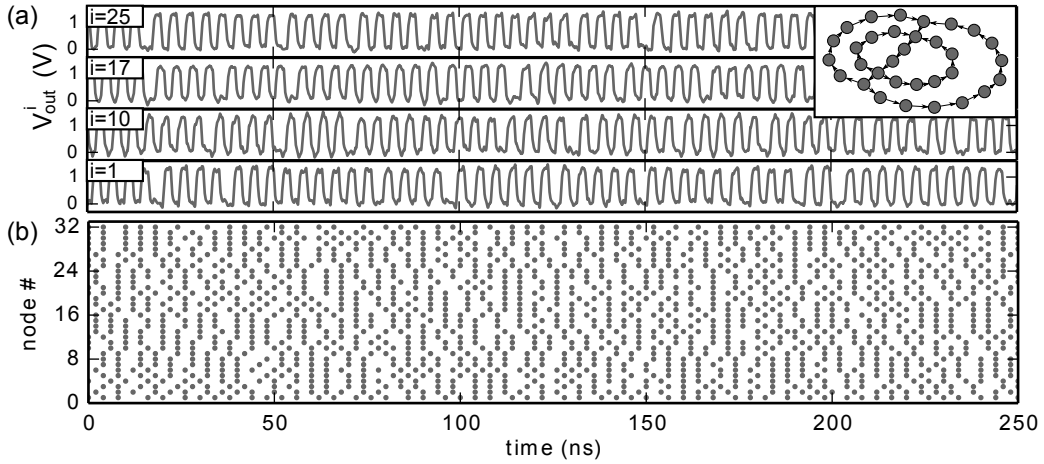


FIGURE 78: Same as Fig. 77, except with $n_{\text{ref}} = 10$ ($T_{\text{ref}} = (2.8 \pm 0.1)$ ns), showing a desynchronized state (synchronization breakdown) of the network dynamics.

dynamics with inter-spike intervals on the order of T_{ref}, as shown in the waveform and raster diagram of Fig. 78. For easy comparison with the previous figure, the time axis is kept the same. The new dynamical state generates excitations constantly, leading to pulsing dynamics with high frequencies close to the maximum frequency allowed by the Boolean neurons, given by $1/T_{\text{ref}}$.

The breakdown is caused by heterogeneity in the loop propagation times at the high-in-degree nodes. With Eq. (100), a maximum time difference $\Delta = \max_{i,j}(|\Delta_i - \Delta_j|) = (2.8 \pm 0.1)$ ns exists in the propagation times T_i of the network loops and leads to a mismatch of the arrival times of pulses. When $\Delta < T_{\text{ref}}$, the refractory time can compensate for the mismatch, by blocking pulses that arrive a time Δ after the first pulse during every period of the clusters T_{cluster}. In this case, the spiking dynamics stays coherent. Otherwise, when $\Delta > T_{\text{ref}}$, the pulse that arrives a time difference Δ after the first pulse will trigger additional pulse trains, leading to incoherent high-frequency spiking.

9.6 ALTERED CLUSTER SYNCHRONIZATION PATTERNS IN BOOLEAN NEURAL NETWORKS

Besides the breakdown for small T_{ref}, the cluster synchronization patterns are also affected for large T_{ref}. When the refractory time is increased to $T_{\text{ref}} = (39 \pm 2)$ ns $\approx 2.3\tau$, the network displays four synchronized clusters (4-cluster state) instead of the 2-cluster state that is inferred from the topology and observed in Fig. 77, as shown in the waveforms and raster diagram in Fig. 79.

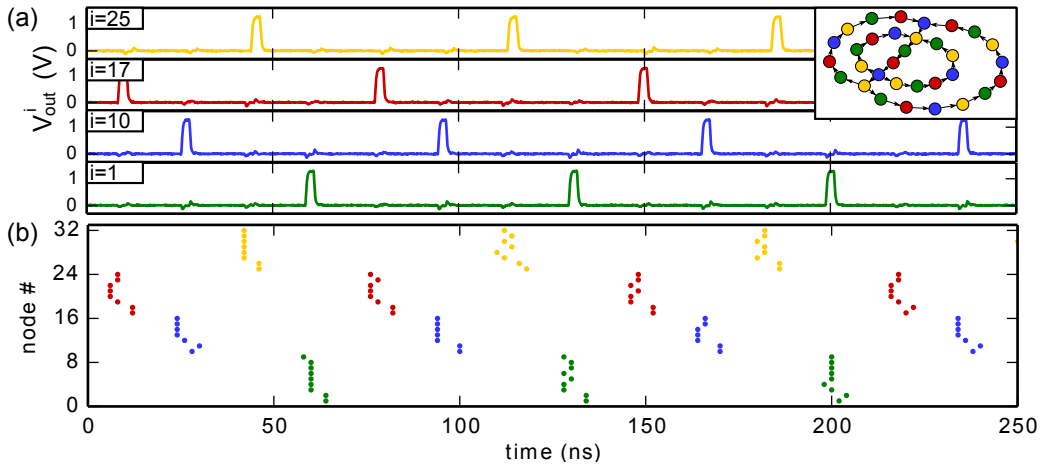


FIGURE 79: Same as Fig. 77, except with $n_{\text{ref}} = 140$ ($T_{\text{ref}} = (39 \pm 2)$ ns), showing four clusters in zero-lag synchronization (4-cluster state).

To understand this behavior, I consider the maximum output frequency of the Boolean neuron, given by $1/T_{\text{ref}}$. When the predicted oscillation frequency, given by $1/T_{\text{cluster}} \approx 1/(\text{GCD} \cdot \tau)$, is above the maximum frequency $1/T_{\text{ref}}$, the network cannot show the predicted cluster state and the dependency on T_{ref} comes into effect. Specifically, stimuli corresponding to higher frequency oscillations ($T < T_{\text{ref}}$) still appear at nodes with a high in-degree, but they are suppressed because these stimuli fall into the refractory periods of earlier excitations. Thus, the GCD has to be recalculated by neglecting some loops, using only the loops that lead to a GCD larger than T_{ref}/τ. Because this pulse-blocking mechanism is based on short oscillation periods, the loops that are effective for the cluster dynamics and used for the calculation are those that lead to the smallest value of the GCD that is greater than T_{ref}/τ, *i.e.*, that lead to the shortest period greater than T_{ref}. Therefore, with the size of loops L_i, the number of clusters is given by

$$\min_{\{L_i\} \in \text{Network}} [\text{GCD}(\{L_i\})] : \text{GCD}(\{L_i\}) > T_{\text{ref}}/\tau. \tag{102}$$

This extended criterion, which combines considerations from both the topology (non-local) and timescale-separation (local), describes successfully the stable synchronization patterns observed in Fig. 79. From the topology in Fig. 76, $\text{GCD}(8, 10, 12, 16) = 2 < T_{\text{ref}}/\tau \approx 2.3$, so that Eq. 102 predicts that a loop is dynamically blocked due to the large refractory period, so that it does not contribute to the network dynamics. Hence, a loop has to be removed from the calculation, leading to the next larger GCD value of $\text{GCD}(8, 12, 16) = 4 > T_{\text{ref}}/\tau \approx 2.3$, which corresponds to the experimental observation of a 4-cluster state. The loop leading to the minimum value of the GCD that satisfies the conditions $GCD > T_{\text{ref}}/\tau$ is the one that is dynamically pruned, which is the loop of 10 nodes. Surprisingly, it is not the shortest loop with 8 nodes but the loop with 10 nodes that first becomes dynamically pruned and removed from the GCD calculation first.

The constraint given by T_{ref} depends on the network topology. For example, a network with predicted zero-lag synchronization (1-cluster state) transitions to a different cluster state already when $T_{\text{ref}}/\tau > 1$. When T_{ref} is further increased, more and more loops lose their effect until the refractory time is longer than the propagation delay through the largest loop; then, spiking dynamics is no longer self-sustained and the network relaxes to the quiescent state.

9.7 CONTROL OF SYNCHRONIZATION PATTERNS IN BOOLEAN NEURAL NETWORKS

A global adjustment of the refractory time T_{ref} of all nodes influences the network dynamics substantially. However, similar modification of the synchronization pattern to those of Section 9.6 can be achieved by adjusting the refractory time of only a selected group of neurons in the network. Hence, local control over the global network dynamics is possible.

I notice that the influence of the refractory time on the network dynamics is most prominent at the nodes with high in-degree. This motivates me to only adjust the refractory times of the nodes with in-degree greater than one, which represents a simple degree-correlation [Bre08a].

First, I investigate the network dynamics for short refractory times. I set the refractory times of nodes to $T_{\text{ref}} = (2.8 \pm 0.1)\,\text{ns}$, a value for which the breakdown of cluster synchronization is observed. When I increase the refractory times of the two nodes with an in-degree greater than one [node 13 and 30 in Fig. 76(a)] to $T_{\text{ref}} = (5.6 \pm 0.2)\,\text{ns}$, the stable synchronization patterns of two clusters reappear as a solution. An initial pulse sent to this network in the quiescent state leads to a 2-cluster synchronization pattern similar to that observed in Fig. 77.

Second, I investigate network dynamics for large refractory times. I set the refractory times of all nodes to $T_{\text{ref}} = (5.6 \pm 0.2)\,\text{ns}$, a value for which a 2-cluster synchronization state is observed. When I now increase the timescales of the two nodes with an in-degree greater than one to $T_{\text{ref}} = (39 \pm 2)\,\text{ns}$, the stable synchronization pattern changes to a 4-cluster state, which I have observed in Fig. 79, when increasing the timescales of all nodes.

Both cases allow for the control of the synchronization patterns locally by a small fraction of the network nodes by adjusting the refractory time of only 2 out of 32 nodes.

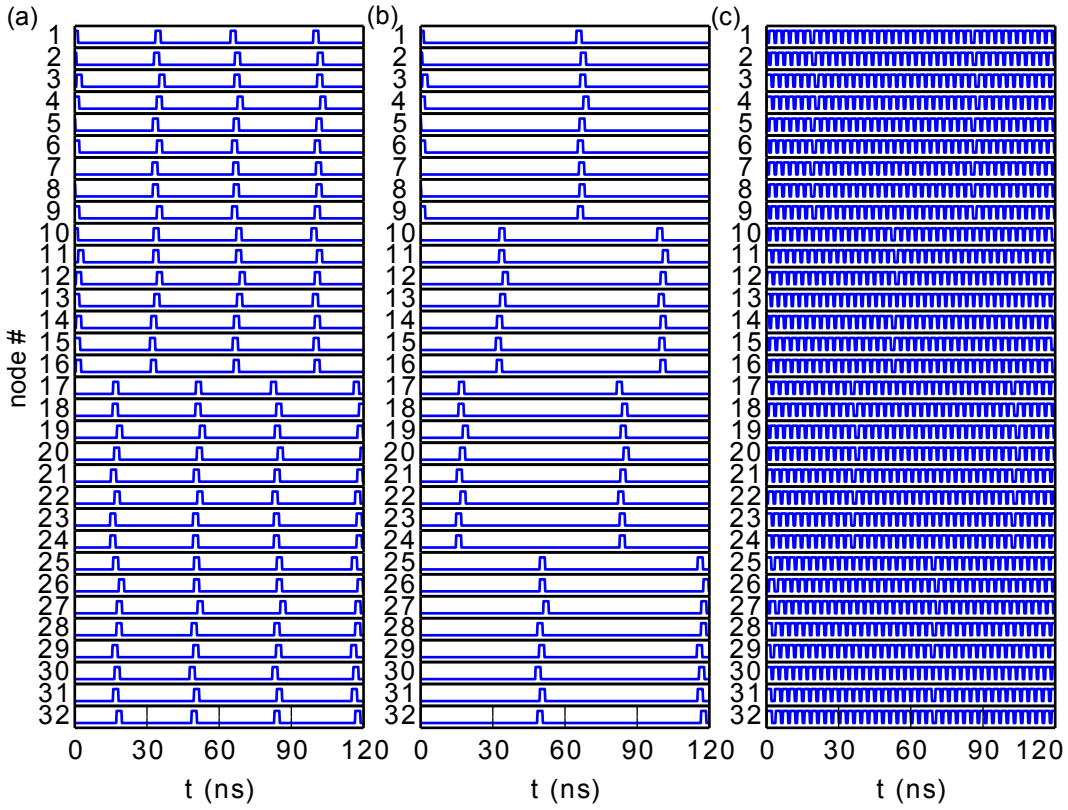


FIGURE 80: Network dynamics calculated from the map for (a) $T_{\rm ref} = (5.6 \pm 0.2)\,{\rm ns}$, (b) $T_{\rm ref} = (39 \pm 2)\,{\rm ns}$, and (c) $T_{\rm ref} = (2.8 \pm 0.1)\,{\rm ns}$. The Boolean waveform of each node is shown versus time. The map is evaluated with a time step $\Delta = 100\,{\rm ps}$, starting with an initial pulse to node 1, and shown after a transient time of 3 μs. The link time delays are $\tau^{(i,j)} = (16.7 \pm 0.6)\,{\rm ns}$.

9.8 NUMERICAL SIMULATION OF BOOLEAN NEURAL NETWORKS

The network dynamics is analyzed theoretically using the Boolean map derived in Sec 8.3.2. The model focuses on the pulse timing of the Boolean neuron by including the mechanism of pulse generation and the refractory period.

I integrate numerically the dynamics of the three experiments of delay-coupled Boolean neuron with the model of the Boolean neuron introduced in Sec 8.3.2. For the specific ring network, I distinguish between nodes by their in-degree. For nodes with in-degree of one, the input $V_{\rm in}^{(j)}(t)$ is given by a delayed version of the output of the connected node $V_{\rm out}^{(i)}(t - \tau^{(i,j)})$ with time delay $\tau^{(i,j)}$. For the two nodes with in-degree greater than one, the input is pre-processed with an OR gate, as shown in Sec. **??**, leading to $V_{\rm in}^{(13)}(t) = V_{\rm out}^{(20)}(t - \tau^{(20,13)}) \vee V_{\rm out}^{(22)}(t - \tau^{(22,13)})$ and $V_{\rm in}^{(30)}(t) = V_{\rm out}^{(9)}(t -$

$\tau^{(9,30)}) \vee V_{\text{out}}^{(13)}(t - \tau^{(13,30)}) \vee V_{\text{out}}^{(15)}(t - \tau^{(15,30)})$, corresponding to the topology in Fig. 69(b).

The link time delays $\tau^{(i,j)} = (16.7 \pm 0.6)\,\text{ns}$ are chosen randomly from a normal distribution with mean of 16.7 ns and standard deviation $\sigma = (0.6\,\text{ns})/3 = 0.2\,\text{ns}$. This value of $\sigma = 0.2\,\text{ns}$ is adjusted to give the best fit with the experimental data. The link time delays have, in addition to heterogeneity, a temporal fluctuation that is a factor of three smaller than the error due to heterogeneity and is hence not included in the model. Similar to the link time delays $\tau^{(i,j)}$, the refractory periods T_{ref} and the pulse widths $T_{\text{pulse}} = (2.2 \pm 0.1)\,\text{ns}$ of the nodes are also obtained from a normal distribution with a mean and standard deviation given by their experimental values.

Figure 80 shows the waveforms generated by the map of the 32 nodes as labeled in Fig. 69(b) for different refractory periods T_{ref} as discussed in the following. The nodes display pulses between V_{low} and V_{high}.

Figure 80(a) and (b) show the waveforms for $T_{\text{ref}} = (5.6 \pm 0.2)\,\text{ns}$ and $T_{\text{ref}} = (39 \pm 2)\,\text{ns}$, corresponding to Fig. 77 and 79. In the theoretical description, the nodes that belong to a cluster generate pulses at approximately the same time, corresponding to near zero-lag cluster synchronization with two and four clusters, respectively, as observed in the experiment, but with no amplitude noise. The map displays the essential features of the experiment, namely the zero-lag cluster synchronization patterns and the period of the oscillations.

For a value of the refractory phase of $T_{\text{ref}} = (2.8 \pm 0.1)\,\text{ns}$, a fast spiking state that is qualitatively similar to the experimental dynamics is also obtained in the simulations as shown in Fig. 80(c) in comparison with Fig. 78. The map displays unsynchronized dynamics with a period on the order of T_{ref}. The period is, however, shorter than in the experiment because, in the model, the experimental processing time h is neglected (the period from the map calculation is $T_{\text{map}} \approx 3.0\,\text{ns}$, whereas the period in the experiment is $T_{\text{exp}} \approx 4.5\,\text{ns}$).

The Boolean map generates dynamics with the essential features of the experimental time series. Differences between experiment and simulations are due to the neglected processing time h, electronic noise, the degradation and low-pass filtering effects in the experiment.

9.9 CONCLUSION

In this chapter, I have connected Boolean neurons into a network with a topology consisting of interconnected rings and have used the resulting networks to study a breakdown and the possibility of control of cluster synchronization dynamics. Cluster synchronization patterns change when the refractory time of the nodes is larger than the link time delays or smaller

than the heterogeneity of the link time delays. For large refractory times, cluster synchronization patterns are modified, and, for short refractory times, cluster synchronization breaks down to an incoherent fast-pulsing state. In both cases, I identify the mechanism leading to the transition and, in the first case, I have put forth a modified GCD criterion that includes the constraints imposed by the refractory time. The synchronization patterns can be controlled by the refractory time of a small fraction of nodes, identified by their in-degree.

The findings in this chapter have two fundamental implications for neuroscience. First, the dynamics of neural networks does not solely depend on the global topology as suggested by Kanter and collaborators [Kan11a, Var12]. I find that, depending on the timescale of the nodes, some links are dynamically pruned, leading to a new effective topology with altered synchronization patterns, as described by Eq. (102). Second, the driver nodes relevant for control can be identified easily by their large in-degree and allow one to control the global network dynamics locally. This is similar to a recent study on the controllability of networks that predicts that the number of driver nodes is given by the network's degree distribution [Liu11].

Various synchronization patterns and more general dynamics are expected for high in-degrees of nodes and for a different choice of the synapses than an OR gate. For example, the flexibility of the logic function will allow implementing inhibiting connections. One step in this direction is the study of synchronization patterns of a network of 80 Boolean neurons that are connected in four coupled populations, which is described in Appendix C.3.

10 SUMMARY AND OUTLOOK

In this thesis, I study the dynamics of autonomous Boolean networks, which are networks of nodes that execute Boolean functions in continuous time without external clocking. In these systems, link time delays, heterogeneity, and non-ideal effects play an important role. Starting from existing work on autonomous Boolean network with chaotic dynamics implemented on a printed circuit board, I integrate this existing design on an electronic chip, which allows for faster and cheaper design cycles and much larger networks.

I study chaotic dynamics in a particularly simple autonomous Boolean network with a topology of only one node with time-delayed feedback, which I design based on guidelines that I develop based on previous studies. This autonomous Boolean network is the building block of a resource-efficient network that I explore as a high-speed physical random number generator, which is, to my knowledge, now used at the National Institute of Standards and Technology (NIST) for quantum communication experiments and will be developed further within a US Army research grant between Duke University and a small company.

I also use autonomous Boolean networks to develop dynamical systems. I first study a periodic Boolean oscillator that is based on a delayed feedback. This periodic oscillator, however, does not include an adjustable coupling strength for the coupling to external signals. As a solution, I study a periodic oscillator that is similar to phase-locked loops, used for clock synchronization, and I find that an internal parameter can be identified with a coupling strength by measuring the synchronization regions. Second, I study a Boolean excitable system, which is a silicon neuron with spiking dynamics with several advantages over existing silicon neurons, such as high speed.

I couple these generic dynamical systems into large networks. Because of large resources on my experimental platform with about 100,000 logic gates, I can study network sizes that are difficult to realize with traditional experimental setups. In a network of periodic oscillators, I find so-called chimera states and a dynamics I discovered called resurgence of chimera states. I find that these networks show complex transient dynamics towards synchrony with a power-law scaling between transient time and phase space volume.

In an artificial neural network, I study so-called cluster synchronization states in interconnected ring networks that generate synchronized spiking

patterns. For this dynamics, I find that an established theory tool breaks down when an internal timescale of neurons is smaller than time delay heterogeneity in the network. I also discover a control strategy for the dynamics of neural networks.

Future work will focus on network-based computing approaches, such as reservoir computing, where a fixed high-dimensional dynamical system, such as a network, processes input information. A linear output layer is then trained to read the reservoir and map the dynamics to a desired output. For such a design, spiking neuron models and delay-line based processors, similar to the Boolean neurons and Boolean oscillators studied here, have already been used successfully in computer simulations for speech recognition and time series prediction [Jae04, Maa02, App11, Sch08k]. Autonomous Boolean networks are especially suited for reservoir computing because of their fast timescale, the possibility to realize large networks, and because they can be implemented together with central processing units (CPUs) for the linear readout layer, forming a compact system-on-a-chip (SoC) device.

A

DELAY LINES REALIZED WITH ELECTRONIC LOGIC CIRCUITS

This appendix discusses the realization of time-delay lines with unclocked logic circuits, in particular with cascaded copier and inverter logic gates. In Sec. A, I discuss the method of implementing delay lines. In Sec. A.2, I use delay lines to measure the gate propagation time. In Sec. A.3, I discuss the difference between copier and inverter-based delay lines. In Sec. A.4, I measure the gate propagation delay of XOR and multiplexer logic gates.[1]

A.1 IMPLEMENTATION OF DELAY LINES

Delay lines in electronic logic circuits can be implemented with various methods, such as by separating logic gates spatially as discussed in Sec. 3.3.2 or by discretizing time and saving the Boolean state to memory [Boa00]. Here, I implement delay lines by utilizing the propagation delay of logic gates.

The propagation time τ_{LG} of CMOS-based logic gates is caused by internal capacitors that take a finite time to be charged and discharged before the output signal reaches the Boolean level [Wes00].

I construct delay lines by cascading logic gates as shown in Fig. 81(a). The resulting time delay for a signal that travels through n cascaded logic gates is then

$$\tau_n = n\tau_{LG}, \tag{103}$$

where τ_{LG} is the delay of a single logic gate including the delay of interconnect between two logic gates.

The logic gates are configured to use a single input and single output, so that they can implement either the Boolean identity or inversion operation, corresponding to a copier or inverter gate. For most implementations of delay lines, I use inverter-based delay lines, which have the advantage of averaging the asymmetry in rise and fall times as discussed in Sec. A.3. However, this makes it necessary that n is an even number to build non-inverting delay lines, which doubles the increments of adjusting time delays. On the other hand, copier-based delay lines are useful when rectification due to differences in rise and fall times are wanted.

1 The method of constructing delay lines is published in Ref. [Ros12].

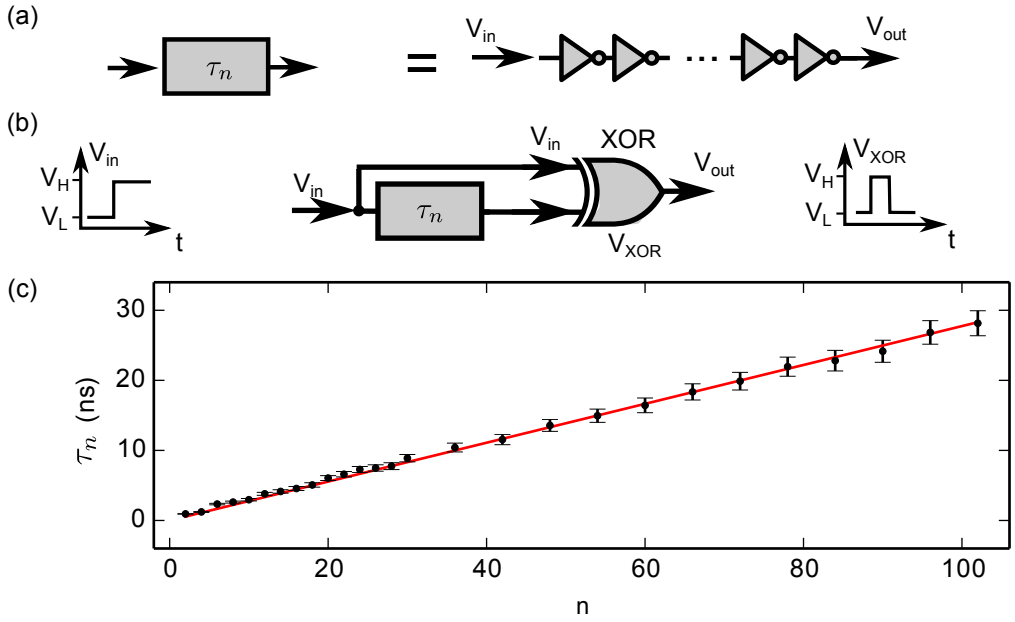


FIGURE 81: (a) Construction of a delay line with cascaded inverter gates. (b) Circuit to measure the time delay. (c) Measurement results of the time delay as a function of n (dots) and fit according to Eq. (103) (line).

The hardware description for an inverter-based delay line can be found in Sec. B.1.

A.2 MEASUREMENT OF THE GATE PROPAGATION DELAY

In this section, I show the linear relation in Eq. (103) and obtain the constant τ_{LG} and the heterogeneity in τ_{LG}. For this, I measure the time delay τ_n resulting from a delay line of n cascaded inverter gates for different n.

Figure 81(b) shows the related setup that includes the delay line and an XOR logic gate. A voltage signal V_{in} is sent into the delay line, where V_{in} describes a single Boolean transition. The output of the delay line is then given by $V_\tau(t) = V_{\mathrm{in}}(t - \tau_n)$ as a delayed version of the input voltage; hence, the two voltages differ in Boolean states for a time period given by the delay τ_n. I measure this difference of Boolean states with an XOR logic gate, which outputs a high Boolean voltage ($V_{\mathrm{XOR}} = V_H$) when the two input voltages differ [see its look-up table in Fig. 2(a)]. The resulting output voltage V_{XOR} of the XOR gate is a pulse with a width corresponding to the length of the delay line, which I measure with an external oscilloscope.

I measure the full width at half maximum (FWHM) of the resulting pulse in V_{XOR} for different n. The generated data is shown in Fig. 81(c) describes the linear relationship of Eq. (103) with the best fit value $\tau_{\mathrm{LG}} = 0.28 \pm 0.01\,\mathrm{ns}$ and an error estimate obtained from the goodness of the fit.

The propagation delay can also be obtained by constructing a ring oscillator and measuring the resulting frequency, as described in Sec. 6.2.5.1.

A.3 DIFFERENCE BETWEEN COPIER- AND INVERTER-BASED DELAY LINES

To understand the difference between copier and inverter-based delay lines, I send a periodic signal into copier and inverter logic gates and measure the duty cycle of input and output. The duty cycle of a Boolean signal is defined as

$$D = \frac{T_H}{T}, \tag{104}$$

where the signal spends a time T_H and T_L in the Boolean high and low state, respectively, and hence the period is $T = T_H + T_L$.

After the signal propagates through a logic gate, the time spent in the high and low Boolean voltage can change ($\tilde{T}_H = T_H + r$, $\tilde{T}_L = T_L - r$) due to a difference in rise and fall time of r, which I call pulse lengthening, leading to an altered duty cycle of

$$\tilde{D} = \frac{T_H + r}{T}. \tag{105}$$

I find that a single logic gate leads to pulse lengthening of $r = 24\,\mathrm{ps}$ for copier logic gates. Figure 82 shows the output waveform of copier-based

delay lines of different length n with a periodic input signal. The pulse lengthening effect increases with n, resulting in a duty cycle of

$$\tilde{D} = \frac{T_H + nr}{T}. \tag{106}$$

When the signal propagates through long enough copier-based delay lines, pulses grow so much that the duty cycle becomes $D = 1$, pulses merge, and the oscillations are pruned. Then, the output stays constantly in the Boolean high state even though the input is oscillatory as shown in Fig. 82(d). I utilize this effect in Chapter 6 as a low-pass filter in phase-locked loops.

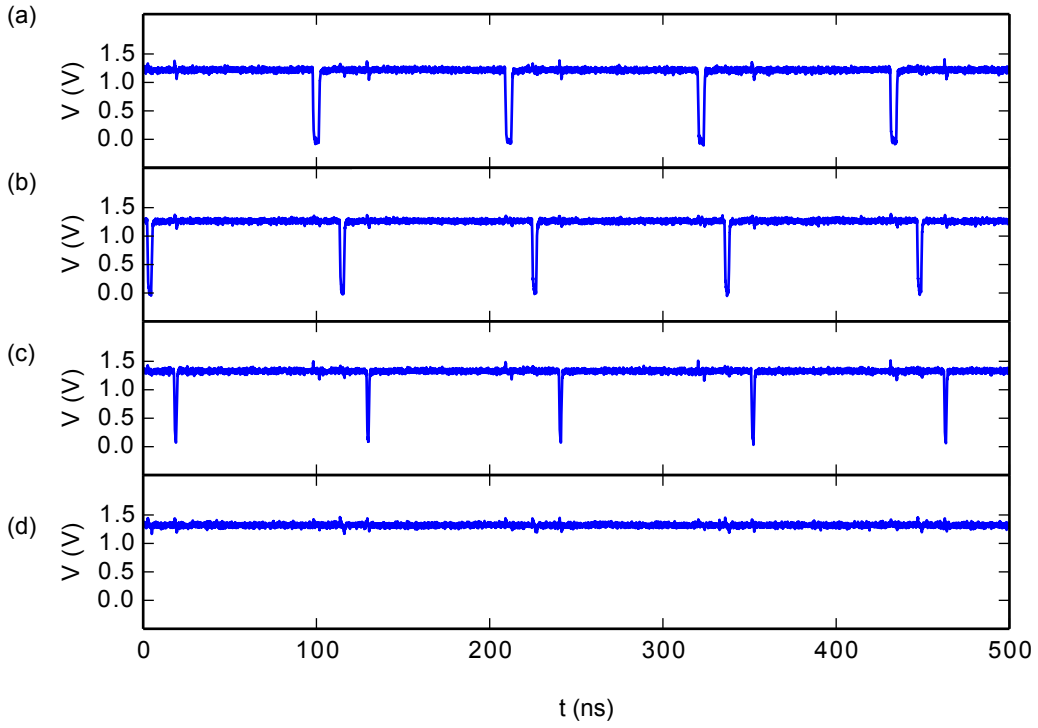


FIGURE 82: Pulse lengthening for an input signal of frequency 9 MHz and duty cycle 96.4%. The input signal shown in (a) propagates through 50, 100, and 150 copier gates, leading to output signals shown in (b), (c) and (d), respectively.

For cascaded inverter logic gates, on the other hand, the pulse lengthening cancels out in every pair of two inverter logic gates. Therefore, chains of inverter gates display a much smaller change in pulse width that I measure to be only $r = \pm 1.4\,\text{ps}$ per pair of inverter logic gates (depending on the specific realization), which is more than two orders of magnitude decreased and more than one order of magnitude smaller than in cascaded copiers. This value of r results from heterogeneity in the pulse lengthening.

type	T (ns)	τ (ns)
inverter	8.31	0.26 ± 0.01
buffer	8.39	0.26 ± 0.01
2-input OR	11.28	0.36 ± 0.02
3-input XNOR	12.58	0.38 ± 0.02
3-input MUX	12.08	0.38 ± 0.02

TABLE 4: Measurements of the period generated by a ring oscillator that includes a ring of 15 logic gates of the type named in the left column and either one inverter gate or one buffer gate to generate inverted delayed feedback. The right column shows the derived gate propagation delay. The open inputs are connected to switches on the FPGA development board.

A.4 DELAY OF DIFFERENT LOGIC GATES

The propagation delay of inverter gates is measured in Sec. A.2 by comparing a Boolean transition and its delayed version with an XOR gate. The delay can also be measured by constructing ring oscillators as described in Sec. 6.2.5.1 and measuring the frequency. With this second method, I measure the propagation delay of different logic gates used in this thesis as shown in Table 4.

Buffer and inverter gates with an in-degree of one have the shortest gate propagation delay. For gates with higher in-degree, the propagation delay increases, which is because of longer paths in the look-up table block for larger in-degrees (see Sec. 3.2.1). Note that the measurement in Sec. A.2 leads to a propagation delay of inverter logic gates of $\tau_{LG} = 0.28 \pm 0.01$ ns, which is 7% higher than the measurement with ring oscillators. This difference is mainly due to the number of logic gates used to construct delay lines. In this section, the delay lines are constructed with 16 logic gates, fitting into a single logic-array block (LAB), hence excluding the delay of wire connections between LABs. On the other hand, the measurement in Sec. A.2 included 100 logic gates to average heterogeneity (see also Sec. 3.2.1).

B

HARDWARE DESCRIPTIONS AND NUMERICAL ALGORITHMS

This appendix shows the hardware description of the autonomous logic circuits in this thesis using the hardware description language Verilog. It also discusses the numerical algorithm for numerical simulations.

The hardware description can be used to implement the circuit on various FPGAs, but it was tested specifically on the Altera Cyclone IV FPGA with model number EP4CE115F29C7N. For an explanation of the syntax of Verilog, see Sec. 3.3.1 and Ref. [Mcn01].

B.1 INVERTER-BASED DELAY LINES

The following hardware description realizes an inverter-based delay line used in most of the following hardware designs.

```
1  module my_delay_line(s_in,s_out);
2  parameter n=20;
3  genvar i;
4  input s_in;
5  output s_out;
6  wire [n-1:0] delay /* synthesis keep */;
7  assign delay[0] = s_in;
8  assign s_out = delay[n-1];
9  generate
10  for (i=0; i < n-1; i=i+1)
11  begin : generate_delay
12   assign delay [i+1] = ~delay [i] ;
13  end
14 endgenerate
15 endmodule
```

This module, called `my_delay_line`, implements a delay line of inverter gates as discussed in Sec. A.1. The hardware module has an input wire `s_in` and output wire `s_out`, where the input signal is sent into the chain of `n` copier gates and the output signal is measured after the last copier gate. The parameter `n` can be defined when this hardware module is initiated. The inverter logic gates are generated with the `assign` statement within a `generate for` loop. For further information see Sec. 3.3.1.

B.2 DELAYED-FEEDBACK XNOR OSCILLATOR

The following hardware description realizes the delayed-feedback XNOR oscillator as introduced in Sec. 4.3. It implements the delay line module defined above.

```
module main(out);
output out;
wire [3:0] net /*synthesis keep*/;
assign net[0] = ~(net[1] ^ net[2] ^ net[3]);
my_delay_line #(10) delay1(net[0], net[1]);
my_delay_line #(6)  delay2(net[0], net[2]);
my_delay_line #(12) delay3(net[0], net[3]);
assign out = net[0];
endmodule
```

Here, the instantiations of `my_delay_line` are called `delay1`, `delay2`, and `delay3`, so that they can be referred in other design tools. The delay lines are instantiated with the parameter `n` using the notation of `#(10)` to set the number of inverter gates included in the delay lines. I include an XNOR gate that is defined with an inversion `~` and XOR `^` operator and has the three inputs `net[1]`, `net[2]`, `net[3]` and the output `net[0]`. The output of the XNOR logic gate `net[0]` is input to the three delay lines that feed back to the inputs of the logic gate. The output port of the FPGA, called `out`, is connected to the output of the XNOR logic gate `net[0]`.

B.3 RANDOM NUMBER GENERATOR

In this section, I discuss the hardware descriptions used for random number generation in Ch. 5.

B.3.1 XOR RING OSCILLATOR

The following is the hardware description for the hybrid Boolean network used for random number generation in Sec. 5.3.

```
module rng(CLOCK_100,word_out);
parameter N=16;
input CLOCK_100;
output reg word_out;
wire [N:0] net /*synthesis keep*/;
reg [3:0] trig;
genvar i;

/*autonomous Boolean nodes*/
```

```
10 assign net[0] =  ~(net[N-1] ^ net[0]  ^ net[1]);
11 assign net[N-1] =  (net[N-2] ^ net[N-1] ^ net[0]);
12 generate
13  for (i=1; i < N-1; i=i+1)
14  begin: generate_ring
15   assign net[i]  = (net[i-1] ^ net[i] ^ net[i+1]);
16  end
17 endgenerate
18
19 /*synchronous Boolean node*/
20 always @ (negedge CLOCK_100)
21 begin
22  trig[0] = net[0];
23  trig[1] = net[4];
24  trig[2] = net[8];
25  trig[3] = net[12];
26 end
27 always @ (posedge CLOCK_100)
28 begin
29  word_out = trig[0] ^ trig[1] ^ trig[2] ^ trig[3];
30 end
31
32 endmodule
```

It has a 1-bit input for the clock and a 1-bit output for the synchronous random bit. The parameter N gives the number of autonomous nodes and is set to $N = 16$, so that the synchronous node gets input from every fourth autonomous node in the network.

Lines 9-17 implement the autonomous Boolean nodes with the XNOR gate in line 10 and the last XOR gate in line 11. The autonomous nodes have outputs given by an vector of N wires `net`. Lines 12-17 implement the remaining $N - 2$ XOR gates using a `for` loop. The XOR gates have have inputs from the left and right neighbor and from itself.

The synchronous node has an output register `word_out` and an input register `trig`. These are implemented using flip-flops with `always@` statements with the wire `clock_100` connected to the clock port. Flip-flops are implemented before and after the synchronous node. The clock can have any frequency, where frequencies that are too high lead to inferior random numbers and clcok frequencies too low do not exploit the achievable speed as discussed in Sec. 5.3. The used hardware board, the Terasic DE2-115, supplies a 50 MHz clock that can be used to generate clock signals of different frequencies using a phase-locked loop using a so-called mega-function.

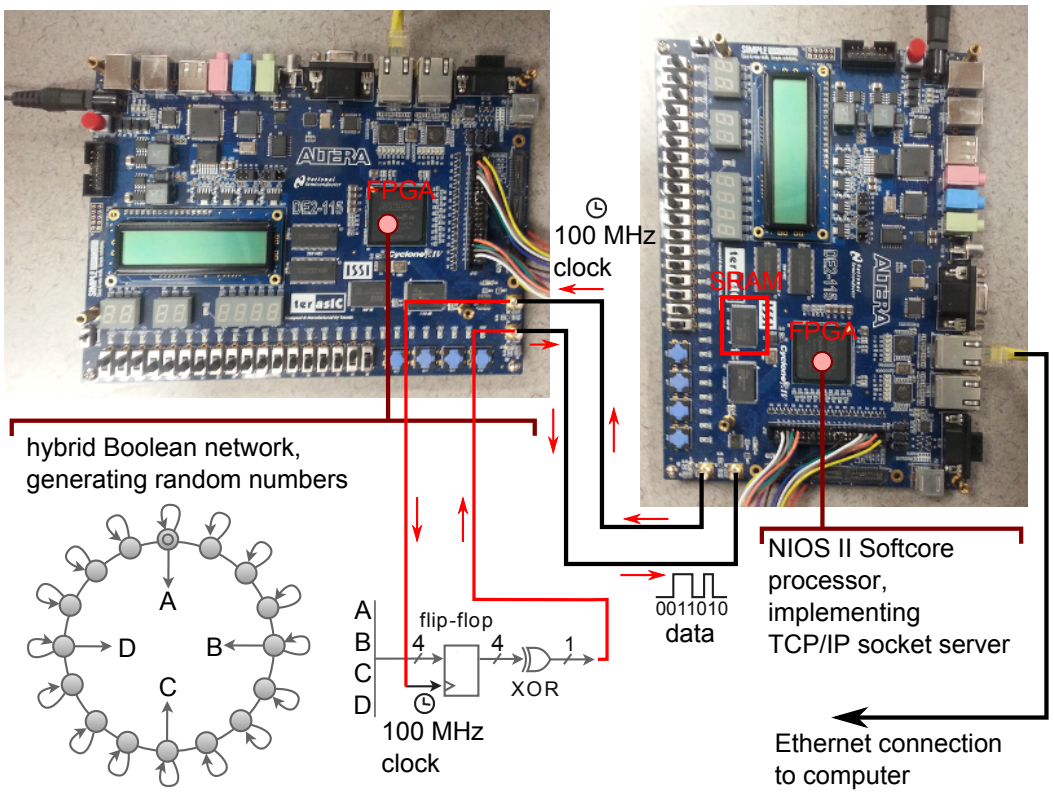


FIGURE 83: Schematic of the transfer of random numbers to the computer. Two FPGA boards are connected via general purpose input outputs (GPIO) to send random numbers from one board to the other at 100 Mbit/s. The receiving board implements a NIOS II softcore processor for the transfer of random numbers via TCP/IP protocol to a computer. The left board implements the hybrid Boolean network shown with a graphic similar to Fig. 22(a).

B.3.2 TRANSFER OF RANDOM NUMBERS TO A COMPUTER

The hybrid Boolean network from the previous section generates a binary stream of random numbers. These are transferred to a computer using another FPGA board, as shown in Fig. 83. The hybrid Boolean network has one input, the 100 MHz clock, and one output, the random number bit stream, which are connected to the SMA input and output connectors on the FPGA board. Another FPGA board generates the clock and receives the random numbers.

This board saves 1 Mbit of data (coming from the other board) into SRAM, then sending this data via TCP/IP protocol to a computer. This is repeated 1000 times to achieve a file size of 1 Gbit. The trasfer protocol requires a processor structure on the FPGA, which is achieved using the NIOS II softcore processor. The processor is programmed to be socket server and a desktop computer is programmed to be the client. The C-code used for both server and client can be found in Ref. [Dono9].

B.3.3 XOR RING NETWORKS IN PARALLEL

For implementing random number generators in parallel, the hardware description in Sec. B.3.1 called `rng` is instantiated 128 times as follows.

```
module rng_parallel(clk,rng_out);
parameter n_rng=128;
input CLOCK_100;
output [127:0] rng_out;
genvar i;
generate
 for (i=0; i < n_rng-1; i=i+1)
 begin: generate_multiple_rngs
  rng inst(CLOCK_100,rng_out[i]);
 end
endgenerate
endmodule
```

A `for` loop implements 128 random number generators.

For the transfer of random numbers, the data is first saved to on-chip memory with word sizes of 128 bits at 100 MHz, which is a FIFO (first in, first out) buffer. Then, the memory is read at a lower rate of 100 Mbit/s to another board as described in Sec. B.3.2 (see also Fig. 31).

B.4 MODIFIED RING OSCILLATORS

The hardware description for a modified ring oscillators is as follows, with a logic circuit shown in Fig. 39 and discussed in Sec. 6.3.

```
module mod_ring_osc(in,out);
parameter n=20;
input in;
output out;
wire delay_in, my_or /*synthesis keep*/;

my_delay_line #(20) delay1( delay_in , out );
assign delay_in = ~my_or;
assign my_or = out | in;
endmodule
```

The code implements an inverter-based delay line that includes 20 inverters as introcuded in Sec. B.1.

B.5 NETWORK MOTIF OF MODIFIED RING OSCILLATORS

The following code shows how two of these oscillators can be coupled mutually by calling the module `mod_ring_osc` twice.

```
module coupled_osc(osc_out);
output [1:0] osc_out;

mod_ring_osc #(20) osc_A(osc_out[1],osc_out[0]);
mod_ring_osc #(20) osc_B(osc_out[0],osc_out[1]);
endmodule
```

The output of each oscillator is input to the other. Here, the coupling is without time delays along the links. Time delays can, however, be included easily with a hardware description similar to Sec. B.7.3.

B.6 BOOLEAN PHASE OSCILLATORS

Here, I discuss the hardware descriptions for Boolean phase oscillators starting with an in-degree of one, then discussing larger in-degrees and non-local networks of Boolean phase oscillators that display chimera states.

B.6.1 BOOLEAN PHASE OSCILLATOR WITH IN-DEGREE ONE

The following shows the hardware description of a Boolean phase oscillator with in-degree one as described in Sec. 6.4.1.

```
module Boolean_phase_osc(in,out);
parameter n = 50;
parameter k = 10;
input in;
output out;

wire [n:0] osc /*synthesis keep*/;
wire mux_out,xor_sig /*synthesis keep*/;

assign out = mux_out;
assign osc[0] = ~mux_out;
assign xor_sig = out ^ in;
assign mux_out = xor_sig ? osc[n-k] : osc[n];

genvar i;
generate
```

```
17  for(i=0;i<n;i=i+1)
18  begin: building_delay
19   assign osc[i+1] = osc[i];
20  end
21 endgenerate
22 endmodule
```

Here, the two parameters `n` and `k` determine the natural frequency and the coupling strength as described in Sec. 6.4.1 and can be adjusted to change the oscillator's properties. The code implements one inverter gate (line 11), a multiplexer (line 13), an XOR gate (line 12), and 50 buffer gates for the delay (line 15-20), leading to 53 logic gates in total. The multiplexer in chooses between the output of the buffer-based delay line `osc[n-k]` and `osc[n]`, leading to a delay difference corresponding to `k` buffer gates.

Boolean phase oscillators can be coupled unidirectionally as follows

```
1 module unidir_coupling(driver,osc);
2 input driver;
3 output osc;
4
5 Boolean_phase_osc #(.n(50),.k(10)) osc_A(driver,osc);
6 endmodule
```

Here, the Boolean signal `driver` is input to the Boolean phase oscillator with output `osc`. The notation `#(.n(50),.k(10))` redefines the parameters within the module. By scanning a range of frequencies of `driver` and comparing the output frequency of `osc`, a devil's staircase can be recorded as discussed in Sec. 6.4.2.

Boolean phase oscillators can be coupled bidirectionally as follows

```
1 module bidir_coupling(osc);
2 output [1:0] osc;
3
4 Boolean_phase_osc #(.n(50),.k(10)) osc_A(osc[1],osc[0]);
5 Boolean_phase_osc #(.n(50),.k(10)) osc_B(osc[0],osc[1]);
6 endmodule
```

By changing the parameters n and k in the two oscillators, the bidirectional coupling plane in Sec. 6.4.4 can be measured. The measurment in that section, however, was automized with labview by implementing several oscillator combinations on the FPGA and selecting and measuring them without changing the hardware implementation on the FPGA.

B.6.2 BOOLEAN PHASE OSCILLATOR WITH LARGE IN-DEGREE

The following hardware description results in a Boolean phase oscillator with an in-degree of 60 as discussed in Ch. 7.

```
module bpo_node(node_out,c_signal,c_disable);
parameter n_delay=30, k=1;
genvar i;
input [59:0] c_signal /*synthesis keep*/;
output node_out;
input [59:0]  c_disable;
wire [n_delay-1:0] main_delay /*synthesis keep*/;
wire [59:0] mux_out /*synthesis keep*/;
wire [59:0] k_delayo /*synthesis keep*/;
wire [59:0] k_delay1 /*synthesis keep*/;
...
wire [59:0] k_delay59 /*synthesis keep*/;

assign node_out = mux_out[0];

// define multiplexer
assign mux_out[0] =  c_disable[0] ? k_delayo[k-1] :
  (c_signal[0] ? mux_out[1] : k_delayo[k-1]);
assign mux_out[1] =  c_disable[1] ? k_delay1[k-1] :
  (c_signal[1] ? mux_out[2] : k_delay1[k-1]);
...
assign mux_out[59] =  c_disable[59] ?  k_delay59[k-1] :
  (c_signal[59] ? main_delay[n_delay-1] :
  k_delay59[k-1]);

// define main main_delay
assign main_delay[0] = ~mux_out[0];
generate
  for(i=0; i<n_delay-1; i=i+1)
  begin: build_main_delay
    assign main_delay[i+1] = main_delay[i];
  end
endgenerate

// define k-coupling delays
assign k_delayo[0] = mux_out[1];
generate
  for(i=0; i<k-1; i=i+1)
  begin: build_k_delayo
    assign k_delayo[i+1] = k_delayo[i];
  end
endgenerate

assign k_delay1[0] = mux_out[2];
```

```
45 generate
46   for(i=0; i<k-1; i=i+1)
47   begin: build_k_delay1
48     assign k_delay1[i+1] = k_delay1[i];
49   end
50 endgenerate
51
52 ...
53
54 assign k_delay59[0] = main_delay[n_delay-1];
55 generate
56   for(i=0; i<k-1; i=i+1)
57   begin: build_k_delay59
58     assign k_delay59[i+1] = k_delay59[i];
59   end
60 endgenerate
61 endmodule
```

The Verilog code for the Boolean phase oscillator with an in-degree of 60 has three connections `node_out`, `c_signal`, and `c_disable`. `node_out` is an output wire with the dynamics of the oscillator. `c_signal` and `c_disable` are vectors of 60 input wires, corresponding to the in-degree of the oscillator. `c_signal` includes the state of the phase comparisons to adjust the delay line and `c_disable` includes signals to disable the input connections used to uncouple oscillators. For example, all wires `c_disable` can be set to the Boolean high state to measure the natural frequency of the oscillator.

The circuit diagram with indicated wire names is shown in Fig. 84. The figure shows that the `c_disable` wires are input to the multiplexers together with the coupling signals `c_signal`. When one `c_disable` wire is chosen at Boolean "1," the longer time delay line of that multiplexer is selected independently of the corresponding `c_signal`. On the other hand, when it is "0," the multiplexer can switch depending on the corresponding `c_signal`. A Boolean "1" `c_signal` signal selects the output of the previous multiplexer or the output of the fixed delay line, corresponding to a coupling input. A Boolean "0" `c_signal` signal selects the longer delay called `k_delay` (see also the explanation of the functionality of the Boolean phase oscillator in Sec. 7.3.1). This functionality of the multiplexer is described in lines 16-24 in the hardware description using the `( ? : )` construct, which is similar to an `if` statement in many programming languages.

Lines 26-33 describe a buffer-based delay line for the constant delay, similar to an inverter-based delay line, discussed in Sec. B.1.

Lines 35-60 define the buffer-based delay lines before the multiplexers with length `k`. Because `k` is 1, the generate statements result in only one buffer each. The Verilog code, however, allows to also implement longer coupling delays by changing the parameter `k`.

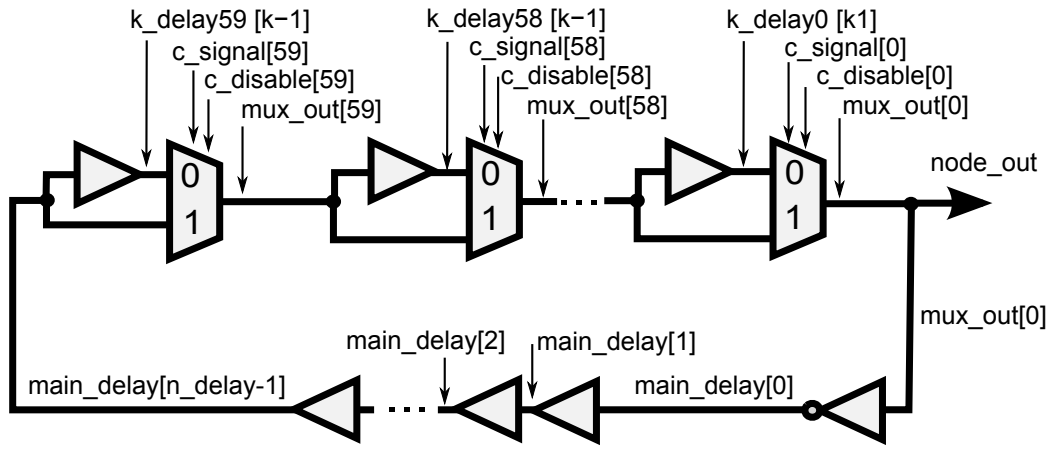


FIGURE 84: Figure similar to Fig. 53 with labels of the wire names in the hardware description.

B.6.3 NON-LOCAL NETWORK OF BOOLEAN PHASE OSCILLATORS

To build a non-local network of Boolean phase oscillators, the module discussed in the previous section is instantiated N times. For a network size of $N = 128$, the hardware description is shown in the following.

```
1  module bpo_network(node_dyn,reset);
2  parameter nr_nodes = 128;
3  output [nr_nodes-1:0] node_dyn;
4  input [29:0] c_range;
5
6  bpo_node #(.n_delay(30),.k(1)) nodeo (node_dyn[0],{
7  node_dyn[127] ^ node_dyn[0],
8  node_dyn[126] ^ node_dyn[0], ...,
9  node_dyn[98]  ^ node_dyn[0],
10 node_dyn[30]  ^ node_dyn[0], ...,
11 node_dyn[1]   ^ node_dyn[0]},
12 {c_range[0], ..., c_range[29], c_range[29],
13 c_range[28], ..., c_range[0]});
14
15 bpo_node #(.n_delay(30),.k(1)) node1 (node_dyn[1],{
16 node_dyn[127] ^ node_dyn[1],
17 node_dyn[126] ^ node_dyn[1], ...,
18 node_dyn[99]  ^ node_dyn[1],
19 node_dyn[31]  ^ node_dyn[1], ...,
20 node_dyn[0]   ^ node_dyn[1]},
21 {c_range[1], ..., c_range[29], c_range[29], ...,
22 c_range[1], c_range[0], c_range[0]});
23
```

```
24 ...
25
26 bpo_node #(.n_delay(30),.k(1)) node127 (node_dyn[127],
27 {node_dyn[126] ^ node_dyn[127], ...,
28 node_dyn[97] ^ node_dyn[127],
29 node_dyn[29] ^ node_dyn[127], ...,
30 node_dyn[0] ^ node_dyn[127]},
31 {c_range[0], ..., c_range[29], c_range[29], ...,
32 c_range[0]});
33 endmodule
```

Here, the module has an output vector of 128 wires that are connected to the Boolean dynamical variables of the nodes and an input vector of 30 wires `c_range` to adjust the coupling range (see also Sec. 7.2.2.1).

The 128 nodes are instantiated with the coupling strength `k`= 1 and constant delay `n_delay`= 30. The latter parameter can be adjusted in the network so that frequency heterogeneity is reduced.

For the first Boolean phase oscillator in lines 6-13, the output wire is connected to `node_dyn[0]`. The 60 node input wires are connected to an XOR phase detector given by `node_dyn[127] ^ node_dyn[0]`, where the first number denotes the number of the oscillator that the signal is compared with. These XOR operations are sorted by node numbers such that the operation with the highest node number comes first. The signals `c_range` that disable input couplings are assembled such that the coupling range is set by the number of 1 bits in `c_range` filled up from the first bit. For example, a coupling radius of $R = 1$ is achieved with `c_range[0]=1` and `c_range[`i`]=0` for $i > 0$, where only the coupling inputs with `node_dyn[127]` and `node_dyn[1]` are activated (for `node_dyn[0]`).

This code is repeated for all 128 oscillators as shown in lines 15-31.

The oscillators are assembled on the chip using the chip planner as discussed in Sec. 54.

For the readout of the network dynamics `node_dyn`, the wire states are saved to on-chip memory and then send to a computer via a RS-232 connection.

B.7 BOOLEAN NEURONS

In this section, I discuss the hardware description of Boolean neurons, of a network motif of Boolean neurons, and of an interconnected ring network of Boolean neurons.

B.7.1 PULSE GENERATOR

The Boolean neurons, as described in Sec. 8.3.1, are based on pulse generators. The hardware description of such a device is shown in the following.

```
1 module pulse_generator(sig_in,pulse_out);
2 parameter n_delay = 5;
3 input sig_in;
4 output pulse_out;
5 reg state,state_inverse;
6 wire delayed_state;
7
8 always @(posedge sig_in)
9   state <= ~state;
10
11 my_delay_line #(n_delay) insto(state,delayed_state);
12 assign pulse_out = state ^ delayed_state;
13 endmodule
```

Here, the `always` construct implements a flip-flop, where the register `state` changes its Boolean value when the input signal `sig_in` has a positive edge. The module `my_delay_line` instantiates a delay line of `n_delay` inverter gates, as introduced in Sec. B.1. The result is the circuit shown in Fig. 69(a) with `n_delay` inverter gates.

B.7.2 BOOLEAN NEURON

Using two pulse generators, a Boolean neuron can be implemented with the following hardware description.

```
1 module Boolean_neuron(in,out);
2 parameter n_ref = 10;
3 parameter n_pulse = 2;
4 input in;
5 output out;
6 wire and_out, refr;
7
8 assign and_out = ~refr & in;
9 pulse_generator #(n_pulse) unito(and_out, out);
10 pulse_generator #(n_ref) unit1(and_out, refr);
11 endmodule
```

The Boolean neuron has an input and output wire and can be configured with parameters `n_ref` and `n_pulse`. It implements the neuron according to Fig. 69(b) with an NAND gate (inversion of AND) in line 8.

B.7.3 NETWORK MOTIF OF TWO BIDIRECTIONALLY-COUPLED BOOLEAN NEURONS

The following shows the hardware description of the bidirectional coupling topology in Sec. 8.4.2 according to the circuit diagram in Fig. 72(b).

```
1 module bidir_coupling(exc_out,reset);
2 output [1:0] exc_out;
3 input        reset;
4 wire [1:0] exc_in, couplingK, couplingC;
5
6 //nodes
7 assign exc_in[0] = (initialpulse | couplingC[1] |
8   couplingK[0]) & ~reset;
9 Boolean_neuron #(10,4) node0(exc_in[0], exc_out[0]);
10 assign exc_in[1] = (                 couplingC[0] |
11   couplingK[1]) & ~reset;
12 Boolean_neuron #(10,4) node1(exc_in[1], exc_out[1]);
13
14 //links
15 my_delay_line #(40) C0(exc_out[0],couplingC[0]);
16 my_delay_line #(40) K0(exc_out[0],couplingK[0]);
17 my_delay_line #(40) C1(exc_out[1]couplingC[1]);
18 my_delay_line #(40) K1(exc_out[1],couplingK[1]);
19 endmodule
```

This hardware module has an output vector of two wires `exc_out` and an input wire `reset`, where the first provides the dynamics of the two Boolean neurons and the second allows to reset the network into the quiescent state. Lines 6-12 instantiate the two neurons and their their input connections according to the circuit diagram in Fig. 72(b). Lines 14-18 implement four delay lines according to the delay links.

B.7.4 CONNECTED RING NETWORK OF BOOLEAN NEURONS

The network of Boolean neurons in Fig. 76 is implemented with the following hardware description

```
1 module network_32nodes(initialpulse,reset,exc_out);
2 parameter nref=10, npulse=4, ndelay=30;
3 input initialpulse;
4 input reset;
5 output [31:0] exc_out;
6 wire [31:0] couplingC, exc_in;
7 genvar i;
8
```

```
9  //define Boolean neurons and link delays
10 generate
11  for(i=0; i <= 31; i=i+1)
12  begin:exc_system_loop
13   my_delay_line #(ndelay) link(exc_out[i],
14     couplingC[i]);
15   Boolean_neuron #(nref,npulse) node(exc_in[i],
16     exc_out[i]);
17  end
18 endgenerate
19
20 //couple topology
21 assign exc_in[0] = (initialpulse |
22   couplingC[15]) &reset;
23 assign exc_in[1] = (couplingC[0]) &~reset;
24 assign exc_in[2] = (couplingC[1]) &~reset;
25 assign exc_in[3] = (couplingC[2]) &~reset;
26 assign exc_in[4] = (couplingC[3]) &~reset;
27 assign exc_in[5] = (couplingC[4]) &~reset;
28 assign exc_in[6] = (couplingC[5]) &~reset;
29 assign exc_in[7] = (couplingC[6]) &~reset;
30 assign exc_in[8] = (couplingC[7]) &~reset;
31 assign exc_in[9] = (couplingC[8]) &~reset;
32 assign exc_in[10] = (couplingC[9]) &~reset;
33 assign exc_in[11] = (couplingC[10]) &~reset;
34 assign exc_in[12] = (couplingC[11]) &~reset;
35 assign exc_in[13] = (couplingC[12]) &~reset;
36 assign exc_in[14] = (couplingC[21] |
37   couplingC[13]) &~reset;
38 assign exc_in[15] = (couplingC[14] | couplingC[24] |
39   couplingC[30]) &~reset;
40 assign exc_in[16] = (couplingC[3]) &~reset;
41 assign exc_in[17] = (couplingC[16]) &~reset;
42 assign exc_in[18] = (couplingC[17]) &~reset;
43 assign exc_in[19] = (couplingC[18]) &~reset;
44 assign exc_in[20] = (couplingC[19]) &~reset;
45 assign exc_in[21] = (couplingC[20]) &~reset;
46 assign exc_in[22] = (couplingC[31]) &~reset;
47 assign exc_in[23] = (couplingC[22]) &~reset;
48 assign exc_in[24] = (couplingC[23]) &~reset;
49 assign exc_in[25] = (couplingC[2]) &~reset;
50 assign exc_in[26] = (couplingC[25]) &~reset;
51 assign exc_in[27] = (couplingC[26]) &~reset;
52 assign exc_in[28] = (couplingC[27]) &~reset;
```

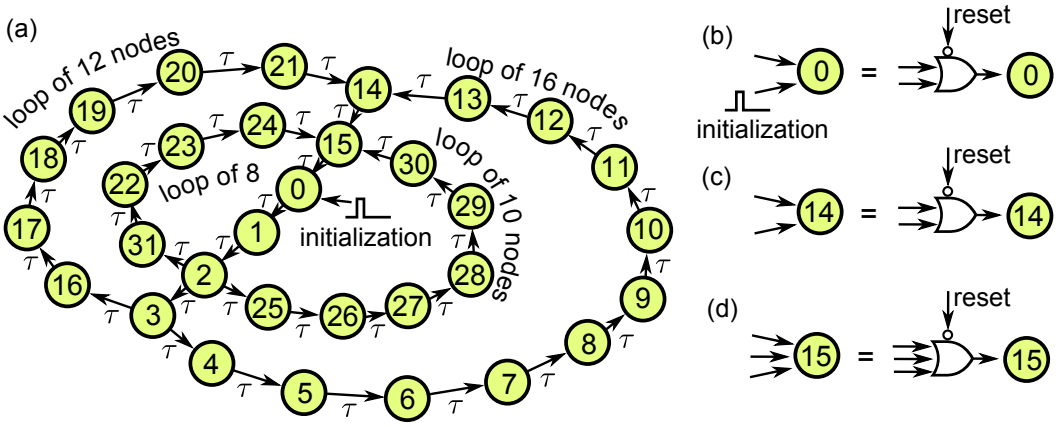


FIGURE 85: Figure similar to Fig. 76. (a) Network topology with node labels according to the hardware description. (b)-(d) The three nodes with an in-degree above one are implemented wiht OR gates. Before each excitation, a reset signal can stop any excitation to reset the network into the quiescent state.

```
53 assign exc_in[29] = (couplingC[28]) &~reset;
54 assign exc_in[30] = (couplingC[29]) &~reset;
55 assign exc_in[31] = (couplingC[2]) &~reset;
56 endmodule
```

The module has three connections, `initialpulse` to initialize the network, which can be generated with a pulse generator and a rising edge signal, for example generated with a mechanical switch; `reset` to reset the network to the resting state; and `exc_out`, which is a vector of 32 wires with the dynamics of the 32 nodes in the network. The network has three parameters, `nref` and `npulse` are used to parameterize the neurons and `ndelay` is the length on the link delay lines.

In lines 10-18, the 32 network nodes and delay links are generated using previous modules, specifically the Boolean neuron above and the delay line in Sec. B.1. Lines 21-55 define the coupling topology according to Fig. 85. The node labeled "0" in Fig. 85 with input wire `exc_in[0]` has an input given by the external wire `initialpulse`, `couplingC[15]`, corresponding to the delayed dynamics of node 15, and `~reset` the Boolean inverse of the reset signal. The Boolean operation is a combination of OR, AND, so that a high Boolean value in either `initialpulse` or `couplingC[15]` can excite the node and a high value in `~reset` will inhibit any excitation as shown in Fig. 85(b). The remaining nodes have a similar construction of input connections.

B.8 NUMERICAL SIMULATION WITH ADAMS-BASHFORTH METHOD

Numerical simulations of a differential equation $\dot{x} = h(x,t)$ approximate the exact solution $x(t)$ by a sequence $\{x_n\}$ corresponding to the times $\{n\mathrm{d}t\}$, where $\mathrm{d}t$ is the time step. I perform numerical simulations with a first order Euler method and a fourth order Adams-Bashforth method depending on the differential equation. The latter method is a two-step predictor-corrector algorithm that first predicts the iteration x_{n+1} from the previous caluclations x_{n-3}, x_{n-2}, x_{n-1}, and x_n, according to

$$x_{n+1} = x_n + \frac{\mathrm{d}t}{24}\left[55h(x_n) - 59h(x_{n-1}) + 37h(x_{n-2}) - 9h(x_{n-3})\right] \tag{107}$$

In the next step, the approximation x_{n+1} is improved using the predicted value, according to

$$x_{n+1} = x_n + \frac{\mathrm{d}t}{24}\left[9h(x_n) + 19h(x_n) - 5h(x_{n-1}) + h(x_{n-2})\right]. \tag{108}$$

The coefficients in front of $h(\cdot)$ are chosen to obtain the highest order of convergence [Pre07].

To handle delay differential equations with a constant delay τ, I add another variable $x(t-\tau) \approx x_{n-N}$ with $N = \lfloor \tau/(\mathrm{d}t) \rfloor$. When $\mathrm{d}t$ is chosen smaller, the approximation becomes better, but computation time increases. Under certain conditions, such as a timescale separation in the differential equation, an algorithm with a variable time step may better choice.

C

ADDITIONAL MATERIAL

C.1 BIAS REDUCTION OF THE XOR OPERATION

For the realization of a random number generator in Chapter 5, I use a 4-input XOR gate to reduce bias and correlations. Here, I evaluate the resulting bias.

For the four inputs of the XOR gate, I assume uncorrelated binary variables $\{x_i\}_{i=0}^{3}$ sampled from identically biased uniform distributions with a probability of a 0 or 1 given by

$$P(x_i = 0) = \frac{1}{2} - b \text{ and } P(x_i = 1) = \frac{1}{2} + b. \tag{109}$$

The resulting probabilities associated with 4-bit words with different counts of 1 are calculated in Table 5.

TABLE 5: Probability associated to all 4-bit words considering bias b

4-bit words	probability
S_0={0000}	$\left(\frac{1}{2}-b\right)^4$
S_1={0001,0010,0100,1000}	$4\left(\frac{1}{2}-b\right)^3\left(\frac{1}{2}+b\right)$
S_2={0011,0101,0110,1001,1010,1100}	$6\left(\frac{1}{2}-b\right)^2\left(\frac{1}{2}+b\right)^2$
S_3={0111,1011,1101,1110}	$4\left(\frac{1}{2}-b\right)\left(\frac{1}{2}+b\right)^3$
S_4={1111}	$\left(\frac{1}{2}+b\right)^4$

I compute the probability

$$\pi_0 = P\left(\bigoplus_{i=0}^{3} x_i = 0\right), \tag{110}$$

which can be used to calculate the bias $b = |\pi_0 - 0.5|$.

Equation (110) can be evaluated by considering the words leading to an output of either 1 or 0. For this, I use that the XOR operation is the parity of the inputs. Therefore, words in S_0, S_2, and S_4 in table 5 lead to a 0 output and hence

$$\pi_0 = P\left([x_3x_2x_1x_0] \in S_0 \cup [x_3x_2x_1x_0] \in S_2 \cup [x_3x_2x_1x_0] \in S_4\right). \tag{111}$$

Assuming uncorrelated bitstreams, these three events are disjoint. Using values from Table 5, I calculate

$$\begin{aligned}\pi_0 &= P([x_3x_2x_1x_0] \in S_0) + P([x_3x_2x_1x_0] \in S_2) + P([x_3x_2x_1x_0] \in S_4), \\ &= \left(\frac{1}{2}-b\right)^4 + 6\left(\frac{1}{2}-b\right)^2\left(\frac{1}{2}+b\right)^2 + \left(\frac{1}{2}+b\right)^4, \\ &= 1/2 + 8b^4. \end{aligned} \tag{112}$$

Therefore, the bias of the 4-input XOR gate is $8b^4$, corresponding to a reduction of a 1% bias at the input to a bias of $8 \cdot 10^{-8}$ at the output. In general, an n-input XOR gate reduced a bias b to a bias $\tilde{b} = 2^{n-1}b^n$, which can be shown by induction.

C.2 CHIMERA STATES WITH RANDOMIZED INITIAL CONDITIONS

To obtain chimera states, the initial conditions usually have to be prepared carefully [Abr04]. Homogeneous initial conditions, on the other hand, do not lead to chimera states in a homogeneous network model, such as the Kuramoto model. I show here, however, that both random and homogeneous initial conditions can lead to chimera states in the model when the phase shift parameter α_{ij} is heterogeneous.

I initialize the simulation of Eq. (75) with random initial conditions ($\phi_i \in [0, 2\pi]$ uniformly distributed) and synchronous initial conditions [$\phi_i = 0$ $(i = 1, ..., N)$] for heterogeneous phase lag parameter $\alpha_{ij} = 0.1 \pm 0.2$ with a Gaussian distribution of mean 0.1 and standard deviation 0.2. The result is shown Fig. 86(a) and (b) leading to a chimera state for both initial conditions as a result of the heterogeneous phase lag parameter.

The finding that random and homogeneous initial conditions can lead to chimera states for heterogeneous networks increases the probability to observe chimera states in nature.

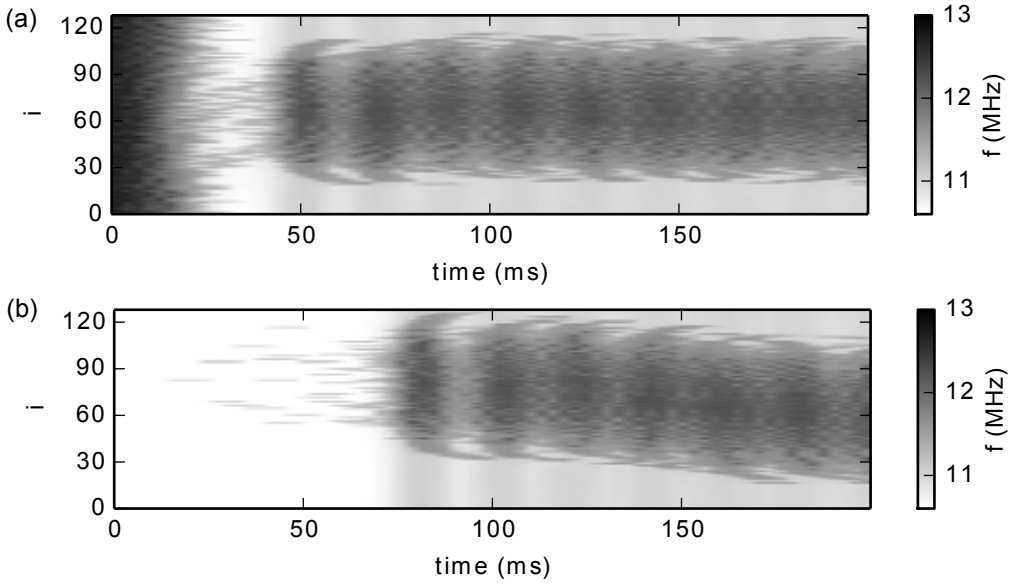


FIGURE 86: Numerical simulation of the Eq. (75) with heterogeneous $\alpha_{i,j}$. (a) Random initial conditions, (b) homogeneous initial conditions. The oscillator index is shifted so that the unsynchronized region of the chimera appears in the middle of the network. Other parameters as in Fig. 5 in the main article.

C.3 CLUSTER SYNCHRONIZATION IN COUPLED NEURAL POPULATIONS

In this section, I study the dynamics of an artificial neural network of 80 excitable nodes. With this, I show that I can build large meta-networks of silicon neurons.[1]

I assemble Boolean neurons in a network of four distinct neural populations, as illustrated in Fig. 87(a) and(b). Dynamical properties of similar network topologies have been investigated theoretically [Vico8, Kop12] because of their similarity to neural circuits such as the thalamic circuitry embedded in the brain [Jono2, Shio3a].

In the experiment, each population consists of 20 excitable nodes, totaling 80 nodes for the entire network. Nodes within a population are connected with probability $p = 0.3$, where links are realized with on-chip wires leading to small link time delays. Nodes of different populations are connected with probability $p = 0.015$ with time-delay links with $\tau = (16.8 \pm 0.6)$ ns. These probabilities lead to strong connections of nodes within a population with negligible link delay and loose connections between nodes of different populations with significant time delay τ.

The dynamics of the artificial neural network is described theoretically by Kanter and collaborators [Kan11a, Var12]. According to the theory, the net-

1 This study is published in Ref. [Ros13b].

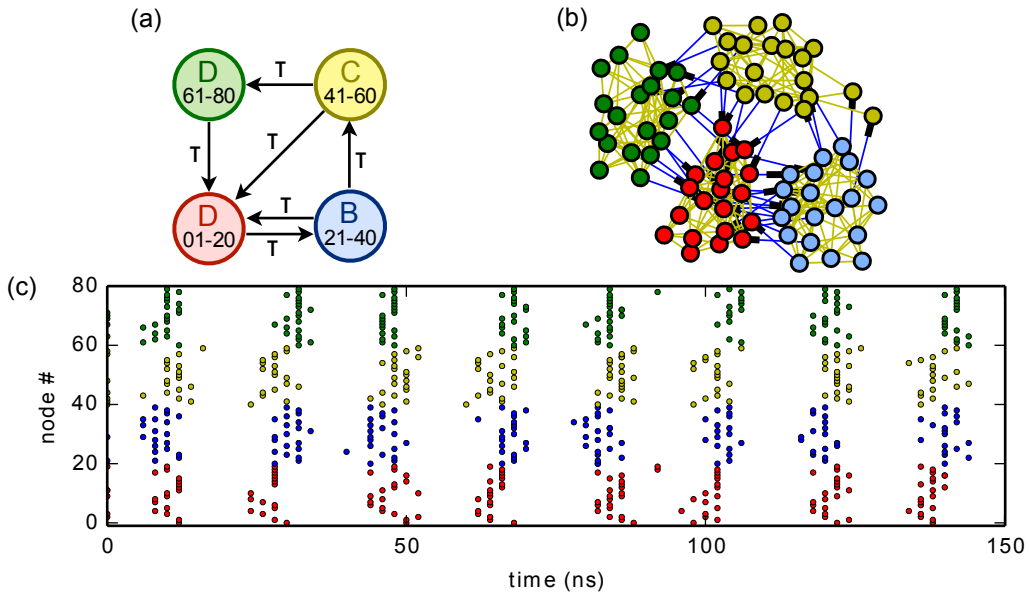


FIGURE 87: (a) Topology of four coupled populations that involves loops of four, three and two elements. (b) Implemented topology, where nodes of the same (different) populations are connected with directed links. An initial pulse is sent to one node to perturb the network out of its quiescent state. (c) Raster diagram of the network for $n_\tau = 60$ $[\tau = (16.8 \pm 0.6)\,\text{ns}]$, $n_{\text{pulse}} = 4$ $[T_{\text{pulse}} = (1.12 \pm 0.04)\,\text{ns}]$, and (b) $n_{\text{ref}} = 20$ $[T_{\text{ref}} = (5.6 \pm 0.2)\,\text{ns}]$.

work dynamics is given by the network topology of the community structure by the greatest common divisor (GCD) of the sizes of directed loops. In the network topology in Fig. 87, inspired by Fig. 1a in Ref. [[Kan11a]], there are three directed loops of two, three, and four neural populations, respectively. Therefore, the theory predicts a number of synchronized zero-lag synchronized clusters of $\text{GCD}(2,3,4) = 1$, i.e., all the populations are predicted to be synchronized with zero time lag (see also Sec. 9.3.2).

The experimental dynamics of the network is reported in the raster diagram of Fig. 87(c), where each circle corresponds to a pulse generated by a node.

I observe that all the artificial neurons of the four populations generate pulse trains with period $\tau \pm \Delta\tau$ with $\Delta\tau = 5\,\text{ns}$. The dispersion $\Delta\tau$ in the period of the pulse trains originates from heterogeneities in the values of the link time delays and limited resolution of the integrated measurement system. The network is near-zero-lag synchronized, which is consistent with the theoretical predictions.

BIBLIOGRAPHY

[Al-06] S. R. Al-araji, Z. M. Hussain, and M. A. Al-qutayri: *Digital Phase Lock Loops* (Springer Verlag, New York, NY, 2006).

[Abe99] J. Abello, P. M. Pardalos, and R. M. G. C: *On maximum clique problems in very large graphs*. DIMACS series **50**, 119 (1999).

[Abr04] D. M. Abrams and S. H. Strogatz: *Chimera States for Coupled Oscillators*. Phys. Rev. Lett. **93**, 174102 (2004).

[Abr08] D. M. Abrams, R. Mirollo, S. H. Strogatz, and D. A. Wiley: *Solvable model for chimera states of coupled oscillators*. Phys. Rev. Lett. **101**, 084103 (2008).

[Ade81] W. J. Adelman and D. E. Goldman: *The biophysical approach to excitable systems* (Springer, New York, 1981).

[Adh11] B. M. Adhikari, A. Prasad, and M. Dhamala: *Time-delay-induced phase-transition to synchrony in coupled bursting neurons*. Chaos **21**, 023116 (2011).

[Ade83] L. M. Adleman, R. L. Rivest, and A. Shamir: *Cryptographic communications system and method*. US Patent 4,405,829 (1983).

[Alb99] R. Albert, H. Jeong, and A.-L. Barabási: *The diameter of the world wide web*. Nature **401**, 130 (1999).

[Alb00] R. Albert, H. Jeong, and A.-L. Barabási: *Error and attack tolerance of complex networks*. Nature **406**, 378 (2000).

[Alb00b] R. Albert and A.-L. Barabási: *Dynamics of Complex Systems: Scaling Laws for the Period of Boolean Networks*. Phys. Rev. Lett. **84**, 5660 (2000).

[Alb02a] R. Albert and A.-L. Barabási: *Statistical mechanics of complex networks*. Rev. Mod. Phys. **74**, 47 (2002).

[Alt10] Altera: *Cyclone III, Device Handbook, Volume 1* (2010).

[Ama12] A. Amann: *Complex Networks Based on Coupled Two-Mode Lasers*. In *Nonlinear Laser Dynamics - From Quantum Dots to Cryptography*, edited by K. Lüdge, chap. 10, pp. 245–267 (WILEY-VCH, Weinheim, 2012).

[Ant93] P. Antognetti, G. Massobrio, and G. Massobrio: *Semiconductor device modeling with SPICE* (McGraw-Hill, New York, 1993).

[App22] E. Appleton: *Automatic synchronization of triode oscillators*. In *Proc. Cambridge Phil. Soc.*, vol. 21, p. 231 (1922).

[App11] L. Appeltant, M. C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. R. Mirasso, and I. Fischer: *Information processing using a single dynamical node as complex system*. Nat. Commun. **2**, 468 (2011).

[Arg05] A. Argyris, D. Syvridis, L. Larger, V. Annovazzi-Lodi, P. Colet, I. Fischer, J. García-Ojalvo, C. R. Mirasso, L. Pesquera, and K. A. Shore: *Chaos-based communications at high bit rates using commercial fibre-optic links*. Nature **438**, 343 (2005).

[Art06] J. Arthur and K. Boahen: *Learning in Silicon: Timing is Everything*. In *Adv. Neur. Inf. Proc. Syst.*, edited by Y. Weiss, B. Schölkopf, and J. Platt, vol. 18, pp. 75–82 (MIT Press, Cambridge, MA, 2006).

[Asm07] S. Asmussen and P. W. Glynn: *Stochastic Simulation: Algorithms and Analysis* (Springer-Verlag, Berlin, 2007).

[Ata10] F. M. Atay, ed.: *Complex Time-Delay Systems*. Understanding Complex Systems (Springer, Berlin, 2010).

[Bae08] C. Baetoniu: *Method and apparatus for true random number generation*. U.S. Patent 7,389,316 (2008).

[Bar03a] A.-L. Barabási and E. Bonabeau: *Scale-free networks*. Sci. Am. **288**, 50 (2003).

[Belo0b] M. J. Bellido-Diaz, J. Juan-Chico, A. J. Acosta, M. Valencia, and J. L. Huertas: *Logical modelling of delay degradation effect in static CMOS gates*. IEE Proc. Circuits Dev. Syst. **147**, 107 (2000).

[Bel59] B. P. Belousov: *A periodic reaction and its mechanism*. Compilation of Abstracts on Radiation Medicine **147** (1959).

[Bes03] R. E. Best: *Phase-Locked Loops* (McGraw-Hill, New York, 2003).

[Bin10] K. Binder and D. W. Heermann: *Monte Carlo simulation in statistical physics: an introduction* (Springer, New York, 2010).

[Bla13] K. Blaha, J. Lehnert, A. Keane, T. Dahms, P. Hövel, E. Schöll, and J. L. Hudson: *Clustering in delay-coupled smooth and relaxational chemical oscillators*. Phys. Rev. E **88**, 062915 (2013).

[Boa00] K. A. Boahen: *Point-to-point connectivity between neuromorphic chips using address events*. IEEE Trans. Circuits Syst. II **47**, 416 (2000).

[Boa05] K. Boahen: *Neuromorphic microchips*. Sci. Am. **292**, 56 (2005).

[Boa06] K. Boahen: *Neurogrid: emulating a million neurons in the cortex*. In *IEEE international conference of the engineering in medicine and biology society* (2006).

[Bra09] S. A. Brandstetter, M. A. Dahlem, and E. Schöll: *Interplay of time-delayed feedback control and temporally correlated noise in excitable systems*. Phil. Trans. R. Soc. A **368**, 391 (2010).

[Bre08a] M. Brede: *Synchrony-optimized networks of non-identical Kuramoto oscillators*. Phys. Lett. A **372**, 2618 (2008).

[Bro08] S. Brown and Z. Vranesic: *Fundamentals of Digital Logic with Verilog Design* (Mc Graw Hill, New York, 2008).

[Buc76] J. Buck: *Synchronous fireflies*. Sci. Am. **234**, 74 (1976).

[Bur06] A. N. Burkitt: *A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input*. Biological Cybernetics **95**, 1 (2006).

[Cak14] C. Cakan, J. Lehnert, and E. Schöll: *Heterogeneous Delays in Neural Networks*. Eur. Phys. J. B **87**, 54 (2014).

[Cal10] K. E. Callan, L. Illing, Z. Gao, D. J. Gauthier, and E. Schöll: *Broadband chaos generated by an optoelectronic oscillator*. Phys. Rev. Lett. **104**, 113901 (2010).

[Cav10] H. L. D. d. S. Cavalcante, D. J. Gauthier, J. E. S. Socolar, and R. Zhang: *On the origin of chaos in autonomous Boolean networks*. Phil. Trans. R. Soc. A **368**, 495 (2010).

[Cha05] M. Chaves, R. Albert, and E. D. Sontag: *Robustness and fragility of Boolean models for genetic regulatory networks*. Journal of theoretical biology **235**, 431 (2005).

[Che10] D. Cheng, H. Qi, and Z. Li: *Analysis and control of Boolean networks: a semi-tensor product approach* (Springer, New York, 2010).

[Che13] X. Cheng, M. Sun, and J. E. S. Socolar: *Autonomous Boolean modelling of developmental gene regulatory networks*. J. R. Soc. Interface **10** (2013).

[Cho07] C. U. Choe, V. Flunkert, P. Hövel, H. Benner, and E. Schöll: *Conversion of Stability in Systems close to a Hopf Bifurcation by Time-delayed Coupling*. Phys. Rev. E **75**, 046206 (2007).

[Cho09] C. U. Choe, T. Dahms, P. Hövel, and E. Schöll: *Controlling synchrony by delay coupling in networks: from in-phase to splay and cluster states*. Phys. Rev. E **81**, 025205(R) (2010).

[Cho14] C. U. Choe, R.-S. Kim, H. Jang, P. Hövel, and E. Schöll: *Delayed-Feedback Control with Arbitrary and Distributed Delay-Time and Non-invasive Control of Synchrony in Networks Coupled with Heterogeneous Delays*. Int. J. Dynam. Control **2**, 2 (2014).

[Cla98] J. R. Clay: *Excitability of the squid giant axon revisited*. Journal of neurophysiology **80**, 903 (1998).

[Cla08] J. R. Clay, D. Paydarfar, and D. B. Forger: *A simple modification of the Hodgkin and Huxley equations explains type 3 excitability in squid giant axons*. Journal of The Royal Society Interface **5**, 1421 (2008).

[Col94] P. Colet and R. Roy: *Digital communication with synchronized chaotic lasers*. Opt. Lett. **19**, 2056 (1994).

[Cov91] T. M. Cover and J. A. Thomas: *Elements of Information Theory* (Wiley, New York, 1991).

[Cuo93] K. M. Cuomo and A. V. Oppenheim: *Circuit implementation of synchronized chaos with applications to communications*. Phys. Rev. Lett. **71**, 65 (1993).

[Dah08c] M. A. Dahlem, G. Hiller, A. Panchuk, and E. Schöll: *Dynamics of delay-coupled excitable neural systems*. Int. J. Bifur. Chaos **19**, 745 (2009).

[Dah12] T. Dahms, J. Lehnert, and E. Schöll: *Cluster and group synchronization in delay-coupled networks*. Phys. Rev. E **86**, 016202 (2012).

[Dar93] M. S. Darby and L. A. Mysak: *A Boolean delay equation model of an interdecadal Arctic climate cycle*. Climate Dyn. **8**, 241 (1993).

[Dav92a] J. M. Davidenko, A. V. Pertsov, R. Salomonsz, W. Baxter, and J. Jalife: *Stationary and drifting spiral waves of excitation in isolated cardiac muscle*. Nature **355**, 349 (1992).

[Dav12] N. Davidson, M. Nixon, E. Ronen, M. Fridman, and A. Friesem: *Phase Locking Large Arrays of Lasers*. In *Conference on Lasers and Electro-Optics 2012*, p. CTu3N.7 (2012).

[Day03] P. Dayan and L. Abbott: *Theoretical neuroscience: computational and mathematical modeling of neural systems*. J. Cogn. Neurosci. **15**, 154 (2003).

[Dee84] D. Dee and M. Ghil: *Boolean difference equations, I: Formulation and dynamic behavior*. SIAM J. Appl. Math. **44**, 111 (1984).

[Dep14] Department of Homeland Security: *Cybersecurity*. http://www.dhs.gov/topic/cybersecurity (2/22/2014).

[Dhu08] O. D'Huys, R. Vicente, T. Erneux, J. Danckaert, and I. Fischer: *Synchronization properties of network motifs: Influence of coupling delay and symmetry*. Chaos **18**, 037116 (2008).

[Dhu11] O. D'Huys, I. Fischer, J. Danckaert, and R. Vicente: *Role of delay for the symmetry in the dynamics of networks*. Phys. Rev. E **83**, 046223 (2011).

[Dic07] M. Dichtl and J. Golić: *High-Speed True Random Number Generation with Logic Gates Only*. In *Cryptographic Hardware and Embedded Systems - CHES 2007*, edited by P. Paillier and I. Verbauwhede, vol. 4727, pp. 45–62 (Springer, 2007).

[Don09] M. J. Donahoo and K. L. Calvert: *TCP/IP sockets in C: practical guide for programmers* (Morgan Kaufmann, 2009).

[Edw00] R. Edwards and L. Glass: *Combinatorial explosion in model gene networks*. Chaos **10**, 691 (2000).

[Edw05] R. Edwards and L. Glass: *A calculus for relating the dynamics and structure of complex biological networks*. Adv. Chem. Phys. **132**, 151 (2005).

[Elo00] M. B. Elowitz and S. Leibler: *A synthetic oscillatory network of transcriptional regulators*. Nature **403**, 335 (2000).

[Eng10] A. Englert, W. Kinzel, Y. Aviad, M. Butkovski, I. Reidler, M. Zigzag, I. Kanter, and M. Rosenbluh: *Zero lag synchronization of chaotic systems with time delayed couplings*. Phys. Rev. Lett. **104**, 114102 (2010).

[Ern09] T. Erneux: *Applied delay differential equations*, vol. 3 (Springer, New York, 2009).

[Far02] I. Farkas, D. Helbing, and T. Vicsek: *Social behaviour: Mexican waves in an excitable medium*. Nature **419**, 131 (2002).

[Fie09] B. Fiedler, V. Flunkert, P. Hövel, and E. Schöll: *Delay stabilization of periodic orbits in coupled oscillator systems*. Phil. Trans. R. Soc. A **368**, 319 (2010).

[Fis93] R. Fisch, J. Gravner, and D. Griffeath: *Metastability in the Greenberg-Hastings model*. Ann. Appl. Prob. pp. 935–967 (1993).

[Fis06] I. Fischer, R. Vicente, J. M. Buldú, M. Peil, C. R. Mirasso, M. C. Torrent, and J. García-Ojalvo: *Zero-Lag Long-Range Synchronization via Dynamical Relaying*. Phys. Rev. Lett. **97**, 123902 (2006).

[Fit55] R. FitzHugh: *Mathematical models of threshold phenomena in the nerve membrane*. Bull. Math. Biol. **17**, 257 (1955).

[Fit61] R. FitzHugh: *Impulses and physiological states in theoretical models of nerve membrane*. Biophys. J. **1**, 445 (1961).

[Flu09] V. Flunkert, O. D'Huys, J. Danckaert, I. Fischer, and E. Schöll: *Bubbling in delay-coupled lasers*. Phys. Rev. E **79**, 065201 (R) (2009).

[Flu10] V. Flunkert, S. Yanchuk, T. Dahms, and E. Schöll: *Synchronizing Distant Nodes: A Universal Classification of Networks*. Phys. Rev. Lett. **105**, 254101 (2010).

[Flu10b] V. Flunkert, S. Yanchuk, T. Dahms, and E. Schöll: *Synchronizing distant nodes: a universal classification of networks*. Phys. Rev. Lett. **105**, 254101 (2010).

[Flu11] V. Flunkert: *Delay-coupled Complex Systems: And Applications to Lasers* (Springer, Heidelberg, 2011).

[Flu13] V. Flunkert, I. Fischer, and E. Schöll: *Dynamics, control and information in delay-coupled systems*. Theme Issue of Phil. Trans. R. Soc. A **371**, 20120465 (2013).

[Fos96] J. Foss, A. Longtin, B. Mensour, and J. Milton: *Multistability and Delayed Recurrent Loops*. Phys. Rev. Lett. **76**, 708 (1996).

[Fri97a] P. Fries, P. R. Roelfsema, A. K. Engel, P. König, and W. Singer: *Synchronization of oscillatory responses in visual cortex correlates with perception in interocular rivalry*. Proc. Natl. Acad. Sci. U.S.A. **94**, 12699 (1997).

[Fri98] E. Friedgut: *Boolean functions with low average sensitivity depend on few coordinates*. Combinatorica **18**, 27 (1998).

[Fri09a] J. Friedrich and W. Kinzel: *Dynamics of recurrent neural networks with delayed unreliable synapses: metastable clustering*. J. Comput. Neurosci. **27**, 65 (2009).

[Fri10] M. Fridman, M. Nixon, N. Davidson, and A. A. Friesem: *Passive phase locking of 25 fiber lasers*. Opt. Lett. **35**, 1434 (2010).

[Fog84] F. Fogelman-Soulie: *Frustration and stability in random Boolean networks*. Discrete Appl. Math. **9**, 139 (1984).

[Gao09] Z. Gao, H. L. D. de S. Cavalcante, S. D. Cohen, R. Zhang, J. E. S. Socolar, and D. J. Gauthier: *Using synchronization of chaos to identify multiple delay times in Boolean-delay systems*. Poster Presentation (2009).

[Ghi85] M. Ghil and A. Mullhaupt: *Boolean delay equations. II. Periodic and aperiodic solutions*. J. Stat. Phys. **41**, 125 (1985).

[Ghi87] M. Ghil, A. Mullhaupt, and P. Pestiaux: *Deep water formation and Quaternary glaciations*. Climate Dyn. **2**, 1 (1987).

[Ghi08] M. Ghil, I. Zaliapin, and B. Coluzzi: *Boolean delay equations. A simple way of looking at complex systems*. Physica D **237**, 2967 (2008).

[Gla98] L. Glass and C. Hill: *Ordered and disordered dynamics in random networks*. Europhys. Lett. **41**, 599 (1998).

[Gla05] L. Glass, T. Perkins, J. Mason, H. T. Siegelmann, and E. R.: *Chaotic Dynamics in an Electronic Model of a Genetic Network*. J. Stat. Phys. **121**, 969 (2005).

[Gla13] L. Glass: Private Communication at Dynamics Days US (2013).

[Gol96] I. Goldberg and D. Wagner: *Randomness and the Netscape Browser*. Dr. Dobb's Journal **21**, 66 (1996).

[Gon08] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabási: *Understanding individual human mobility patterns*. Nature **453**, 779 (2008).

[Gon12] X. Gong and J. E. S. Socolar: *Quantifying the complexity of random Boolean networks*. Phys. Rev. E **85**, 066107 (2012).

[Gop14] R. Gopal, V. K. Chandrasekar, A. Venkatesan, and M. Lakshmanan: *Observation and characterization of chimera states in coupled dynamical systems with nonlocal coupling* (2014). ArXiv:1403.4022.

[Gre78] J. M. Greenberg and S. Hastings: *Spatial patterns for discrete models of diffusion in excitable media*. SIAM J. Appl. Math. **34**, 515 (1978).

[Hag12] A. Hagerstrom, T. E. Murphy, R. Roy, P. Hövel, I. Omelchenko, and E. Schöll: *Experimental Observation of Chimeras in Coupled-Map Lattices*. Nature Physics **8**, 658 (2012).

[Haj98] A. Hajimiri and T. H. Lee: *A General Theory of Phase Noise in Electrical Oscillators*. IEEE J. Solid-St. Circ. **33**, 179 (1998).

[Har01] S. L. Harmer, S. Panda, and S. A. Kay: *Molecular bases of circadian rhythms*. Annu. Rev. Cell Dev. Biol. **17**, 215 (2001).

[Har11] T. Harayama, S. Sunada, K. Yoshimura, P. Davis, K. Tsuzuki, and A. Uchida: *Fast nondeterministic random-bit generation using on-chip chaos lasers*. Phys. Rev. A **83**, 031803(R) (2011).

[Har12] T. Harayama, S. Sunada, K. Yoshimura, J. Muramatsu, K. I. Arai, A. Uchida, and P. Davis: *Theory of fast nondeterministic physical random-bit generation with chaotic lasers*. Phys. Rev. E **85**, 046215 (2012).

[Hau06] B. Hauschildt, N. B. Janson, A. G. Balanov, and E. Schöll: *Noise-induced cooperative dynamics and its control in coupled neuron models*. Phys. Rev. E **74**, 051906 (2006).

[Hau07a] C. Hauptmann, O. E. Omel'chenko, O. V. Popovych, Y. L. Maistrenko, and P. A. Tass: *Control of spatially patterned synchrony with multisite delayed feedback*. Phys. Rev. E **76**, 066209 (2007).

[Hei01b] T. Heil, I. Fischer, W. Elsäßer, J. Mulet, and C. R. Mirasso: *Chaos Synchronization and Spontaneous Symmetry-Breaking in Symmetrically Delay-Coupled Semiconductor Lasers*. Phys. Rev. Lett. **86**, 795 (2001).

[Hei11] S. Heiligenthal, T. Dahms, S. Yanchuk, T. Jüngling, V. Flunkert, I. Kanter, E. Schöll, and W. Kinzel: *Strong and weak chaos in nonlinear networks with time-delayed couplings*. Phys. Rev. Lett. **107**, 234102 (2011).

[Hei13] S. Heiligenthal, T. Jüngling, O. D'Huys, D. A. Arroyo-Almanza, M. C. Soriano, I. Fischer, I. Kanter, and W. Kinzel: *Strong and weak chaos in networks of semiconductor lasers with time-delayed couplings*. Phys. Rev. E **88**, 012902 (2013).

[Hic11] K. Hicke, O. D'Huys, V. Flunkert, E. Schöll, J. Danckaert, and I. Fischer: *Mismatch and synchronization: Influence of asymmetries in systems of two delay-coupled lasers*. Phys. Rev. E **83**, 056211 (2011).

[Hod48] A. L. Hodgkin: *The local electric changes associated with repetitive action in a medullated axon*. J. Physiol. **107**, 165 (1948).

[Hod52] A. L. Hodgkin and A. F. Huxley: *A Quantitative Description of Membrane Current and its Application to Conduction and Excitation in Nerve*. J. Physiol. **117**, 500 (1952).

[Hop82] J. J. Hopfield: *Neural networks and physical systems with emergent collective computational abilities*. Proc. Natl. Acad. Sci. U.S.A. **79**, 2554 (1982).

[Hor89] P. Horowitz, W. Hill, and T. C. Hayes: *The art of electronics*, vol. 2 (Cambridge university press, Cambridge, 1989).

[Hoe05] P. Hövel and E. Schöll: *Control of unstable steady states by time-delayed feedback methods*. Phys. Rev. E **72**, 046203 (2005).

[Hoe09] P. Hövel, M. A. Dahlem, and E. Schöll: *Control of synchronization in coupled neural systems by time-delayed feedback*. Int. J. Bifur. Chaos **20**, 813 (2010).

[Hoe10b] P. Hövel: *Control of Complex Nonlinear Systems with Delay*. Springer Theses (Springer, Heidelberg, 2010).

[Hsu99] T.-Y. Hsu, B.-J. Shieh, and C.-Y. Lee: *An all-digital phase-locked loop (ADPLL)-based clock recovery circuit*. IEEE Journal of Solid-State Circuits **34**, 1063 (1999).

[Huy86] C. Huygens: *The pendulum clock or geometrical demonstrations concerning the motion of pendula as applied to clocks* (Iowa State University Press, Ames, IA, 1986). Translated by R. Blackwell.

[Iha93] S. Ihara: *Information theory for continuous systems*, vol. 2 (World Scientific, Singapore, 1993).

[Ike79] K. Ikeda: *Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system*. Optics communications **30**, 257 (1979).

[Ike82] K. Ikeda, K. Kondo, and O. Akimoto: *Successive Higher-Harmonic Bifurcations in Systems with Delayed Feedback*. Phys. Rev. Lett. **49**, 1467 (1982).

[Ill11] L. Illing, C. D. Panda, and L. Shareshian: *Isochronal chaos synchronization of delay-coupled optoelectronic oscillators*. Phys. Rev. E **84**, 016213 (2011).

[Ind11] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. van Schaik, R. Etienne-Cummings, T. Delbruck, S. C. Liu, P. Dudek, P. Häfliger, S. Renaud, J. Schemmel, G. Cauwenberghs, J. Arthur, K. Hynna, F. Folowosele, S. Saighi, T. Serrano-Gotarredona, J. Wijekoon, Y. Wang, and K. Boahen: *Neuromorphic silicon neuron circuits*. Front. Neurosci. **5**, 73 (2011).

[Izh06] E. M. Izhikevich and R. FitzHugh: *FitzHugh-Nagumo model*. Scholarpedia **1**, 1349 (2006). revision 123664.

[Izh07] E. M. Izhikevich: *Dynamical Systems in Neuroscience* (MIT Press, Cambridge, 2007).

[Jae04] H. Jaeger and H. Haas: *Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication*. Science **304**, 78 (2004).

[Jeo00] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási: *The large-scale organization of metabolic networks*. Nature **407**, 651 (2000).

[Jeo04] Y. Jeong, S. Jung, and J. Liu: *A CMOS impulse generator for UWB wireless communication systems*. In *Int. Symp. Circuits and Syst.*, vol. 4, pp. 129 – 132 (2004).

[Jon02] E. G. Jones: *Thalamic circuitry and thalamocortical synchrony*. Phil. Trans. R. Soc. B **357**, 1659 (2002).

[Jun99] B. Jun and P. Kocher: *The Intel random number generator*. White paper prepared for Intel Corporation, http://decuslib.com/decus/vmslt99a/sec/intelrng.pdf (1999).

[Jus09] W. Just, A. Pelster, M. Schanz, and E. Schöll: *Delayed Complex Systems*. Theme Issue of Phil. Trans. R. Soc. A **368**, 301 (2010).

[Kan10] I. Kanter, Y. Aviad, I. Reidler, E. Cohen, and M. Rosenbluh: *An optical ultrafast random bit generator*. Nature Photonics **4**, 58 (2010).

[Kan11] I. Kanter, M. Zigzag, A. Englert, F. Geissler, and W. Kinzel: *Synchronization of unidirectional time delay chaotic networks and the greatest common divisor*. Europhys. Lett. **93**, 60003 (2011).

[Kan11a] I. Kanter, E. Kopelowitz, R. Vardi, M. Zigzag, W. Kinzel, M. Abeles, and D. Cohen: *Nonlocal mechanism for cluster synchronization in neural circuits*. Europhys. Lett. **93**, 66001 (2011).

[Kat98] H. Kato: *A Dynamic Formulation of Ring Oscillator as Solitary-Wave Propagator*. IEEE Trans. Circuits Syst. I **45**, 98 (1998).

[Kat07] J. Katz and Y. Lindell: *Introduction to modern cryptography: principles and protocols* (CRC Press, Boca Raton, 2007).

[Kau69] S. A. Kauffman: *Metabolic stability and epigenesis in randomly constructed genetic nets*. J. Theoret. Biol. **22**, 437 (1969).

[Kau93] S. A. Kauffman: *The origins of order: Self organization and selection in evolution* (Oxford University Press, New York, 1993).

[Kau03] S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein: *Random Boolean Networks models and the yeast transcriptional network*. Proc. Natl. Acad. Sci. U.S.A. **100**, 14796 (2003).

[Kau04] S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein: *Genetic networks with canalyzing Boolean rules are always stable*. Proc. Natl. Acad. Sci. U.S.A. **101**, 17102 (2004).

[Kea12] A. Keane, T. Dahms, J. Lehnert, S. A. Suryanarayana, P. Hövel, and E. Schöll: *Synchronisation in networks of delay-coupled type-I excitable systems*. Eur. Phys. J. B **85**, 407 (2012).

[Kee09] J. P. Keener and J. Sneyd: *Mathematical physiology: Cellular physiology*, vol. 1 (Springer Verlag, New York, 2009).

[Khe94] M. Khellah, S. Brown, and Z. Vranesic: *Minimizing interconnection delays in array-based FPGAs*. In *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 181–184 (IEEE, 1994).

[Kin09] W. Kinzel, A. Englert, G. Reents, M. Zigzag, and I. Kanter: *Synchronization of networks of chaotic units with time-delayed couplings*. Phys. Rev. E **79**, 056207 (2009).

[Koc96] L. Kocarev and U. Parlitz: *Generalized synchronization, predictability, and equivalence of unidirectionally coupled dynamical systems*. Phys. Rev. Lett. **76**, 1816 (1996).

[Kop12] E. Kopelowitz, M. Abeles, D. Cohen, and I. Kanter: *Sensitivity of global network dynamics to local parameters versus motif structure in a cortexlike neuronal model*. Phys. Rev. E **85**, 051902 (2012).

[Kou05] Y. C. Kouomou, P. Colet, L. Larger, and N. Gastaud: *Chaotic breathers in delayed electro-optical systems*. Phys. Rev. Lett. **95**, 203903 (2005).

[Kum99] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins: *Trawling the Web for emerging cyber-communities*. Comput. Netw. **31**, 1481 (1999).

[Kur84] Y. Kuramoto: *Chemical Oscillations, Waves, and Turbulence* (Springer, New York, 1984).

[Kur02a] Y. Kuramoto and D. Battogtokh: *Coexistence of Coherence and Incoherence in Nonlocally Coupled Phase Oscillators.* Nonlin. Phen. in Complex Sys. **5**, 380 (2002).

[Kuz98] I. A. Kuznetsov: *Elements of applied bifurcation theory*, vol. 112 (Springer, New York, 1998).

[Kyr11] Y. N. Kyrychko, K. B. Blyuss, and E. Schöll: *Amplitude death in systems of coupled oscillators with distributed-delay coupling*. Eur. Phys. J. B **84**, 307 (2011).

[Lad13] J. Ladenbauer, J. Lehnert, H. Rankoohi, T. Dahms, E. Schöll, and K. Obermayer: *Adaptation controls synchrony and cluster states of coupled threshold-model neurons*. Phys. Rev. E **88**, 042713 (2013).

[Lar10] L. Larger and J. M. Dudley: *Optoelectronic chaos*. Nature **465**, 41 (2010).

[Lar13] L. Larger, B. Penkovsky, and Y. L. Maistrenko: *Virtual Chimera States for Delayed-Feedback Systems*. Phys. Rev. Lett. **111**, 054103 (2013).

[Leh11] J. Lehnert, T. Dahms, P. Hövel, and E. Schöll: *Loss of synchronization in complex neural networks with delay*. Europhys. Lett. **96**, 60013 (2011).

[Lev01] I. B. Levitan and L. K. Kaczmarek: *The Neuron: Cell and Molecular Biology: Cell and Molecular Biology* (Oxford University Press, New York, 2001).

[Li11] X. Li, A. B. Cohen, T. E. Murphy, and R. Roy: *Scalable parallel physical random number generator based on a superluminescent LED*. Opt. Lett. **36**, 1020 (2011).

[Li13] W. Li, I. Reidler, Y. Aviad, Y. Huang, H. Song, Y. Zhang, M. Rosenbluh, and I. Kanter: *Fast Physical Random-Number Generation Based on Room-Temperature Chaotic Oscillations in Weakly Coupled Superlattices*. Phys. Rev. Lett. **111**, 044102 (2013).

[Lin04] B. Lindner, J. García-Ojalvo, A. B. Neiman, and L. Schimansky-Geier: *Effects of noise in excitable systems*. Phys. Rep. **392**, 321 (2004).

[Liu04] F.-Y. Lin and J.-M. Liu: *Chaotic Radar Using Nonlinear Laser Dynamics*. IEEE J. Quantum Electron. **40**, 815 (2004).

[Liu11] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási: *Controllability of complex networks*. Nature **473**, 167 (2011).

[Loc02] A. Locquet, C. Masoller, and C. R. Mirasso: *Synchronization regimes of optical-feedback-induced chaos in unidirectionally coupled semiconductor lasers*. Phys. Rev. E **65**, 56205 (2002).

[Lor63] E. N. Lorenz: *Deterministic nonperiodic flow*. J. Atmos. Sci. **20**, 130 (1963).

[Lu04] J. Lü, X. Yu, and G. Chen: *Chaos synchronization of general complex dynamical networks*. Physica A **334**, 281 (2004).

[Maa01] W. Maass and C. M. Bishop: *Pulsed neural networks* (MIT press, Cambridge, 2001).

[Maa02] W. Maass, T. Natschläger, and H. Markram: *Real-time computing without stable states: A new framework for neural computation based on perturbations*. Neural Comp. **14**, 2531 (2002).

[Mar01] E. Marder and D. Bucher: *Central pattern generators and the control of rhythmic movements*. Curr. Biol. **11**, R986 (2001).

[Mar10] E. A. Martens, C. R. Laing, and S. H. Strogatz: *Solvable model of spiral wave chimeras*. Phys. Rev. Lett. **104**, 044101 (2010).

[Mar13] E. A. Martens, S. Thutupalli, A. Fourrière, and O. Hallatschek: *Chimera States in Mechanical Oscillator Networks*. Proc. Natl. Acad. Sci. U.S.A. **110**, 10563 (2013).

[Mas01] C. Masoller: *Anticipation in the Synchronization of Chaotic Semiconductor Lasers with Optical Feedback*. Phys. Rev. Lett. **86**, 2782 (2001).

[Mas08] C. Masoller, M. C. Torrent, and J. García-Ojalvo: *Interplay of subthreshold activity, time-delayed feedback, and noise on neuronal firing patterns*. Phys. Rev. E **78**, 041907 (2008).

[Mas09a] C. Masoller, M. C. Torrent, and J. García-Ojalvo: *Dynamics of globally delay-coupled neurons displaying subthreshold oscillations*. Phil. Trans. R. Soc. A **367**, 3255 (2009).

[Max09] C. Maxfield: *FPGAs World Class Designs* (Newnes, Burlington, 2009).

[Mcc43] W. S. McCulloch and W. Pitts: *A logical calculus of the ideas immanent in nervous activity*. Bull. Math. Biol. **5**, 115 (1943).

[Mcn01] M. T. Y. McNamara and et al.: *IEEE Standard Verilog Hardware Description Language*. The Institute of Electrical and Electronics Engineers, Inc. IEEE Std 1364-2001 (2001).

[Mel11] S. Meloni, N. Perra, A. Arenas, S. Gómez, Y. Moreno, and A. Vespignani: *Modeling human mobility responses to the large-scale spreading of infectious diseases*. Sci. Rep. **1** (2011).

[Mes96] T. Mestl, C. Lemay, and L. Glass: *Chaos in high-dimensional neural and gene networks*. Physica D **98**, 33 (1996).

[Mes97] T. Mestl, R. J. Bagley, and L. Glass: *Common Chaos in Arbitrarily Complex Feedback Networks*. Phys. Rev. Lett. **79**, 653 (1997).

[Met49] N. Metropolis and S. Ulam: *The Monte Carlo method*. J. Am. Statist. Assoc. **44**, 335 (1949).

[Mik12] T. Mikami, K. Kanno, K. Aoyama, A. Uchida, T. Ikeguchi, T. Harayama, S. Sunada, K.-i. Arai, K. Yoshimura, and P. Davis: *Estimation of entropy rate in a fast physical random-bit generator using a chaotic semiconductor laser with intrinsic noise*. Phys. Rev. E **85**, 016211 (2012).

[Mil67] S. Milgram: *The small world problem*. Psychol. Today **2**, 60 (1967).

[Milo2] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon: *Network motifs: simple building blocks of complex networks*. Science **298**, 824 (2002).

[Mir90a] R. E. Mirollo and S. H. Strogatz: *Synchronization of Pulse-Coupled Biological Oscillators*. SIAM J. Appl. Math. **50**, 1645 (1990).

[Mono2] J. M. Montoya and R. V. Sole: *Small world patterns in food webs*. J. Theor. Biol. **214**, 405 (2002).

[Moo97] C. Z. Mooney: *Monte Carlo Simulation* (Sage, Thousand Oaks, 1997).

[Muc10] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela: *Community structure in time-dependent, multiscale, and multiplex networks*. Science **328**, 876 (2010).

[Nag62] J. Nagumo, S. Arimoto, and S. Yoshizawa.: *An active pulse transmission line simulating nerve axon.* Proc. IRE **50**, 2061 (1962).

[Nako8] S. Nakamoto: *Bitcoin: A peer-to-peer electronic cash system*. http://s.kwma.kr/pdf/Bitcoin/bitcoin.pdf (2008).

[Naso4] M. P. Nash and A. V. Panfilov: *Electromechanical model of excitable tissue to study reentrant cardiac arrhythmias*. Prog. Biophys. Mol. Biol. **85**, 501 (2004).

[Nep12] T. Nepusz and T. Vicsek: *Controlling edge dynamics in complex networks*. Nature Physics **8**, 568 (2012).

[New01c] M. E. J. Newman: *The structure of scientific collaboration networks*. Proc. Natl. Acad. Sci. U.S.A. **98**, 404 (2001).

[New05] M. E. Newman: *Power laws, Pareto distributions and Zipf's law*. Contemporary physics **46**, 323 (2005).

[New10] M. E. J. Newman: *Networks: An Introduction* (Oxford University Press, New York, 2010).

[Nix11] M. Nixon, M. Friedman, E. Ronen, A. Friesem, N. Davidson, and I. Kanter: *Synchronized Cluster Formation in Coupled Laser Networks*. Phys. Rev. Lett. **106**, 223901 (2011).

[Nix12] M. Nixon, M. Fridman, E. Ronen, A. A. Friesem, N. Davidson, and I. Kanter: *Controlling Synchronization in Large Laser Networks*. Phys. Rev. Lett. **108**, 214101 (2012).

[Nko13] S. Nkomo, M. R. Tinsley, and K. Showalter: *Chimera States in Populations of Nonlocally Coupled Chemical Oscillators*. Phys. Rev. Lett. **110**, 244102 (2013).

[Nor07] J. Norrell, B. Samuelsson, and J. E. S. Socolar: *Attractors in continuous and Boolean networks*. Phys. Rev. E **76**, 046122 (2007).

[Nor09] J. Norrell and J. E. Socolar: *Boolean modeling of collective effects in complex networks*. Phys. Rev. E **79**, 061908 (2009).

[Oht90] T. Ohta, A. Ito, and A. Tetsuka: *Self-organization in an excitable reaction-diffusion system: Synchronization of oscillatory domains in one dimension*. Phys. Rev. A **42**, 3225 (1990).

[Oli11] N. Oliver, M. C. Soriano, D. W. Sukow, and I. Fischer: *Dynamics of a semiconductor laser with polarization-rotated feedback and its utilization for random bit generation*. Opt. Lett. **36**, 4632 (2011).

[Ome10] O. E. Omel'chenko, M. Wolfrum, and Y. L. Maistrenko: *Chimera states as chaotic spatiotemporal patterns*. Phys. Rev. E **81**, 065201 (2010).

[Ome10a] O. E. Omel'chenko, M. Wolfrum, and Y. L. Maistrenko: *Chimera states as chaotic spatiotemporal patterns*. Phys. Rev. E **81**, 065201(R) (2010).

[Ome11] I. Omelchenko, Y. Maistrenko, P. Hövel, and E. Schöll: *Loss of coherence in dynamical networks: spatial chaos and chimera states*. Phys. Rev. Lett. **106**, 234102 (2011).

[Ome12] I. Omelchenko, B. Riemenschneider, P. Hövel, Y. L. Maistrenko, and E. Schöll: *Transition from spatial coherence to incoherence in coupled chaotic systems*. Phys. Rev. E **85**, 026212 (2012).

[Ome12a] O. E. Omel'chenko, M. Wolfrum, S. Yanchuk, Y. L. Maistrenko, and O. Sudakov: *Stationary patterns of coherence and incoherence in two-dimensional arrays of non-locally-coupled phase oscillators*. Phys. Rev. E **85**, 036210 (2012).

[Ome13] I. Omelchenko, O. E. Omel'chenko, P. Hövel, and E. Schöll: *When Nonlocal Coupling Between Oscillators Becomes Stronger: Patched Synchrony or Multichimera States*. Phys. Rev. Lett. **110**, 224101 (2013).

[Opp97] A. V. Oppenheim: *Signals and Systems* (Prentice-Hall, Upper Saddle River, 1997).

[Ott08] E. Ott and T. M. Antonsen: *Low dimensional behavior of large systems of globally coupled oscillators*. Chaos **18**, 037113 (2008).

[Pan12] A. Panchuk, D. P. Rosin, P. Hövel, and E. Schöll: *Synchronization of coupled neural oscillators with heterogeneous delays*. Int. J. Bif. Chaos **23**, 1330039 (2013).

[Pan13] M. J. Panaggio and D. M. Abrams: *Chimera states on a flat torus*. Phys. Rev. Lett. **110**, 094102 (2013).

[Pan14] M. J. Panaggio and D. M. Abrams: *Chimera states: Coexistence of coherence and incoherence in networks of coupled oscillators* (2014). ArXiv:1403.6204.

[Pec90] L. M. Pecora and T. L. Carroll: *Synchronization in chaotic systems*. Phys. Rev. Lett. **64**, 821 (1990).

[Pec98] L. M. Pecora and T. L. Carroll: *Master Stability Functions for Synchronized Coupled Systems*. Phys. Rev. Lett. **80**, 2109 (1998).

[Pei09] M. Peil, M. Jacquot, Y. K. Chembo, L. Larger, and T. Erneux: *Routes to chaos and multiple time scale dynamics in broadband bandpass nonlinear delay electro-optic oscillators*. Phys. Rev. E **79**, 026208 (2009).

[Per14] N. Perlroth: *Experts Find a Door Ajar in an Internet Security Method Thought Safe*. New York Times. http://bits.blogs.nytimes.com/2014/04/08/flaw-found-in-key-method-for-protecting-data-on-the-internet/ (April 8 2014).

[Pet93] V. Petrov, V. Gaspar, J. Masere, and K. Showalter: *Controlling chaos in the Belousov-Zhabotinsky reaction*. Nature **361**, 240 (1993).

[Pik01] A. S. Pikovsky, M. G. Rosenblum, and J. Kurths: *Synchronization, A Universal Concept in Nonlinear Sciences* (Cambridge University Press, Cambridge, 2001).

[Pik08] A. Pikovsky and M. Rosenblum: *Partially Integrable Dynamics of Hierarchical Populations of Coupled Oscillators*. Phys. Rev. Lett. **101**, 264103 (2008).

[Plo00] R. Plonsey and R. C. Barr: *Bioelectricity A Quantitative Approach* (Kluwer Academic, New York, 2000).

[Pol20] B. Van der Pol: *A theory of the amplitude of free and forced triode vibrations*. Radio Rev. **1**, 701 (1920).

[Pol27] B. Van der Pol and J. Van der Mark: *Frequency demultiplication*. Nature **120**, 363 (1927).

[Pol28] B. van der Pol and J. Van der Mark: *LXXII. The heartbeat considered as a relaxation oscillation, and an electrical model of the heart*. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **6**, 763 (1928).

[Pom09] A. Pomerance, E. Ott, M. Girvan, and W. Losert: *The effect of network topology on the stability of discrete state models of genetic control*. Proc. Natl. Acad. Sci. U.S.A. **106**, 8209 (2009).

[Pop11] O. V. Popovych, S. Yanchuk, and P. A. Tass: *Delay- and Coupling-Induced Firing Patterns in Oscillatory Neural Loops*. Phys. Rev. Lett. **107**, 228102 (2011).

[Pre07] W. H. Press: *Numerical recipes 3rd edition: The art of scientific computing* (Cambridge university press, Cambridge, 2007).

[Pas01] R. Pastor-Satorras and A. Vespignani: *Epidemic Spreading in Scale-Free Networks*. Phys. Rev. Lett. **86**, 3200 (2001).

[Ran80] R. H. Rand and P. J. Holmes: *Bifurcation of periodic motions in two weakly coupled van der Pol oscillators*. Int. J. Nonlin. Mech. **15**, 387 (1980).

[Ray96] J. W. S. B. Rayleigh: *The theory of sound*, vol. 2 (Macmillan, London, 1896).

[Rei99] C. Reichhardt and F. Nori: *Phase locking, Devil's staircases, Farey trees, and Armold tongues in driven vortex lattices with periodic pinning*. Phys. Rev. Lett. **82**, 414 (1999).

[Rei09] I. Reidler, Y. Aviad, M. Rosenbluh, and I. Kanter: *Ultrahigh-Speed Random Number Generation Based on a Chaotic Semiconductor Laser*. Phys. Rev. Lett. **103**, 024102 (2009).

[Rib08] A. S. Ribeiro, S. A. Kauffman, J. Lloyd-Price, B. Samuelsson, and J. E. S. Socolar: *Mutual information in random Boolean models of regulatory networks*. Phys. Rev. E **77**, 011901 (2008).

[Rin94] J. L. Ringo, R. W. Doty, S. Demeter, and P. Y. Simard: *Time is of the essence: a conjecture that hemispheric specialization arises from interhemispheric conduction delay*. Cerebr. Cortex **4**, 331 (1994).

[Rin98] J. Rinzel and G. B. Ermentrout: *Analysis of neural excitability and oscillations*. In *Methods in neuronal modeling*, edited by C. Koch and I. Segrev, vol. 2, pp. 251–292 (MIT press, Cambridge, 1998).

[Rob03] S. Robinson: *Still guarding secrets after years of attacks, RSA earns accolades for its founders*. SIAM News **36**, 1 (2003).

[Rod99] E. Rodriguez, N. George, J. P. Lachaux, J. Martinerie, B. Renault, and F. J. Varela: *Perception's shadow: long-distance synchronization of human brain activity*. Nature **397**, 430 (1999).

[Roe97] P. R. Roelfsema, A. K. Engel, P. König, and W. Singer: *Visuomotor integration is associated with zero time-lag synchronization among cortical areas*. Nature **385**, 157 (1997).

[Ron07] D. Rontani, A. Locquet, M. Sciamanna, and D. S. Citrin: *Loss of time-delay signature in the chaotic output of a semiconductor laser with optical feedback*. Opt. Lett. **32**, 2960 (2007).

[Ron09] D. Rontani, M. Sciamanna, A. Locquet, and D. S. Citrin: *Multiplexed encryption using chaotic systems with multiple stochastic-delayed feedbacks*. Phys. Rev. E **80**, 066209 (2009).

[Ron12] D. Rontani, D. P. Rosin, D. J. Gauthier, and E. Schöll: *Autonomous Time-Delayed Boolean Networks using FPGAs*. In *Proc. 2012 Internat. Symposium on Nonlinear Theory and its Applications (NOLTA2012), Palma de Mallorca*, pp. 391–394 (IEICE, Japan, 2012).

[Ros05] E. Rossoni, Y. Chen, M. Ding, and J. Feng: *Stability of synchronous oscillations in a system of Hodgkin-Huxley neurons with delayed diffusive and pulsed coupling*. Phys. Rev. E **71**, 061904 (2005).

[Ros11a] D. P. Rosin, K. E. Callan, D. J. Gauthier, and E. Schöll: *Pulse-train solutions and excitability in an optoelectronic oscillator*. Europhys. Lett. **96**, 34001 (2011).

[Ros12] D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll: *Excitability in autonomous Boolean networks*. Europhys. Lett. **100**, 30003 (2012).

[Ros13] D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll: *Control of synchronization patterns in neural-like Boolean networks*. Phys. Rev. Lett. **110**, 104102 (2013).

[Ros13a] D. P. Rosin, D. Rontani, and D. J. Gauthier: *Ultrafast physical generation of random rumbers using hybrid Boolean networks*. Phys. Rev. E **87**, 040902(R) (2013).

[Ros13b] D. P. Rosin, D. Rontani, D. J. Gauthier, and E. Schöll: *Experiments on autonomous Boolean networks*. Chaos **23**, 025102 (2013).

[Ros14] D. P. Rosin, D. Rontani, and D. J. Gauthier: *Synchronization of coupled Boolean phase oscillators*. Phys. Rev. E **89**, 042907 (2014).

[Ros14a] D. P. Rosin, D. Rontani, N. D. Haynes, E. Schöll, and D. J. Gauthier: *Transient scaling and resurgence of chimera states in coupled Boolean phase oscillators* (2014). Submitted, arXiv:1405.1950.

[Ruk01] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo: *A statistical tests suite for random and pseudorandom number generators for cryptographic applications*. NIST Special Publication 800-22 (2001).

[Rul95] N. F. Rulkov, M. M. Sushchik, L. S. Tsimring, and H. D. I. Abarbanel: *Generalized synchronization of chaos in directionally coupled chaotic systems*. Phys. Rev. E **51**, 980 (1995).

[Sas82] N. Sasaki: *Higher harmonic generation in CMOS/SOS ring oscillators*. IEEE Trans. Electron Dev. **29**, 280 (1982).

[Sch84] H. G. Schuster: *Deterministic Chaos* (Physik-Verlag, Weinheim, 1984).

[Sch06i] G. Schneider and D. Nikolić: *Detection and assessment of near-zero delays in neuronal spiking activity*. J. Neurosci. Methods **152**, 97 (2006).

[Sch07] E. Schöll and H. G. Schuster, eds.: *Handbook of Chaos Control* (Wiley-VCH, Weinheim, 2008). Second completely revised and enlarged edition.

[Sch08k] B. Schrauwen, M. D'Haene, D. Verstraeten, and J. V. Campenhout: *Compact hardware liquid state machines on FPGA for real-time speech recognition*. Neural Networks **21**, 511 (2008).

[Sch08] E. Schöll, G. Hiller, P. Hövel, and M. A. Dahlem: *Time-delayed feedback in neurosystems*. Phil. Trans. R. Soc. A **367**, 1079 (2009).

[Sch11e] S. J. Schiff: *Neural Control Engineering: The Emerging Intersection Between Control Theory and Neuroscience* (MIT Press, Cambridge, 2011).

[Sch13] E. Schöll: *Synchronization in delay-coupled complex networks*. In *Advances in Analysis and Control of Time-Delayed Dynamical Systems*, Ed. by J.-Q. Sun, Q. Ding, chap. 4, pp. 57–83 (World Scientific, Singapore, 2013).

[Sch14a] L. Schmidt, K. Schönleber, K. Krischer, and V. García-Morales: *Coexistence of synchrony and incoherence in oscillatory media under nonlinear global coupling*. Chaos **24**, 013102 (2014).

[Sel10] A. I. Selverston: *Invertebrate central pattern generator circuits*. Phil. Trans. R. Soc. B **365**, 2329 (2010).

[Seno9a] D. V. Senthilkumar, J. Kurths, and M. Lakshmanan: *Inverse synchronizations in coupled time-delay systems with inhibitory coupling*. Chaos **19**, 023107 (2009).

[Shio3a] S. Shipp: *The functional logic of cortico-pulvinar connections*. Phil. Trans. R. Soc. B **358**, 1605 (2003).

[Shmo2] I. Shmulevich, E. R. Dougherty, and W. Zhang: *From Boolean to probabilistic Boolean networks as models of genetic regulatory networks*. Proc. IEEE **90**, 1778 (2002).

[Shmo4] I. Shmulevich and S. Kauffman: *Activities and sensitivities in Boolean network models*. Phys. Rev. Lett. **93**, 048701 (2004).

[Sie14] J. Sieber, O. E. Omel'chenko, and M. Wolfrum: *Controlling Unstable Chaos: Stabilizing Chimera States by Feedback*. Phys. Rev. Lett. **112**, 054102 (2014).

[Sin11] S. Singh: *The code book: the science of secrecy from ancient Egypt to quantum cryptography* (Random House LLC, New York, 2011).

[Sma12] A. G. Smart: *Exotic chimera dynamics glimpsed in experiments*. Phys. Today **65**, 17 (2012).

[Smi35] H. M. Smith: *Synchronous flashing of fireflies*. Science **32**, 151 (1935).

[Sny12] D. Snyder, A. Goudarzi, and C. Teuscher: *Finding Optimal Random Boolean Networks for Reservoir Computing*. In *Artificial Life*, vol. 13, pp. 259–266 (2012).

[Soboo] M. I. Sobhy and A. R. Shehata: *Chaotic radar systems*. In *IEEE MTT-S Int. Microwave Symp. Dig.*, vol. 3, pp. 1701–1704 (IEEE, 2000).

[Soco3] J. E. S. Socolar and S. A. Kauffman: *Scaling in Ordered and Critical Random Boolean Networks*. Phys. Rev. Lett. **90**, 068702 (2003).

[Sol96] R. V. Solé, S. C. Manrubia, B. Luque, J. Delgado, and J. Bascompte: *Phase transitions and complex systems: Simple, nonlinear models capture complex systems at the edge of chaos*. Complexity **1**, 13 (1996).

[Soro7] F. Sorrentino and E. Ott: *Network synchronization of groups*. Phys. Rev. E **76**, 056114 (2007).

[Sor13] M. C. Soriano, J. García-Ojalvo, C. R. Mirasso, and I. Fischer: *Complex photonics: Dynamics and applications of delay-coupled semiconductors lasers*. Rev. Mod. Phys. **85**, 421 (2013).

[Ste93a] O. Steinbock, V. Zykov, and S. C. Müller: *Control of spiral-wave dynamics in active media by periodic modulation of excitability*. Nature **366**, 322 (1993).

[Ste99] P. S. G. Stein, S. Grillner, A. Selverston, and D. G. Stuart: *Neurons, networks, and motor behavior* (MIT press, Cambridge, 1999).

[Sto01] T. Stojanovski, J. Pihl, and L. Kocarev: *Chaos-based random number generators. Part II: practical realization*. IEEE Trans. Circuits Syst. I **48**, 382 (2001).

[Str93] S. H. Strogatz and I. Stewart: *Coupled oscillators and biological synchronization*. Sci. Am. **269**, 102 (1993).

[Str94a] S. H. Strogatz: *Nonlinear Dynamics and Chaos* (Westview Press, Cambridge, 1994).

[Str00] S. H. Strogatz: *From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators*. Physica D **143**, 1 (2000).

[Str05a] S. H. Strogatz, D. Abraham, A. D. McRobbie, B. Eckhardt, and E. Ott: *Crowd synchrony on the Millennium Bridge*. Nature **438**, 43 (2005).

[Sun07] B. Sunar, W. J. Martin, and D. R. Stinson: *A provably secure true random number generator with built-in tolerance to active attacks*. IEEE Trans. Comp. **56**, 109 (2007).

[Sun13] M. Sun, X. Cheng, and J. E. S. Socolar: *Causal structure of oscillations in gene regulatory networks: Boolean analysis of ordinary differential equation attractors*. Chaos **23**, 025104 (2013).

[Sun13a] J. Q. Sun and G. Ding: *Advances in Analysis and Control of Time-Delayed Dynamical Systems* (World Scientific, Singapore, 2013).

[Tal10] N. N. Taleb: *The Black Swan:: The Impact of the Highly Improbable Fragility* (Random House LLC, New York, 2010).

[Tel08] T. Tél and Y.-C. Lai: *Chaotic transients in spatially extended systems*. Phys. Rep. **460**, 245 (2008).

[Tin12] M. R. Tinsley, S. Nkomo, and K. Showalter: *Chimera and phase cluster states in populations of coupled chemical oscillators*. Nature Physics **8**, 662 (2012).

[Uch08] A. Uchida, K. Amano, M. Inoue, K. Hirano, S. Naito, H. Someya, I. Oowada, T. Kurashige, M. Shiki, S. Yoshimori, K. Yoshimura, and P. Davis: *Fast physical random bit generation with chaotic semiconductor lasers*. Nature Photonics **2**, 728 (2008).

[Ued05] H. R. Ueda, S. Hayashi, W. Chen, M. Sano, M. Machida, Y. Shigeyoshi, M. Iino, and S. Hashimoto: *System-level identification of transcriptional circuits underlying mammalian circadian clocks*. Nature Genetics **37**, 187 (2005).

[Uhl06] P. J. Uhlhaas and W. Singer: *Neural Synchrony in Brain Disorders: Relevance for Cognitive Dysfunctions and Pathophysiology*. Neuron **52**, 155 (2006).

[Ujj13] S. R. Ujjwal and R. Ramaswamy: *Chimeras with multiple coherent regions*. Phys. Rev. E **88**, 032902 (2013).

[Van98a] G. D. VanWiggeren and R. Roy: *Communication with chaotic lasers*. Science **279**, 1198 (1998).

[Var01] F. Varela, J. P. Lachaux, E. Rodriguez, and J. Martinerie: *The brainweb: Phase synchronization and large-scale integration*. Nat. Rev. Neurosci. **2**, 229 (2001).

[Var12] R. Vardi, A. Wallach, E. Kopelowitz, M. Abeles, S. Marom, and I. Kanter: *Synthetic reverberating activity patterns embedded in networks of cortical neurons*. Europhys. Lett. **97**, 066002 (2012).

[Var12a] R. Vardi, R. Timor, S. Marom, M. Abeles, and I. Kanter: *Synchronization with mismatched synaptic delays: A unique role of elastic neuronal latency*. Europhys. Lett. **100**, 48003 (2012).

[Ved06] V. Vedral and J. Friedman: *Introduction to quantum information science*, vol. 1 (Oxford University Press, Oxford, 2006).

[Vic08] R. Vicente, L. L. Gollo, C. R. Mirasso, I. Fischer, and P. Gordon: *Dynamical relaying can yield zero time lag neuronal synchrony despite long conduction delays*. Proc. Natl. Acad. Sci. U.S.A. **105**, 17157 (2008).

[Vla94] Vladimirescu: *The SPICE book* (John Wiley & Sons, Inc., Hoboken, 1994).

[Neu66] J. Von Neumann and A. W. Burks: *Theory of self-reproducing automata* (University of Illinois Press, Urbana, 1966).

[Wac95b] A. Wacker, S. Bose, and E. Schöll: *Transient spatio-temporal chaos in a reaction-diffusion model*. Europhys. Lett. **31**, 257 (1995).

[Wal65] C. C. Walker and A. W. R.: *On Temporal Characteristics of Behavior in Certain Complex Systems*. Kybernetik **3**, 100 (1965).

[Wal03] H.-O. Walther: *The solution manifold and< i> C</i>< sup> 1</sup>-smoothness for differential equations with state-dependent delay*. J. Diff. Eq. **195**, 46 (2003).

[Wat98] D. J. Watts and S. H. Strogatz: *Collective dynamics of 'small-world' networks*. Nature **393**, 440 (1998).

[Way10] M. A. Wayne and P. G. Kwiat: *Low-bias high-speed quantum random number generator via shaped optical pulses*. Opt. Express **18**, 9351 (2010).

[Wei14] L. Weicker, T. Erneux, L. Keuninckx, and J. Danckaert: *Analytical and experimental study of two delay-coupled excitable units*. Phys. Rev. E **89**, 012908 (2014).

[Wer97] T. Werner and V. Akella: *Asynchronous processor survey*. Computer **30**, 67 (1997).

[Weso0] N. H. E. Weste and K. Eshraghian: *Principles of CMOS VLSI Design: A Systems Perspective* (Addison Wesley, Boston, 2000), 2nd edn. Chapter 4.4.

[Whi57] J. S. White: *A t-test for the serial correlation coefficient*. Ann. Math. Stat. **28**, 1046 (1957).

[Wic13] M. Wickramasinghe and I. Z. Kiss: *Spatially organized dynamical states in chemical oscillator networks: Synchronization, dynamical differentiation, and chimera patterns*. PloS one **8**, e80586 (2013).

[Wij95] M. C. Wijffels, C. J. Kirchhof, R. Dorland, and M. A. Allessie: *Atrial fibrillation begets atrial fibrillation A study in awake chronically instrumented goats*. Circulation **92**, 1954 (1995).

[Wil00] R. J. Williams and N. D. Martinez: *Simple rules yield complex food webs*. Nature **404**, 180 (2000).

[Wil13] C. R. S. Williams, T. E. Murphy, R. Roy, F. Sorrentino, T. Dahms, and E. Schöll: *Experimental Observations of Group Synchrony in a System of Chaotic Optoelectronic Oscillators*. Phys. Rev. Lett. **110**, 064104 (2013).

[Win67] A. T. Winfree: *Biological rhythms and the behavior of populations of coupled oscillators*. J. Theor. Biol. **16**, 15 (1967).

[Win80] A. T. Winfree: *The Geometry of Biological Time* (Springer, New York, 1980).

[Woh95] T. M. H. Wohlleben and A. J. Weaver: *Interdecadal climate variability in the subpolar North Atlantic*. Climate Dyn. **11**, 459 (1995).

[Wol83] S. Wolfram: *Statistical mechanics of cellular automata*. Rev. Mod. Phys. **55**, 601 (1983).

[Wol09] K. Wold and C. H. Tan: *Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings*. Int. J. Reconf. Comp. **2009**, 501672 (2009).

[Wol11] M. Wolfrum and O. E. Omel'chenko: *Chimera states are chaotic transients*. Phys. Rev. E **84**, 015201 (2011).

[Wol11a] M. Wolfrum, O. Omel'chenko, S. Yanchuk, and Y. Maistrenko: *Spectral properties of chimera states*. Chaos **21**, 013112 (2011).

[Yao96] X. S. Yao and L. Maleki: *Optoelectronic oscillator for photonic systems*. IEEE J. Quantum Electron. **32**, 1141 (1996).

[Zak14] A. Zakharova, M. Kapeller, and E. Schöll: *Chimera Death: Symmetry Breaking in Dynamical Networks*. Phys. Rev. Lett. **112**, 154101 (2014).

[Zal03] I. Zaliapin, V. Keilis-Borok, and M. Ghil: *A Boolean delay equation model of colliding cascades. Part I: Multiple seismic regimes*. J. Stat. Phys. **111**, 815 (2003).

[Zal03a] I. Zaliapin, V. Keilis-Borok, and M. Ghil: *A Boolean delay equation model of colliding cascades. Part II: Prediction of critical transitions*. J. Stat. Phys. **111**, 839 (2003).

[Zhao9a] R. Zhang, H. L. D. de S. Cavalcante, Z. Gao, D. J. Gauthier, J. E. S. Socolar, M. M. Adams, and D. P. Lathrop: *Boolean Chaos*. Phys. Rev. E **80**, 045202 (2009).

[Zil09] R. Zillmer, N. Brunel, and D. Hansel: *Very long transients, irregular firing, and chaotic dynamics in networks of randomly connected inhibitory integrate-and-fire neurons*. Phys. Rev. E **79**, 031909 (2009).

ACKNOWLEDGMENTS

I thank my supervisors Prof. Daniel J. Gauthier and Prof. Eckehard Schöll, who have been excellent mentors and supported me greatly during my doctoral studies. Prof. Daniel J. Gauthier always pointed me towards the significant topics and stressed the importance of rigor with both experimental and theoretical studies. Prof. Eckehard Schöll encouraged me since I started studying physics after highschool; with his immense knowledge in theoretical physics, he guided me towards developing the theoretical backbone of this thesis.

I am indebted to Prof. Damien Rontani, who as a then postdoctoral researcher accompanied me on the endeavor of realizing autonomous logic circuits on FPGAs and was involved with most of the research in this thesis. He also supported me as a third advisor.

I acknowledge helpful conversations at conferences and laboratory visits with Prof. Leon Glass, Dr. Philipp Hövel, Prof. Raj Roy, Prof. Ed Ott, Prof. Ido Kanter and Prof. Yang-Yu Liu.

I want to thank the present and former members of the research groups of Prof. Daniel Gauthier and Prof. Eckehard Schöll for the great working atmosphere. In particular, I thank Dr. Seth Cohen, Dr. Kristine Callan, Hannah E. Guilbert, Bonnie L. Schmittberger, Margaret E. Shea, Nicholas D. Haynes, Judith Lehnert, Dr. Kathy Lüdge and Andrew Keane for interesting discussions.

Lastly, I thank my parents, step parents, and my brother for their support and encouragement.

This work was supported by the U.S. Army Research Office Grants W911NF-11-1-0451, W911NF-12-1-0099, W911NF-12-1-0424, U.S. Office of Naval Research Grant N00014-07-1-0734, and the DFG in the framework of SFB 910.