TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY
Julkaisu 852 • Publication 852

Ismo Hänninen

Computer Arithmetic on Quantum-Dot Cellular Automata Nanotechnology

Tampere 2009

Tampereen teknillinen yliopisto. Julkaisu 852
Tampere University of Technology. Publication 852

Ismo Hänninen

Computer Arithmetic on Quantum-Dot Cellular Automata Nanotechnology

Thesis for the degree of Doctor of Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB222, at Tampere University of Technology, on the 7th of December 2009, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2009

ISBN 978-952-15-2269-7 (printed)
ISBN 978-952-15-2289-5 (PDF)
ISSN 1459-2045

ABSTRACT

The traditional digital technologies are reaching their performance limits, and the desired growth of computing power can be continued only by adopting emerging circuit technologies and new design approaches into use. This thesis provides a practical view on the emerging quantum-dot cellular automata (QCA) nanotechnology, presenting techniques to construct high-density, performance optimized, and noise tolerant basic arithmetic circuits. Several novel arithmetic units are proposed, described at the logic, the pipeline, and the layout level, and verified using quantum mechanical simulation.

Design analysis shows, that on the self-latching QCA nanotechnology, the basic serial and parallel arithmetic structures for addition, or multiplication, typically have comparable latency, but the throughput follows the degree of parallelism. The circuit area is dominated by the passive wiring overhead, characterized by a square-law dependency on operand word length. Hierarchical probabilistic analysis shows, that bit-stage level macro component reliability has about linear effect on the total reliability, while component types affect whole with weights determined by the word length, and the wiring dominates also the reliability. The power dissipation is analyzed near the ultimate limit of computation efficiency, using the Landauer's principle, showing that irreversible information erasures consume significant power on molecular QCA, severely limiting the operating frequency.

The studies in this thesis show, that QCA systems have to address emerging technology characteristic, that have not had much impact in traditional engineering work. Design optimization has to be started from the reliability and power challenges, which will determine the feasibility of any planned system.

PREFACE

The work presented in this thesis has been carried out at the Department of Computer Systems at Tampere University of Technology, Tampere, Finland, where the author held an assistant's position during the years 2007–2009.

I would like to express my deepest gratitude to my supervisor Prof. Jarmo Takala for his guidance and encouragement during the nano-arithmetic research, which was at the time a one-man undertaking apart from that. Grateful acknowledgments go also to Prof. Peeter Ellervee and Prof. Sorin Cotofana for their constructive reviews, improving the manuscript of the thesis.

Sincere thanks to my esteemed teaching colleagues, Prof. Marko Hännikäinen, Prof. Timo D. Hämäläinen, Jarno Vanne, M.Sc., and Erno Salminen, M.Sc. Together we have seen to the education of future computer engineers, giving them the basic tools for project management, scientific writing, and computer arithmetic. I would also like to recognize all the other fellows at the department for inspiring conversations on a multitude of topics and a friendly environment, and the staff who helped with all the practical matters, Mrs. Johanna Reponen, Ms. Irmeli Lehto, and Timo Rintakoski, M.Sc.

Finally, I am obliged to my mother Marjatta Hänninen, and parents-in-law Eliisa and Mauri Rintanen for their continuous support to our family during this endeavor. My heartfelt gratitude to my dear wife Susa and sons Konsta and Otso, for enduring the absence of the husband and the father during the many conference trips and the frequent inattention at other times. Susa and the little bears, thanks to you, and praises to the Heavenly Father for you!

Orivesi, November 2009
Ismo Hänninen

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AM	Array Multiplier
AOI	And-Or-Inverter
ATPG	Automatic Test Pattern Generation
CMOS	Complementary Metal Oxide Semiconductor
CSA	Carry-Save Adder
EQCA	Extended Quantum-Dot Cellular Automata
FA	Full Adder
FFT	Fast Fourier Transform
GALS	Globally Asynchronous, Locally Synchronous
HA	Half Adder
IC	Integrated Circuit
IEEE	The Institute of Electrical and Electronics Engineers
ITM	Ideal Transfer Matrix
ITRS	International Technology Roadmap for Semiconductors
LSB	Least Significant Bit
MAC	Multiply-Accumulate
MSB	Most Significant Bit
PIP	Propagated Instruction Processor
PLA	Programmable Logic Array

PTM	Probabilistic Transfer Matrix
RAM	Random Access Memory
RCA	Ripple Carry Adder
RMQDA	Restricted Minima Quantum-Dot Array
SA	Serial Adder
SBSA	Serial Bit-Stream Analyzer
SCQCA	Split Current Quantum-Dot Cellular Automata
SD	Signed-Digit
SFA	Summand and Full Adder Block
SIMD	Single Instruction, Multiple Data
TSC	Totally Self-Checking
ULP	Unit in the Last Position, the weight of LSB bit
Verilog	Verilog Hardware Description Language
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VLSI	Very Large Scale Integration
QCA	Quantum-Dot Cellular Automata

LIST OF SYMBOLS

Quantum-Dot Cellular Automata

’0’	bit with value zero
’1’	bit with value one
in	input binary signal
in_i	ith input binary signal
out	output binary signal
out_i	ith output binary signal

Binary Adders

a	input operand bit
b	input operand bit
c	input operand bit
c_{in}	input carry bit
c_{out}	output carry bit
s	output sum bit
$M(a,b,c)$	three-input majority function defined as $M(a,b,c) = (a \wedge b) \vee (b \wedge c) \vee (a \wedge c)$
$\wedge$	logical AND operation
$\vee$	logical OR operation

n	input operand word length (bits)
a_i	ith bit of the input operand word (0 is LSB)
b_j	jth bit of the input operand word (0 is LSB)
s_k	kth bit of the sum output word (0 is LSB)
A(3:0)	4-bit input operand word
B(3:0)	4-bit input operand word
S(4:0)	5-bit output sum word

Binary Multipliers

n	input operand word length (bits)
a_i	ith bit of the first input operand word (0 is LSB)
b_j	jth bit of the second input operand word (0 is LSB)
A	first input operand word (multiplicand)
A_i	first input operand word, on index i of a data word set
B	second input operand word (multiplier)
B_j	second input operand word, on index j of a data word set
L	latency of the complete result word (clock cycles)
L_{LSB}	latency of the least significant result bit (clock cycles)
L_{MSB}	latency of the most significant result bit (clock cycles)

Serial-Parallel Multiplier

M	output multiplication result word
M_k	output multiplication result word, on index k of a data word set
$M_{k,i}$	ith bit of the output multiplication result word, on index k of a data word set

$M_{k,LSB}$	LSB bit of the output multiplication result word, on index k of a data word set
$M_{k,MSB}$	MSB bit of the output multiplication result word, on index k of a data word set
m_k	kth bit of the multiplication result output word (0 is LSB)
s_j	summand bit, cell index j
sum_j	output sum bit, cell index j
$carry_j$	output carry bit, cell index j

Array Multiplier

M	output multiplication result word
M_k	output multiplication result word, on index k of a data word set
m_k	kth bit of the multiplication result word (0 is LSB)
$s_{i,j}$	summand bit, cell on column i and row j
$sum_{i,j}$	output sum bit, cell on column i and row j
$carry_{i,j}$	output carry bit, cell on column i and row j

Radix-4 Multiplier

n_A	multiplicand operand word A length (bits)
n_B	multiplier operand word B length (bits)
n_P	output result word P length (bits)
P	output multiplication result word
M	internal multiple word
S	internal sum vector
C	internal carry vector
$+A$	positive multiple word (the multiplicand)

$+2A$	doubled multiple word
$-A$	negated multiple word
$-2A$	doubled negated multiple word
p_i	ith bit of the multiplication result output word (0 is LSB)
m_i	ith bit of the internal multiple word (0 is LSB)
s_j	jth bit of the sum vector (0 is LSB)
c_k	kth bit of the carry vector (0 is LSB)
$a_i^{(2)}$	ith bit of the doubled multiple word (0 is LSB)
$a_i^{(M)}$	ith bit of the negated multiple word (0 is LSB)
$a_i^{(2M)}$	ith bit of the doubled negated multiple word (0 is LSB)
c_{inter}	internal inter-digit carry bit
c_{intra}	internal intra-digit carry bit
$>$	1-bit arithmetic shift to right
$>>$	2-bit arithmetic shift to right
sel_{double}	mux control signal for selecting doubled multiple (active high)
sel_{negate}	mux control signal for selecting negated multiple (active high)
sel_{null}	mux control signal for selecting zero multiple (active high)
sel_{-A}	internal control signal for selecting negated multiple (active high)
sel_{-2A}	internal control signal for selecting doubled negated multiple (active high)
sel_{+2A}	internal control signal for selecting doubled multiple (active high)
L_{total}	latency of the complete result word (clock cycles)

Reliability Analysis

n	input operand word length (bits)
a_i	ith bit of the input operand word (0 is LSB)
b_j	jth bit of the input operand word (0 is LSB)
L_i	ith row parallel component level
$\otimes$	Kronecker (alternatively tensor) product
$\star$	element-wise matrix multiplication
P_i	PTM of the parallel components on row i
v	vector representing the probabilities of each input case
R	total reliability, confidence of no failures

Ripple Carry Adder

s_k	kth bit of the sum output word (0 is LSB)
c_{out}	output carry bit
P_{RCA}	PTM of the complete ripple carry adder
$P_{RCA,ideal}$	ITM of the complete ripple carry adder
P_{FA}	PTM of a full adder
P_{IW}	PTM of an input wire block
P_{OW}	PTM of an output wire block
p	uniform signal error probability
p_{FA}	uniform full adder component error probability
p_W	uniform wire component error probability
a	full adder contribution in the reliability approximation
b	wire block contribution in the reliability approximation

Array Multiplier

$s_{i,j}$	summand bit, cell on column i and row j
$sum_{i,j}$	output sum bit, cell on column i and row j
$carry_{i,j}$	output carry bit, cell on column i and row j
m_k	kth bit of the multiplication result word (0 is LSB)
P_{AM}	PTM of the complete array multiplier
$P_{AM,ideal}$	ITM of the complete array multiplier
P_{SFA}	PTM of a summand and full adder block
P_W	PTM of a wire block
P_F	PTM of a fanout block
P_X	PTM of a wire crossing block
P_I	PTM of an ideal wire
$P_{C,TR}$	PTM of the cell at top-right corner
$P_{C,T}$	PTM of the cells in the middle of top row
$P_{C,TL}$	PTM of the cell at top-left corner
$P_{C,R}$	PTM of the cells in the middle of rightmost column
$P_{C,G}$	PTM of the general cells in the middle
$P_{C,L}$	PTM of the cells in the middle of leftmost column
$P_{C,BR}$	PTM of the cell at bottom-right corner
$P_{C,B}$	PTM of the cells in the middle of bottom row
$P_{C,BL}$	PTM of the cell at bottom-left corner
p_A	uniform active logic error probability
p_W	uniform wire component error probability

Power Analysis

n	input operand word length (bits)
E_{sig}	logical signal minimum energy content
E_{dis}	energy loss due to single bit erasure
k_B	the Boltzmann constant, 1.3807×10^{-23} J/K
T	temperature in Kelvin degrees

LIST OF PUBLICATIONS

This thesis is a monograph, which contains some unpublished material, but is mainly based on the work already published in the following peer-reviewed international publications. In the text, these publications are referred to as [P1], [P2],..., [P8].

[P1] I. Hänninen and J. Takala, "Robust adders based on quantum-dot cellular automata," in *Proceedings of the IEEE International Conference on Application-specific Systems, Architectures and Processors*, Montréal, QC, Canada, Jul. 8-11, 2007, pp. 391–396.

[P2] I. Hänninen and J. Takala, "Pipelined array multiplier based on quantum-dot cellular automata," in *Proceedings of the European Conference on Circuit Theory and Design*, Seville, Spain, Aug. 26-30, 2007, pp. 938-941.

[P3] I. Hänninen and J. Takala, "Binary multipliers on quantum-dot cellular automata," *Facta Universitatis*, vol. 20, no. 3, pp. 541-560, Dec. 2007. [Online]. Available: http://factaee.elfak.ni.ac.yu/fu2k73/15hanninen.html

[P4] I. Hänninen and J. Takala, "Reliability of n-bit nanotechnology adder," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, Montpellier, France, Apr. 7-9, 2008, pp. 34-39.

[P5] I. Hänninen and J. Takala, "Arithmetic design on quantum-dot cellular automata nanotechnology," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, ser. Lecture Notes in Computer Science, M. Bereković, N. Dimopoulos, and S. Wong, Eds. Berlin/Heidelberg, Germany: Springer, 2008, vol. 5114, pp. 43-52.

[P6] I. Hänninen and J. Takala, ”Reliability of a QCA array multiplier,” in *Proceedings of the IEEE Conference on Nanotechnology*, Arlington, TX, USA, Aug. 18-21, 2008, pp. 315–318.

[P7] I. Hänninen and J. Takala, ”Binary adders on quantum-dot cellular automata,” to appear in *Journal of Signal Processing Systems* (Springer, New York, NY, USA). [Online]. Available: http://dx.doi.org/10.1007/s11265-008-0284-5

[P8] I. Hänninen and J. Takala, ”Radix-4 recoded multiplier on quantum-dot cellular automata,” in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, ser. Lecture Notes in Computer Science, K. Bertels, N. Dimopoulos, C. Silvano, and S. Wong, Eds. Berlin/Heidelberg, Germany: Springer, 2009, vol. 5657, pp. 118–127.

1. INTRODUCTION

The computing power offered by the integrated circuit (IC) technology has grown exponentially for nearly five decades, following the *Moore's Law* (original 1965 [1], updated later [2]): the number of primitive components per chip doubles every eighteen to twenty-four months, consequently doubling also the number of the available memory bits and computation operations per time unit. This trend, broken down in detail in the International Technology Roadmap for Semiconductors (ITRS) reports [3], has enabled the modern-day world and plays a prominent role in the development of all fields of science and engineering. However, the progress of IC industry is in danger of halting soon, if only traditional computer hardware technologies are utilized.

1.1 The Necessary Technology Transition

In about a decade, the traditional digital circuit technologies are reaching their practical and theoretical limits, as the beneficial continuous downscaling of electronics becomes more challenging. Most technologies, like the nearly everywhere present complementary metal oxide semiconductor (CMOS), use transistors as current switches, representing binary information as currents and voltages. However, these primitive devices have several problems when they get really small: the on/off levels become inadequate, the leakage currents significant, the resistance high, the charge quantized, and the wires very large in comparison with the active devices. [3]

In the long run, the most severe problem is the heat generation, as the circuit capacitances are charged to a potential and again discharged to ground, usu-

ally wasting nearly all of the energy contained in the logic signal. This is already a problem with the present technologies, but if molecular device densities are reached, the problem becomes truly unmanageable: on the operational frequencies of hundreds of gigahertz, with each transistor moving only *one* electron across one volt potential, the power densities reach the *megawatt* per square centimeter region. Careful adiabatic charging can lower the power dissipation, but not enough to enable true molecular electronics. [4]

Several *emerging technologies* have been proposed to replace the current semiconductor transistor approach. Quantum-dot cellular automata nanotechnology (QCA) is one of the foremost candidates, offering robust ways to reach circuit densities 10^{11} to 10^{12} devices/cm^2 and clock frequencies several orders of magnitude higher than the expected technological peak of the CMOS. The concept was introduced in the early 1990s [5, 6] and has already been demonstrated in laboratory environment with small proof-of-concept systems [7–9], but adopting QCA into general use requires still considerable advances both in the design methodology and the manufacturing processes.

1.2 Bringing the Emerging Technologies Into Use

There is a definite need for pressing the emerging technologies into service, but this is very challenging, due to novel characteristics complicating both the digital design process and the actual physical manufacturing. Early research into QCA circuits and systems has demonstrated that these two levels of engineering work are much more tightly coupled than on the traditional technologies. The digital designer has to aim at an optimized end product while being well aware of the underlying implementation technology.

Design optimization can be characterized with several comparable metrics of performance and cost, which are affected by the chosen computation algorithm, hardware structure, and logical components. The relationship between the designer choices and the quality of the results is governed by novel principles, on the emerging technologies. For example, on QCA a logic signal is

propagated by the copy-operation from a cell automaton to another, and with high operating frequencies, the automata wires are relatively long in comparison with the distance that the signal can propagate during a clock cycle. Thus, the length of a wire translates directly into significant delay and distinct number of pipeline stages, which has tremendous effect on which types of arithmetic units are practical. Another important issue is the underlying imperfect cellular interaction, which also has to be compensated by applying sub-gate level pipelining.

The emerging technologies are inherently unreliable, both the manufacturing processes and the runtime operation of a chip inside a computer system. Circuit primitives will deviate from the traditional, practically deterministic behavior and operate with stochastic characteristics, raising general *design-for-reliability* to top concern. Improvements can be aimed at various design levels, typically introducing redundancy or reconfigurability into the system, with a significant cost increase. The unreliable physical layer has to be taken into account from the start of the design work, or getting a complete system to run might turn out to be so expensive, that the gains of applying new technology would be totally lost. Fault-tolerant design requires profound understanding about the relationship between physical implementation and design abstractions, and this understanding can be found with the aid of reliability analysis techniques.

Power consumption and the resulting heat dissipation already set the performance limit of the traditional digital circuits, but this is an even more dominating cost factor for the emerging technologies: there is room for a tremendous number of devices in the nanoworld, having possibly molecular device densities. The combined heat generation must not exceed what we can cool off, and as each device is allowed to dissipate less and less, we possibly need a paradigm shift from irreversible computing to reversible computing. The already present *design-for-power* trend reaches unprecedented weight in the industry, when the reversibility aspects have to be addressed.

1.3 Objectives and Research Statements

The objective of this thesis is to find more efficient methods of designing and analyzing arithmetic circuits on the QCA technology, and solutions to the following problem statements are presented:

How to design cost-efficient basic arithmetic circuits, accounting for the imperfect cellular interaction inherent to QCA? The arithmetic circuits on QCA have been studied only very little, leaving the analysis of the traditional design metrics of performance and complexity superficial. Especially, the technology-inherent noise coupling interaction has not been taken into account, or has been avoided with an unjustified performance penalty, preventing the use of the resulting basic blocks to construct multi-bit arithmetic [10]. Several standard arithmetic structures have not been adapted to QCA, at all.

Which factors contribute to the measurable design characteristics on QCA? The effects of algorithmic, structural, and component choices on performance and cost metrics have previously been analyzed only very little, especially the gains vs. the costs of parallelism in the computation [11–14]. Neither the contribution of active and passive circuitry, nor the role of the operand word length, has been established.

How reliable should the QCA primitive devices be, to enable the construction of large arithmetic units? The existing designs have not been adequately analyzed, to find out where the costly reliability improvements could be most beneficially aimed at [15]. Especially, macroblock level analysis, needed to hierarchically model large arithmetic units, has not been conducted before, and it has not been established how the underlying device failure rate requirements scale with the operand word length. There has been very little work on the dependencies of architectural decisions and reliability, on QCA.

What is the role of irreversible power dissipation in QCA arithmetic? The dissipation characteristics of large designs, including macrocomponent level contributions, have not been analyzed although some work has been conducted on the primitive device level power and energy dynamics [16, 17].

Especially, operating frequency limitations of complete arithmetic units on molecular QCA have not been determined and the major limiting factor, information erasure dissipation (irreversibility), has not been identified before. It has not been clear, whether the costly reversible computing paradigm is necessary, for reaching maximum performance on the technology.

1.4 Main Contributions

This thesis provides a practical view on digital design work on QCA, using several case studies to illustrate the various aspects of constructing highly parallel (in a general sense, including several co-existing data sets in a pipeline) hardware structures with this computing paradigm, offering potential nanotechnology implementations. The studied arithmetic units are sufficiently massive to reveal the fundamental characteristics, while retaining enough structural regularity to enable modeling and simulation to some extent. In short, the main contributions are described in the following:

Efficient arithmetic designs for QCA. The presented techniques allow the construction of high-density, performance optimized, and noise tolerant basic arithmetic circuits on QCA. Several novel arithmetic units are designed, utilizing the inherent characteristics of the technology, and aiming at modularity and customization to varying operand word lengths. The designs are described at the logic, the pipeline, and the layout level, and verified with quantum mechanical simulation.

- Novel binary adder units for QCA technology:
  - Robust full adder, with minimized area using only one QCA fabrication layer.
  - Robust serial adder, with minimized area and latency, and maximum throughput.
  - Pipelined ripple carry adder, with noise robustness, minimized area and latency, and maximum throughput.

- Novel binary multiplier units for QCA technology:
  - Serial-parallel multiplier, with noise robustness.
  - Pipelined array multiplier, with noise robustness and highest performance-area efficiency.
  - Radix-4 multiplier, with noise robustness and customizable degree of parallelism (the most flexible algorithm on QCA, thus far).

Design analysis factors on QCA arithmetic. The analysis of the presented arithmetic designs and cases reported in the literature leads to the following conclusions, on the relationship between the degree of parallelism and the traditional design metrics, on pipelined QCA:

- The basic serial and pipelined arithmetic structures have equal latency, but the throughput differs, following the degree of general parallelism.
- Passive wiring overhead typically dominates the circuit area, with a square-law dependency on the operand word length.

Reliability analysis and improvement on QCA arithmetic. The reliability levels of complete arithmetic units are established via probabilistic analysis, hierarchically constructing the total failure rate from the failure rates of the underlying components. For the important cases of the pipelined ripple carry adder and the pipelined array multiplier, the following conclusions are drawn:

- Bit-stage level macro component (a full adder, for example) reliability has about linear effect on the total reliability.
- Component types affect the total reliability with different weights, depending on the operand word length.
- Passive wiring overhead dominates also the reliability.

- Large operand length arithmetic on the failure-rich tecnologies is not feasible, unless a multi-level redundancy scheme is developed for tolerating very high primitive component failure rates. (However, complete fault-tolerant design methodology is out of the scope of the thesis.)

Electrical power analysis on QCA arithmetic. The power dissipation of complete arithmetic units is analyzed near the ultimate limit of computation efficiency, using the Landauer's principle [18]. Based on the pipelined ripple carry adder, the pipelined array multiplier, and the serial-parallel multiplier, the following conclusions are drawn for molecular QCA:

- Irreversible information erasures consume significant power, as opposed to the situation on traditional technologies.
- Clock frequencies of the designs are limited much lower than the expected switching speeds of the primitive devices.
- Reaching the full technology potential requires reversible computing principles, to be incorporated into the designs.

All the achieved results, modeling, designing and optimization work presented as the contribution of this thesis were developed by the author. The work was reported earlier in eight publications [P1]– [P8] and the author was the main author in all of them. Consequently, some chapters contain verbatim extracts from those papers while the copyrights of the extracts are retained by the respective copyright holders.

1.5 Thesis Outline

An introduction to the quantum-dot cellular automata computing paradigm is given in Chapter 2, describing the basic circuit constructs, interconnects, signal dynamics, clocking approaches, physical implementation variants, available modeling approaches, and digital design efforts. Next, novel arithmetic

units, adders and multipliers, are proposed at the logic, the pipeline, and the QCA layout level, including performance and cost comparison with previous design proposals, in Chapters 3 and 4, respectively.

Chapter 5 presents a reliability study based on probabilistic transfer matrices, which are used to formulate the component failure rates of two of the proposed arithmetic designs, the pipelined ripple carry adder and the array multiplier. Chapter 6 continues with a power analysis at the fundamental Landauer's limit, determining the absolute minimum power dissipation that the laws of the nature allow for the QCA ripple carry adder, serial-parallel multiplier, and array multiplier. Finally, Chapter 7 concludes the thesis.

2. QUANTUM-DOT CELLULAR AUTOMATA

The quantum-dot cellular automata (QCA) concept is very intuitive: we have bistable cellular automata, which are operated under clocked control. There are various ways to construct the physical cells and apply the clocking, but the implementation technologies are still under development. As a result, this treatment stays mostly at an abstract level without the physical details. A survey on the history of the development of the general cellular automata concept can be found in [19].

Chapter Contents. The general principles of constructing the circuit primitives are introduced in Sec. 2.1 and Sec. 2.2, the signal dynamics behavior described in Sec. 2.3, and the clocking approaches enabling inherent pipelining summarized in Sec. 2.4. This account is valid for all QCA implementations based on the *electrostatic interaction*, which is believed to offer the best performance, in comparison with other possible approaches (which utilize the magnetic coupling). For completeness, the QCA physical implementation variants are introduced in Sec. 2.5, and the simulation models described briefly in Sec. 2.6. The conclusion of the chapter follows in Sec. 2.7 with a survey into general digital design for QCA.

2.1 QCA Basics

The information storage and transport on electrostatic quantum-dot cellular automata [6] is based on the local position of charged particles inside a small section of the circuit, called a cellular automaton (which is essentially a structured charge container), and there is no electrical particle current in the circuit

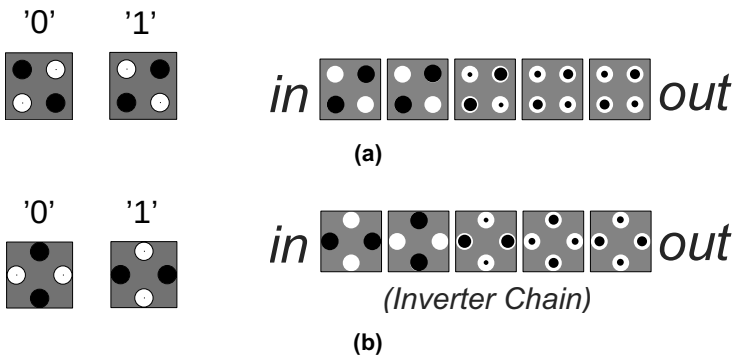


Fig. 1. *QCA cell polarizations and wires: a) type 1 cell, direct non-inverting wire, and b) type 2 cell, inverter chain wire. The four quantum-dots of a cell function as localization centers for electrons: white dot denotes empty center, black dot denotes a center with an electron, and smaller black dots denote smaller localization likelihood during switching. These binary cells are the standard approach, but cells with other dot arrangements can be constructed.*

at all. The QCA cell has a limited number of quantum-dots, which the particles can occupy, and these dots are arranged such that the cell can have only two polarizations (two degenerate quantum mechanical ground states), representing binary value zero or one. A cell can switch between the two states by letting the charged particles tunnel between the dots quantum mechanically.

The cells exchange information by classical Coulombic interaction. An input cell forced to a polarization drives the next cell into the same polarization, since this combination of states has minimum energy in the electric field between the charged particles in neighboring cells. Information is copied and propagated in a wire consisting of the cell automata. Figure 1 shows the available two cell types and the corresponding wires.

The QCA cells can form the primitive logic gates shown in Fig. 2. The simplest structure, the inverter, is usually formed by placing the cells with only their corners touching. The electrostatic interaction is inverted, because the quantum-dots of different polarizations are misaligned between the cells. The other gates are usually based on a three-input majority gate of the cell type one, relaxing to minimum energy between the input and output cells, having the truth table shown in Table 1. The gate performs the two-input AND-

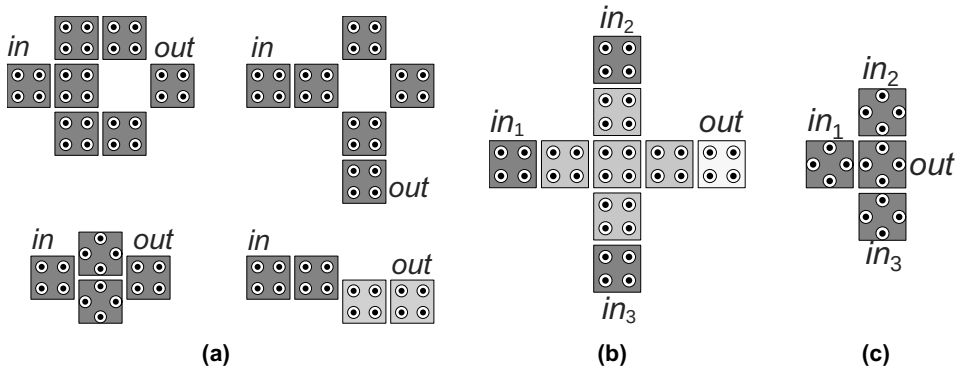


Fig. 2. *QCA primitive logic gates: a) different inverters, b) 3-input majority gate, and c) 3-input minority gate. Gray levels indicate different clocking zones, required to achieve reliable operation without noise coupling (see Sec. 2.3).*

operation when the third input is fixed at logical zero, and the two-input OR-operation when the third input is fixed at logical one. This completes a universal logic set, capable of implementing any combinatorial computation. [5]

The majority gate is classified as a threshold gate, where the sum of weighted inputs has to exceed a given threshold value before the output is set (an introduction to threshold logic can be found in [20]). Synthesis of general majority logic for the nanotechnologies was considered in [21, 22], while an optimal representation of 3-minterm Boolean logic using majority gates was developed in [23, 24]. An alternative universal logic approach is based on a very similar *minority gate*, constructed with the cell type two [25].

Table 1. *Truth tables of the QCA majority and minority gates, enabling AND, OR, NAND, and NOR operations.*

Majority Gate

	in_1	in_2	in_3	*out*
Two-input	0	0	0	0
AND	0	0	1	0
($in_1 = 0$)	0	1	0	0
	0	1	1	1
Two-input	1	0	0	0
OR	1	0	1	1
($in_1 = 1$)	1	1	0	1
	1	1	1	1

Minority Gate

	in_1	in_2	in_3	*out*
Two-input	0	0	0	1
NAND	0	0	1	1
($in_1 = 0$)	0	1	0	1
	0	1	1	0
Two-input	1	0	0	1
NOR	1	0	1	0
($in_1 = 1$)	1	1	0	0
	1	1	1	0

The QCA cells can be used to construct also more complex gate-level primitives, which could make circuit synthesis easier. An *and-or-inverter* (AOI) gate has been proposed in [26,27], and other programmable multi-input gates constructed using several majority voters in [28,29]. For increased robustness against defects and faults, a *block majority gate* was proposed in [30,31], and 3×3 cell grid *tile-based design* in [32–35].

It should be noted, that QCA is usually classical digital computing, not quantum computing: the binary information is contained in the classical degrees of freedom, instead of the superposition of states, while the quantum effects are used only to enable switching. Still, it is possible to construct both true quantum computers and analog information processors based on QCA [36]. Allowing multi-state cells enables also discrete ternary logic, based on the extended quantum-dot cellular automata (EQCA) [37,38].

2.2 Interconnects

Physical distance translates directly into timing delay and distinct number of pipeline stages, on QCA. Another important characteristic of the interconnects is the capability to create signal wire crossing in several ways.

Coplanar Crossing. The two basic cell types are orthogonal and can be positioned to have minimal interaction with each other, enabling the coplanar wire crossing shown in Fig. 3(a). One of the wires uses only cell type one, while the other uses only cell type two, resulting in the wires operating independently on the same fabricational layer. With this approach, it is possible to implement all the logic and interconnects on a single QCA layer, with no counterpart in the traditional technologies. The problem of the coplanar crossing is the high sensitivity to manufacturing faults: misplaced cells can easily break the symmetrical arrangement, leading to unwanted signal coupling between the two wires [39]. A recent effort to increase the robustness of the structure against defects and thermal effects can be found in [40,41].

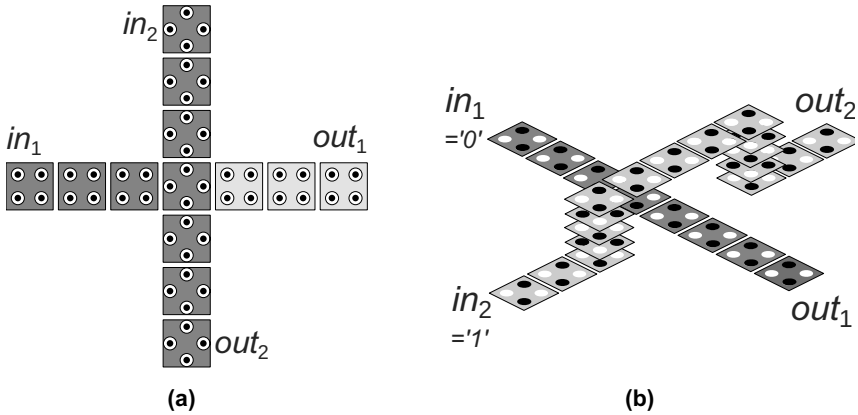


Fig. 3. *QCA wire crossings: a) coplanar crossing, with the gray levels indicating different clocking zones, required to achieve reliable operation without noise coupling (see Sec. 2.3), and b) multi-layer crossing.*

Multi-Layer Crossing. A traditional multi-layer crossing shown in Fig. 3(b) can be constructed with either cell type, as long as the vertical distance of the wires is large enough to prevent signal leaking from one layer to another, and there is a way to create vias of stacked cells between the layers. This approach is more tolerant to misplaced cells than the coplanar crossing, but requires an implementation technology with many active QCA layers on top of each other [42]. Another possibility is to create the vias and crossing layer using another technology, for example CMOS. The multi-layer technologies have not been demonstrated yet, and although the required precision of cell placement is not as high as with the pure single-layer technology, there are various other problems, which are likely to be as challenging. Due to this, the designs presented in this study use only the coplanar crossing.

Logical Crossing. Physical wire crossings can be eliminated by replacing them with logic gates, using node duplication, adjusting the timing of the crossing signals, or utilizing crossing-minimizing routing algorithms. Typically, this causes both area and performance penalty, but can be used to alleviate the manufacturing challenges considerably [43–47].

2.3 Signal Dynamics

The ideal QCA cell has very symmetrical and local interaction, but in real implementations, the electric field and quantum coherence transmit the interaction farther than the nearest neighbor. This weak interaction was earlier believed to be adequately canceled out by distance and layout symmetry, but a recent study clarified that the lack of *timing* symmetry can create sneak noise paths in crucial circuit structures. There is unwanted signal coupling between circuit sections, and even inside the primitive circuit elements. The problem is severe, as the QCA cells affect their neighbors in a very non-linear, bistable way: a small change in the polarization of a cell causes a much larger change in an un-polarized cell next to it, amplifying both the correct and the unwanted signals. Newly polarized cells provide positive feedback, which drives the injection point of the signal to an even stronger polarization [10,48]. This implementation problem was anticipated earlier by an esteemed physicist [49].

The coplanar wire crossing is extremely sensitive to noise coupling: In static case (when all the cells have settled to a polarization), the wire of the cell type two causes a symmetrical and effectively self-canceling interaction to the output side of the other wire, but in dynamical case (when the signals are arriving to the crossing), the polarization is at first present only at the input side, so that the compensating interaction from the output side is missing. This small unbalance causes the signal to couple into the output segment of the type one wire, followed by rapid copying from cell to cell and amplification, making the later compensation insignificant. When the real signal of the type one wire arrives to the crossing, it will not be strong enough to switch the output segment already settled in strong erroneous polarization.

The majority gate is also sensitive to signal timing, as the requirement for fair majority voting is, that all the input signals must be present with equal magnitude, when the active center cell starts to switch. If there is glitching in the center cell, the erroneous polarization gets copied to the output section of the gate, which effectively functions as a noise amplifier. However, the problem with both of these circuit primitives can be fixed with timing constraints:

the real input signals must be present and driving strongly, when the crucial circuit section begins to switch. QCA clocking schemes potentially offer the capability for this. (Explanation of the clocking follows in Sec. 2.4.)

Zone Clocking. A simple arrangement of the QCA clocking zones ensures, that the real signals beat the noise signals racing through the circuit [10, 48]. A coplanar wire crossing functions correctly, when the output section of the cut wire is switched only after the other parts have firmly settled. Similarly, a majority gate functions reliably when it is placed on three clocking zones: the first zone secures the inputs, the second zone performs majority voting, and the third zone latches the result. Such zone assignments are shown with different gray levels in Fig. 2 and 3.

Wave Clocking. There is no robust method of avoiding noise coupling under wave clocking (coarse inhomogeneous clocking field), because we cannot control precisely, which cells are contained in the switching section at each moment. Careful design and precise manufacturing might enable the coplanar wire crossing and the majority gate to work, but it is still unknown, if this can be reliably achieved [50]. In view of this, only designs based on the robust clocking zone approach are considered in this thesis.

2.4 Clocking Approaches and Pipelining

On QCA, a clocking mechanism determines via an electric field when the cells are un-polarized, latch their input values, and start driving other cells. It is used both for designing sequential circuits, creating pipelines, and forcing the circuit to stay in the quantum mechanical ground state, which depends on the inputs of the circuit, and represents the correct computational result and successful signal propagation. The clock provides also additional energy, enabling true signal gain on this nanotechnology. Non-clocked cell arrays seem unpractical, since they relax to the ground state too slowly [51].

A large array of cells switching at the same time can get stuck in a local energy minimum of the combined electric field, called a *kink* state, never reaching the

ground state, producing an erroneous computation result. To prevent this, the active phase of the clock is applied only to a small section of the circuit at each time instant, making the probability of the kink state to diminish [6]. The practical section size set by this phenomenon is not yet determined, but background thermal fluctuations set another upper limit: on molecular QCA, a single majority gate would function up to the temperature of 450 K, and a wire segment of 50 cells would still operate correctly at room temperature [52].

Clocking Structure. The section size can be restricted by dividing the cell array into zones controlled by different clocks, discrete clock phases for adjacent zones. This *zone clocking* [6] is simple and provides exact control for the timing of the circuit (over single QCA cell), but requires very fine-grained clocking circuit, possibly unpractical to manufacture on the molecular technologies. The clocking circuit underneath the cell layer can be made more coarse-grained and much easier to manufacture, if we increase the spacing between the clocking wires, and apply an inhomogeneous, smoothly graded electric field over the QCA plane. Multi-phase clocking signals cause the active clocking field to travel through the circuit in a continuous manner, creating a wave of computation. This *wave clocking* [53] approach was developed especially for molecular QCA, but it provides only coarse control over the timing of the circuit (grouping tens of cells together). A two-dimensional approach applicable for systolic structures was proposed in [54, 55].

Clocking Waveforms. The waveforms of the usual *Landauer-type clocking* [6, 53] are simple, and rely only on adiabatic switching principles to limit the dissipation of changing the state of a cell: the clock transition speed is limited, enabling the re-use of the signal energy already present in the circuit. Figure 4(a) shows a wire spanning four clocking zones and Fig. 4(b) the corresponding Landauer clock waveforms. During a complete cycle, each zone goes through the four phases (*Release*, *Hold*, *Switch*, and *Relaxed*, defined in [6]), and the wire effectively implements a stage of a micropipeline or a register. Since only one of the zones can hold a valid bit during a clock fraction, four zones are needed to construct one logical pipeline stage. This scheme is simple, but when applied to a circuit of normal irreversible logic, it leads

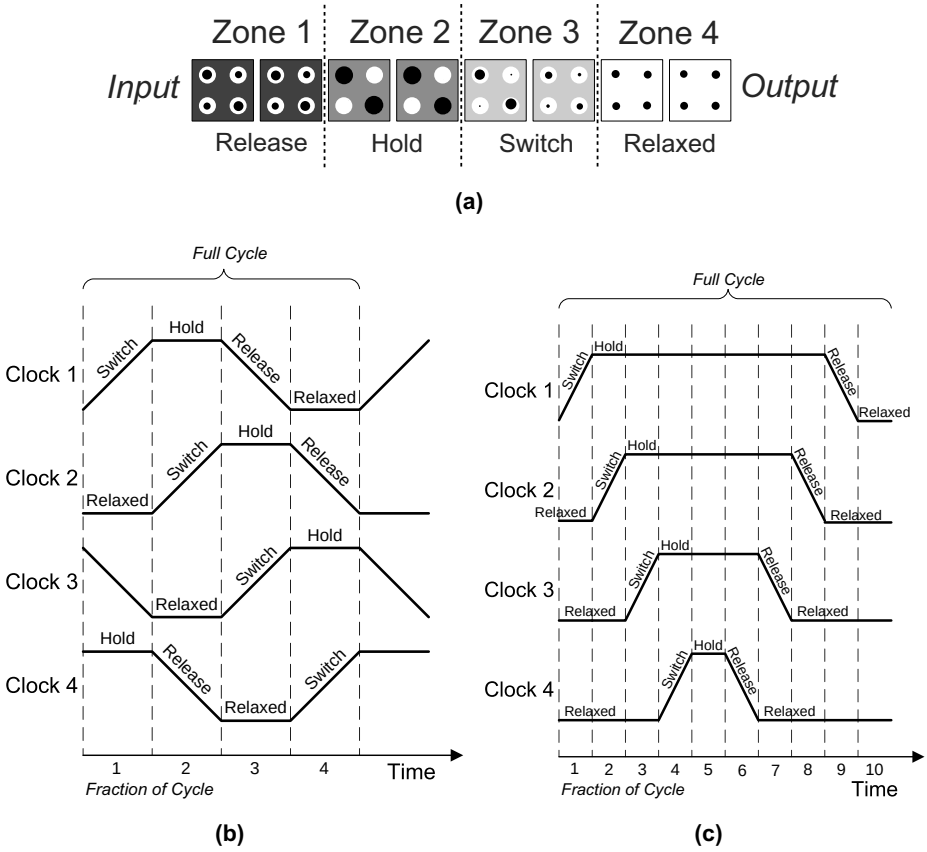


Fig. 4. *QCA zone clocking: a) a wire spanning four clocking zones, shown at Landauer clock fraction three, b) Landauer clocking waveforms, and c) Bennett clocking waveforms.*

to energy dissipation caused by erasing information [18], which becomes the dominant factor on molecular QCA with high operating frequencies [17].

As the ultimate power limits of irreversible computing are quickly reached, *Bennett-type clocking* [16,56] was recently proposed to achieve fully reversible operation on QCA, while using cheap irreversible logic gates. The underlying principle (originating from [57]) is that we first compute the results by latching the cell array, from the input side to the output side, and then uncompute by letting the array to relax to an unpolarized state, from the output side to the input side, eliminating most of the bit erasures and limiting the dissipation to

either the inputs or the outputs of the logic circuit. This scheme makes *any* QCA circuit fully reversible, without additional circuit area or complexity cost, while the penalty is paid fully in the timing. Figure 4(c) illustrates the Bennett clock waveforms, making a circuit spanning four clocking zones to operate reversibly, using more than twice the number of clock fractions, in comparison with the simple Landauer scheme. The major drawback is that the natural pipelining is lost, when all of the clocking zones are occupied by the same bit (one element of a data set in a pipeline). An improved floorplan to achieve general space/time tradeoff was proposed in [58].

2.5 Physical Implementations

There are various ways to construct the physical QCA cells with bistability and dominating local interaction, and apply the clocking, but the material systems are still very early in their development. Most of the research has been on intercellular electrostatic interaction, promising the best performance, implemented with metal-dot, semiconductor, or molecular approach. However, in the near future, the first large-scale circuits might be realized with magnetic interaction, with lower performance, but less challenging manufacturing [59].

An introduction to early work on the QCA implementation challenges can be found in [60], while image charge neutralization was considered in [61, 62], cell-level improvements to reach the ground state more robustly in [63, 64], and a three-dimensional layer approach to remove the metastable states proposed in [65]. *Restricted minima quantum-dot arrays* (RMQDA) concept, based on passive non-clocked wires and active clocked amplifier segments, was proposed in [66], and *split current QCA* (SCQCA), a technology based on resonant tunneling currents, was proposed in [67]. In the following, the main approaches to physical QCA implementation are summarized, the first three based on electrostatic interaction and the fourth on magnetic interaction.

Metal-Island QCA. The realization based on Coulomb blockade effect in tunnel junction connected metallic islands has been demonstrated in labora-

tory environment, but since the feature size is fairly large (around one micron dot-to-dot distance), the quantum mechanics needed for operation are present only in cryogenic sub-Kelvin temperatures. The experiments have been typically conducted at 15 mK [68], using liquid helium cooling, while 300 mK has been the highest reached temperature. This approach has enabled small proof-of-concept circuits (low-temperature prototypes of the future molecular systems), showing that the bistable cellular automata can be realized and switched by controlling the tunneling of electrons, but it is not practical to construct large circuits this way. [7–9, 69–82]

Semiconductor QCA. The semiconductor realization of the electrostatic cells promises a way to mass manufacturing of QCA circuits, since the industry has already decades of experience with the constantly improving processes. The challenge here is the need to reach extremely small feature size, for the circuits to operate above the sub-Kelvin temperatures. The future lithography processes might reach small enough granularities to enable a somewhat higher operating temperature, up to about 77 K [83], which could be acceptable in a supercomputer. Possible application is the classical part of a quantum computer, with state-of-the-art cooling system. [84–90]

Molecular QCA. The most desired implementation of QCA is based on single molecules acting as cells (transition metal atoms as quantum dots), as the molecules are naturally small enough (< 1 nm dot-to-dot distance) to enable the quantum effects at the room temperature, and they also lead to extremely high device densities. The challenge is again the manufacturing. First, large amounts of the right kind of molecules should be created, which seems very possible with chemical synthesis. Second, these cells should be deposited with unprecedented precision to form a circuit layout, which turns out to be very difficult. A promising way to do this might be high-resolution electron beam lithography combined with DNA self-assembly. [6, 52, 53, 91–107]

Magnetic QCA. The other flavour of QCA uses bistable magnetic cells to store information, and magnetic coupling to transport it between the cells. The benefit of this is the availability of the crucial quantum effects also at high temperatures and relatively large feature sizes, which simplifies the manufac-

turing process very much. There have been several proposals to implement magnetic QCA, but common for all of them is the limited operating speed, for nanomagnets around 10–100 MHz. [108–116]

The designs presented in this thesis have been verified with a simulation model of the electrostatic QCA, but in principle, they should be relatively easy to adapt also to the magnetic technology. For example, the standard wire construct of a magnetic variant might be an inversion chain, which a design automation tool should handle correctly while translating wire blocks between the technologies. However, the tools do not exist, yet.

2.6 Simulation Models

High-abstraction level design models for QCA have not been much studied, but the logic behavior can be coarsely explored with simple digital bistable approximations. However, since the QCA circuits are still at such an early stage of research, more faithful modeling has to be done at the quantum mechanical level. As full quantum mechanics simulation is not computationally feasible for systems of even tens of the cellular automata, several approximations have been proposed for more practical circuit verification, but it is not yet clear, how well these approximations really match the physical implementations. Complete treatment of the quantum mechanical modeling of QCA circuits can be found in [36], while the main approaches are summarized in the following. The few technology-dependent design and simulation tools available are M-AQUINAS [85], Q-BART [117, 118], and QCADesigner [119–123], while a Bayesian network based probabilistic approach has been described in [124–128] and a Hopfield neural network based simulator in [129]. Classical SPICE tool based modeling has been proposed in [130–132].

The starting point is the *full quantum mechanical model*, where the fully coherent system is modeled by the many-body Schrödinger equation. This is feasible for a couple of cells only, since the computational burden scales exponentially with the number of cells, preventing use in system verification [133].

The problem can be slightly alleviated by approximating the cells as *coupled two-state systems*, which raises the circuit size limit to 10–15 cells. The way to model larger QCA circuits of practical size is the *Hartree-Fock approximation*, which models the intracellular dynamics quantum mechanically and the intercell interactions with classical Coulombic coupling [36]. The state variables of the Hartree-Fock approximation scale linearly with system size, but error is introduced by failure to model the time-dependent dynamics correctly, losing sight of the intercell correlations. An intermediate approximation accounting for second-order correlations was recently implemented in [122].

Implementation of the Hartree-Fock Approximation. The current state-of-the-art implementation of the Hartree-Fock approximation is included in the freely available QCADesigner tool [119–123], internally using coherence vector formalism, based on the density matrix approach. Each QCA cell is considered as a simple two-state system, represented by a Hamiltonian matrix. The charge neutral cells interact through a quadrupole-quadrupole moment, which decays inversely as a power of five of the distance between the cells, making it possible to limit the effective computation neighborhood.

The cell Hamiltonian is projected into a vector describing the energy environment of the cell on a Pauli spin basis. This is used to form the steady state coherence vector, describing the stationary energy state of the cell. The coherence vector represents the density matrix of a cell (as a Pauli spin projection, the polarization the third component), which is used in the equation of particle motion. For each cell, the simulation evaluates the equation of motion (a partial differential equation) using an explicit time marching algorithm.

Computational Complexity of the Hartree-Fock Approximation. The simulation model of an arbitrary circuit design (for example, a full adder unit) consists of a large array of parallel QCA cells, each having a polarization state and an effective neighborhood of interacting cells. The main parameters of the explicit time marching algorithm are the time step and total running time, which limit the accuracy and the reached evolution stage of the modeled system. For each time step taken, the computation has to be carried across the complete cell array; for each cell, this is equivalent to evaluating the energy

environment vector, the steady state coherence vector, and the equation of quantum mechanical particle motion of the coherence vector.

The main burden of computation is directly determined by the number of cells in the array, and the effective neighborhood of a cell. Typical QCA designs require that at least next-to-neighbor interactions have to be included in the energy environment of a cell, or some of the phenomena needed for the circuit to operate are missed completely. This means that the effective neighborhood consists of at least 24 cells, each of these containing two electrons participating in the combined energy state. On each simulation step, after updating the energy environment and steady state vectors of a cell, the coherence vector of the cell is stepped forward in time. The existing simulation engine evaluates the partial differential equation with the Euler method or the Runge-Kutta method, which are local constant complexity operations.

2.7 Digital Design

In addition to the development of the underlying implementation technology, there has been a considerable amount of research into circuit and systems design for QCA. The early research aims to shorten the time required to master the technology, utilizing the special characteristics from the physical layout to high level system design; the coplanar wire crossing, sub-gate level pipelining, processing-in-wire, and memory-in-motion paradigms enable very cost efficient logic, but they also present new design challenges.

Memories. Although QCA is inherently a self-latching technology, where even simple wires function as shift registers, several general purpose structures have been proposed to enable traditional RAM type addressable memory. Most of the proposed designs circulate the stored bits continuously inside a QCA wire loop, dynamically refreshing the value, and implementing a serial memory (an addressable shift register) [134–136], a fully parallel memory with one bit per loop [42, 120], or a serial-parallel hybrid [137–139]. More complex H-structured memory and the corresponding execution model have been proposed in [140–142], while the loop-based memory approach has been

replaced with a line-based approach, requiring special clocking zones and timing, in [143, 144]. A scheme to verify the timing of the basic memory structures using Verilog modeling has been presented in [145].

Programmable logic. Nanoscale circuits and architectures will have to tolerate higher percentages of defects than the current circuits, and the necessary redundancy can be flexibly provided by a programmable computing platform. Reconfigurable logic has been proposed for QCA in [29, 47, 83, 146–150].

Processors. In addition to the basic logic, more complex designs for QCA have been proposed, aimed at signal processing tasks. A serial bit-stream analyzer (SBSA) for telecommunications was proposed in [151, 152], an approach to synthesize energy minimizing QCA arrays for vision computing in [153], a SIMD array propagated instruction processor (PIP) for image processing in [154], a hybrid fast Fourier transform (FFT) processor with data permutation implemented using QCA in [155], and basic blocks for stochastic computing for signal processing in [156]. A general purpose microprocessor Simple 12 with universal clocking floorplan was proposed in [48, 117, 118, 157–159], and a design based on accumulator architecture in [42, 119, 160].

Design Methodology. Abstraction level can be raised to cope with large system complexity and to develop a systematic design flow, but this extremely important area has been only preliminary studied for QCA. A top-down methodology utilizing modeling with the standard VHDL language was described in [152], and an environment for behavioral and logic level QCA design (named HDLQ), generating Verilog code, was proposed in [161]. Tile-based modular approach was presented in [32–35, 135], and methodical construction of sequential systems in [162–164]. Globally asynchronous, locally synchronous (GALS) design approach was adapted to QCA to achieve layout-timing independent operation in [165–167].

Design automation. Coping with design complexity requires software tools, which transform a high-level digital system description into a valid description of the physical QCA layout, solving the general challenges of logic synthesis, placing and routing. Simple 3-minterm Boolean logic reduction to op-

timal majority logic was developed in [23, 24], a genetic algorithm based optimizer in [168], synthesis of symmetric functions in [169] and mathematical cellular automata logic in [170], and evolutionary synthesis of small QCA circuits in [129, 171]. A logic-level area cost model for synthesis was proposed in [172], while a true multi-level threshold/majority/minority network synthesis methodology and a tool (named MALS) were presented in [21, 22, 173, 174]. Physical design automation was divided into steps of clock zone partitioning, zone placement, and cell placement, providing the first partitioning and placement algorithm for automatic QCA layout generation, in [175–178]. Since wire crossings are crucial constructs on QCA, algorithmic approaches to crossing minimization have been developed in [43–46, 175].

Testing, defects, and faults. The defects and faults of the physical QCA implementations have not been experimentally established, but they are expected to turn out crucial to design feasibility. The basic types of QCA defects (cell displacement, misalignment, and omission) have been characterized and a testing scheme proposed in [179–182], the effects of scaling the size of the cells were inspected in [183, 184], and the defect tolerance properties of tile-based design were analyzed in [34, 35] and sequential systems in [162–164]. The effects of clock shifts and timing errors were simulated in [10, 48, 185, 186], switching error likelihood determined using probabilistic networks in [128], general displacement tolerance studied in [39, 187, 188], and thermal robustness of the wire crossing schemes predicted in [40, 41]. Logic-level fault masking in crossbar-based programmable logic array (PLA) was considered in [150] and the applicability of N-detect stuck-at fault testing for QCA designs in [189]. Faults in logically reversible one dimensional QCA arrays were studied in [190–192], utilizing the bijective nature of the gates to simplify testing significantly. Automatic test pattern generation (ATPG) for combinatorial QCA logic was developed in [193–195]. For detecting system run-time errors, synthesis tool support for totally self-checking (TSC) threshold logic circuits was presented in [173], and triple modular redundancy with shifted operands was proposed as a circuit improvement in [196]. Block gates were proposed to obtain fault tolerance in [30, 31].

3. BINARY ADDERS

Previous arithmetic designs for QCA have not been optimized or the effects of optimization evaluated, leading to inefficiency and unacceptable penalty paid for the noise robustness (the dynamic problem of the physical cellular automata described in Sec. 2.3). The approaches presented here allow the construction of high-density, performance optimized, and noise tolerant basic adder circuits, utilizing the inherent characteristics of QCA technology, and aiming at modularity and customization to varying operand word lengths. Based on the novel designs and the literature, the relationship between the degree of parallelism (in the general sense of the term, including several co-existing data sets) and the traditional design metrics of performance and cost are analyzed, on pipelined QCA. This was reported earlier in [P1] and [P7].

Chapter Contents. The previous work on digital design and adders on QCA is summarized in Sec. 3.1, while the novel full adder, serial adder, and ripple carry adder units are described in Sec. 3.2, Sec. 3.3, and Sec. 3.4, respectively. The verification approach utilized for the proposed designs is reported in Sec. 3.5, and the design analysis presented in Sec. 3.6. The conclusion of the chapter follows in Sec. 3.7.

3.1 Previous Work

In addition to the development of the underlying implementation technology, there has been a considerable amount of research into circuit and systems design for QCA. The early research aims to shorten the time required to master this emerging technology, and even to guide its development with actual per-

formance and cost metrics. The special characteristics of QCA have not been encountered in traditional technologies, but in the nanotechnology implementations they affect everything, from the physical layout to high level system design. The coplanar wire crossing, ultra-fine-grained pipelining below logic gate level, processing-in-wire, and memory-in-motion paradigms enable very cost efficient logic, but they also present new design challenges. Introduction to the well-develop field of general computer arithmetic, including the addition algorithms and hardware structures, can be found in books [197, 198].

The articles introducing the concept of QCA paradigm contained already outlines of a combinatorial full adder unit, based on the cellular automata [5], followed by a more feasible clocked design in [6]. Some of the first multi-bit adders were the ripple carry adder introduced in [199, 200] and the bit-serial adder introduced [201,202]. Optimal majority logic representation for the full adder and the corresponding ripple carry adder were described in [200], while general majority logic optimization was developed in [23, 24], and an updated bit-serial adder described in [202]. The performance of the multi-bit adders was initially explored in [14], and more advanced structures with reduced latency, the carry lookahead adder and the conditional sum adder, were adapted to QCA and analyzed in detail in [11, 203–205].

Recently, it was discovered that the imperfect cellular interaction renders most of the previous QCA designs unreliable under dynamic conditions (see Sec. 2.3). The problem can be fixed with systematic placing of the clocking zones, but the approach has tremendous effect on the cost and performance of arithmetic units, which have to address it at very low level. Paying excessive penalty, a robust full adder and serial adder were proposed in [10, 48].

3.2 Robust Full Adder

The basic block used to construct nearly all larger arithmetic units is the single-bit full adder. In the following sections, the proposed noise rejecting design is described at the logic, the pipeline, and the QCA layout level.

3.2.1 Logical Structure and Pipeline

The proposed Full Adder (FA) structure in Fig. 5(a) follows the minimal 3-input majority gate based logical formulation presented in [200], which consists of three majority gates and two inverters. The sum and carry outputs, s and c_{out} respectively, are computed as follows:

$$\begin{aligned} s &= M[\overline{c_{out}}, c_{in}, M(a,b,\overline{c_{in}})] \; ; \\ c_{out} &= M(a,b,c_{in}) \end{aligned} \tag{1}$$

where a and b are the input operands, c_{in} is the carry input, and $M()$ is the majority function defined as

$$M(a,b,c) = (a \wedge b) \vee (b \wedge c) \vee (a \wedge c) \tag{2}$$

where $\wedge$ and $\vee$ denote the logical AND and OR operations, respectively.

The computation of the sum requires two levels of majority logic, thus the total latency of the sum is two clock cycles (eight clock fractions, following the definition given in Sec. 2.4). The design forms a two-stage pipeline, which can compute with two different co-existing operand sets in parallel. The carry is computed in one clock cycle, as it can be formed using only one majority gate. This is beneficial for designing arithmetic units like the ripple carry adder, since the propagating carries usually set the performance limit.

3.2.2 Layout

Possible QCA layouts for the full adder can be formed with various different cell arrangements and geometries (including mirroring), but the inherent pipelining and clocking zone restrictions have to be taken into account, to minimize the latency and increase the robustness (the related QCA signal dynamics were described in Sec. 2.3, and the general clocking and pipelining approaches summarized in Sec. 2.4). The layout in Fig. 5(b) is organized in such a way that the coplanar crossovers consume as few clocking zones as

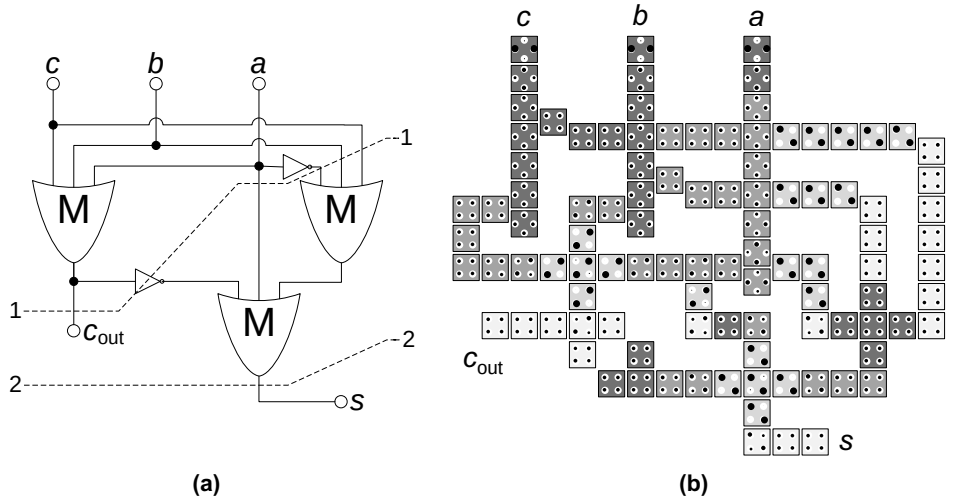


Fig. 5. *Full adder: a) logical structure, with the dotted lines denoting the pipeline stages (full clock cycles), and b) proposed QCA layout.*

possible. Avoiding noise coupling requires advancing one clocking zone on the horizontal output side of the crossings. The majority gates are laid out in three clocking zones, which ensures that the majority voting considers all the input signals at the same moment in time, and the cells on the output side do not function as a noise amplifier during the computation.

The inverters can be formed inside one clocking zone, except the unaligned inverters used to translate the cell segments before and after the wire crossings. These structures have weaker signal coupling causing susceptibility to noise amplification, and the output side has to be advanced by one zone.

The number of cells in a clocking zone is limited to ensure that the array reaches its energetic ground state, which is also the correct logical result state. The longest continuous path inside a zone is eleven cells, as the crossing wires present only minimal coupling.

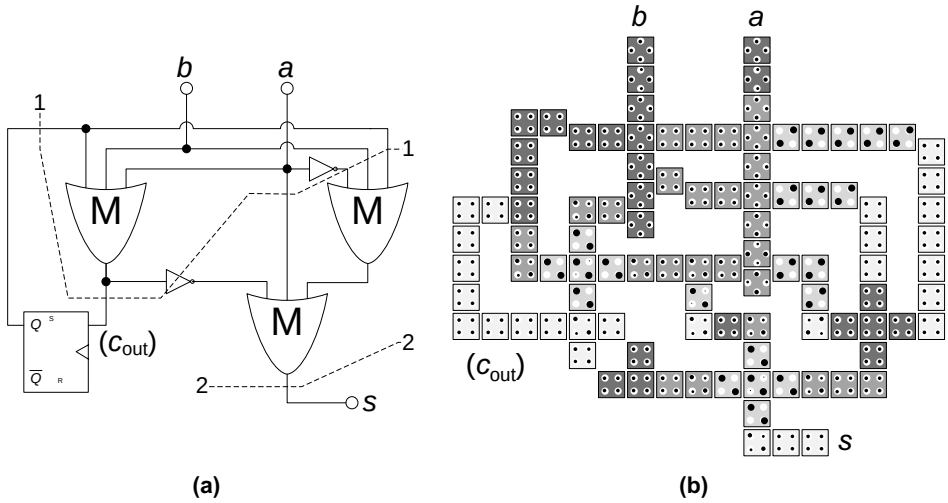


Fig. 6. *Serial adder: a) logical structure, with the dotted lines denoting the pipeline stages (full clock cycles), and b) proposed QCA layout.*

3.3 Robust Serial Adder

The simplest multi-bit arithmetic unit is the serial adder, a straightforward application of a full adder. In the following sections, the proposed cost-efficient design is described at the logic, the pipeline, and the QCA layout level.

3.3.1 Logical Structure and Pipeline

The textbook Serial Adder (SA) in Fig. 6(a) is formed by connecting the carry output of a full adder to feed the carry input. On QCA, the register shown in the carry loop can be removed, as the clocked full adder itself separates the intermediate values by the internal pipeline. The proposed design requires an additional startup clock cycle to clear the carry pipeline, since a mux to select between the initial and the feedback carry would require a pipeline stall.

The operand bits are fed into the adder serially on consecutive cycles, and the sum will appear after two cycle latency, in serial form on consecutive cycles. The intermediate carries are available with one cycle latency and can be fed back without a delay, so the adder can be operated without pipeline stalls.

3.3.2 Layout

The layout in Fig. 6(b) is based on the dense and robust full adder design introduced in Sec. 3.2, which guarantees avoiding most noise coupling. The design is modified to include a carry feedback, a short wire segment assigned the same clocking zone that is outputting the carry. Reliable routing is achieved by placing the interconnect far from other circuits parts, which have the same clocking zone assignment as the driver of the carry feedback. Interference from all other zones is rejected, and there are no wire crossings outside the full adder part.

The serial adder has a maximum of twelve cells in a single continuous clocking zone, forming the carry feedback. This limited array is expected to stay near the energetic ground state, propagating signals correctly.

3.4 Pipelined Ripple Carry Adder

The standard structure for the addition of binary words in the two's complement representation is the ripple carry adder, which utilizes several connected full adders in the computation. In the following sections, the proposed pipelined implementation is described at the logic, the pipeline, and the QCA layout level. The structure can be used also as a subtracter simply by negating the subtrahend and adding the result to the minuend, or by replacing the full adder components with their one-bit subtracter variant. [197, 198]

3.4.1 Logical Structure and Pipeline

The textbook Ripple Carry Adder (RCA) in Fig. 7(a) is formed by connecting full adders in series to form a carry path. On QCA, each full adder stage consumes one clock cycle to produce a carry, and the total delay of an n-bit adder is $(n+1)$ clock cycles, because the last stage requires an additional cycle to produce the last sum bit.

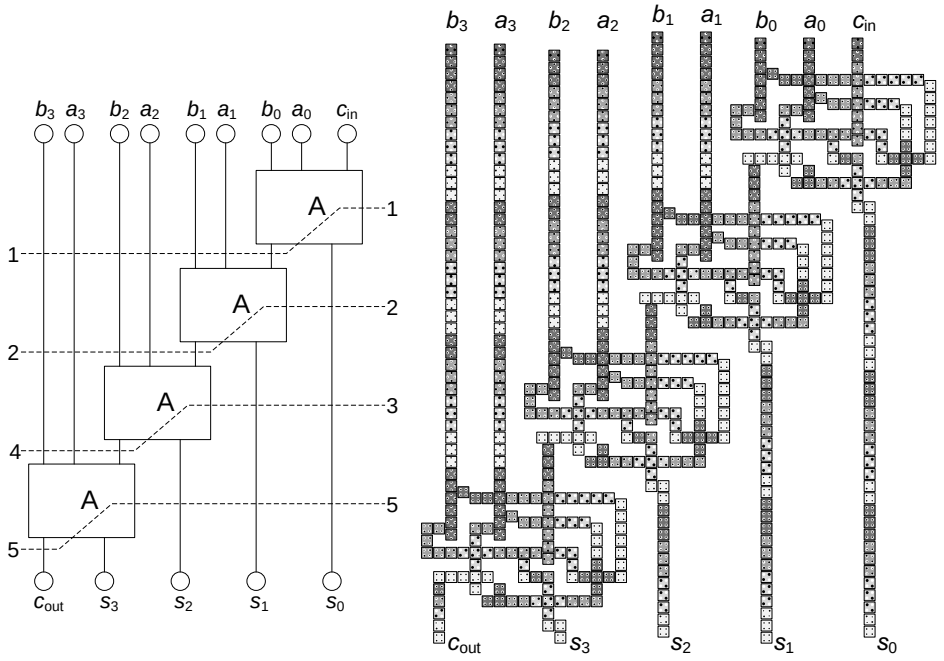


Fig. 7. *Ripple carry adder (4-bit case): a) logical structure, with the dotted lines denoting the pipeline stages (full clock cycles), and b) proposed QCA layout.*

The full adders operate on consequent clock cycles, as a result of the pipelining, and this sets a requirement to delay the input operand bits. Without this synchronization, the different additions coexisting in the pipeline would be mixed, producing only garbage as result. Each full adder stage must receive the operand bits at the same time as the previous stage outputs its carry, and also the sum outputs must be delayed so that the result can be obtained in parallel form, when the last stage finishes computation. The bits are delayed with a chain of cells propagating the signals in parallel, shifting on each cycle.

Each adder stage is ready to start a new computation as soon as it has produced an output carry. An n-bit pipelined ripple carry adder is computing n additions co-existing in the pipeline, and after the startup latency of $(n+1)$ clock cycles, the results are obtained on consequent cycles.

3.4.2 Layout

The 4-bit layout in Fig. 7(b) is based on the dense and robust full adder design introduced in Sec. 3.2, which guarantees avoiding noise coupling to some extent. However, care must be taken in placing these blocks adjacent to each other or the pipeline wiring, to avoid inter-block disturbance. This is relatively easily achieved by keeping the distances larger than two cells, or in the case of smaller spacing, assigning the adjacent wires non-interfering clocking zones, preventing a wire unintentionally driving a neighbor. Note that there are no wire crossings outside the full adder blocks.

An n-bit ripple carry adder has a maximum of eleven cells in a single continuous clocking zone, and the full adder stages and operand/result pipelines are organized in uniform fashion. This leads to a modular structure, which can be reliably customized to variable operand word lengths.

3.5 Verification

The logical correctness of the designs was checked with paper-and-pencil analysis, while the handcrafted layouts were simulated with the QCADesigner tool. The modifications during the iterative construction of the layouts included changing the number of cells used in the wiring, adjusting the locations of the wire crossings and the logic gates, and experimenting with various clocking zone arrangements and spacings between independent circuit sections, separated by the clocking zones.

The layout simulation with the QCADesigner tool utilized the coherence vector engine [119–123]. The computation is based on the Hartree-Fock approximation, which models each QCA cell as a single coherent quantum mechanical system, while the interaction between the cells is modeled with only classical electrostatic mechanism. The simulation is marched forward in a time-dependent manner, which reveals the circuit sections that are susceptible to noise coupling and amplification, giving early prediction of signal race

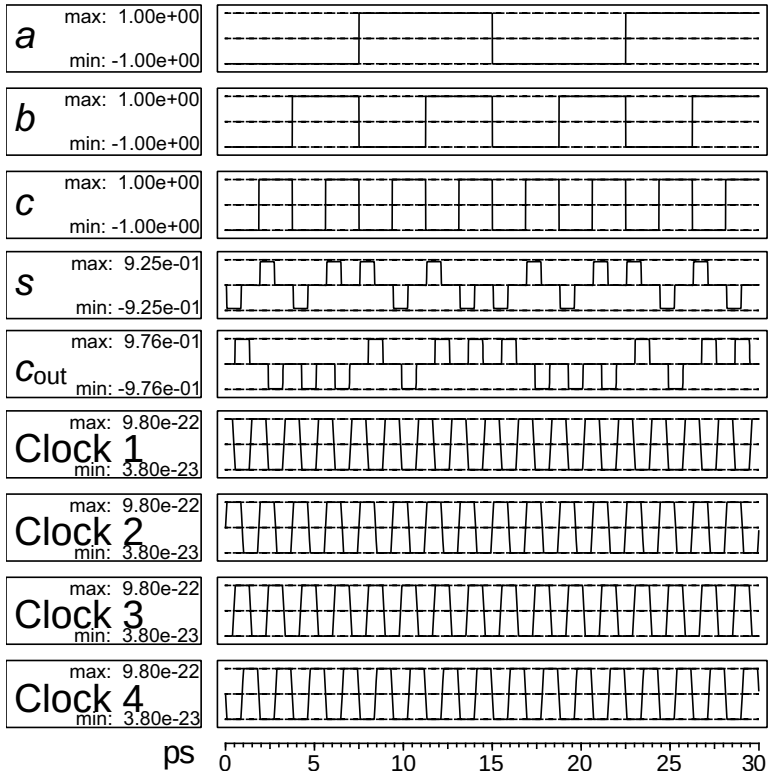


Fig. 8. *Simulation waveforms of the full adder, with sum latency of two cycles and carry latency of one cycle (30 ps shown, clock frequency 533 GHz). The waveforms of the individual logic signals show the polarization levels of the input and output cells, with the value range from -1 to 1, the extremities corresponding to the binary values 0 and 1, and the value 0 corresponding to the un-polarized state.*

conditions. This heavy model was necessary, because most other approaches are based on time-independent approximations, which completely fail to predict the noise paths. Part of the simulation results are shown in Figs. 8 and 9, where the signal waveforms correspond to the polarization of the output cells.

The simulated layouts were based on a QCA cell sized 18×18 nm, with 4 quantum dots each having a diameter of 5 nm, and the distance between the center of cells was 20 nm (corresponding to a semiconductor technology, enabling comparison with the previous design proposals). The parameters used in QCADesigner version 2.0.3 coherence vector engine were the defaults:

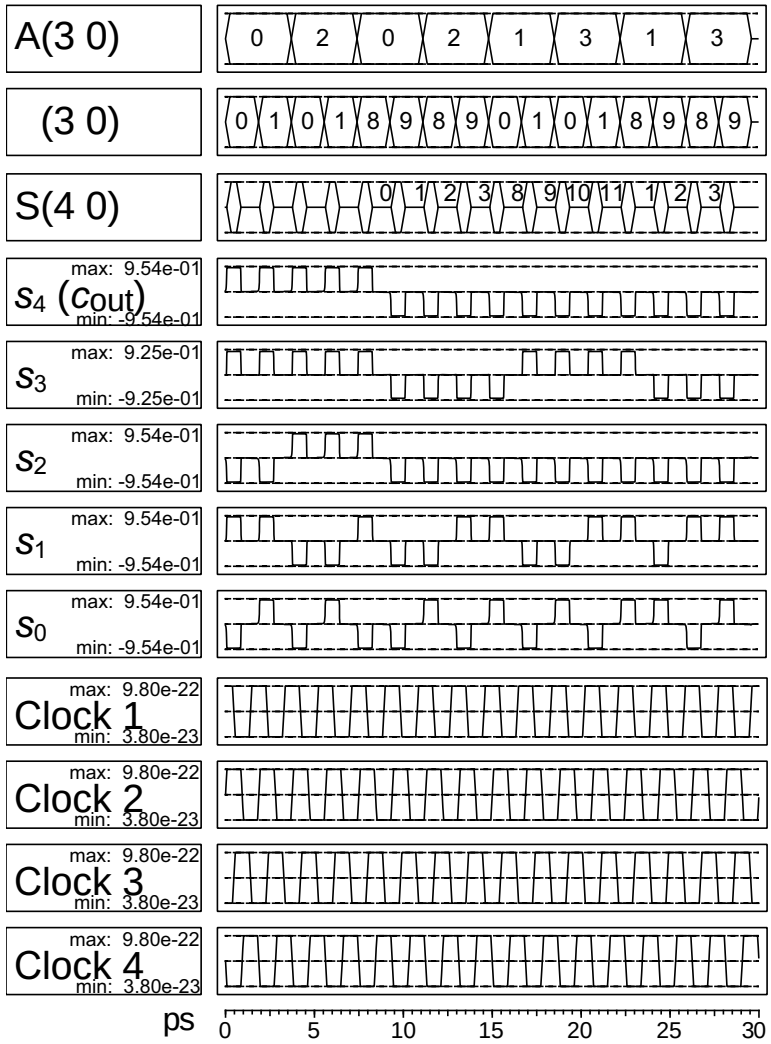


Fig. 9. *Simulation waveforms of the 4-bit ripple carry adder (30 ps shown, clock frequency 533 GHz). The waveforms of the individual logic signals show the polarization levels of the output cells, with the value range from -1 to 1, the extremities corresponding to the binary values 0 and 1, and the value 0 corresponding to the un-polarized state. A(3:0) and B(3:0) are the 4-bit input operand words, while S(4:0) is the 5-bit output sum word.*

temperature 1 K, relaxation time 1 fs, time step 0.1 fs, clock high 9.8×10^{-22} J, clock low 3.8×10^{-23} J, clock shift 0, clock amplitude factor 2, radius of effect 80 nm, relative permittivity 12.9, layer separation 11.5 nm, Euler method, and randomized simulation order. The QCA clock frequency was varied by changing the number of stimulus vectors and adjusting the simulation time.

Reaching the compact full adder layout required numerous iterations of modifying the structure and running the simulation, as the design was slowly optimized by hand. Exhaustive runs covering all the cases and different values in the pipeline was possible for the full adder and the serial adder, and the signal levels both inside the circuits and at the outputs were verified to settle strongly to correct values. The *n*-bit pipelined ripple carry adder was simulated exhaustively on small word lengths only, since the coherence vector based approach is computationally very expensive. The cases of two-, three-, four-, and eight-bit units were run through with all possible combinations of operand values, and larger units were tested with various operand sizes and representative input cases, since the number of states grows exponentially, preventing exhaustive runs. The results were logically correct and the signal levels unaffected by the word length, reaching extremely high simulated clock frequencies (several thousands of GHz), indicating that unwanted signal coupling is controlled throughout the circuits, and the designs are robust.

3.6 Design Analysis

The practicality of a design can be characterized with several metrics. The performance of arithmetic units is typically expressed as the *latency* before getting any finished computation results out, and the *throughput*, describing the number of results produced during a period of time. The engineering objective is to get high performance, while keeping low the traditional costs: *design complexity* and *circuit area*. In the following sections, the novel designs are analyzed in respect to these metrics and compared with the corresponding cases reported in the literature, specified in Table 2. (The important characteristics of reliability and power are treated in Ch. 5 and Ch. 6, respectively.)

Table 2. *Compared adder designs.*

Full Adder Designs:	**Serial Adder Designs:**
Proposed Full Adder	Proposed Serial Adder
Multi-Layer Full Adder [14]	Multi-Layer Serial Adder [14]
Single Layer Full Adder [200]	Single Layer Serial Adder [202]
Robust Full Adder [10]	Robust Serial Adder [10]

Ripple Carry Adder Designs:
Proposed Ripple Carry Adder
Multi-Layer Ripple Carry Adder [14]
Single Layer Ripple Carry Adder [200]
Robust Ripple Carry Adder [10]

3.6.1 Full Adders

Area. Figure 10(a) presents the layout areas of the adders, expressed as the relative size of the rectangular shape in comparison with the area of a QCA cell. The novel full adder occupies 12% more area than the smallest of all previous designs [14], which requires multiple cell layers and is still susceptible to noise coupling. Limiting to single layer designs, the proposed full adder is 38% smaller than the previous smallest one [200], which has been demonstrated to malfunction because of noise issues [10]. The novel full adder is 73% smaller than the only previous design considering the noise paths [10].

Latency. As a combinational full adder is not considered feasible on QCA, here is comparison of only clocked circuits, which can be reliably kept near the ground state. The fastest ones of the previous full adders [14, 200] produce both the sum and the carry in one clock cycle, and the previous structure avoiding the noise paths has a latency of three clock cycles for both outputs [10]. The proposed full adder has the noise coupling issues corrected and also produces the carry in one clock cycle. The sum takes two cycles, but this does not affect much the latency of larger arithmetic units, because the carry path is usually the critical one.

Throughput. The pipelined full adders have a throughput of one result per clock cycle on the carry output. After the startup latency, the constant throughput of the sum output is also one result per cycle, as long as the pipeline of the adder structure is kept filled with operand bits.

3.6.2 Serial Adders

Area. The serial adders occupy constant circuit areas shown in Fig. 10(a), regardless of the operand word length. The implementation based on the proposed dense full adder is 24% smaller than the previous smallest multi-layer case [14], although the older full adder unit itself is smaller. This is because the novel design has very short carry feedback path, which does not require any additional space. The proposed design is also 48% smaller than the previous smallest single layer case [202]. The novel unit is 76% smaller than the only previous design considering the noise paths [10].

Latency. The delay of the serial adders grows linearly with the operand word length, based on the carry delay of the underlying full adder: the fastest designs complete a carry at each clock cycle, thus they can use it in the next bit addition at once, and compute an n-bit addition in n cycles [14, 202]. The novel serial adder has also this minimal carry latency, although the sum pipeline has one additional stage, causing the n-bit addition latency to be $(n+1)$ clock cycles. The previous noise rejecting design has a carry delay of three cycles, so the pipeline has to be always stalled before another bit addition can be started: the n-bit addition takes $3n$ cycles [10].

Throughput. The throughput of all the serial adders is inversely proportional to the operand word length. The fastest units, like the novel design, use as many clock cycles as there are bits in the operand word to compute a single result, and the next computation starts only after the current one is finished: the result throughput is $1/n$ [14, 202]. The previous noise rejecting design is very inefficient because of the pipeline stalls, having a throughput as low as $1/(3n)$ [10]. Table 3 summarizes the characteristics of the adder designs.

3.6.3 Ripple Carry Adders

Area. The ripple carry adders consist of active logic and passive wiring (although clocked on QCA), which both consume area dependent on the operand word length: the number of full adders, and consequently their total area,

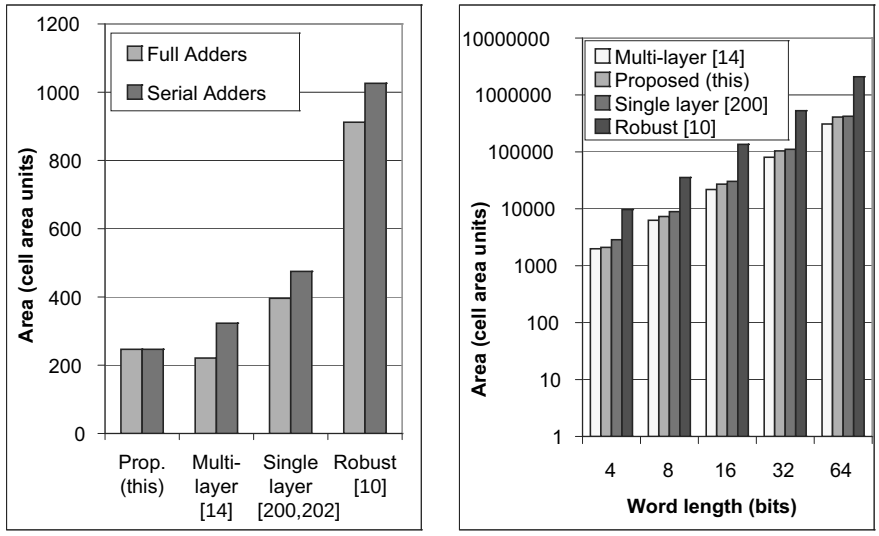


Fig. 10. *Area of the adders: a) full/serial adders and b) ripple carry adders.*

grows linearly, but for each new full adder block, the pipeline wiring overhead grows also in every row above and below, resulting in a square-law behavior. The total area is dominated by this wiring: in a 16-bit unit the wiring consumes nearly 90% of the area, settling in larger units asymptotically to about 99%, as shown for the proposed design in Fig. 11. A simple rectangular model was used, since the area trend changed very little, even if the wire blocks were compacted and the rectangular shape lost. The area follows always closely a second-order polynom of the operand word length.

The other designs have also this square-law dependence, as shown in Fig. 10(b). The greatest differences between various designs occur at large operand word lengths. In a 64-bit case with the operand and result pipelines, the proposed robust ripple carry adder is 32% larger than the smallest of all designs [14], which occupies multiple layers and still lacks in noise tolerance. The novel design is 3% smaller than the smallest of the single layer cases [200], demonstrated unreliable [10]. There are no previous ripple carry adder designs considering the noise paths, but if one would be based on the older robust full adder unit [10], the design proposed here would be about 80% smaller.

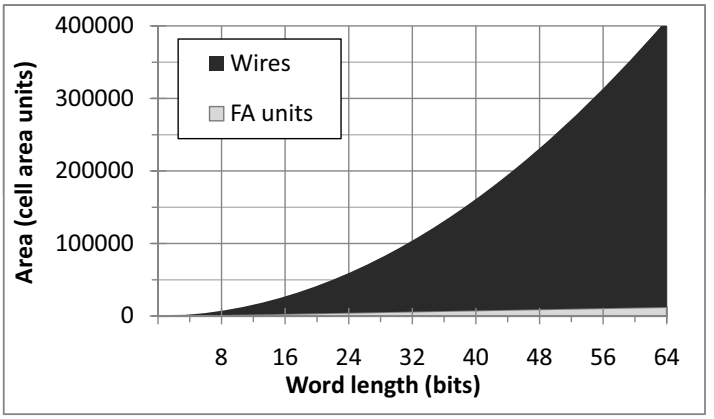


Fig. 11. *Square-law wiring and linear active area of the proposed ripple carry adder.*

Latency. The delay of all QCA ripple carry adders grows linearly with the operand word length, based on the carry delay of the underlying full adder: the fastest designs complete a carry each clock cycle, so the next full adder stage can use it with one cycle delay, resulting in computation of n-bit addition in n cycles [14, 200]. The novel ripple carry adder also has this minimal latency carry path, although the sum pipelines have one additional stage, causing the n-bit addition latency to be $(n+1)$ clock cycles. If a ripple carry adder would be based on the older robust full adder unit [10], the deeper pipeline would cause the n-bit addition to take $3n$ cycles.

Throughput. The QCA ripple carry adders achieve a constant throughput, because of the pipelined operation and computing several additions with co-existing data sets. The units produce a result on every clock cycle, regardless of the operand word length, as long as the pipeline is kept full [14, 200]. If a ripple carry adder would be based on the older robust full adder unit [10], the constant throughput could be also achieved, as the pipeline stalls affecting the serial adder could be avoided with the absence of feedback loops. Table 3 summarizes the characteristics of the different adder designs.

Noise coupling. The previous ripple carry adders do not consider the sneak noise paths, and many of these designs have also been verified to fail under more accurate simulation [10, 48]. So far, only the unit proposed here addresses this, verified by extensive time-dependent simulation.

Table 3. *Comparison of the performance and area of the n-bit adders.*

Design	**Latency** (cycles)	**Throughput** (results per cycle)	**Area** (cells)
Proposed SA	$n+1$	$1/n$	250
Multi-Layer SA [14]	n	$1/n$	320
Single Layer SA [202]	n	$1/n$	480
Robust SA [10]	$3n$	$1/(3n)$	1030
Proposed RCA	$n+1$	1	$98n^2$
Multi-Layer RCA [14]	n	1	$72n^2$
Single Layer RCA [200]	n	1	$98n^2$
Robust RCA [10]	$3n$	1	$504n^2$

3.7 Summary

Novel adder units were described at the logic, the pipeline, and the QCA layout level, and verified with quantum mechanical simulation. The proposed adders achieve tolerance against the technology-inherent noise coupling, while performing at least as well as the previous design proposals for QCA. High area-efficiency is achieved by the dense layouts, which can be used as building blocks in the construction of larger arithmetic units. Novel binary adder units for QCA technology:

- Robust full adder, with minimized area using only one fabrication layer.
- Robust serial adder, with minimized area and latency, and maximum throughput.
- Pipelined ripple carry adder, with noise robustness, minimized area and latency, and maximum throughput.

Based on the novel designs and the compared cases found in the literature, the basic serial and pipelined adder designs typically have the same latency, while the throughput is more affected by the degree of parallelism. Passive wiring overhead dominates the circuit area, with square-law dependence on the operand word length.

4. BINARY MULTIPLIERS

Previous arithmetic designs for QCA are inefficient and pay unacceptable penalty for the noise robustness (the dynamic signal problem of the physical cellular automata was described in Sec. 2.3). The approaches presented here allow the construction of high-density, performance optimized, and noise tolerant basic multiplier circuits, utilizing the inherent technology characteristics, while aiming at modularity and customization to varying operand word lengths. Based on the novel designs and the literature, the relationship between the degree of parallelism (in the general sense of the term, including several co-existing data sets) and the traditional design metrics of performance and cost is analyzed, on pipelined QCA. This work was reported earlier in [P2], [P3], and [P8].

Chapter contents. The previous work on QCA multipliers is summarized in Sec. 4.1, while the novel implementations of the serial-parallel, array, and radix-4 multiplier units are described in Sec. 4.2, Sec. 4.3, and Sec. 4.4, respectively. The verification approach utilized for the proposed designs is reported in Sec. 4.5, and the design analysis presented in Sec. 4.6. The conclusion of the chapter follows in Sec. 4.7.

4.1 Previous Work

There has been a considerable amount of research into arithmetic circuits on QCA, aiming to solve the challenges of more general digital design. The studied multiplier units are sufficiently massive to reveal the fundamental technology characteristics, while retaining enough structural regularity to enable

modeling and simulation to some extent. Introduction to the well-develop field of general computer arithmetic, including the multiplication algorithms and hardware structures, can be found in books [197, 198].

The first multiplier proposal for QCA was the serial-parallel structure [202], processing one of the operands as a parallel word and the other bit-serially. The logical design space and variations of this structure were thoroughly charted in [12, 205], and a quasi-modular parallel tree multiplier presented in [13], but these designs do not solve the radius-of-effect noise coupling problem described in Sec. 2.3. Thus, failure can be expected under dynamic signal conditions [10, 48].

In addition to the basic serial-parallel and parallel tree structures, there have not been any other multiplier proposals for QCA. This is the motivation behind adapting two other standard approaches, the cellular array and the radix-4 recoded multiplier, for the technology, presented in this thesis. The proposed serial-parallel and array multipliers use unsigned and the radix-4 multiplier two's complement number representation, and the multiplier and multiplicand operand lengths are independent.

4.2 Serial-Parallel Multiplier

The basic multiplication algorithm can be collapsed to construct a textbook serial-parallel multiplier, utilizing a single row of full adders to sequentially accumulate the partial products [197]. In the following sections, the proposed noise rejecting implementation is described at the logic, the pipeline, and the QCA layout level.

4.2.1 Logical Structure

The textbook serial-parallel multiplier (the first QCA adaptation in [202], recently named carry delay multiplier and optimized in [12, 205]) is formed by a chain of identical functional units, corresponding to a summation row in

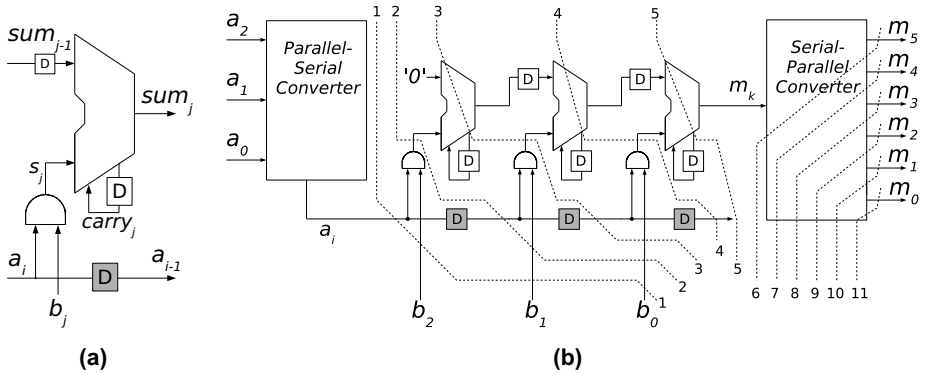


Fig. 12. *Logical structure of the serial-parallel multiplier: a) multiplier cell, and b) complete 3-bit multiplier, with the dotted lines denoting the pipeline stages (full clock cycles).*

the paper-and-pencil multiplication algorithm with unsigned n-bit binary input operands $A = (a_{n-1}, \ldots, a_1, a_0)$ and $B = (b_{n-1}, \ldots, b_1, b_0)$, resulting in a $2n$-bit output word $M = (m_{2n-1}, \ldots, m_1, m_0)$, where a_0, b_0, and m_0 are the least significant bits, respectively [197]:

$$
\begin{array}{cccccccc}
 & & & & a_{n-1} & \cdots & a_1 & a_0 \\
\times & & & & b_{n-1} & \cdots & b_1 & b_0 \\
\hline
 & & & & a_{n-1}b_0 & \cdots & a_1b_0 & a_0b_0 \\
 & & & a_{n-1}b_1 & \cdots & a_1b_1 & a_0b_1 & 0 \\
 & & & & \vdots & & 0 & 0 \\
+ & a_{n-1}b_{n-1} & \cdots & a_1b_{n-1} & a_0b_{n-1} & 0 & 0 & 0 \\
\hline
 & m_{2n-1} & \cdots & \cdots & \cdots & \cdots & m_1 & m_0
\end{array}
\tag{3}
$$

The multiplier cell in Fig. 12(a) computes a single bit multiplication of bit a_i from the serial operand and bit b_j from the parallel operand with an AND-gate, forming a *summand* s_j, which is combined by a full adder with a sum output sum_{j-1} from a cell to the left, and a previous carry output $carry_j$. Corresponding to a row in the paper-and-pencil equation, the complete multiplier unit formed by the chain of units in Fig. 12(b) forms a *partial product*, and sums it with the previous one. The final result is available in serial form on the output m_k on the right, on consecutive clock cycles.

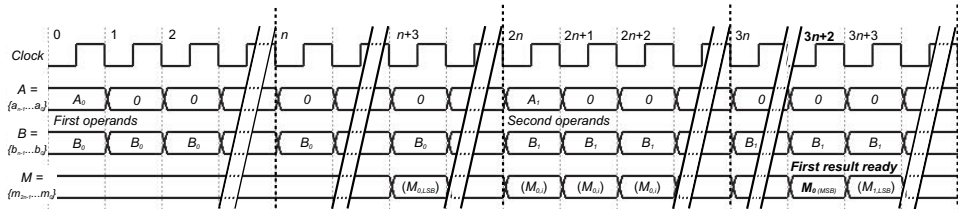


Fig. 13. *Timing of the complete n-bit serial-parallel multiplier.*

4.2.2 Pipeline

The serial operand cannot reach every multiplier cell on the same clock cycle, since distance translates directly into timing on QCA. The operand has to be fed through a shifting pipeline, spreading the computation of a partial product both in space and time. The critical path goes through every full adder, the implementation having a carry latency of one and a sum latency of two clock cycles, enabling serial operation without pipeline stalls. The summands are formed at consequent pipeline stages on consequent cycles, with several data sets co-existing in the pipeline. The total latency is defined by the critical path propagating the accumulated carries to the final result, growing linearly with the operand word length n, the least significant bit (LSB) appearing after $L_{LSB} = n + 3$, and the most significant bit (MSB) after $L_{MSB} = 3n + 2$ clock cycles. (With optimized structure, $L_{MSB} = 2n$ cycles [12, 205]).

The registers shown with shaded boxes in Fig. 12 correspond to the delays required by the serial operand wiring, while the other registers can be absorbed into the clocked adder units (the clocking and pipelining approaches were described in Sec. 2.4). Figure 13 shows the input/output timing of the complete n-bit unit, including the data format converters, showing the precise setting and holding of the parallel input words and additional zero values. The total latency L of the parallel output is $L = L_{MSB} = 3n + 2$ cycles.

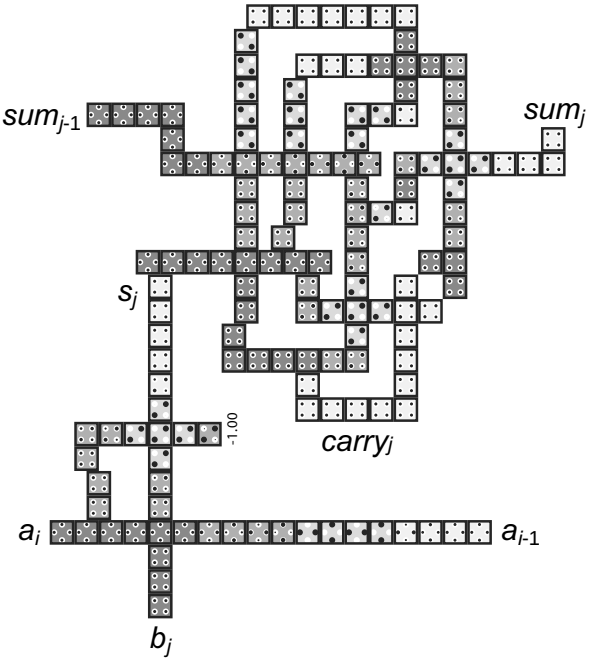


Fig. 14. *The proposed QCA layout of the serial-parallel multiplier cell.*

4.2.3 Layout

The functional cell of the multiplier is based on the serial adder implementation described in Sec. 3.3, following minimal majority logic formulation presented in [200]. The layout shown in Fig. 14 has at most a section of 12 QCA cells switching at the same time, forming the carry feedback loop of the serial adder. The bottom-left majority gate performs the AND-operation producing the summand, which the serial adder receives at the same instant as the sum from the left cell and the carry from previous cycle. In the bottom portion, the serial operand bit a_i is routed to the next cell, crossing over operand bit b_j line, consuming one clock cycle.

An n-bit multiplier is formed by combining n multiplier cells to the regular chain shown in Fig. 15. The wiring of the parallel operand B is shown on the bottom-left, delaying each bit according to the input stage on the chain. Figure 15(b) shows a 16-bit multiplier unit with considerable wiring overhead,

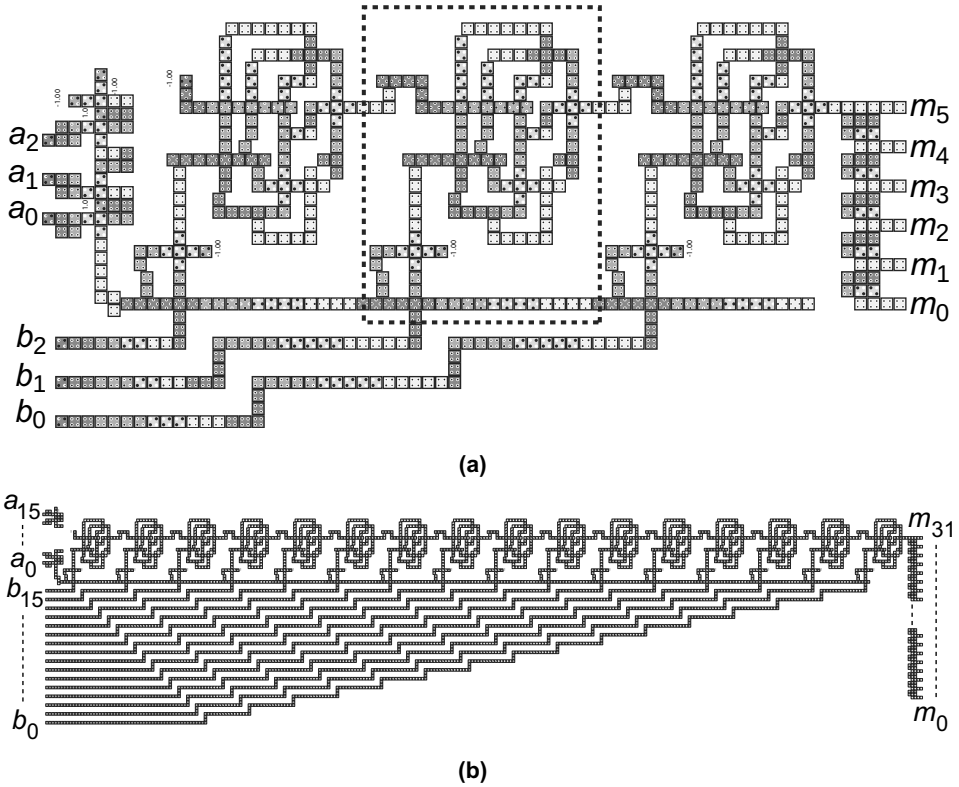


Fig. 15. *The proposed QCA layout of the serial-parallel multiplier, including the format converters and distribution wiring: a) 3-bit multiplier, having a latency of 11 clock cycles, and b) 16-bit multiplier, having a latency of 50 clock cycles.*

but without this wiring, the inputs would be spread on a large area, which is not practical, if there is a compact bus feeding the unit. The parallel-serial converter of operand A is located on the top-left, while the bit-serial result m_k emerges from the right end of the chain, followed by the serial-parallel converter producing the result M. Inter-block disturbances are avoided by keeping the circuit part distances larger than two cells, or in the case of smaller spacing, assigning the adjacent wires non-interfering clocking zones, preventing a component unintentionally driving a neighbor (the related signal dynamics were described in Sec. 2.3 and Sec. 2.4).

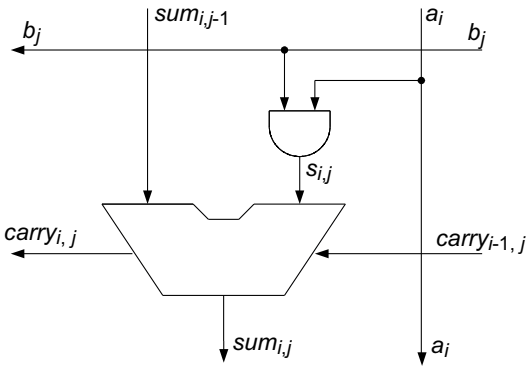


Fig. 16. *Logical structure of the array multiplier cell.*

4.3 Pipelined Array Multiplier

The systolic multiplier structure with maximum parallelism (in the form of co-existing data sets) available for inherently micro-pipelined technology is the textbook cellular array multiplier, constructed using customized full adder based cells [197]. In the following sections, the proposed cost-efficient implementation is described at the logic, the pipeline, and the QCA layout level.

4.3.1 Logical Structure

The textbook array multiplier [197] and the proposed implementation is formed by a regular lattice of identical functional units, following the paper-and-pencil multiplication algorithm with unsigned n-bit binary input operands $A = (a_{n-1}, \ldots, a_1, a_0)$ and $B = (b_{n-1}, \ldots, b_1, b_0)$, resulting in a $2n$-bit output $M = (m_{2n-1}, \ldots, m_1, m_0)$, where a_0, b_0, and m_0 are the least significant bits, respectively. The computation is mapped straight on the hardware, the smallest unit with all the necessary functionality having three-bit operands.

Figure 16 shows the logical structure of the multiplier cell, which computes a single bit multiplication of bits a_i and b_j using an AND-gate, forming a *summand* $s_{i,j}$, which is combined by a full adder with a previous sum output $sum_{i,j-1}$ from a cell above, and a previous carry output $carry_{i-1,j}$ from a cell to the right. Corresponding to a row in the paper-and-pencil Eq. 3, each row

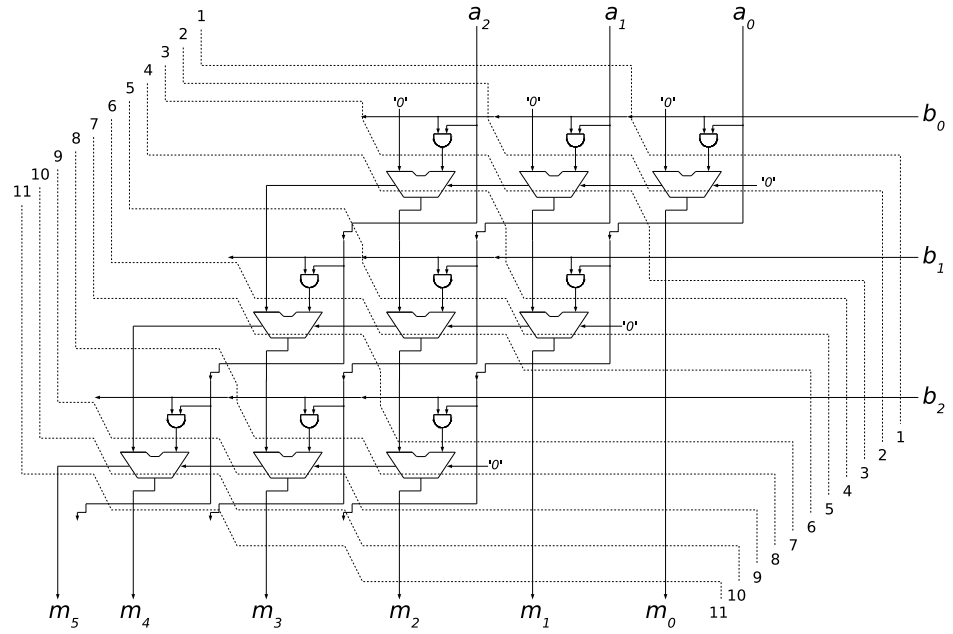


Fig. 17. *Logical structure of a 3-bit array multiplier, with the dotted lines denoting the pipeline stages.*

in the multiplier array shown in Fig. 17 forms a *partial product*, and sums it with the output of a row above. The final result is available in parallel form on the outputs $sum_{i,j}$ of the bottom cells in each column, and the most significant bit (MSB) appears on the last carry output $carry_{n-1,n-1}$.

The logic of the outer perimeter cells can be optimized by replacing the full adders with only two valid input bits with half adders, for a small linear reduction of circuit area, following the operand word length. However, the current design feeds zero values into the extra inputs of the full adders, resulting in equivalent logical operation.

4.3.2 Pipeline

The critical path of a cell is formed by the full adder, which has a carry latency of one clock cycle and a sum latency of two cycles, while the summand of another data set is formed in parallel. The dependencies between the multiplier

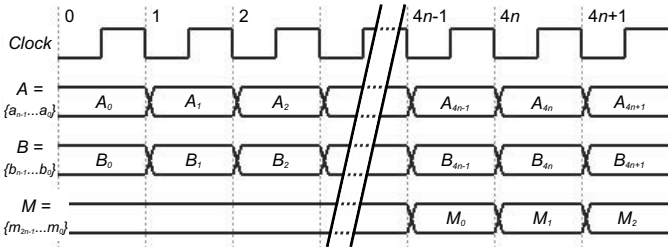


Fig. 18. *Timing of the complete n-bit array multiplier.*

cells dictate that the multiplication proceeds from top-right to bottom-left corner, the diagonal critical path latency L of the array growing linearly with the operand word length n: $L = 4n - 1$ cycles.

The input operands must be delayed according to the computational order of the cells in different rows and columns. The least significant bit (LSB) pair (a_0, b_0) can be fed into the array instantly, but the other operand bits a_i must be delayed by i cycles, and the bits b_j by $3j$ cycles. The multiplier outputs have to be synchronized into a parallel word, by delaying each sum output corresponding to result bit $m_k, k = 0 \ldots (2n-2)$ by $(4(n-1) - 2j - i)$ cycles, where j is the row and i the column index of the output cell in the paper-and-pencil organization, while the last carry output, the MSB result bit m_{2n-1}, must be delayed by one cycle. The dotted lines shown in Fig. 17 correspond to the signal delays and the pipeline stages (the clocking and pipelining approaches were described in Sec. 2.4), while Fig. 18 shows the input/output timing of the complete n-bit unit.

4.3.3 Layout

The multiplier cell is a modification of the full adder implementation described in Sec. 3.2, following the minimal majority logic formulation presented in [200]. The layout shown in Fig. 19 has been optimized for multiplier use by placing the inputs and outputs so that the wires between the cells are kept short, and the number of wire crossings is minimal (the related signal dynamics were described in Sec. 2.3). The largest section of QCA cells

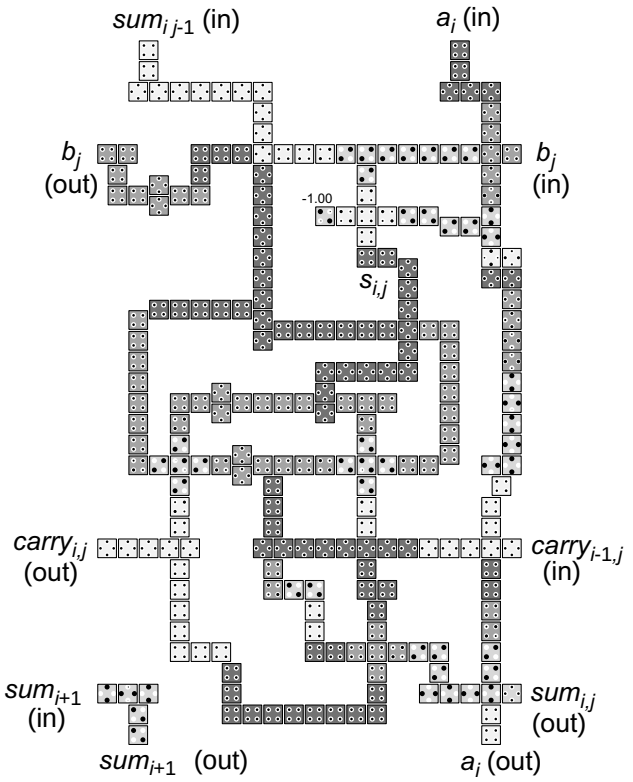


Fig. 19. *The proposed QCA layout of the array multiplier cell, with a delay of 2 clock cycles for the output $sum_{i,j}$, 1 cycle for the output $carry_{i,j}$, 3 cycles for operand a_i path, and 1 cycle for operand b_j path.*

switching at the same time is limited to 20 cells, placed in the continuous zone forming the sum input distribution network. The topmost majority gate performs the AND-operation producing the summand $s_{i,j}$, which the full adder receives at the same instant as the upper sum input $sum_{i,j-1}$ and the right side carry input $carry_{i-1,j}$. On the right side, the operand bit a_i is routed to the next row, crossing over the operand bit b_j, the carry input and the sum output $sum_{i,j}$, consuming three clock cycles. In the top portion, the operand bit b_j is routed to the next column on the left, crossing over the operand bit a_i and the sum input, consuming one clock cycle.

An n-bit multiplier is formed by combining n^2 multiplier cells to the regular array shown in Fig. 20 and Fig. 21. The rectangular shape was handcrafted

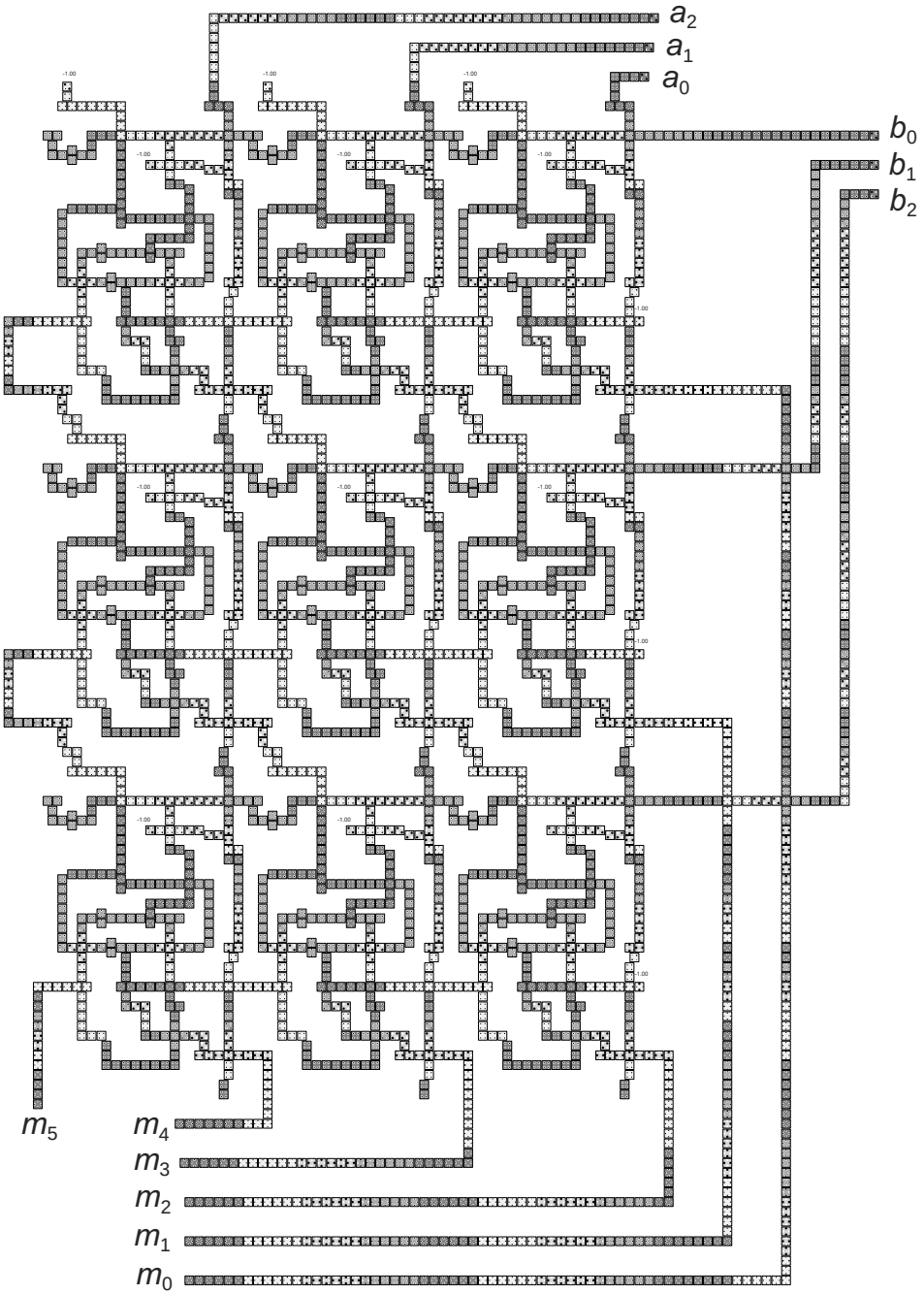


Fig. 20. *The proposed QCA layout of a 3-bit array multiplier, including the operand and result delay lines, having a latency of 11 clock cycles.*

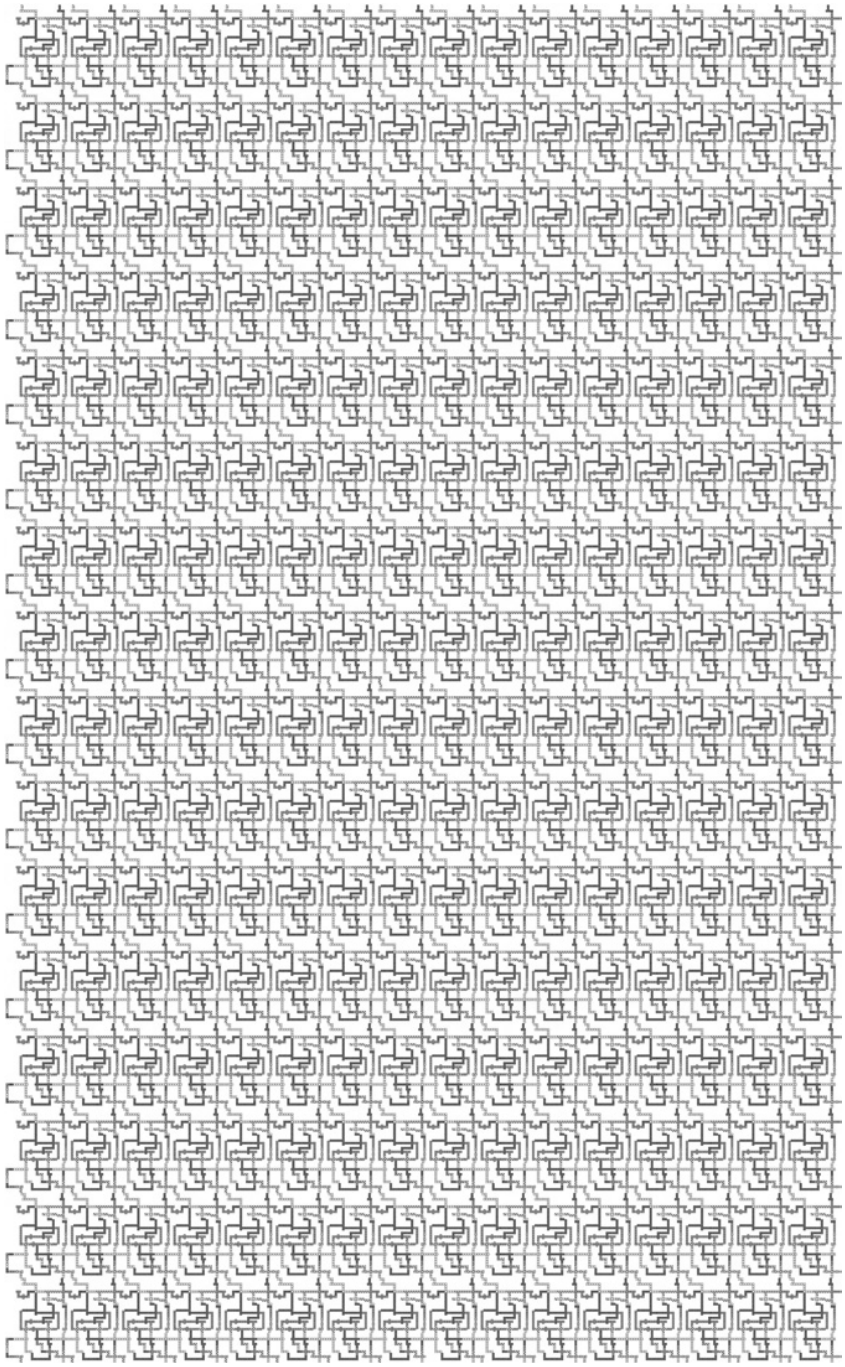


Fig. 21. *The proposed QCA layout of a 16-bit array multiplier, without the operand and result delay lines, having a latency of 63 clock cycles.*

iteratively, as the routing and the clocking zones of the multiplier cell were placed to ensure that noise interaction appears only after the correct signals have firmly settled. Inter-block disturbances are avoided by keeping the circuit part distances larger than two cells, or in the case of smaller spacing, assigning the adjacent wires non-interfering clocking zones (the related signal dynamics were described in Sec. 2.3 and Sec. 2.4). The operand bits are delayed before entering the array, the wiring shown in Fig. 20: operand A is located in the top portion, without any wire crossings, and operand B on the right side, crossing some of the multiplier output signals. Half of the outputs emerge from the right side of the array, and the others from the bottom, where the results bits are gathered and synchronized into the parallel word M.

4.4 Radix-4 Multiplier

The basic algorithm can be replaced with a standard, but more advanced algorithm, to construct the textbook radix-4 modified Booth multiplier [197]. In the following sections, the algorithm is introduced and the proposed scalable design is described at the logic, the pipeline, and the QCA layout level.

4.4.1 Radix-4 Multiplication Algorithm

The popular radix-4 modified Booth's algorithm, originating from [206], was chosen for implementation, since it handles two's complement numbers directly, and enables a systolic structure with no control overhead or feedback signals, which is an important simplification on QCA. The approach scans the multiplier word in groups of several bits, with an overlapping lookbehind reference bit, as shown in Fig. 22, and transforms it into a high-radix signed-digit (SD) number, with the digits representing the *operations* needed to perform the high-radix multiplication. The benefit of scanning two bits together is that the number of cycles needed to accumulate the partial products into the final result is halved, in comparison with the direct sequential algorithm.

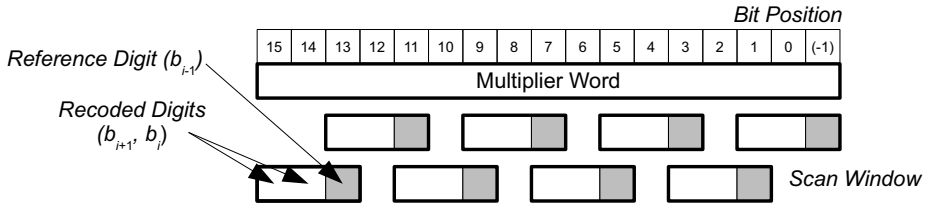


Fig. 22. *Radix-4 overlapped scanning, processing two new multiplier bits and a look-behind reference bit together, applicable sequentially or in parallel.*

The recoding function shown in Table 4 enables skipping over continuous sequences of zero or one bits in the multiplier word, replacing a sequence of operations with just two additions and shifting, while handling also isolated ones efficiently, as opposed to the standard radix-2 Booth algorithm, which can double the number of required operations in the worst case. Since this transformation has no dependence between adjacent digits, all the bit groups can be recoded in parallel, opening a way to compute also the partial products in parallel, which would not be possible with canonical SD recoding, offering the maximum number of zero digits and shift operations.

4.4.2 Structural Design

The proposed multiplier is a pipelined systolic structure with no feedback signals, since this approach matches very well the inherently pipelined technology by avoiding wire latency cost and control overhead (for a recent comparable implementation on CMOS, see [207]). The unit achieves the halved latency and doubled throughput promised by the radix-4 algorithm, but does not benefit from skipping of continuous bit sequences in the multiplier word, which on traditional technologies gives a variable performance boost depending on the particular multiplier word. The reason for this is that the novel implementation utilizes an ultra-low latency carry-save adder (CSA) to avoid pipeline stalls in accumulating the partial products: an n-bit summation step takes only one clock cycle, about the same time it takes to shift the accumulated value two bit positions, and much less than the time needed for a longer shift. This is because the physical distance translates directly into latency

Table 4. *Recoding function, which converts three original multiplier bits (b_{i+1}, b_i, and the lookbehind reference b_{i-1}) into the corresponding radix-4 operation.*

Multiplier bits			**Comment**	**Operation**
b_{i+1}	b_i	b_{i-1}		
0	0	0	string of zeros	0
0	1	0	single one	$+A$
1	0	0	start of ones	$-2A$
1	1	0	start of ones	$-A$
0	0	1	end of ones	$+A$
0	1	1	end of ones	$+2A$
1	0	1	single zero	$-A$
1	1	1	string of ones	0

and additional pipeline stages, on the technology (the related QCA signal dynamics were described in Sec. 2.3, and the general clocking and pipelining approaches summarized in Sec. 2.4). In the case that the adder had lower throughput than the other components, the zero/shift operations could be utilized to skip the adder completely, but with considerable control cost.

The top-level structure of the multiplier unit is shown in Fig. 23(a) and a QCA layout in Fig. 23(b). The main design objective was to achieve continuous operation without stalling the pipeline under any circumstances, since this would deteriorate the performance and require complex control, which seems currently very unpractical on QCA due to the timing restrictions. The original input operands (A, B) and the multiplication result (P) are in parallel data format, having word lengths n_A, n_B, and $n_P = n_A + n_B$.

The unit processes the multiplicand A as a full parallel word, and the multiplier B sequentially in the recoded radix-4 format ($\lceil n_B/2 \rceil$ digits). The multiplier digits are internally expressed with three mux control signals, which are used to select the correct multiple from the five alternatives ($+A$, $+2A$, $-A$, $-2A$, 0), produced by the combination of two's complementer and multiple distribution network. The multiples are accumulated with the sequential carry-save adder ($n_A + 1$ bits wide), which is also shifting the partial product two positions to the right on each clock cycle. The sum and carry vectors enter serially into the vector merge adder, which computes the final bi-

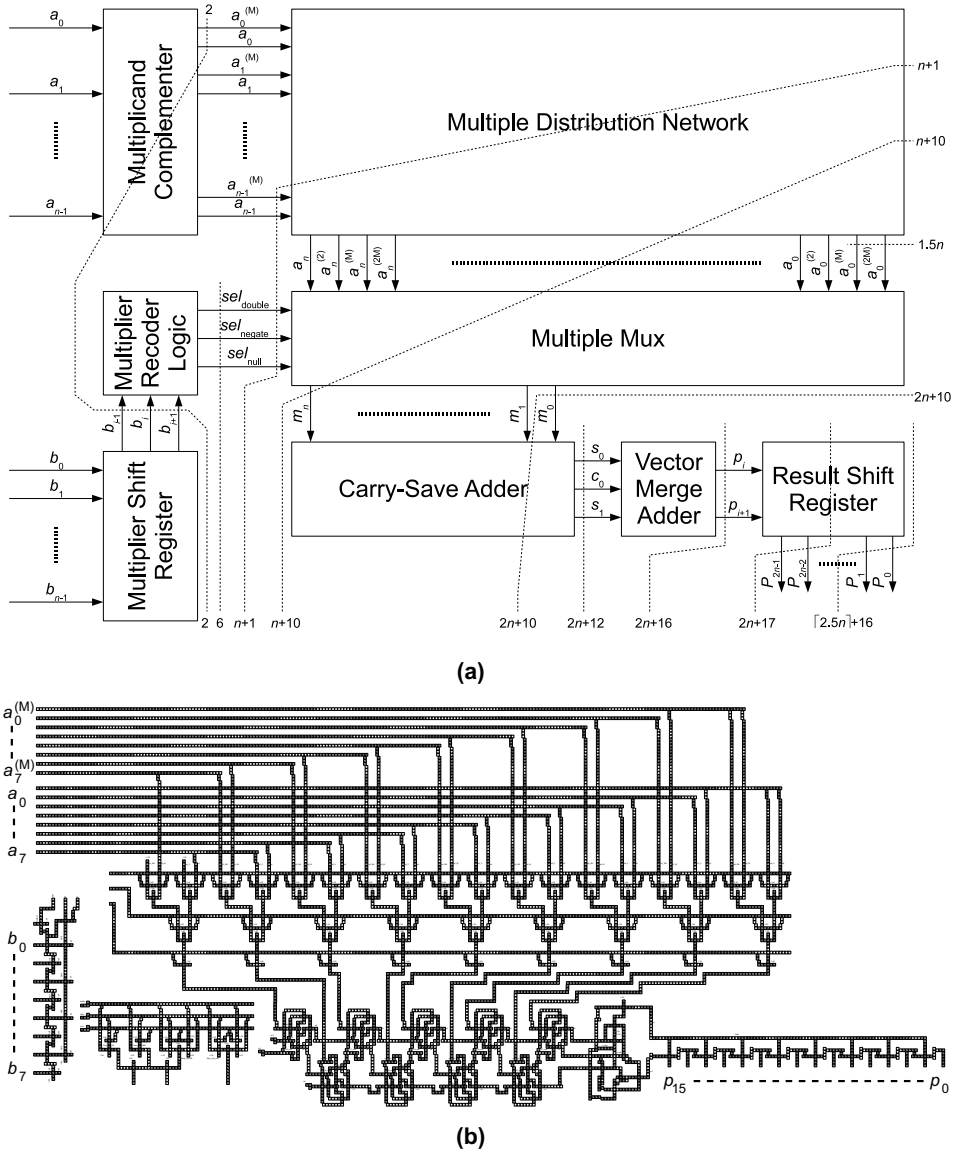


Fig. 23. *Proposed radix-4 recoded multiplier unit: a) block diagram, with the dotted lines denoting the pipeline stages (full clock cycles), and b) QCA layout with 8-bit operands.*

Table 5. *Latency and area of the separate components.*

Component	**Latency** (clock cycles)	**Area** (cell area units)
Multiplier shift register	2	$100n_B + 100$
Multiplier recoder logic	4	1650
Multiplicand complementer	n_A	$200n_A$
Multiple distribution network	$(n_A+1)/2+6$	$140n_A^2 + 500n_A + 200$
Multiple selection mux	$(n_A+1)/2+3$	$900n_A + 900$
Carry-save adder	$\lceil (n_A+1)/2 \rceil + 1$	$360n_A + 700$
Vector merge adder	4	700
Result shift register	$\lceil (n_A+n_B)/2 \rceil - 1$	$200\lceil (n_A+n_B)/2 \rceil$

nary product, operating sequentially and propagating the carries. The latency of the complete unit has a linear dependence on the operand word length: $L_{total} = 2n_A + n_B/2 + 16$ (exact with even number lengths).

4.4.3 Implementation

The multiplier unit consists of eight top-level modules, shown in Fig. 23(a): two blocks for multiplier word recoding (shift register and recoder logic), three blocks for creating and selecting the multiples (complementer, distribution network, and mux), and three blocks for accumulating the partial product and converting it into the final result (carry-save adder, vector merge adder, and shift register). Table 5 summarizes the operand length dependencies of the latency and area of the separate components, but the complete unit runs the pipelined bit slices of neighboring blocks in parallel, effectively hiding the internal delays. The latencies in the following description are initial.

Multiplier Shift Register

The n_B-bit multiplier word B is initially fed into the shift register, shown in Fig. 24, for conversion from parallel format into serial stream of bit groups. The output is produced from the least significant bit (LSB) side of the register, which effectively provides a 3-bit sliding window into the multiplier word,

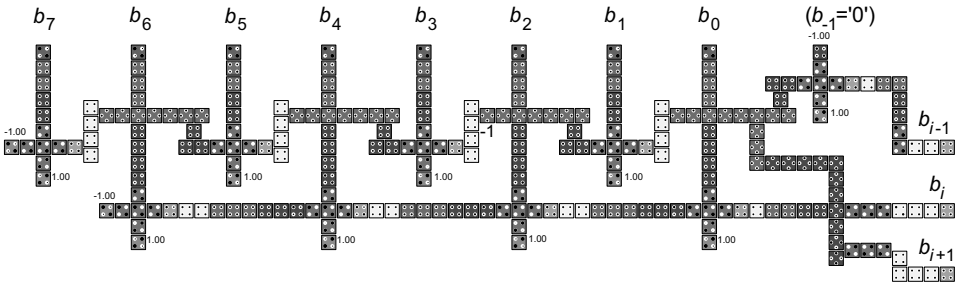


Fig. 24. *Multiplier shift register, 8-bit QCA layout.*

by shifting two bits per clock cycle (one overlapping reference bit). After a constant initial latency, the bit groups enter sequentially into the recoder block. The unit could be modified to produce all the bit groups in parallel.

Multiplier Recoder Logic

Three multiplier bits are transformed into the radix-4 SD format, with the first level of the recoder logic creating one-hot multiple select signals (sel_{+2A}, sel_{-A}, sel_{-2A}, sel_{null}), and the second level combining these to form the mux control signals (sel_{double}, sel_{negate}, sel_{null}), as shown in Fig. 25. The two levels could be combined for optimization, which might be beneficial if the block

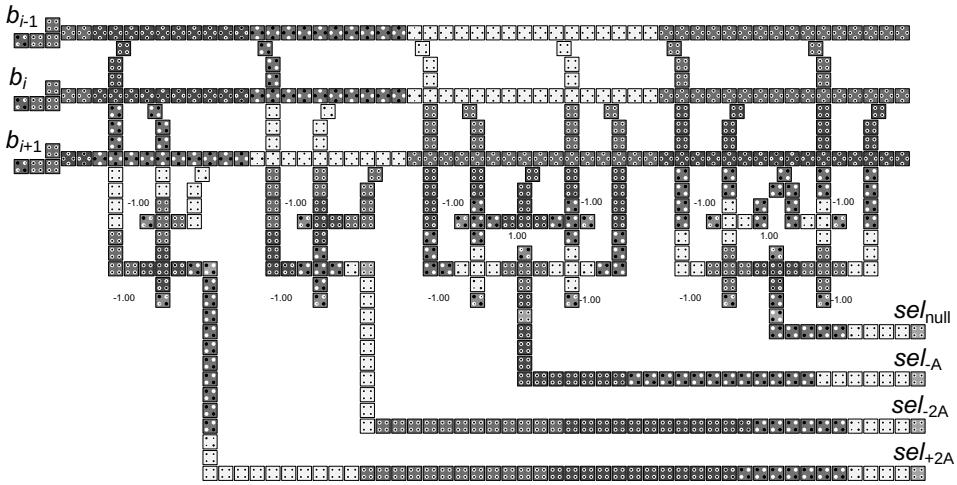


Fig. 25. *Multiplier recoder first logic level, QCA layout.*

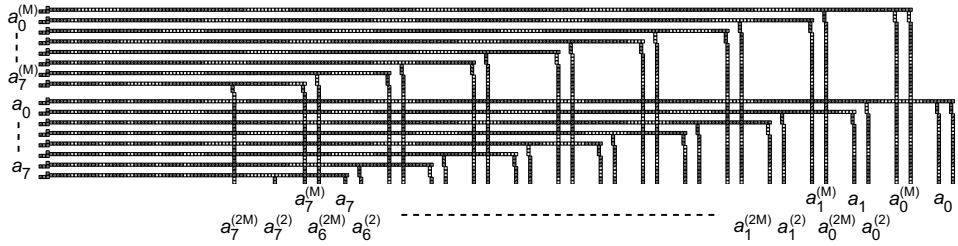


Fig. 26. *Multiple distribution network, 8-bit QCA layout.*

was used as a bit slice for a fully parallel recoder. The constant area and latency are quite insignificant, in the proposed design.

Multiplicand Complementer

The n_A-bit multiplicand word A is initially fed into the block forming the negated multiple ($-A$). This is two's complementation, which requires bit-inversion and the addition of unit in the last position (ULP, the weight of the LSB bit), so there is carry rippling with the associated linear latency, in respect to the word length. The block is, however, fully pipelined, and as such will cause only initial latency in the operation of the complete design. One's complement could be computed fully in parallel, with constant latency, and the addition of ULP postponed to the carry-save adder, for improvement.

Multiple Distribution Network

The multiple distribution network shown in Fig. 26 has the simple multiples (A, $-A$) as input, and produces the doubled multiples ($2A$, $-2A$) by wiring that implements the 1-bit shift and extends all the multiples into $(n_A + 1)$ length, most significant bit MSB sign-extended, least significant bit as zero. The outputs are the four multiple words, interleaved by the bit position for the following mux bit slices. The wiring as a whole has a linear latency on the operand word length, but in the complete design, this latency is intertwined into the timing of the bit slices of the following blocks (that is, hidden). The square-law area of this network dominates the area of the complete multiplier.

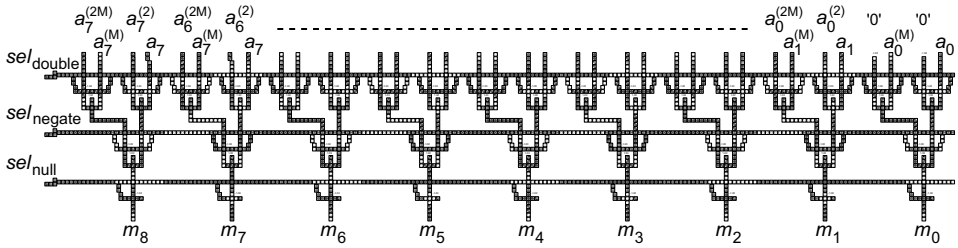


Fig. 27. *Multiple selection mux, QCA layout with 9-bit operands.*

Multiple Selection Mux

The mux block with (n_A+1) slices with four data bit inputs on each, shown in Fig. 27, selects the correct multiple from the input words (A, $-A$, $2A$, $-2A$) or produces the zero-multiple, based on the three control signals provided by the multiplier recoder (sel_{double}, sel_{negate}, sel_{null}). The critical path of the block is formed by the select signal wires, having linear latency on the operand length, but again, this latency in hidden in the complete pipelined multiplier.

Carry-Save Adder

The sequentially operating (n_A+1)-bit carry-save adder, shown in Fig. 28, accumulates the multiples ($+A$, $+2A$, $-A$, $-2A$, or 0) received from the mux block, shifting the stored partial product two digits to the right on each clock cycle. The accumulated value is internally expressed in carry-save format, having two bits for each digit position, and the summation step effectively combines three binary operands (new multiple M, sum vector S, and carry vector C) into the new value. The benefit of this adder type is the ability to compute the whole (n_A+1)-bit accumulation in one clock cycle, matching the input data rate and enabling the continuous operation of the pipeline through the whole multiplier unit. A ripple-carry adder on QCA would have a linear delay, and the best lookahead and conditional sum adders logarithmic delay on the operand word length, stalling the pipeline completely between each addition, because of the dependence on the feedback value.

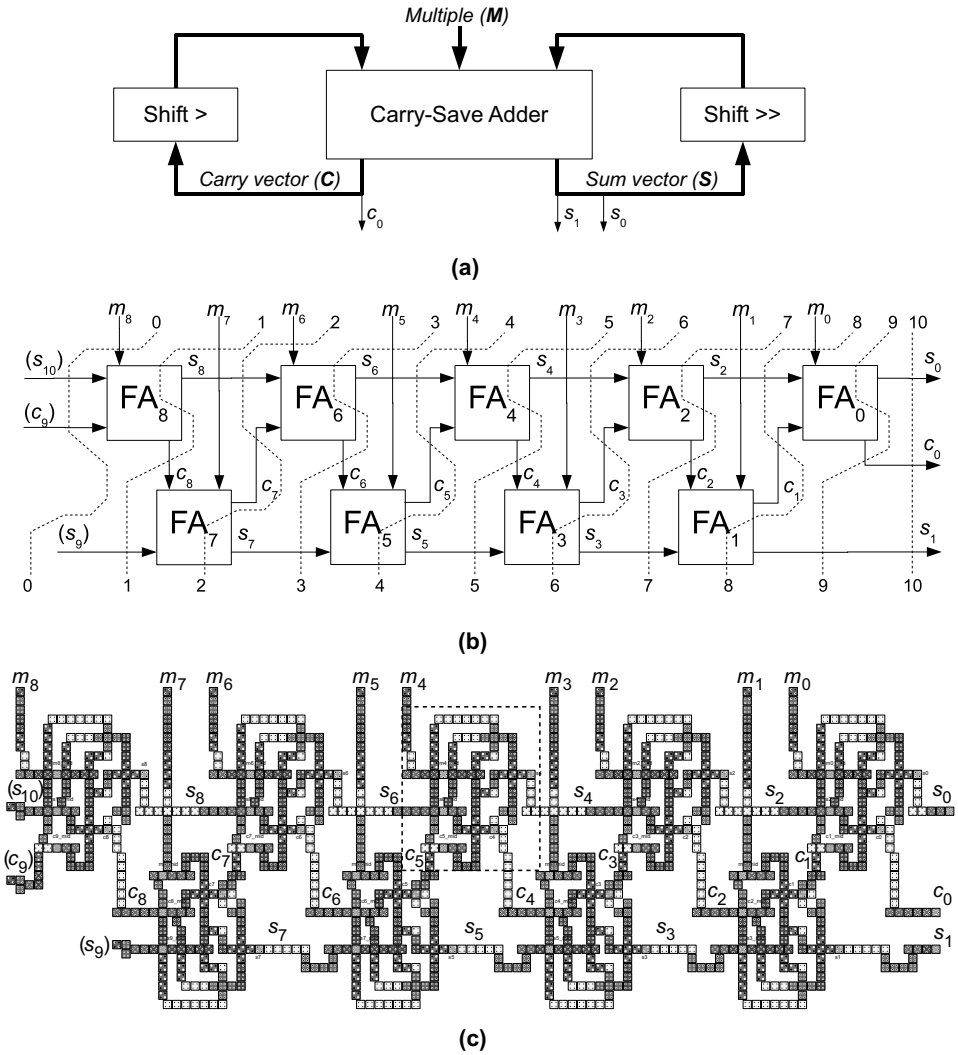


Fig. 28. *Sequential carry-save adder with shifting: a) conceptual structure, b) logic with full adder (FA) components, with the dotted lines denoting the pipeline stages (full clock cycles, not showing the inherent registers), and c) QCA layout with 9-bit operand.*

The carry-save adder consists of two-digit wide slices that compute with different co-existing data sets in parallel, to obtain the continuous data flow when using the QCA full adder blocks described in Sec. 3.2 as (3,2)-counters. The full adder has a carry latency of one clock cycle and sum latency of two cycles (the presented fastest noise rejecting design), which matches the two-bit

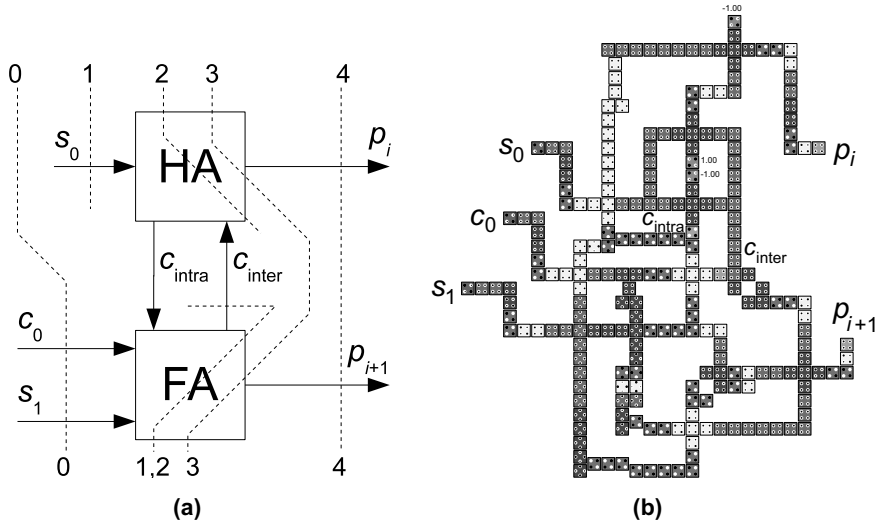


Fig. 29. *Sequential vector merge adder: a) logical structure using half adder (HA) and full adder (FA) components, with the dotted lines denoting the pipeline stages (full clock cycles), and b) optimized QCA layout.*

shifting very well: there is just enough time for the critical path to produce a digit for the next slice, to be used on the next clock cycle. Because of the shifting, a carry bit is moved one digit position to LSB direction, and the sum bit, requiring longer computation time, two digit positions to LSB direction. An accumulation step is spread in time, so that each digit slice is computing with different operand, on the same clock cycle.

Vector Merge Adder

The partial product received from the LSB end of the carry-save adder is in carry-save format (radix-4 digits), which has to be processed into standard non-redundant binary number (radix-2). The vector merge adder, shown in Fig. 29, adds the sequential stream of sum and carry vector bits with proper weights, propagating the carries from the LSB side towards MSB side. On each clock cycle, two bits from the sum vector (s_0, s_1) and one bit from the carry vector (only c_0, since c_{-1} does not exist) is transformed into two bits of the final result (p_i, p_{i+1}), and fed into the result shift register.

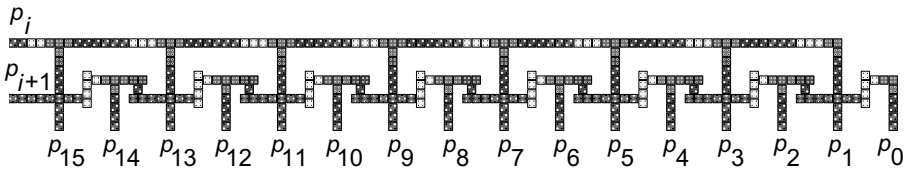


Fig. 30. *Result shift register, 16-bit QCA layout.*

The vector merge adder consists logically of a half adder (HA) and a full adder (FA) component, but the existing QCA designs for these blocks could not be utilized, since their combination would have a total latency of two clock cycles, stalling the pipeline. To match the data rate of the pipeline and solve the carry dependence between two bit positions (since two radix-4 digits are summed each cycle), a layout-level optimization of the combination of half and full adder was designed, to obtain bi-directional carry exchange during one clock cycle. This was possible without breaking the clocking zone rules, which would not be the case when combining two full adders (the related signal dynamics were described in Sec. 2.3 and Sec. 2.4).

Result Shift Register

The two neighboring result bits (p_i, p_{i+1}), arriving together from the vector merge adder, are interleaved by the 2-bit shift register, shown in Fig. 30. This is a simple serial-to-parallel conversion, producing the parallel result word P.

4.5 Verification

The logical correctness of the multiplier designs was checked with paper-and-pencil analysis, while the handcrafted layouts were simulated with the QCADesigner tool. The modifications during the iterative construction of the layouts included changing the number of cells used in the wiring, adjusting the locations of the wire crossings and the logic gates, and experimenting with various clocking zone arrangements and spacings between independent circuit sections, separated by the clocking zones.

The layout simulation with the QCADesigner utilized the coherence vector engine [119–123]. The simulation proceeds in time-dependent manner, which is able to reveal the circuit sections that are susceptible to noise coupling and amplification, giving early prediction of signal race conditions. This heavy model was necessary, because most other approaches are based on time-independent approximations, which completely fail to predict the noise paths.

The simulated layouts were based on a QCA cell sized 18×18 nm, with 4 quantum dots each having a diameter of 5 nm, and the distance between the center of cells was 20 nm (corresponding to a semiconductor technology, enabling comparison with the previous design proposals). The parameters used in QCADesigner version 2.0.3 coherence vector engine were the defaults: temperature 1 K, relaxation time 1 fs, time step 0.1 fs, clock high 9.8×10^{-22} J, clock low 3.8×10^{-23} J, clock shift 0, clock amplitude factor 2, radius of effect 80 nm, relative permittivity 12.9, layer separation 11.5 nm, Euler method, and randomized simulation order. The QCA clock frequency was varied by changing the number of stimulus vectors and the total simulation time.

Serial-Parallel and Array Multiplier. The multiplier cells were simulated exhaustively, covering all the pipeline states, 3-bit multipliers with all possible input combinations, and larger n-bit units with various operand sizes and representative input cases, since the exponentially growing number of states prevents exhaustive runs. Correct multiplication results were obtained with extremely high simulated clock frequencies (several thousands of gigahertz), indicating that unwanted signal coupling is controlled throughout the circuits, and the designs are robust with all operand word lengths.

Radix-4 Multiplier. The fixed size modules and the bit slices of larger blocks were simulated exhaustively, covering all the internal pipeline states, and the combined blocks were tested with a number of representative cases, exhaustive runs prevented by the exponential number of systems states. Correct functionality was obtained with extremely high simulated clock frequencies (up to one thousand gigahertz), indicating that unwanted signal coupling is controlled throughout the circuit, and the design is robust.

Table 6. *Compared multiplier designs.*

Proposed Radix-4 Multiplier
Serial-Parallel Multiplier [202]
Optimized Serial-Parallel Multiplier [12, 205]
Proposed Array multiplier

4.6 Design Analysis

The basic performance metrics (*latency*, *throughput*) and the traditional cost metrics (*design complexity*, *circuit area*) of the multipliers are considered in the following sections. The novel designs are analyzed and compared with the corresponding cases reported in the literature, specified in Table 6. (The reliability and power characteristics are treated in Ch. 5 and Ch. 6.)

The main characterizing parameter of the different multiplier designs is the degree of computation parallelism (in the general sense including pipelining and co-existing data sets). The serial-parallel multiplier accumulates the partial products with time-shared hardware, fully parallel in respect to the multiplicand operand, bit-serial in respect to the multiplier operand, while the array multiplier accumulates the partial products with dedicated adder rows, fully parallel in respect to both operands. The radix-4 recoded multiplier is situated between the two previous proposals. In the following treatment, the word length of both operands is n bits, leading to a $2n$-bit multiplication result. The degree of parallelism translates directly to the performance and cost metrics, compared in the next sections and summarized in Table 7 on p. 72.

4.6.1 Circuit Area

The circuit area of the compared multiplier units grows with a square-law dependence on the operand word length, expressed in Fig. 31 as the relative size of the rectangular shape in comparison with the area of a single QCA cell. The smallest design is the serial-parallel multiplier, while the radix-4 multiplier is about five times as large, on each corresponding operand length.

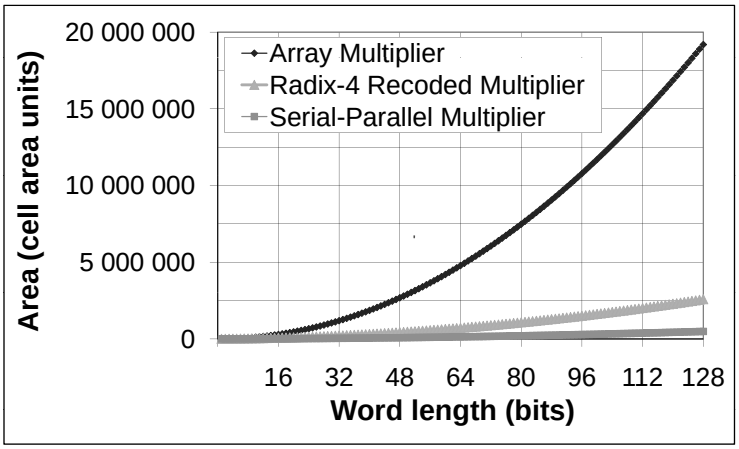
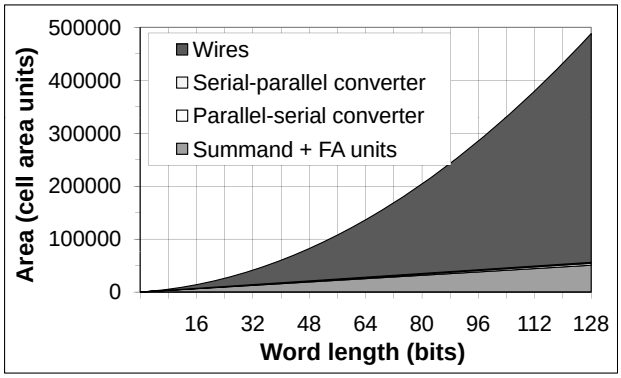


Fig. 31. *Circuit area of the multiplier designs, expressed in equivalent QCA cell area units, for various operand lengths*

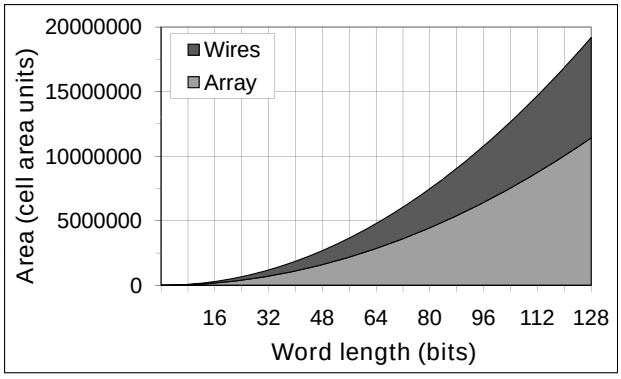
The largest design is the massive array multiplier, reaching over 40 times the size of the previous bit-serial design. The area ratios settle to these constants asymptotically, with the growing operand word length.

The previous proposals of the serial-parallel multiplier [12,202,205] promised linearly increasing area compared to the length of the operand word, but this was accomplished by leaving out the wires distributing the parallel operand across the unit, originating from a compact bus. As this wiring is usually necessary, it is included in this analysis, and the designs have been optimized for latency and area, under the single-layer technology constraint. The contribution of active and passive circuitry for each design is shown in Fig. 32.

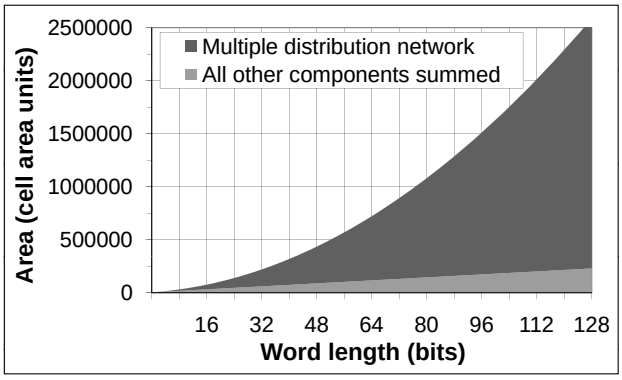
The practical serial-parallel design occupies a square-law area, and the percentage of wiring dominates, from 16-bit to larger units, reaching 90% in the 128-bit case, as shown in Fig. 32(a). Also the array multiplier grows with square-law, but it has equally fast growing terms for both the active area and the wiring, the overhead settling to a constant 40%, as shown in Fig 32(b). The array multiplier grows much faster than the serial-parallel unit (even without the linear advantage): a 16-bit array is 20 times as large as the corresponding serial-parallel unit, and the ratio of the areas settles asymptotically, from 128-bit units upwards, to the array being about 40 times as large.



(a)



(b)



(c)

Fig. 32. *Area of the structures, expressed in equivalent QCA cell area units: a) serial-parallel multiplier, b) array multiplier, and c) radix-4 multiplier.*

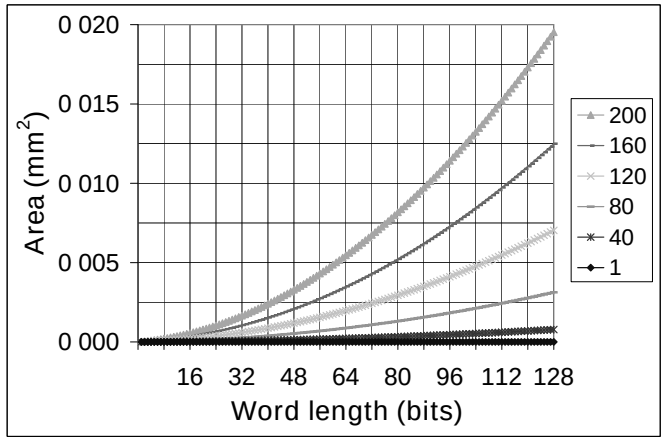
Both of the simple multiplier structures suffer from the need to synchronize the operand words to the pipelined bit slices with extensive delay wiring (square-law), while the active circuitry of the serial-parallel design grows only linearly and the core of the array multiplier with quadruple dependence on the operand length. The active circuitry of the novel radix-4 unit is very small and also limited to linear growth, while the huge multiple distribution network consumes nearly all of the area, as shown in Fig 32(c). The compared designs are heavily area-optimized and will not yield to further layout-level improvements, leaving architectural and algorithmic approaches as the only way to reduce the high wiring overhead.

Figure 33 shows some feature size specific absolute areas (mm^2) for the designs. The described area relations hold for different implementations, but the absolute area has also a square-law growth with the technology specific QCA cell width (nm). The unit cell sizes 1–10 nm correspond to predicted molecular and 10–200 nm to semiconductor QCA technologies.

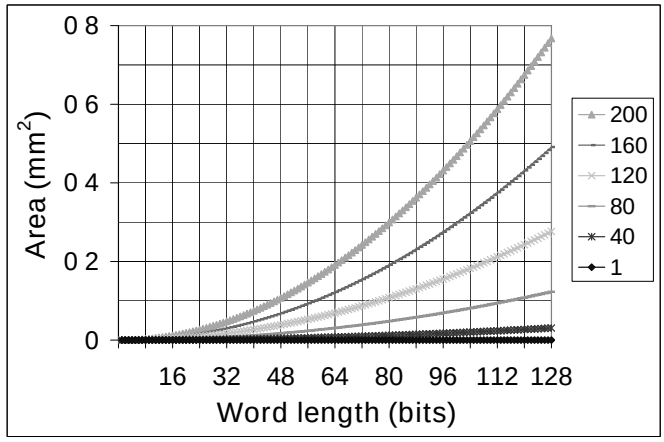
4.6.2 Performance

The latency of the compared multipliers is in the linear regime, with respect to the operand word length. In this sense, the previous serial-parallel multiplier [12, 202, 205] is the fastest unit, having finished the computation in $2n$ clock cycles at best. The radix-4 algorithm offers theoretically halved latency, but the practical delays inherent to the proposed implementation increase the latency to about $2.5n$ cycles. The array multiplier finishes last, having a delay of about $4n$ cycles.

The multiplier designs have tremendous difference in throughput, usually considered the most important performance metric, when there are a lot of back-to-back computations to perform. After the initial latencies are behind and the pipelines are filled, the clear winner is the array multiplier, which produces a new multiplication result on each clock cycle, achieving constant throughput, independent of the operand word length. The radix-4 unit produces a new result once in every n cycles, which is the theoretical maximum



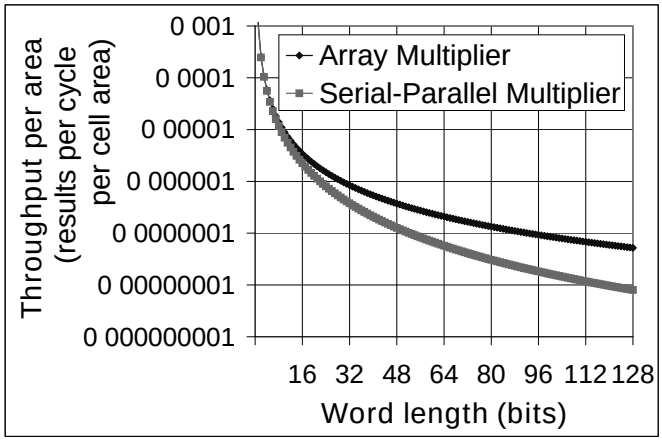
(a)



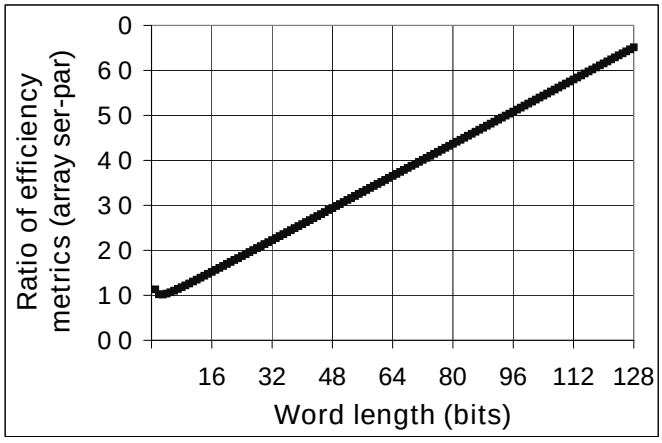
(b)

Fig. 33. *The areas for some implementation technologies, according to cell width in nanometers: a) serial-parallel multiplier and b) array multiplier. Typical CMOS multipliers at 65 nm (for example the designs in [208]) can reach about the same size as the largest 200 nm cell QCA multipliers, while the smaller QCA cell sizes offer area reduction by several orders of magnitude.*

throughput when the partial products are accumulated sequentially. The least amount of results is obtained from the serial-parallel multiplier, finishing a computation once in every $2n$ cycles, producing the result bit-serially, the throughput proportional to the inverse of the result word length.



(a)



(b)

Fig. 34. *The performance-area efficiency of the multipliers: a) throughput per area unit, and b) the ratio of the efficiency metrics (array per serial-parallel).*

An important measure of cost efficiency is how many multiplication results can be obtained per clock cycle, for a unit circuit area invested in the multiplier. The throughput per area metric decreases faster than inversely to the square of the word length, as shown in Fig. 34(a). On very small word lengths (2–4 bit operands) the designs generally have the same efficiency, but with larger operands, the array multiplier is linearly better than the serial-parallel: a 128-bit array produces 6.5 times more results than a serial-parallel unit, per

invested clock cycle and unit area, as shown in Fig. 34(b). The radix-4 multiplier efficiency is in the same regime as the serial-parallel design.

4.6.3 Architectural Aspects

The basic serial-parallel multiplier and the array multiplier are structurally very simple, since they are situated at the extreme ends on the axis of parallel computation, in the general sense, including pipelining and co-existing data sets. On QCA, the array multiplier can be considered having the highest degree of parallelism, since the inherent pipelining and distance translating into delay make the construction of a traditional fully parallel tree multiplier unpractical [197].

In the proposed radix-4 design, selecting a compromise in the degree of parallelism has a cost in the structural complexity: new hardware blocks are needed for recoding the multiplier, generating and distributing several multiples, and accumulating the partial products with low latency carry-save addition and vector merging. The general overhead, nearly all in the multiple distribution wiring, increases the area over the previous sequential design, but the expected doubled throughput is achieved. The radix-4 design and also the array multiplier utilize the underlying full adders with 100% efficiency, while the serial-parallel design can feed the bit slices with new operands only in about 75% of the clock cycles.

The radix-4 recoded multiplier has several properties, that can be extended to obtain variable degrees of parallel computation and tolerance against malfunctioning low-level hardware with physical manufacturing defects and runtime faults. Instead of recoding only one digit of the multiplier per cycle, all of them could be obtained at once, and a tree of carry-save adders used to sum several multiples into the partial product in parallel. Sequential operation with several adders offers the option of module level redundancy to increase the reliability, requiring a runtime control mechanism to be developed. The possibilities, including extension into radix-8, have a huge design space to be explored, but there are already some pointers available for the direction.

Table 7. *Comparison of the QCA multipliers, on word length n operands.*

Design	**Latency** (cycles)	**Throughput** (results per cycle)	**Area** (cells)
Radix-4 multiplier	$2.5n+16$	$1/n$	$140n^2$
Serial-parallel multiplier [202], optimized in [12, 205]	$3n+2$ $2n$	$1/(2n)$	$26n^2$
Array multiplier	$4n-1$	1	$1100n^2$

4.7 Summary

Novel multiplier units were described at the logic, the pipeline, and the QCA layout level, and verified with quantum mechanical simulation. The proposed multipliers achieve tolerance against the technology-inherent noise coupling, while performing at least as well as the previous design proposals for QCA. High area-efficiency is achieved by the dense layouts, which can be used as building blocks in the construction of larger designs. Novel binary multiplier units for QCA technology:

- Serial-parallel multiplier, with noise robustness.
- Pipelined array multiplier, with noise robustness and highest performance-area efficiency.
- Radix-4 multiplier, with noise robustness and customizable degree of parallelism (the most flexible algorithm on QCA, thus far).

As with the adders, the basic serial and pipelined multipliers typically have the same latency, while the throughput is strongly dependent on the degree of parallelism (generally, with pipelining and co-existing data sets). Passive wiring overhead dominates the circuit area, with square-law dependence on the operand word length, in all of the designs. Increasing the algorithmic complexity improves throughput and offers new degrees of freedom for design exploration, not available for the simple extremities of parallelism.

5. RELIABILITY ANALYSIS

The reliability of large QCA designs has not been analyzed before, leaving the technology requirements and the significance of architectural decisions unknown. Next, the reliability levels of two complete arithmetic units are established via probabilistic analysis, hierarchically constructing the total failure rate from the failure rates of the underlying components. It is shown, that in typical arithmetic units, the macro-level component (bit stage) reliability has about linear effect on total reliability, while the component types have contribution weights set by the operand word length. Passive wiring overhead dominates also the reliability, and a multi-level redundancy scheme appears as a necessary requirement, for tolerating very high primitive component failing rates. However, complete fault-tolerant design methodology is out of the scope of this treatment. This work was reported earlier in [P4] and [P6].

Chapter contents. The previous work on QCA reliability is summarized in Sec. 5.1 and the theory of probabilistic transfer matrices presented in Sec. 5.2. The application of the method and the analysis results on the novel pipelined ripple carry adder are described in Sec. 5.3, while Sec. 5.4 presents the application of the method and the analysis results on the novel array multiplier. The conclusion of the chapter follows in Sec. 5.5.

5.1 Previous Work

The nanotechnologies are very sensitive to manufacturing defects and runtime faults, which are always present when matter is manipulated into very fine-grained structures, as becomes apparent also from the detailed challenges de-

scribed on the International Technology Roadmap for Semiconductors (ITRS) [3]. There has been some research into countering these issues at the level of QCA technology, but there is no complete fault-tolerant design methodology. The primitive devices have been studied, but it has been unknown, how reliable the elements should be, to enable functioning larger constructs. The proposals to build arithmetic and even larger circuits do not consider this, and the key reliability design problem is not yet adequately solved. The contribution here concentrates on the analysis side, but for completeness, also the efforts on QCA specific design work are summarized. In the scope exceeding QCA, the general fault-tolerant techniques have been developed over a long period of time, and an overview of the schemes is given in [209].

One of the first reliability studies on QCA proposed block gates to obtain fault tolerance [30, 31], and the run-time error behavior of a metal-dot technology shift-register, implemented in laboratory environment, was analyzed in [9] and revisited in [82]. Thermodynamic stability was inspected with a statistical mechanical model in [52], and triple modular redundancy with shifted operands was proposed as a circuit improvement in [196]. The scaling properties of the QCA primitives were inspected in [183, 184], and fault simulations shown in [39, 188]. A promising approach to combat the faults with a tile-based design approach was recently analyzed in [32, 33, 35], and the sensitivity of sequential structures in [163, 164]. Robustness of the coplanar wire crossing, including alternative implementations, was inspected in [40, 41].

The severe noise path problem related to the imperfect radius of interaction of the practical cellular automata was inspected in [50], and the resulting phenomena were explained in [10]. This issue has tremendous effect on the cost and performance of arithmetic units, which have to be designed to deal with it at very low level. Failure to address this renders most of the previous proposals of QCA implementations as unreliable, and the only previous designs solving the problem pay excessive penalty for the robustness [10, 48].

The key reliability problem is not yet adequately solved, and there is no methodology to design a complete robust system. However, recently the requirements that a full adder (FA) unit sets on the underlying devices was

analyzed in [15, 210]. The largest acceptable failure rates were established via probabilistic transfer matrices of the primitive components (logic gates and short wire segments), giving a concrete goal for the developers of the physical and gate-level QCA nanotechnology implementations of single bit addition [15, 210].

This novel work extends the probabilistic analysis to complete multi-bit arithmetic units, the pipelined ripple carry adder and the pipelined array multiplier. The contribution of the passive wiring and active full adders to the overall reliability are compared, including the operand word length effects. The results are used in conjunction with the previous study, to transform the architectural requirements into the failure rates of the low-level circuit primitives.

5.2 Probabilistic Transfer Matrices

The reliability of a combinatorial circuit can be computed accurately, if the structure of the circuit and the error rates of the components are known. Each component has it's own probabilistic transfer matrix (PTM), which states all the probabilities of an input combination leading to an output combination. The individual matrices can be combined into the PTM of a complete circuit: the common PTM of parallel components is formed from the smaller PTMs via Kronecker (alternatively tensor) products (here denoted by $\otimes$), and the common PTM of components in series is formed from the smaller PTMs with matrix-matrix products. The approach can be applied at various abstraction and hierarchy levels, and there is no need to have actual physical parameters included, although they can be utilized in low-level analysis. [211–213]

The produced collective matrix describes the probabilistic relations between each distinct input combination and each distinct output combination of the complete system, and this data structure has to be further processed to determine the desired reliability parameters. In the case of the simple *total reliability*, this is achieved as follows: the reliability of a circuit is found by multiplying the PTM element-wise with a corresponding ideal transfer ma-

trix (ITM), with zero failure probabilities, thus setting to zero all the elements representing erroneous input-output cases. The resulting matrix is multiplied left-wise by a vector containing the probabilities of each input case to the circuit, producing a vector describing the probability of occurence of each correct input-output case, with the specified input probability distribution. The total reliability is simply the sum of the vector elements. [211–213]

5.3 Pipelined Ripple Carry Adder

The reliability of the binary multi-bit addition is established, when executed on the standard ripple carry adder (RCA) structure presented earlier. The contributions of passive wiring and active computing hardware are compared, including the word length effects.

5.3.1 Probabilistic Formulation

The probability transfer matrix approach can be applied to the QCA ripple carry adder, which in principle is a purely combinatorial structure, although the QCA implementation leads to pipelining at very fine-grained level. The design has three types of components in this probabilistic analysis, as shown in Fig. 35: full adders, input wire blocks, and output wire blocks. The component probabilistic transfer matrices are shown in Fig. 36. An n-bit RCA has n rows, each consisting of a full adder and $(n-1)$ wire blocks, and the PTMs are first constructed for each row of the parallel components using the Kronecker product, for the 4-bit unit:

$$\begin{aligned} P_1 &= P_{IW} \otimes P_{IW} \otimes P_{IW} \otimes P_{FA} \; ; \\ P_2 &= P_{IW} \otimes P_{IW} \otimes P_{FA} \otimes P_{OW} \; ; \\ P_3 &= P_{IW} \otimes P_{FA} \otimes P_{OW} \otimes P_{OW} \; ; \\ P_4 &= P_{FA} \otimes P_{OW} \otimes P_{OW} \otimes P_{OW} \end{aligned} \tag{4}$$

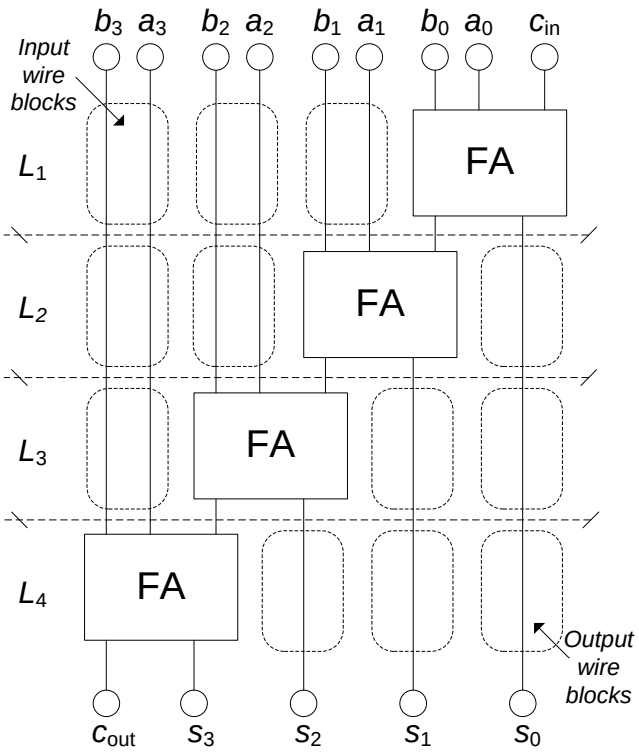


Fig. 35. *Ripple carry adder, dependencies of the components following the logic/layout composition with 4-bit operand length.*

where P_{FA}, P_{IW}, and P_{OW} are the PTMs of full adder, input wire block, and output wire block, respectively. The row PTMs in series are combined into the PTM of the complete RCA unit using the matrix product:

$$P_{RCA} = P_1 P_2 P_3 P_4. \tag{5}$$

An ideal transfer matrix ($P_{RCA,ideal}$) of the RCA is constructed similarly, from ideal component matrices with failure rate $p = 0$. The probability transfer matrix and the ideal transfer matrix are multiplied element-wise ($\star$) to remove the reliability mass corresponding to failures, and the resulting matrix is then

Input cases (b,a,c):	Output cases (c_{out},s): 00	01	10	11
000	$1-p$	$p/3$	$p/3$	$p/3$
001	$p/3$	$1-p$	$p/3$	$p/3$
010	$p/3$	$1-p$	$p/3$	$p/3$
011	$p/3$	$p/3$	$1-p$	$p/3$
100	$p/3$	$1-p$	$p/3$	$p/3$
101	$p/3$	$p/3$	$1-p$	$p/3$
110	$p/3$	$p/3$	$1-p$	$p/3$
111	$p/3$	$p/3$	$p/3$	$1-p$

$= P_{FA}$

(a)

Input cases (b,a):	Output cases (b,a): 00	01	10	11
00	$1-p$	$p/3$	$p/3$	$p/3$
01	$p/3$	$1-p$	$p/3$	$p/3$
10	$p/3$	$p/3$	$1-p$	$p/3$
11	$p/3$	$p/3$	$p/3$	$1-p$

$= P_{IW}$

(b)

Input cases (s):	Output cases (s): 0	1
0	$1-p$	p
1	p	$1-p$

$= P_{OW}$

(c)

Fig. 36. *Probabilistic transfer matrices of the components, with error probability p: a) full adder (P_{FA}), b) input wire block (P_{IW}), and c) output wire block (P_{OW}).*

left-multiplied with a vector $v = [1/2^{2n+1}, \ldots, 1/2^{2n+1}]$ containing the equal probabilities of each input case. The total reliability is the sum of the elements of the resulting vector:

$$R = \sum (v(P_{RCA} \star P_{RCA,ideal})). \tag{6}$$

5.3.2 Reliability Approximation

The PTM approach is computationally complex since the matrix dimensions grow rapidly with the number of inputs and outputs of a stage: a j-input k-output PTM has size $[2^j \times 2^k]$, having 2^{j+k} elements. This results in exponential number of real multiplications and multiply-accumulate (MAC) op-

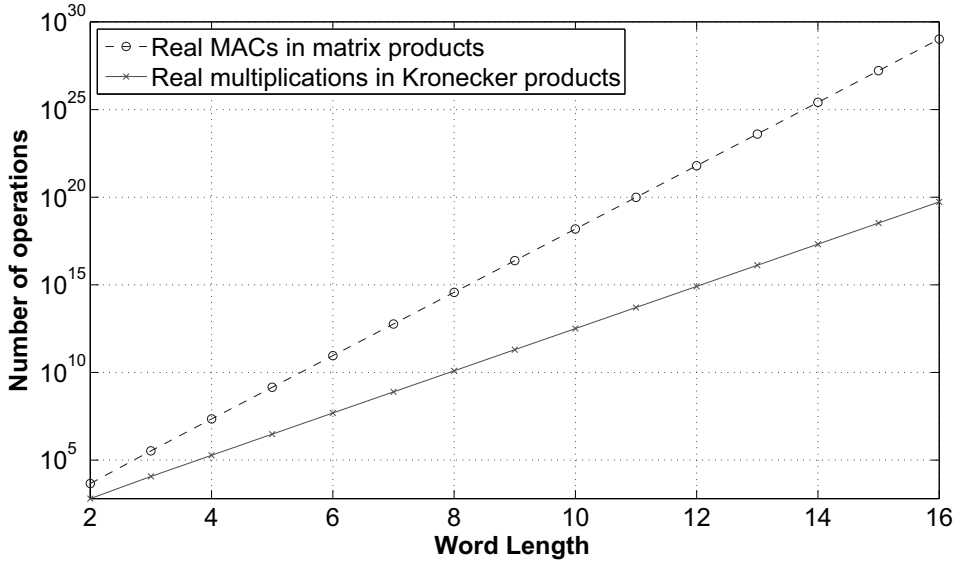


Fig. 37. *Complexity of forming the exact probability matrix of the complete RCA.*

erations in computing the total reliability of an RCA, as shown in Fig. 37, although the analysis of a single full or serial adder alone would be only a constant complexity operation with relatively small matrices [15, 210]. Fast computation on a personal computer is possible only up to the operand word length of six bits, where the largest matrix has $2^{13+12} \approx 33.6 * 10^6$ elements. Forming the RCA matrices takes five minutes for a component reliability, with numerical Matlab on a Pentium M 2.13GHz, 2 GB of RAM, WinXP SP2.

The exact PTM computation was run on word lengths $n = 1 \ldots 6$, to obtain a large set of data on various component failure rates, evenly stepped in the range $0 \ldots 10^{-3}$, simplified by using the same error probability p_W for both of the wire block types (although it should be noted that this causes a minor bias in the resulting data set). The data was then used to construct a second degree polynomial approximation of the RCA reliability, given in Table 8.

The approximation matches very well the small word length data, the deviation from the exact computation being around 0.001%, and based on the trend of the prediction error having quadruple increase with the operand word length, a 128-bit adder reliability is overestimated less than 4%, as long as the

Table 8. *RCA reliability approximation.*

Total reliability $R = a + b + 1$	
Component	**Contribution**
Full adder	$a = (-0.98n - 0.04)p_{FA}$
Wire block	$b = (-0.89n^2 + 0.80n + 0.11)p_W$

component failure rate does not exceed 10^{-4}, and the peak overestimation of 8% is reached around component failure rate 3×10^{-4}. Since the model is tuned for the medium range of component error rates, from there on, larger component failure rates tend towards the opposite prediction error, growing rapidly with the operand word: a 128-bit adder reliability is underestimated at worst by 45%, as long as the component failure rate does not exceed 10^{-3}, which can be considered a justified assumption of an acceptable worst case limit for digital QCA computing [15]. After this point, the pessimistic trend continues even more steeply, and the model cannot be used for reliability prediction anymore. This model is technology independent, but the chosen component hierarchy sets the parameter interpretation, as defined in Sec. 5.3.1.

5.3.3 Analysis Results

The total reliability R of a fixed operand length RCA depends linearly on the failure rate of the full adder block p_{FA} and the failure rate of the wire block p_W. However, the relative weights of the two factors are heavily affected by the operand word length n of the unit, as shown in Table 8: the full adder has a weight determined by a linear function of the word length, while the wire block has a weight of a quadratic dependence on the word length. Thus the wiring dominates the reliability from 3-bit to larger units, if the component failure rates are same for the different block types. This is consistent with the complete RCA circuit area proportions of the blocks, linear vs. quadratic behavior with the operand word length.

Table 9. *RCA 99% level reliability requirements for the component blocks.*

Word length	**Component failure rate**		
	Failing FAs, ideal wires (best case)	Ideal FAs, failing wires (theoretical)	Failing FAs, failing wires (worst case)
1	1.00×10^{-2}	-	-
2	4.90×10^{-3}	5.37×10^{-3}	2.51×10^{-3}
3	3.16×10^{-3}	1.78×10^{-3}	1.14×10^{-3}
4	2.39×10^{-3}	9.12×10^{-4}	6.61×10^{-4}
5	2.00×10^{-3}	5.37×10^{-4}	4.27×10^{-4}
6	1.60×10^{-3}	3.55×10^{-4}	2.82×10^{-4}
8	1.27×10^{-3}	1.98×10^{-4}	1.71×10^{-4}
16	6.36×10^{-4}	4.65×10^{-5}	4.34×10^{-5}
32	3.18×10^{-4}	1.13×10^{-5}	1.09×10^{-5}
64	1.59×10^{-4}	2.78×10^{-6}	2.73×10^{-6}
128	7.97×10^{-5}	6.91×10^{-7}	6.85×10^{-7}

On the other hand, to achieve certain total reliability level of a complete system, the *required* reliability of a component depends on where this component is used, and how many instances there are. A system of four full adders and wiring blocks achieves a desired total reliability level with fairly modest failure rate requirements for the basic blocks, but a system of 128 full adders will not reach the same total reliability level, if the underlying block failure rate of the smaller unit is allowed. Thus, a large arithmetic unit requires much better components than a small unit.

Table 9 shows the component failure rates required to get a 99% total reliability for the RCA. In the assumed best case, the wire blocks are nearly ideal, leaving the full adders as the only failing components (column 2); a purely theoretical case has ideal full adders and failing wires (column 3). In the worst case, the wire blocks are as likely to fail as the full adders (column 4). Multi-bit addition has the reliability dominated by the wire blocks, while the full adders have one or two orders of magnitude smaller contribution. Table 10 shows a similar trend for a 99.999% total reliability level.

Table 10. *RCA 99.999% level reliability requirements for the component blocks.*

Word length	**Component failure rate**		
	Failing FAs, ideal wires (best case)	Ideal FAs, failing wires (theoretical)	Failing FAs, failing wires (worst case)
1	1.00×10^{-5}	-	-
2	4.90×10^{-6}	5.37×10^{-6}	2.51×10^{-6}
3	3.16×10^{-6}	1.74×10^{-6}	1.12×10^{-6}
4	2.40×10^{-6}	9.12×10^{-7}	6.61×10^{-7}
5	2.00×10^{-6}	5.25×10^{-7}	4.17×10^{-7}
6	1.58×10^{-6}	3.55×10^{-7}	2.82×10^{-7}
8	1.27×10^{-6}	1.98×10^{-7}	1.71×10^{-7}
16	6.36×10^{-7}	4.65×10^{-8}	4.34×10^{-8}
32	3.18×10^{-7}	1.13×10^{-8}	1.09×10^{-8}
64	1.59×10^{-7}	2.78×10^{-9}	2.73×10^{-9}
128	7.97×10^{-8}	6.91×10^{-10}	6.85×10^{-10}

The RCA wire blocks are expected to reach very high reliability in practical implementations, since they can be usually hardened by increasing the width of the lines as shown in the layout in Fig. 38 (on QCA, wires of multiple parallel cells). However, the reliability of multi-bit arithmetic is still quite demanding: even with ideal wire blocks, a 64-bit ripple carry adder with a total reliability of 99% requires a full adder with a failure rate smaller than 0.00016, corresponding to a component reliability of 99.98%. This translates into circuit primitive (logic gate and short wire segment) failure rate of about 10^{-6}, which might still be reached on the QCA nanotechnology [15, 210].

The RCA total reliability of 99%, with the full adders and the wire blocks failing with equal rate, requires a component reliability of about 99.9997%. This translates into circuit primitives, which would have to be nearly as reliable as traditional technology components, the failure rate at least 10^{-8} [15, 210]. This might already be out of reach of the fault-abundant future technologies. (The current CMOS transistor failure rates are around 10^{-11} to 10^{-10} [3].)

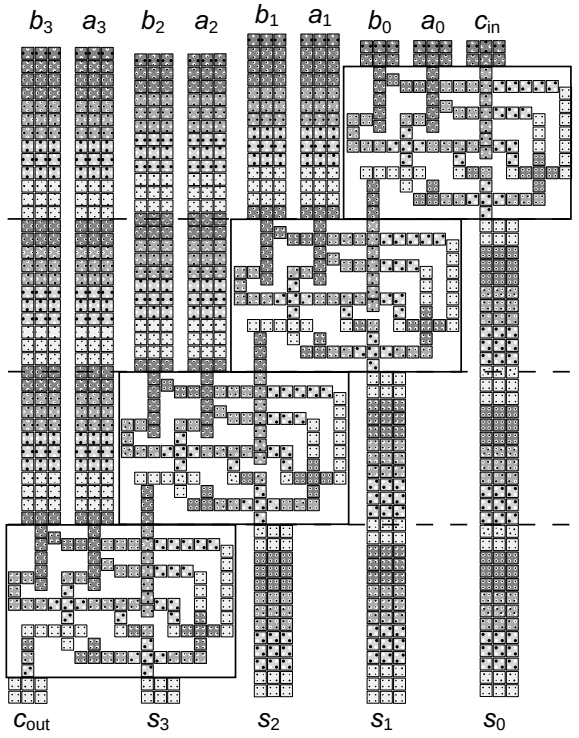


Fig. 38. *Ripple carry adder with hardened wire blocks, three cells wide.*

5.4 Array Multiplier

The reliability of the binary multi-bit multiplication is established, when executed on the massively parallel array multiplier (AM) structure. The previously presented layout level implementation on QCA is analyzed, comparing the contribution of the passive wiring and the active computing hardware.

5.4.1 Probabilistic Formulation

The array multiplier is in principle a combinatorial structure, compliant to the PTM decomposition method. A multiplier cell, having the logical structure shown in Fig. 39(a) and the QCA layout shown in Fig. 39(b), is formed by components of several types, schematically separated in Fig. 40(a): a summand and full adder block (SFA), straight wire blocks (W), fanout wire blocks

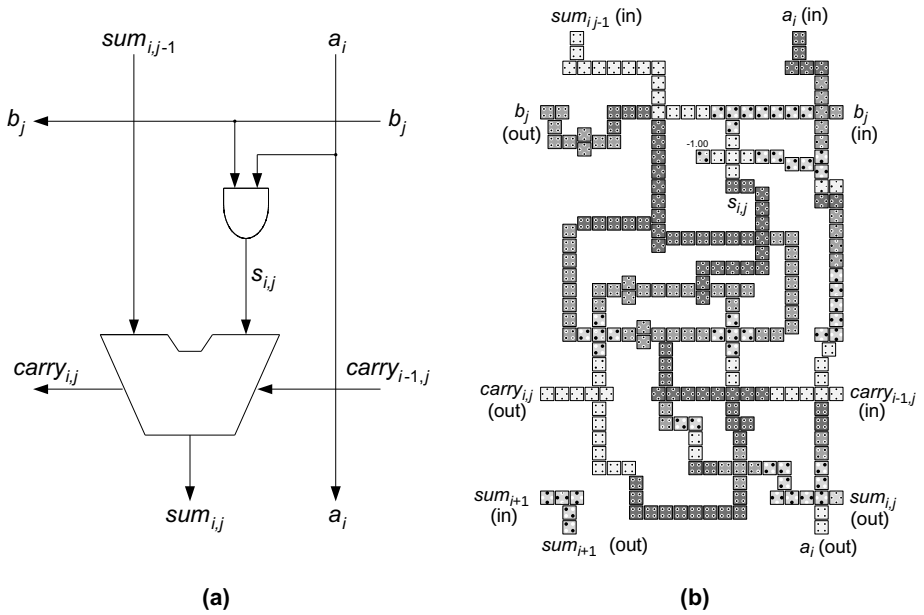


Fig. 39. *Multiplier cell: a) logical structure, and b) QCA layout.*

(F), and crossing wire blocks (X) (in the perimeter cells also crossing and terminate block (XT) and null sources (N)), having the PTMs shown in Fig. 41. The PTM of the general multiplier cell (C_G) is formed according to the dependencies of the blocks in six levels, as shown in Fig. 40(b), with ideal wires I as formal placeholders for inputs not on the lowest level:

$$\begin{aligned} P_0 &= P_I \otimes P_X \otimes P_I \,; \\ P_1 &= P_I \otimes P_F \otimes P_F \otimes P_I \,; \\ P_2 &= P_X \otimes P_I \otimes P_I \otimes P_W \otimes P_I \,; \\ P_3 &= P_W \otimes P_I \otimes P_I \otimes P_I \otimes P_X \,; \\ P_4 &= P_I \otimes P_{SFA} \otimes P_I \,; \\ P_5 &= P_I \otimes P_I \otimes P_X \,; \\ P_{C,G} &= P_0 P_1 P_2 P_3 P_4 P_5 . \end{aligned} \tag{7}$$

The PTM of the whole array is formed by combining the cell PTMs according to the computational order, as shown in Fig. 42 for a 3-bit unit. Nine cell

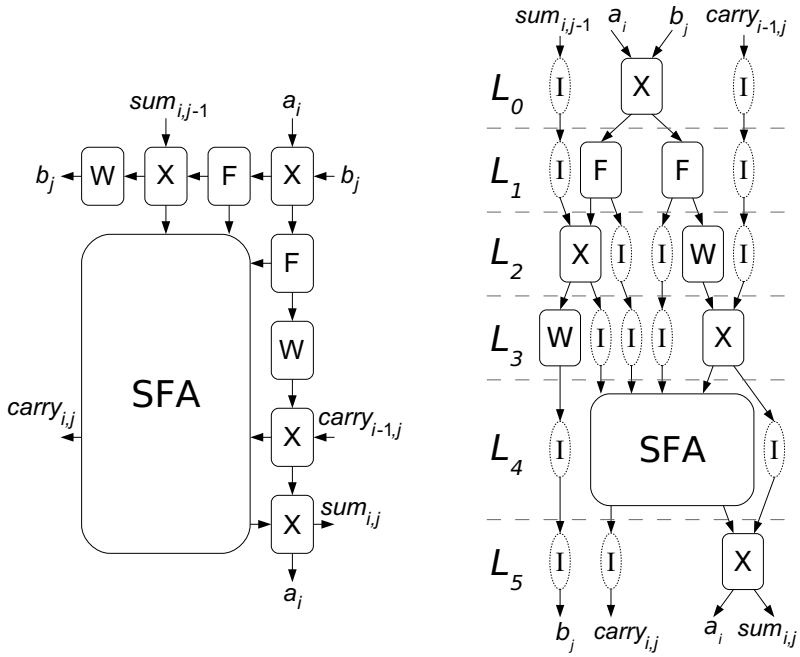


Fig. 40. *Multiplier cell C_G: a) layout components and b) dependency graph. SFA: summand and full adder. W: straight wire. F: fanout. X: wire crossing. I: ideal wire as formal placeholder for inputs not at the lowest level.*

PTMs are needed as the number of inputs and outputs varies according to the position in the array, leading to the following decomposition of P_{AM}:

$$
\begin{aligned}
P_0 &= P_I \otimes P_I \otimes P_{C,TR} \otimes P_I \otimes P_I \; ; \\
P_1 &= P_I \otimes P_{C,T} \otimes P_I \otimes P_I \otimes P_I \otimes P_I \; ; \\
P_2 &= P_{C,TL} \otimes P_I \otimes P_{C,R} \otimes P_I \otimes P_I \; ; \\
P_3 &= P_I \otimes P_I \otimes P_{C,G} \otimes P_I \otimes P_I \otimes P_I \otimes P_I \; ; \\
P_4 &= P_{C,L} \otimes P_I \otimes P_{C,BR} \otimes P_I \otimes P_I \; ; \\
P_5 &= P_I \otimes P_I \otimes P_{C,B} \otimes P_I \otimes P_I \otimes P_I \; ; \\
P_6 &= P_{C,BL} \otimes P_I \otimes P_I \otimes P_I \otimes P_I \; ; \\
P_{AM} &= P_0 P_1 P_2 P_3 P_4 P_5 P_6 . \qquad (8)
\end{aligned}
$$

Input (s_{in}, b, a, c_{in}):	Output (c_{out}, s_{out}): 00	01	10	11
0000	$1-p_A$	$p_A/3$	$p_A/3$	$p_A/3$
0001	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
0010	$1-p_A$	$p_A/3$	$p_A/3$	$p_A/3$
0011	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
0100	$1-p_A$	$p_A/3$	$p_A/3$	$p_A/3$
0101	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
0110	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
0111	$p_A/3$	$p_A/3$	$1-p_A$	$p_A/3$
1000	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
1001	$p_A/3$	$p_A/3$	$1-p_A$	$p_A/3$
1010	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
1011	$p_A/3$	$p_A/3$	$1-p_A$	$p_A/3$
1100	$p_A/3$	$1-p_A$	$p_A/3$	$p_A/3$
1101	$p_A/3$	$p_A/3$	$1-p_A$	$p_A/3$
1110	$p_A/3$	$p_A/3$	$1-p_A$	$p_A/3$
1111	$p_A/3$	$p_A/3$	$p_A/3$	$1-p_A$

$= P_{SFA}$

(a)

Input :	Output : 0	1
0	$1-p_W$	p_W
1	p_W	$1-p_W$

$= P_W$

(b)

Input:	Output (out_1, out_2): 00	01	10	11
0	$1-p_W$	$p_W/3$	$p_W/3$	$p_W/3$
1	$p_W/3$	$p_W/3$	$p_W/3$	$1-p_W$

$= P_F$

(c)

Input (in_1, in_2):	Output (out_1, out_2): 00	01	10	11
00	$1-p_W$	$p_W/3$	$p_W/3$	$p_W/3$
01	$p_W/3$	$p_W/3$	$1-p_W$	$p_W/3$
10	$p_W/3$	$1-p_W$	$p_W/3$	$p_W/3$
11	$p_W/3$	$p_W/3$	$p_W/3$	$1-p_W$

$= P_X$

(d)

Fig. 41. *Probabilistic transfer matrices of the macro components of the multiplier cell, with error probability p_A for the active logic and p_W common for all wiring: a) summand and full adder (P_{SFA}), b) wire block (P_W), c) fanout block (P_F), and d) wire crossing block (P_X).*

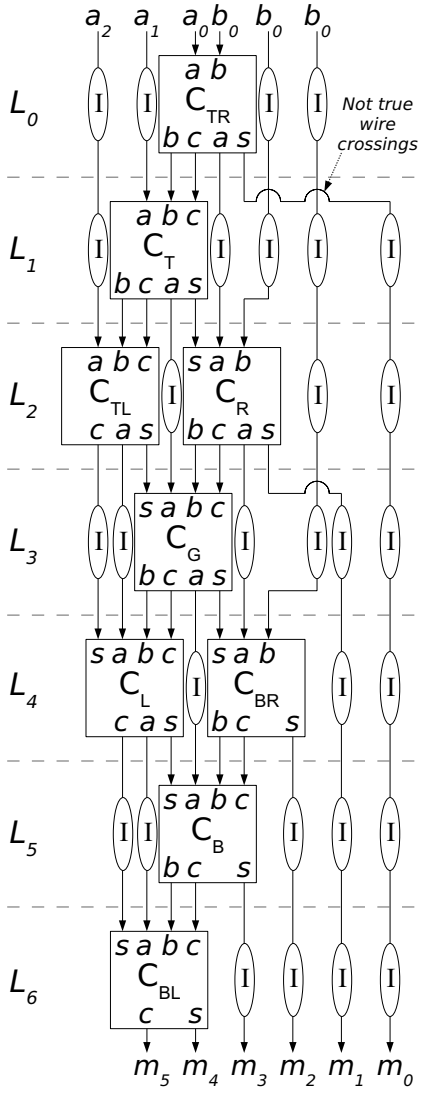


Fig. 42. *Dependency graph of a 3-bit array multiplier. The general array uses nine cell types with varying number of input and output ports, determined by the position in the array. C_{TR}: cell at top-right corner. C_T: cells in the middle of top row. C_{TL}: cell at top-left corner. C_R: cells in the middle of rightmost column. C_G: general cells in the middle. C_L: cells in the middle of leftmost column. C_{BR}: cell at bottom-right corner. C_B: cells in the middle of bottom row. C_{BL}: cell at bottom-left corner. I: ideal wire as formal placeholder for inputs not at the lowest level.*

An ideal transfer matrix $P_{AM,ideal}$ is constructed similarly, from ideal component matrices with failure rate $p_A = p_W = 0$. The P_{AM} and $P_{AM,ideal}$ matrices are multiplied element-wise (denoted by $\star$) to remove the reliability mass corresponding to erroneous outputs, and the resulting matrix is then multiplied left-wise by a vector $v = [1/2^{2n}, \ldots, 1/2^{2n}]$, containing the equal probabilities of each input case. The total reliability is simply the sum of the elements of the resulting vector:

$$R = \sum v(P_{AM} \star P_{AM,ideal}). \tag{9}$$

5.4.2 Analysis Results

The total reliability R of a fixed operand length AM depends nearly linearly on the failure rate of the summand and full adder block p_A and the common failure rate of the various wire blocks p_W, but the wiring has four to five times as much effect as the active block. This is mostly due to the fact that the pass-through wiring accumulates error for all the four output signals of a cell, while the active part affects only two of the signals. The other reason is that in this formulation, a large number of wire blocks is used for every active block.

Figure 43 shows the total reliability of a 3-bit array, with various active and passive component failure rates. In the assumed best case, the wire blocks are nearly ideal: a 99% total reliability level for the AM is reached, if the active logic failure rate p_A is kept below 0.00116 (above reliability of 99.9%). In the worst case, the wire blocks are as likely to fail as the active blocks: a 99% level requires the common failure rate ($p_A = p_W$) to stay below 0.00019 (above reliability of 99.99%). The worst case is defined this way, since the wiring cannot be less reliable than the large active component, when they share the same underlying technology.

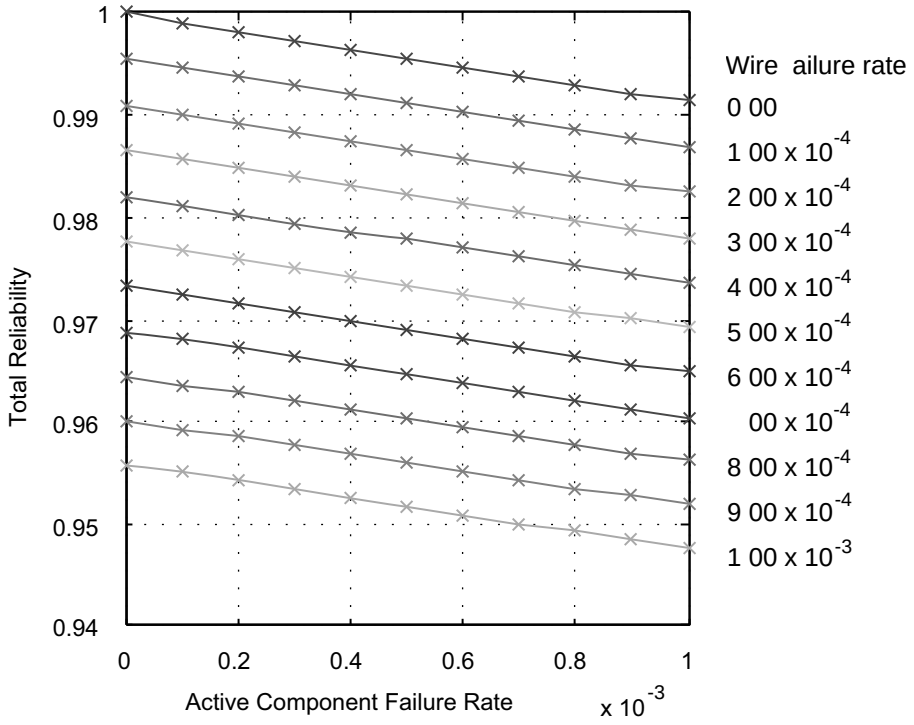


Fig. 43. *Array multiplier: total reliability vs. active component failure rate, each line with a fixed wire failure rate shown on the right.*

The requirements for the active component can be transformed into practical gate level failure rates, where the primitives should have about two decades higher reliability than the larger block [15, 210]: a 99.9% SFA block needs about 99.999% level gate reliability (failure rate of 10^{-5}), which might be quite acceptable. A 99.99% SFA requires gate primitives with 99.9999% reliability (failure rate of 10^{-6}), which might not be reached at the physical design level alone.

5.5 Summary

The reliability of complete arithmetic units was established via probabilistic analysis, hierarchically constructing the total failure rate from the failure rates

of the underlying components. Based on the the pipelined ripple carry adder and the pipelined array multiplier, it was concluded, that in the typical basic structures, the bit slice level component reliability has about linear effect on the total reliability, while the different component types have contribution weights set by the operand word length. Passive wiring overhead dominates strongly also the reliability, as the circuit area.

It was shown, that the total failure rate of a complete arithmetic unit increases orders of magnitude faster than the increasing failure rate of the underlying device-level primitives. Thus, a strong multi-level redundancy scheme is needed for tolerating the defective and fault-abundant implementation technologies. However, a complete fault-tolerant design methodology was left out of the scope of this treatment.

6. POWER ANALYSIS

The power characteristics of large QCA designs have not been analyzed, leaving unknown the maximum operating frequencies and the role of irreversibility dissipation. Next, three complete arithmetic units are analyzed near the ultimate limit of computation efficiency, using the Landauer's principle. It is shown, that information erasure causes power dissipation, which significantly limits the maximum operating frequency of molecular QCA arithmetic units, and thus, reaching the full technology potential requires reversible computing principles to be adopted. This work was reported earlier in [P3] and [P7].

Chapter contents. The previous work on QCA power characteristics is summarized in Sec. 6.1 and the theory of irreversibility dissipation in Sec. 6.2. The analysis of the novel pipelined ripple carry adder is presented in Sec. 6.3, while the analysis of the novel serial-parallel and array multiplier is presented in Sec. 6.4. The conclusion of the chapter follows in Sec. 6.5.

6.1 Previous Work

Previous power analyses concentrated on the energetic characteristics of the QCA nanotechnology and the primitive circuit elements: a quantum mechanical treatment in [17] showed that irreversible bit erasures really dissipate energy on QCA, as implied by the second law of thermodynamics according to Landauer [18], while [214–216] analyzed the energetics with a semi-classical model, suitable for predicting the coarse dissipation of metal-dot implementations. An elegant and straightforward approach to create fully reversible circuits with Bennett clocking was introduced in [16, 56], following the ideas

of Bennett [57], achieving ultra low power dissipation without any additional circuit complexity, while other approaches and trade-offs for reversible QCA design were outlined in [217]. The power dissipation and reversibility of primitive QCA circuits and clocking approaches have been analyzed with a simple mechanical model in [218], and a worst-case estimation model for irreversible operation and abrupt switching with leakage and switching power components was presented in [219].

In this thesis, the power dissipation in larger systems and at architectural level is considered. The analysis is based on the thermodynamic necessity to dissipate energy, when a bit of information is irreversibly erased, and the approach was proven valid for QCA in [17]. The novel study connects the structure of arithmetic units to the inevitable dissipation in the actual circuit layouts, and predicts the limits of performance and operating frequencies. The aim is to establish the key parameters of QCA arithmetic; the bit erasure power dissipation of the previously presented designs is analyzed, demonstrating the inevitable tight limits for the operating frequencies of the units, under irreversible operation. Fully reversible operation of the pipelined ripple carry adder is also outlined.

6.2 Irreversibility Power

The thermal noise floor sets a requirement for a signal to have an energy content of at least $E_{sig} = 100k_BT$ (where k_B is the Boltzmann constant, and T the temperature in Kelvin degrees), to obtain a decent error probability of 3.72×10^{-44} [220]. However, there is no fundamental need to dissipate this energy on every step of computation, like the conventional technologies do; the laws of physics require a dissipation of about 140 times smaller energy, and *only* when a bit of information is lost [18]. On QCA, most of the signal energy can be transferred from cell to cell and re-used, making the unavoidable erasure dissipation a significant factor. This ultimate limit of energy efficiency can be reached, since there is no need to move electron currents and charge circuit capacitances [17].

The power analysis presented here is based on the thermodynamical requirement of heat generation in the active QCA layer, which is expected to dominate the power. The underlying clock network, possibly implemented with CMOS technology, is also a source of dissipation, although a minor one, since it can be charged adiabatically, and the clock generator can be placed completely off-chip, to ease cooling. [16, 56]

The Landauer's principle [18], resulting from a thermodynamic consideration of energy and entropy of the system, declares, that losing a bit of information about the system state leads inevitably to dissipating $E_{dis} = k_B T \ln 2$ of energy into the environment, computed in this thesis at the room temperature. The circuits presented in this study are based on irreversible logic, losing information at each step of computation, and their lowest power limits are set by this law of nature, when operated with the normal Landauer-type clocking [6, 53].

6.3 Pipelined Ripple Carry Adder

The power characteristics of the standard ripple carry adder are analyzed in respect to the ultimate physical limit of computation, the Landauer's principle. Also, fully reversible operation is described, based on the Bennett-type clocking approach.

6.3.1 Bit Erasure Energy

The ripple carry adder discards information in each functional unit, a full adder, at each clock cycle. A full adder, having the truth table shown in Table 11, compresses eight distinct input combinations into four different output values in quite an unbalanced way. Three input cases are mapped onto a single output case twice, leading to the loss of two bits of system state. This is very strong state compression, causing considerable power dissipation, as the total number of bit erasures in n-bit addition is directly, and the energy efficiency inversely, proportional to the operand word length, as shown in Fig. 44.

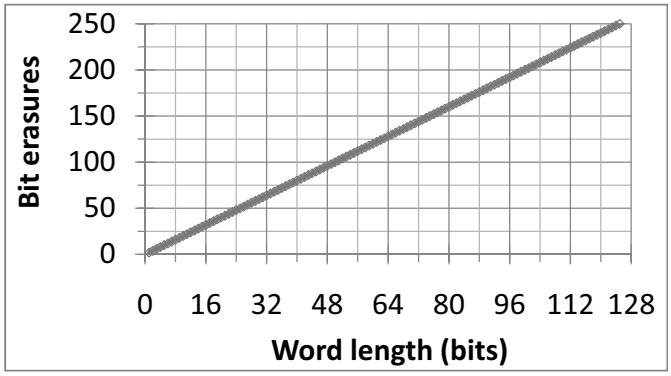
Table 11. *Truth table of the full adder, showing the unbalanced eight-to-four input-output mapping, resulting in a strong state compression.*

Mapping	Inputs			Outputs	
	a	b	c_{in}	c_{out}	s
One-to-one	0	0	0	0	0
Three-to-one	0	0	1	0	1
	0	1	0		
	1	0	0		
Three-to-one	0	1	1	1	0
	1	0	1		
	1	1	0		
One-to-one	1	1	1	1	1

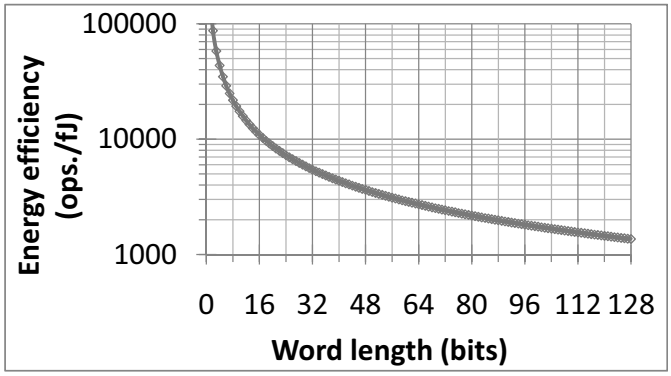
6.3.2 Power Density

The power density is determined by the place and computation time of the bit erasures, in the case of the full and serial adders, simply located in the same part of the circuit on consequent clock cycles. In the pipelined ripple carry adder, a single n-bit addition is spread on n different full adders, operating on consequent cycles, but since the structure computes n additions with the different data sets co-existing in the pipeline, the power density of the active region (the full adders) is exactly the same as in the other units, and not dependent on the operand word length. The minimum reachable *hot spot* power is very high, as quite a lot of information is lost on small circuit area. Resulting implementation specific power densities for various molecular QCA feature sizes are shown in Fig. 45(a). (Surprisingly, multiplier units could achieve much lower power densities: less information is lost, since the input operands can be made available for reversing the computation, as a free by-product of the multiplier organization. This is considered in the following Sec. 6.4.)

Efficient spreading of the heat could tremendously alleviate the power problem in the ripple carry adder. If the heat could be distributed all over the structure, growing with a square-law, the energy on area unit diminishes inversely to the square of the operand word length, as shown in Fig. 45(b). However, since the heat transfer characteristics of QCA implementations are still unknown, the discussion here is limited to the hot spot power.



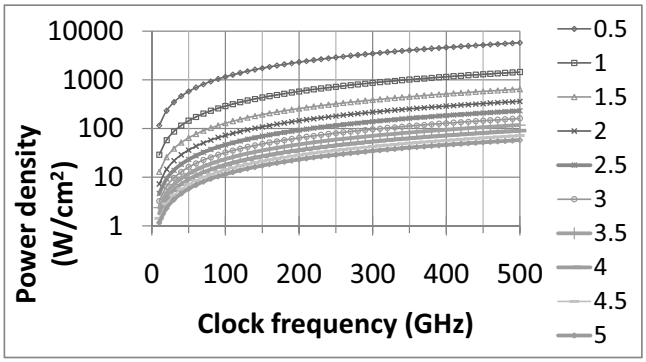
(a)



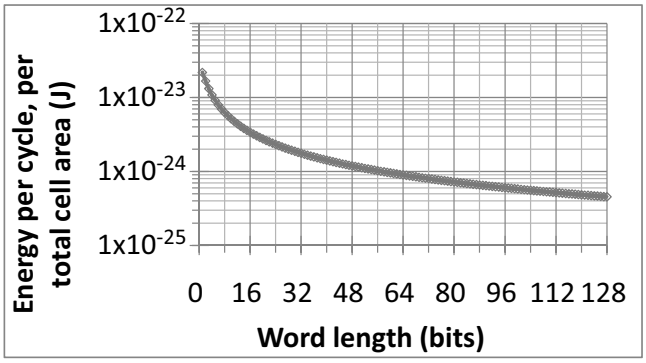
(b)

Fig. 44. *Bit erasures in n-bit addition: a) total number and b) resulting energy efficiency, defined as complete n-bit addition operations per unit energy.*

The power density is highly dependent on the cell size of the technology, limiting the maximum operating frequency, as we can easily cool only about 100 watts of heat per square centimeter (the actual value used by the ITRS [3] is 198 W/cm^2, but here it is conservatively assumed that about half should be reserved for power components other than the active QCA layer). This is a problem with molecular implementations, while the more coarse technologies are limited by other issues. In view of heat generation, a coarse featured technology might be more feasible than the smallest possible; the maximum



(a)



(b)

Fig. 45. *Power and energy comparison on QCA: a) power density on the active area of the adders, according to the cell width (size in nanometers on the right), and b) erasure energy spread on the total area of the ripple carry adder.*

clock frequencies of the adders, for various molecular QCA technologies, are shown in Fig. 46. The limits found for the adders are surprisingly low: on a 1 nm cell technology, the bound settles to about 35 GHz, and on 3 nm technology, to about 310 GHz. Although these limits are very high frequencies for conventional technologies, they are definitely restricting for emerging devices predicted to have switching in the terahertz regime; increasing the performance of arithmetic by raising the clock frequency will reach its limits very fast, compared to the approach of increasing the level of parallelism.

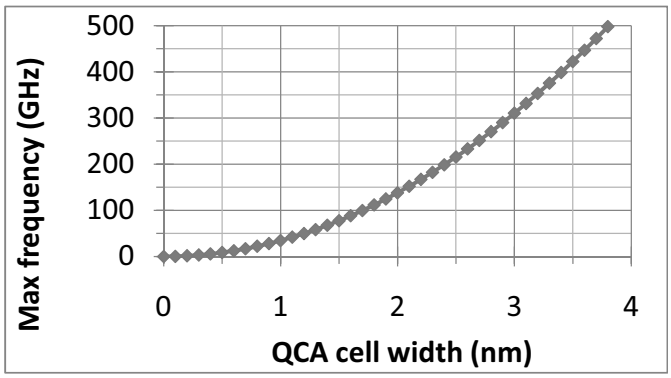


Fig. 46. *Maximum operating frequency of the adders on molecular QCA technologies of various feature sizes, under the cooling restriction of 100 W/cm^2.*

It can be concluded, that when the irreversibility power is the limiting factor, the smaller the QCA cell the slower the circuit. However, this is true for any technology: the denser the circuit, the higher the power density, and thus, the lower the top operating frequency. Although the ITRS report on the emerging research devices projects 61 GHz circuit speed for future transistors, they would have this same ultimate limitation when scaled to molecular dimensions (the projected densities are at least two orders of magnitude lower) [3].

6.3.3 Reversible Design

The Landauer's principle, bit erasure requires always dissipation [18], limits the operating frequency of these irreversible arithmetic units, if the designs are implemented with the very much sought-for molecular QCA. The only way to reach smaller power densities and higher clock frequencies is to adapt reversible computing principles into the designs, and avoid discarding information about the state of the system [57]. The main problem is the heavy cost of this, and the huge design space formed by the possible approaches [217].

Reversible design requires usually reversible logic elements, but on QCA, it is possible to avoid this additional cost by applying Bennett-type clock-

ing [16, 56], which makes all circuits logically and physically reversible. This approach has the drawback of disabling the natural pipelined operation, and the full cost of reversibility is directed at the structural performance of the arithmetic unit. There is no direct area cost, but the complicated clocking waveforms are more difficult to produce than the standard Landauer-type waveforms (see the clocking description in Sec. 2.4).

Bennett clocking is quite challenging to apply on sequential designs, due to the complex control between steps of computing and uncomputing. This is why it is not considered feasible on the serial adder, but the pipelined ripple carry adder, on the other hand, is at the logic level a purely combinatorial structure, which is ready for this scheme of reversibility. The cell structure does not require any modifications, and the clocking waveforms can be simply changed and the pipelined operation lost, ending up with an implementation that is fully reversible, dissipating energy only at the inputs or the outputs. The end result might perform better than an irreversible adder, but this is not guaranteed, as illustrated by the following estimate:

1 nm QCA-cell based ripple carry adder. An irreversible 4-bit RCA, the design presented in Sec. 3.4 with Landauer clocking, has the bit erasure limited maximum operating frequency of 35 GHz, as uncovered in Sec. 6.3.2, while possible other power components, like the clocking network, are assumed to be sufficiently minor below this speed. Performance gain is attempted via Bennett clocking, in which case the reversible adder can have the maximum frequency allowed by the minor power components other than information erasure.

Structural performance loss. An irreversible Landauer clocked pipelined RCA finishes an addition operation on every full clock cycle after the initial latency, producing at most 35 billion results per second. A reversible Bennett clocked 4-bit unit reserves the complete structure for one addition for nine cycles: five cycles for forward computing and four cycles for backwards reversal. It is important to notice, that a pipelined unit has constant throughput, while the throughput (defined as results per cycle) of a Bennett clocked n-bit adder decreases inversely

proportional to the operand word length, just like the maximum allowed frequency of the critical path of a purely combinatorial traditional arithmetic unit. Thus, a 4-bit Bennett clocked RCA must have at least 9×35 GHz $= 315$ GHz operating frequency, to reach the *same* computing performance, results per second, as a pipelined unit.

Example 1: minor power components set sufficiently high frequency limit, 500 GHz. The reversible adder clock speed can be raised over the minimum limit of 315 GHz required to compensate the huge throughput loss due to the lost pipelined operation. At the allowed 500 GHz, the unit produces $500/9 \approx 55.6$ billion results per second, thus achieving a computing performance improvement of $100 \times (55.6 - 35)/35 \approx 59$ % in comparison with the irreversible design.

Example 2: minor power components set too tight frequency limit, 200 GHz. The reversible adder clock speed cannot be raised over the minimum limit of 315 GHz required to compensate the structural throughput loss. At the allowed 200 GHz, the unit produces $200/9 \approx 22.2$ billion results per second, having a computing performance $100 \times (22.2 - 35)/35 \approx -37$ % worse than the original irreversible design.

Whether Bennett-type clocking enables so high frequencies, that the structural performance loss is fully compensated and processing power gained, is still unknown, since unknown power components, regardless of how infinitesimal they are, will set the limit after the irreversibility factor has been removed. However, the fundamental physical limit of energy dissipation in the bit erasures can be circumvented, and Bennett clocking might be the easiest way to do this. The scheme is unique for QCA, as most other technologies, including CMOS, require also considerable modifications to the circuit structure.

6.4 Serial-Parallel and Array Multiplier

The power characteristics of the standard multipliers are analyzed in respect to the ultimate physical limit of computation, the Landauer's principle. Also, fully reversible operation based on Bennett clocking is considered.

6.4.1 Bit Erasure Energy

The multipliers discard information in each functional unit, at each clock cycle. A combination of a two-input AND-gate and a full adder compresses 16 distinct input combinations into four different output values in quite an unbalanced way, most information lost when seven input cases are mapped onto a single output case, causing a loss of three bits of system state in one multiplier cell. This is luckily not the real case, because the input operand bits a_i and b_j are either held intact for the whole computation or fed with the outputs to the next functional unit. Forwarding the inputs as garbage signals is a costly way to preserve system state, but as a free by-product of the multiplier organization, these bits are available for reversing the computation, and thus, the multiplier cell loses only one bit of information. At the end of an n-bit multiplication, also the original input operands must be discarded, erasing $2n$ bits. The total number of bit erasures is directly, and the energy efficiency inversely, proportional to the square of the word length, as shown in Fig. 47.

6.4.2 Power Density

The place and timing of the bit erasures in an n-bit multiplication is different for each design, in principle affecting the expected power density, but in practice, evened out by the array computing n multiplications with different data sets co-existing in the pipeline, while the serial-parallel unit computes a single spread multiplication. There is also a difference in the hardware utilization, as the array can run with 100% of the functional units computing all the time, while the serial-parallel structure reaches a utilization of about 75%. This is due to the fact that the last pipeline stages of the bit-serial approach form a bottleneck, forcing the previous stages to stall or compute with zero inputs.

The minimum reachable power density is found by normalizing the total erasure energy with the computation time and the area of the multiplier. The metric does not depend on the word length of the unit, only on the structure

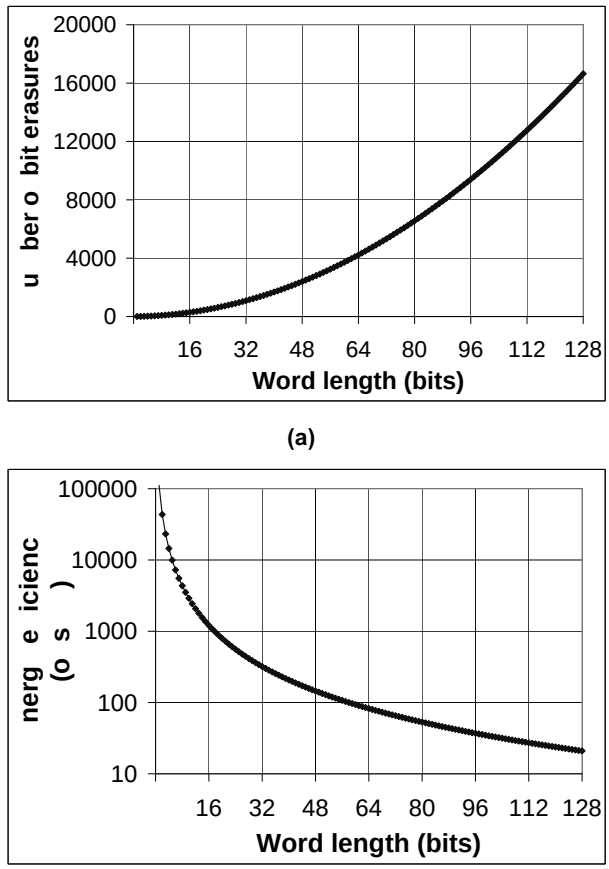
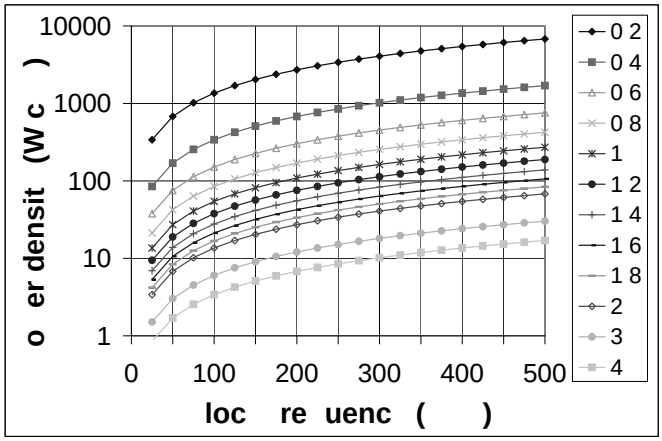


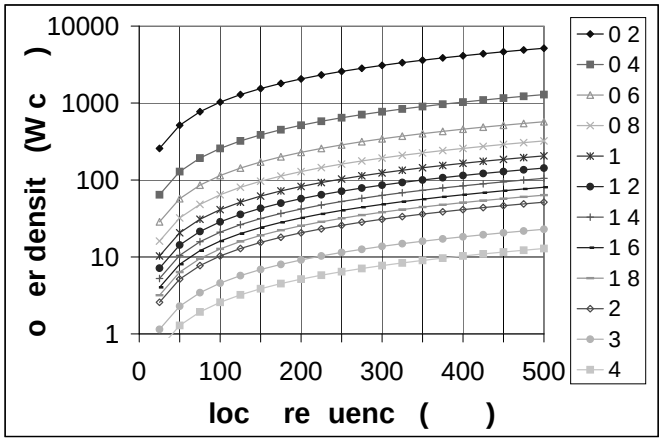
Fig. 47. *The bit erasures in n-bit multiplication, when the operand bits are available with intermediate outputs: a) total number, and b) resulting energy efficiency, defined as complete n-bit multiplications per unit energy.*

and the operating frequency; the array has a smaller value than the serial-parallel unit, on the same clock rate. Implementation specific power densities for various molecular QCA feature sizes are shown in Fig. 48.

The power density is highly dependent on the cell size of the technology, limiting the maximum operating frequency, as we can easily cool only about 100 watts of heat per square centimeter (about half of the ITRS [3] value of 198 W/cm^2, to make a conservative reservation for power components other than the active QCA layer). This is a problem with molecular implementations,



(a)



(b)

Fig. 48. *The power density of the multiplier designs on molecular QCA technologies, according to the cell width (size in nanometers on the right): a) serial-parallel multiplier, and b) array multiplier.*

while the more coarse technologies are limited by other issues. In view of heat generation, a coarse featured technology might be more feasible than the smallest possible; the maximum clock frequencies of the multipliers on various molecular QCA technologies are shown in Fig. 49.

In the presented area optimized implementations, the array multiplier enables higher operating frequencies than the serial-parallel multiplier, even with the

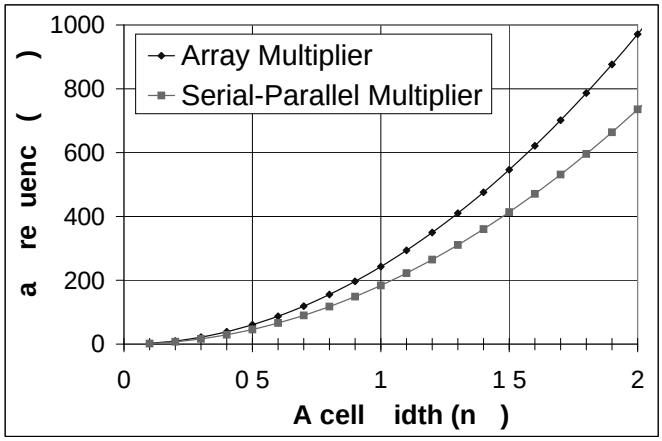


Fig. 49. *The maximum operating frequency of the multiplier structures on molecular QCA technologies of various feature sizes, under the cooling restriction of 100 W/cm^2.*

bit-serial structure having several idle cycles reducing the power density. On a 1 nm cell technology, the maximum frequency of the array is 240 GHz, while the serial-parallel structure reaches 180 GHz. Although these are very high frequencies for conventional technologies, they are quite restricting limits on a technology, which is predicted to have switching in the terahertz regime.

6.4.3 Reversible Design

The Landauer's principle, bit erasure requires always dissipation [18], limits the operating frequency of also the irreversible multiplier units, if the designs are implemented with the very much sought-for molecular QCA. Smaller power densities and higher clock frequencies could be reached by adopting reversible computing [57], based on reversible logic gates, QCA specific Bennett clocking [16, 56], or some combination of these approaches [217].

Bennett clocking (as described in Sec. 2.4) is very appealing method of creating reversible logic circuits from combinatorial (pipelined on QCA) or systolic designs, since modifications to the circuit structure can be avoided or at least limited. Reversible logic gates are not needed, but pipelining and the inherent structural performance is lost. Bennett clocking is quite challenging

to apply on sequential designs, due to the complex control between steps of computing and uncomputing, which is the reason it is not very suitable for the serial-parallel or radix-4 multiplier. A natural target is the systolic pipelined array multiplier, which is at the logic level a purely combinatorial structure.

The computing performance offered by a Bennett clocked reversible array multiplier might surpass an irreversible implementation, but this is not guaranteed, since it is still unknown, whether other power dissipation components than information erasure, regardless of how infinitesimal they are, set too tight limits after the irreversibility factor has been removed. Thus, whether Bennett-type clocking alone enables so high clock frequencies, that the structural performance loss is fully compensated and processing power gained, remains an open question, although the fundamental physical limit of energy dissipation in the bit erasures can be circumvented. (See the explanatory examples on a reversible ripple carry adder in Sec. 6.3.3.)

6.5 Summary

The power dissipation of complete arithmetic units was analyzed near the ultimate limit of computation efficiency, using the Landauer's principle directly. Based on the pipelined ripple carry adder, the pipelined array multiplier, and the serial-parallel multiplier, it was concluded, that irreversible information erasure consumes significant electrical power on molecular QCA. The bit erasures limit the allowed clock frequencies of the designs much lower than the expected switching speeds of the primitive devices would enable.

Based on the identified power limitation, it was concluded, that the full technology potential of QCA cannot be reached by the traditional approach of purely irreversible digital design. The desired higher operating frequencies are available only, if the bit erasure dissipation can be reduced by incorporating reversible computing principles into the designs. More generally, similar density-speed-reversibility tradeoffs are necessary also for *all* other circuit technologies offering molecular implementations.

7. CONCLUSIONS

In this thesis, computer arithmetic units have been developed for the promising quantum-dot cellular automata nanotechnology, and the characteristics of the designs evaluated with several metrics. Understanding these structures is crucial for establishing design methodology for the nanotechnology, and enabling optimization and design automation, in the long run.

7.1 Main Results

Novel modular arithmetic units were designed and described at the logic, the pipeline, and the QCA layout level, and verified with quantum mechanical simulation. The adder and multiplier designs were compared with previous proposals for QCA, showing that the presented designs achieve noise rejection with little or no performance/area penalty. In several cases, the presented units had the best performance or the smallest circuit area.

The performance metrics indicated, that basic serial and pipelined arithmetic structures typically have the same latency, on the studied self-latching technology, while the difference is in the throughput. The circuit area is dominated by the passive wiring overhead, characterized with a square-law dependency on the operand word length.

The reliability analysis, based on the probabilistic transfer matrix method, showed the urgent need for a multi-level redundancy scheme to be developed for large operand length arithmetic. Typically, macro component reliability had about linear effect on the total reliability, while the component types affected the total reliability with different weights, depending on the word

length. Wiring overhead dominated also the reliability. (The study was limited to analysis, leaving actual fault tolerant design methodology out of the scope of this treatment.)

The power analysis, at the fundamental Landauer's limit, showed that reaching the full technology potential of QCA requires reversible computing principles to be adopted into the designs. Irreversible bit erasures consume significant power on the molecular implementations, limiting the clock frequencies of the designs much lower than the expected switching speed of the technology. Generally, similar density-speed-reversibility tradeoffs are necessary for all other circuit technologies offering molecular implementations.

To conclude, the studies in this thesis show that QCA nanotechnology circuits and systems have to address several phenomena, that have not had much impact in traditional engineering work, yet. Design optimization has to be started from the reliability and power challenges, which will determine if the planned system will be practical to manufacture and really works, under the specified operating conditions. These challenges have been identified and broken down in the component analyses of this thesis, but actual design methodology to address them remains to be developed.

7.2 Future Development

The work on arithmetic structures matching the fine-grained pipelining of QCA is not complete. At this point, a good candidate for further research is the proposed Radix-4 recoded multiplier unit, which offers a lot of possibilities for introducing redundancy to achieve tolerance against the manufacturing defects and the runtime faults. However, the conducted studies give rise to some predictions about the future of computing systems more generally.

Reliability. The conducted reliability studies present a fresh view on the topic, serving as a starting point for developing the analysis methods further. Design automation to generate the reliability decomposition of a system should be incorporated in emerging tools, enabling easy design space explo-

ration. In the scope of computing systems generally, complete fault-tolerant design methodology must and will be developed, to enable functioning systems based on failing parts. Whether these systems process digital data, or data in some other representation format, remains to be seen.

Reversibility. The electrical power studies of this thesis also have a new view on the topic, inspiring the development of the logical block level irreversibility analysis methods further. Design automation to compute the effects of information erasure should be incorporated in the emerging tools, again, aiming at easy design space exploration. In the long run, information processing systems will have incorporate at least partially reversible computation methods to limit the heat generation, while reversibility is also a strict precondition for the visioned quantum computing with superposed state variables. However, a lot of basic research is needed, before the huge design space of redundancy and reversibility degrees of freedom is adequately understood.

Bibliography

[1] G. Moore, "Cramming more components onto integrated circuits," *Proc. IEEE*, vol. 86, no. 1, pp. 82–85, Jan. 1998.

[2] E. Mollick, "Establishing Moore's law," *IEEE Ann. Hist. Comput.*, vol. 28, no. 3, pp. 62–75, Jul.–Sep. 2006.

[3] International Technology Roadmap for Semiconductors. (2007) 2007 ITRS report. [Online]. Available: http://www.itrs.net/Links/2007ITRS/Home2007.htm

[4] C. Lent, "Molecular quantum-dot cellular automata," in *IEEE Workshop Signal Processing Systems Design and Implementation*, Banff, AB, Canada, Oct. 2–4, 2006, keynote talk.

[5] C. Lent, P. Tougaw, and W. Porod, "Quantum cellular automata: The physics of computing with arrays of quantum dot molecules," in *Proc. Workshop Phys. Comp.*, Dallas, TX, USA, Nov. 17–20, 1994, pp. 5–13.

[6] C. Lent and P. Tougaw, "A device architecture for computing with quantum dots," *Proc. IEEE*, vol. 85, no. 4, pp. 541–557, Apr. 1997.

[7] G. Snider, A. Orlov, I. Amlani, G. Bernstein, C. Lent, J. Merz, and W. Porod, "Quantum-dot cellular automata," in *Digest Pap. Microprocesses and Nanotechnol. Conf.*, Yokohama, Japan, Jul. 6–8, 1999, pp. 90–91.

[8] A. Orlov, R. Kummamuru, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Clocked quantum-dot cellular automata devices: Experimental studies," in *Proc. IEEE Conf. Nanotechnology*, Maui, HI, USA, Oct. 28–30, 2001, pp. 425–430.

[9] R. Kummamuru, A. Orlov, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Operation of a quantum-dot cellular automata (QCA) shift register and analysis of errors," *IEEE Trans. Electron Devices*, vol. 50, no. 9, pp. 1906–1913, Sep. 2003.

[10] K. Kim, K. Wu, and R. Karri, "The robust QCA adder designs using composable QCA building blocks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 1, pp. 176–183, Jan. 2007.

[11] H. Cho and E. Swartzlander, "Adder designs and analyses for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 374–383, May 2007.

[12] ——, "Serial parallel multiplier design in quantum-dot cellular automata," in *Proc. IEEE Symp. Computer Arithmetic*, Montepellier, France, Jun. 25–27, 2007, pp. 7–15.

[13] S.-W. Kim and E. Swartzlander, "Parallel multipliers for quantum-dot cellular automata," in *Proc. IEEE Nanotechnol. Materials Devices Conf.*, Traverse City, MI, USA, Jun. 2–5, 2009, pp. 68–72.

[14] R. Zhang, K. Walus, W. Wang, and G. Jullien, "Performance comparison of quantum-dot cellular automata adders," in *Proc. IEEE Int. Symp. Circuits Syst.*, Kobe, Japan, May 23–26, 2005, pp. 2522–2526.

[15] T. Dysart and P. Kogge, "Analyzing the inherent reliability of moderately sized magnetic and electrostatic QCA circuits via probabilistic transfer matrices," *IEEE Trans. VLSI Syst.*, vol. 17, no. 4, pp. 507–516, Apr. 2009.

[16] C. Lent, M. Liu, and Y. Lu, "Bennett clocking of quantum-dot cellular automata and the limits to binary logic scaling," *Nanotechnol.*, vol. 17, no. 16, pp. 4240–4251, Aug. 2006.

[17] J. Timler and C. Lent, "Maxwell's demon and quantum-dot cellular automata," *J. Appl. Phys.*, vol. 94, pp. 1050–1060, Jul. 2003.

[18] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J. Res. Dev.*, vol. 5, pp. 183–191, Jul. 1961.

[19] P. Sarkar, "A brief history of cellular automata," *ACM Comput. Surv.*, vol. 32, no. 1, pp. 80–107, Mar. 2000.

[20] D. Hampel and R. O. Winder, "Threshold logic," *IEEE Spectr.*, vol. 8, no. 5, pp. 32–39, May 1971.

[21] R. Zhang, P. Gupta, L. Zhong, and N. Jha, "Threshold network synthesis and optimization and its application to nanotechnologies," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 24, no. 1, pp. 107–118, Jan. 2005.

[22] R. Zhang, P. Gupta, and N. Jha, "Majority and minority network synthesis with application to QCA-, SET-, and TPL-based nanotechnologies," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 26, no. 7, pp. 1233–1245, Jul. 2007.

[23] R. Zhang, K. Walus, W. Wang, and G. Jullien, "A method of majority logic reduction for quantum cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 4, pp. 443–450, Dec. 2004.

[24] K. Walus, G. Schulhof, G. Jullien, R. Zhang, and W. Wang, "Circuit design based on majority gates for applications with quantum-dot cellular automata," in *Rec. Asilomar Conf. Signals, Systems and Computers*, vol. 2, Pacific Grove, CA, USA, Nov. 7–10, 2004, pp. 1354–1357.

[25] S. Roy and B. Saha, "Minority gate oriented logic design with quantum-dot cellular automata," in *Proc. Int. Conf. on Cellular Automata for Research and Industry*, Perpignan, France, Sep. 20–23, 2006, pp. 646–656.

[26] M. Momenzadeh, J. Huang, M. Tahoori, and F. Lombardi, "Characterization, test, and logic synthesis of and-or-inverter (AOI) gate design for QCA implementation," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 24, no. 12, pp. 1881–1893, Dec. 2005.

[27] J. Huang, M. Momenzadeh, M. B. Tahoori, and F. Lombardi, "Design and characterization of an and-or-inverter (AOI) gate for QCA implementation," in *Proc. ACM Great Lakes Symp. VLSI*, Boston, MA, USA, Apr. 26–28, 2004, pp. 426–429.

[28] W. Townsend and J. Abraham, "Complex gate implementations for quantum dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, Munich, Germany, Aug. 16–19, 2004, pp. 625–627.

[29] E. Ganesh, L. Kishore, and M. Rangachar, "Study of complex gate structures in quantum cellular automata technology for FPGA applications," in *IET-UK Int. Conf. Information and Communication Technology in Electrical Sciences*, Chennai, TamilNadu, India, Dec. 20–22, 2007, pp. 789–794.

[30] A. Fijany and B. Toomarian, "New design for quantum dots cellular automata to obtain fault tolerant logic gates," *J. Nanopart. Res.*, vol. 3, no. 1, pp. 27–37, Feb. 2001.

[31] ——, "Quantum dots cellular automata: Fault tolerant universal logic gates," *J. Nanotechnol.*, 2001. [Online]. Available: http://hdl.handle.net/2014/16544

[32] J. Huang, M. Momenzadeh, L. Schiano, M. Ottavi, and F. Lombardi, "Tile-based QCA design using majority-like logic primitives," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 1, no. 3, pp. 163–185, Oct. 2005.

[33] J. Huang, M. Momenzadeh, L. Schiano, and F. Lombardi, "Simulation-based design of modular QCA circuits," in *Proc. IEEE Conf. Nanotechnology*, vol. 2, Nagoya, Japan, Jul. 11–15, 2005, pp. 533–536.

[34] J. Huang, M. Momenzadeh, and F. Lombardi, "Defect tolerance of QCA tiles," in *Proc. Design, Automation & Test in Europe Conf.*, vol. 1, Munich, Germany, Mar. 6–10, 2006, pp. 1–6.

[35] ——, "On the tolerance to manufacturing defects in molecular QCA tiles for processing-by-wire," *J. Electron. Testing*, vol. 23, no. 2–3, pp. 163–174, Jun. 2007.

[36] G. Tóth, "Correlation and coherence in quantum-dot cellular automata," Ph.D. dissertation, Univ. Notre Dame, IN, USA, Jul. 2000. [Online]. Available: http://optics.szfki.kfki.hu/ toth/Thesis.pdf

[37] I. Bajec, N. Zimic, and M. Mraz, "Towards the bottom-up concept: Extended quantum-dot cellular automata," *Microelectron. Eng.*, vol. 83, no. 4–9, pp. 1826–1829, Apr.–Sep. 2006.

[38] ——, "The ternary quantum-dot cell and ternary logic," *Nanotechnol.*, vol. 17, no. 8, pp. 1937–1942, Apr. 28, 2006.

[39] G. Schulhof, K. Walus, and G. Jullien, "Simulation of random cell displacements in QCA," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 3, no. 1, pp. 1–14, Apr. 2007.

[40] S. Bhanja, M. Ottavi, F. Lombardi, and S. Pontarelli, "QCA circuits for robust coplanar crossing," *J. Electron. Testing*, vol. 23, no. 2–3, pp. 193–210, Jun. 2007.

[41] ——, "Novel designs for thermally robust coplanar crossing in QCA," in *Proc. Design, Automation & Test in Europe Conf.*, vol. 1, Munich, Germany, Mar. 6–10 2006.

[42] K. Walus, G. Schulhof, and G. Jullien, "High level exploration of quantum-dot cellular automata (QCA)," in *Rec. Asilomar Conf. Signals, Systems and Computers*, vol. 1, Pacific Grove, CA, USA, Nov. 7–10, 2004, pp. 30–33.

[43] A. Chaudhary, D. Chen, X. Hu, M. Niemier, R. Ravichandran, and K. Whitton, "Fabricatable interconnect and molecular QCA circuits," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 26, no. 11, pp. 1978–1991, Nov. 2007.

[44] A. Chaudhary, D. Chen, X. S. Hu, K. Whitton, M. Niemier, and R. Ravichandran, "Eliminating wire crossings for molecular quantum-dot cellular automata implementation," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, USA, Nov. 6–10, 2005, pp. 565–571.

[45] W. Chung, B. Smith, and S. Lim, "Node duplication and routing algorithms for quantum-dot cellular automata circuits," *IEE Proc. Circuits Devices Syst.*, vol. 153, no. 5, pp. 497–505, Oct. 2006.

[46] B. S. Smith and S. K. Lim, "QCA channel routing with wire crossing minimization," in *Proc. ACM Great Lakes Symp. VLSI*, Chicago, IL, USA, Apr. 17–19, 2005, pp. 217–220.

[47] C. Graunke, D. Wheeler, D. Tougaw, and J. Will, "Implementation of a crossbar network using quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 4, no. 4, pp. 435–440, Jul. 2005.

[48] K. Kim, K. Wu, and R. Karri, "Towards designing robust QCA architectures in the presence of sneak noise paths," in *Proc. Design, Automation & Test in Europe Conf.*, Munich, Germany, Mar. 7–11, 2005, pp. 1214–1219.

[49] R. Landauer, "Information is physical," in *Proc. Workshop Physics and Computation*, Dallas, TX, USA, Oct. 2–4, 1992, pp. 1–4.

[50] Z. Patitz, N. Park, M. Choi, and F. Meyer, "QCA-based majority gate design under radius of effect-induced faults," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Monterey, CA, USA, Oct. 3–5, 2005, pp. 217–225.

[51] L. Bonci, M. Gattobigio, G. Iannaccone, and M. Macucci, "Monte-carlo simulation of clocked and non-clocked QCA architectures," *J. Comput. Electron.*, vol. 1, no. 1–2, pp. 49–53, Jul. 2002.

[52] Y. Wang and M. Lieberman, "Thermodynamic behavior of molecular-scale quantum-dot cellular automata QCA wires and logic devices," *IEEE Trans. Nanotechnol.*, vol. 3, no. 3, pp. 368–376, Sep. 2004.

[53] E. Blair and C. Lent, "Quantum-dot cellular automata: An architecture for molecular computing," in *Proc. Int. Conf. Simulation of Semiconductor Processes and Devices*, Boston, MA, USA, Sep. 3–5, 2003, pp. 14–18.

[54] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Two-dimensional schemes for clocking/timing of QCA circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 1, pp. 34–44, Jan. 2008.

[55] ——, "Clocking and cell placement for QCA," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 343–346.

[56] M. Liu and C. Lent, "Power dissipation in clocked quantum-dot cellular automata circuits," in *Annu. Device Research Conf. Digest*, vol. 1, Santa Barbara, CA, USA, Jun. 20–22, 2005, pp. 123–124.

[57] C. Bennett, "Logical reversibility of computation," *IBM J. Res. Dev.*, vol. 17, pp. 525–532, Nov. 1973.

[58] S. Frost-Murphy, E. DeBenedictis, and P. Kogge, "General floorplan for reversible quantum-dot cellular automata," in *Proc. ACM Int. Conf. Computing Frontiers*, Ischia, Italy, May 7–9, 2007, pp. 77–81.

[59] G. Bernstein, "Quantum-dot cellular automata by electric and magnetic field coupling," in *Proc. IEEE Custom Integrated Circuits Conf.*, San Jose, CA, USA, Sep. 21–24, 2003, pp. 223–229.

[60] T. Cole and J. C. Lusth, "Quantum-dot cellular automata," *Prog. Quantum Electron.*, vol. 25, no. 4, pp. 165–189, 2001.

[61] J. Lusth, "Symmetric versus asymmetric charge neutralization in quantum-dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, Maui, HI, USA, Oct. 28–30, 2001, pp. 380–385.

[62] ——, "Balancing QCA logic gates under image charge neutralization," in *Proc. IEEE Conf. Nanotechnology*, Washington, DC, USA, Aug. 26–28, 2002, pp. 347–350.

[63] J. C. Lusth, C. B. Hanna, and J. C. Díaz-Vélez, "Eliminating non-logical states from linear quantum-dot-cellular automata," *Microelectron. J.*, vol. 32, no. 1, pp. 81–84, Jan. 2001.

[64] D. Kelly and J. Lusth, "Logic devices for partitioned quantum-dot cells," in *Proc. IEEE Conf. Nanotechnology*, Maui, HI, USA, Oct. 28–30, 2001, pp. 374–379.

[65] J. Liang and J. Lusth, "3-dimensional configuration to promote timely settling of quantum-dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, vol. 2, San Francisco, CA, USA, Aug. 11–14, 2003, pp. 770–773.

[66] E. Schuchman and J. Lusth, "Computing with restricted minima systems," in *Proc. IEEE Conf. Nanotechnology*, Washington, DC, USA, Aug. 26–28, 2002, pp. 359–362.

[67] K. Walus, R. Budiman, and G. Jullien, "Split current quantum-dot cellular automata—modeling and simulation," *IEEE Trans. Nanotechnol.*, vol. 3, no. 2, pp. 249–255, Jun. 2004.

[68] K. K. Yadavalli, A. O. Orlov, J. P. Timler, C. S. Lent, and G. L. Snider, "Fanout gate in quantum-dot cellular automata," *Nanotechnol.*, vol. 18, no. 37, Sep. 19, 2007.

[69] G. Toth and C. Lent, "Quasiadiabatic switching for metal-island quantum-dot cellular automata," *J. Appl. Phys.*, vol. 85, no. 5, pp. 2977–2984, Mar. 1, 1999.

[70] I. Amlani, A. Orlov, G. Toth, G. Bernstein, C. Lent, and G. Snider, "Digital logic gate using quantum-dot cellular automata," *Science*, vol. 284, no. 5412, pp. 289–291, Apr. 9, 1999.

[71] A. Orlov, I. Amlani, G. Toth, C. Lent, G. Bernstein, and G. Snider, "Experimental demonstration of a binary wire for quantum-dot cellular automata," *Appl. Phys. Lett.*, vol. 74, no. 19, pp. 2875–2877, May 10, 1999.

[72] I. Amlani, A. Orlov, R. Kummamuru, G. Bernstein, C. Lent, and G. Snider, "Experimental demonstration of a leadless quantum-dot cellular automata cell," *Appl. Phys. Lett.*, vol. 77, no. 5, pp. 738–740, Jul. 31, 2000.

[73] A. Orlov, I. Amlani, R. Kummamuru, R. Ramasubramaniam, G. Toth, C. Lent, G. Bernstein, and G. Snider, "Experimental demonstration of clocked single-electron switching in quantum-dot cellular automata," *Appl. Phys. Lett.*, vol. 77, no. 2, pp. 295–297, Jul. 10, 2000.

[74] A. Orlov, R. Kummamuru, R. Ramasubramaniam, G. Toth, C. Lent, G. Bernstein, and G. Snider, "Experimental demonstration of a latch in clocked quantum-dot cellular automata," *Appl. Phys. Lett.*, vol. 78, no. 11, pp. 1625–1627, Mar. 12, 2001.

[75] G. Snider, A. Orlov, R. Kummamuru, R. Ramasubramaniam, I. Amlani, G. Bernstein, C. Lent, J. Merz, and P. Wolfgang, "Quantum-dot cellular automata: Introduction and experimental overview," in *Proc. IEEE Conf. Nanotechnology*, Maui, HI, USA, Oct. 28–30, 2001, pp. 465–470.

[76] A. Orlov, R. Kummamuru, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Clocked quantum-dot cellular automata shift register," *Surf. Sci.*, vol. 532, pp. 1193–1198, Jun. 10, 2003.

[77] R. Kummamuru, A. Orlov, J. Timler, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "A quantum-dot cellular automata shift register," in *Device Research Conf.*, Notre Dame, IN, USA, Jun. 25–27, 2001, pp. 103–104.

[78] R. Kummamuru, A. Orlov, G. Toth, J. Timler, R. Rajagopal, C. Lent, G. Bernstein, and G. Snider, "Power gain in a quantum-dot cellular automata (QCA) shift register," in *Proc. IEEE Conf. Nanotechnology*, Maui, HI, USA, Oct. 28–30, 2001, pp. 431–436.

[79] R. Kummamuru, J. Timler, G. Toth, C. Lent, R. Ramasubramaniam, A. Orlov, G. Bernstein, and G. Snider, "Power gain in a quantum-dot cellular automata latch," *Appl. Phys. Lett.*, vol. 81, no. 7, pp. 1332–1334, Aug. 12, 2002.

[80] R. Kummamuru, A. Orlov, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Experimental demonstration of a QCA shift register and analysis of errors," in *Digest Int. Electron Devices Meeting*, San Francisco, CA, USA, Dec. 8–11, 2002, pp. 95–98.

[81] M. Liu and C. Lent, "High-speed metallic quantum-dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, vol. 2, San Francisco, CA, USA, Aug. 11–14, 2003, pp. 465–468.

[82] ——, "Reliability and defect tolerance in metallic quantum-dot cellular automata," *J. Electron. Testing*, vol. 23, no. 2–3, pp. 211–218, Jun. 2007.

[83] M. Crocker, X. S. Hu, M. Niemier, M. Yan, and G. Bernstein, "PLAs in quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 7, no. 3, pp. 376–386, May 2008.

[84] M. Chen and W. Porod, "Design of gate-confined quantum-dot structures in the few-electron regime," *J. Appl. Phys.*, vol. 78, no. 2, pp. 1050–1057, Jul. 15, 1995.

[85] P. Tougaw and C. Lent, "Dynamic behavior of quantum cellular automata," *J. Appl. Phys.*, vol. 80, no. 8, pp. 4722–4736, Oct. 15, 1996.

[86] M. Governale, M. Macucci, G. Iannaccone, C. Ungarelli, and J. Martorell, "Modeling and manufacturability assessment of bistable quantum-dot cells," *J. Appl. Phys.*, vol. 85, no. 5, pp. 2962–2971, Mar. 1, 1999.

[87] C. Ungarelli, S. Francaviglia, M. Macucci, and G. Iannaccone, "Thermal behavior of quantum cellular automaton wires," *J. Appl. Phys.*, vol. 87, no. 10, pp. 7320–7325, May 15, 2000.

[88] M. Macucci, G. Iannaccone, S. Francaviglia, and B. Pellegrini, "Semiclassical simulation of quantum cellular automaton circuits," *Int. J. Circuit Theory Appl.*, vol. 29, no. 1, pp. 37–47, Jan.–Feb. 2001.

[89] M. Macucci, M. Gattobigio, L. Bonci, G. Iannaccone, F. Prins, C. Single, G. Wetekam, and D. Kern, "A QCA cell in silicon-on-insulator technology: Theory and experiment," *Superlattices Microstruct.*, vol. 34, no. 3–6, pp. 205–211, Sep.–Dec. 2003.

[90] K. Walus, R. Budiman, and G. Jullien, "Impurity charging in semiconductor quantum-dot cellular automata," *Nanotechnol.*, vol. 16, no. 11, pp. 2525–2529, Nov. 2005.

[91] C. Lent, "Molecular electronics—bypassing the transistor paradigm," *Science*, vol. 288, no. 5471, pp. 1597–1599, Jun. 2, 2000.

[92] C. Lent, B. Isaksen, and M. Lieberman, "Molecular quantum-dot cellular automata," *J. Am. Chem. Soc.*, vol. 125, no. 4, pp. 1056–1063, Jan. 29, 2003.

[93] C. Lent and B. Isaksen, "Clocked molecular quantum-dot cellular automata," *IEEE Trans. Electron Devices*, vol. 50, no. 9, pp. 1890–1896, Sep. 2003.

[94] E. Blair and C. Lent, "An architecture for molecular computing using quantum-dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, San Francisco, CA, USA, Aug. 11–14, 2003, pp. 402–405.

[95] J. Jiao, G. Long, F. Grandjean, A. Beatty, and T. Fehlner, "Building blocks for the molecular expression of quantum cellular automata. Isolation and characterization of a covalently bonded square array of two ferrocenium and two ferrocene complexes," *J. Am. Chem. Soc.*, vol. 125, no. 25, pp. 7522–7523, Jun. 25, 2003.

[96] Z. Li, A. Beatty, and T. Fehlner, "Molecular QCA cells. 1. Structure and functionalization of an unsymmetrical dinuclear mixed-valence complex for surface binding," *Inorg. Chem.*, vol. 42, no. 18, pp. 5707–5714, Sep. 8, 2003.

[97] Z. Li and T. Fehlner, "Molecular QCA cells. 2. Characterization of an unsymmetrical dinuclear mixed-valence complex bound to a Au surface by an organic linker," *Inorg. Chem.*, vol. 42, no. 18, pp. 5715–5721, Sep. 8, 2003.

[98] H. Qi, S. Sharma, Z. Li, G. Snider, A. Orlov, C. Lent, and T. Fehlner, "Molecular quantum cellular automata cells. Electric field driven switching of a silicon surface bound array of vertically oriented two-dot molecular quantum cellular automata," *J. Am. Chem. Soc.*, vol. 125, no. 49, pp. 15 250–15 259, Dec. 10, 2003.

[99] H. Qi, A. Gupta, B. Noll, G. Snider, Y. Lu, C. Lent, and T. Fehlner, "Dependence of field switched ordered arrays of dinuclear mixed-valence complexes on the distance between the redox centers and the size of the counterions," *J. Am. Chem. Soc.*, vol. 127, no. 43, pp. 15 218–15 227, Nov. 2, 2005.

[100] W. Hu, K. Sarveswaran, M. Lieberman, and G. Bernstein, "High-resolution electron beam lithography and DNA nano-patterning for molecular QCA," *IEEE Trans. Nanotechnol.*, vol. 4, no. 3, pp. 312–316, May 2005.

[101] Y. Lu and C. S. Lent, "Theoretical study of molecular quantum-dot cellular automata," *J. Comput. Electron.*, vol. 4, no. 1-2, pp. 115–118, Apr. 2005.

[102] Y. Lu, M. Liu, and C. Lent, "Molecular electronics–from structure to circuit dynamics," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 62–65.

[103] ——, "Molecular quantum-dot cellular automata: From molecular structure to circuit dynamics," *J. Appl. Phys.*, vol. 102, no. 3, Aug. 1, 2007.

[104] Z. Jin, "Fabrication and measurement of molecular quantum cellular automata (QCA) device," Master's thesis, Univ. Notre Dame, IN, USA, 2006.

[105] M. Manimaran, G. Snider, C. Lent, V. Sarveswaran, M. Lieberman, Z. Li, and T. Fehlner, "Scanning tunneling microscopy and spectroscopy investigations of QCA molecules," *Ultramicrosc.*, vol. 97, no. 1–4, pp. 55–63, Oct.–Nov. 2003.

[106] I. Lee, V. Sarveswaran, M. Lieberman, and E. Greenbaum, "Characterization of a single molecular QCA cell by Q-control enhanced amplitude modulation atomic force microscopy," *Ultramicrosc.*, vol. 106, no. 8–9, pp. 735–741, Jun.–Jul. 2006.

[107] M. Niemier, M. Crocker, X. Hu, and M. Lieberman, "Using CAD to shape experiments in molecular QCA," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, USA, Nov. 5–9, 2006, pp. 907–914.

[108] R. Cowburn and M. Welland, "Room temperature magnetic quantum cellular automata," *Science*, vol. 287, no. 5457, pp. 1466–1468, Feb. 25, 2000.

[109] G. Csaba and W. Porod, "Simulation of field coupled computing architectures based on magnetic dot arrays," *J. Comput. Electron.*, vol. 1, no. 1-2, pp. 87–91, Jul. 2002.

[110] G. Csaba, A. Imre, G. Bernstein, W. Porod, and V. Metlushko, "Nanocomputing by field-coupled nanomagnets," *IEEE Trans. Nanotechnol.*, vol. 1, no. 4, pp. 209–213, Dec. 2002.

[111] M. Parish and M. Forshaw, "Physical constraints on magnetic quantum cellular automata," *Appl. Phys. Lett.*, vol. 83, no. 10, pp. 2046–2048, Sep. 8, 2003.

[112] ——, "Magnetic cellular automata (MCA) systems," *IEE Proc. Circuits Devices Syst.*, vol. 151, no. 5, pp. 480–485, Oct. 2004.

[113] G. Bernstein, A. Imre, V. Metlushko, A. Orlov, L. Zhou, L. Ji, G. Csaba, and W. Porod, "Magnetic QCA systems," *Microelectron. J.*, vol. 36, no. 7, pp. 619–624, Jul. 2005.

[114] A. Imre, G. Csaba, L. Ji, A. Orlov, G. Bernstein, and W. Porod, "Majority logic gate for magnetic quantum-dot cellular automata," *Science*, vol. 311, no. 5758, pp. 205–208, Jan. 13, 2006.

[115] K. Nikolić and M. Forshaw, "Molecular magnetic quantum cellular automata," Univ. College London, Physics & Astronomy Dept., Tech. Rep., Jun. 2004.

[116] M. Niemier, M. Alam, X. S. Hu, G. Bernstein, W. Porod, M. Putney, and J. DeAngelis, "Clocking structures and power analysis for nanomagnet-based logic devices," in *Proc. Int. Symp. Low Power Electronics And Design*, Portland, OR, USA, Aug. 27–29, 2007, pp. 26–31.

[117] M. Niemier, M. Kontz, and P. Kogge, "A design of and design tools for a novel quantum dot based microprocessor," in *Proc. Design Automation Conf.*, Los Angeles, CA, USA, Jun. 5–9, 2000, pp. 227–232.

[118] M. Niemier, "Designing digital systems in quantum cellular automata," Master's thesis, Univ. Notre Dame, IN, USA, 2000.

[119] K. Walus and G. Jullien, "Design tools for an emerging SoC technology: Quantum-dot cellular automata," *Proc. IEEE*, vol. 94, no. 6, pp. 1225–1244, Jun. 2006.

[120] K. Walus, T. Dysart, G. Jullien, and R. Budiman, "QCADesigner: A rapid design and simulation tool for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 26–31, Mar. 2004.

[121] K. Walus, G. Schulhof, and G. A. Jullien, "Implementation of a simulation engine for clocked molecular QCA," in *Proc. Canadian Conf. Electrical and Computer Engineering*, Ottawa, ON, Canada, May 7–10, 2006, pp. 2128–2131.

[122] F. Karim, A. Navabi, K. Walus, and A. Ivanov, "Quantum mechanical simulation of QCA with a reduced Hamiltonian model," in *Proc. IEEE Conf. Nanotechnology*, Arlington, TX, USA, Aug. 18–21, 2008, pp. 327–330.

[123] (2007) QCADesigner website. Univ. Calgary ATIPS Laboratory. [Online]. Available: http://www.qcadesigner.ca

[124] S. Srivastava and S. Bhanja, "Hierarchical probabilistic macromodeling for QCA circuits," *IEEE Trans. Comput.*, vol. 56, no. 2, pp. 174–190, Feb. 2007.

[125] S. Bhanja and S. Sarkar, "Probabilistic modeling of QCA circuits using Bayesian networks," *IEEE Trans. Nanotechnol.*, vol. 5, no. 6, pp. 657–670, Nov. 2006.

[126] S. Srivastava and S. Bhanja, "Bayesian macromodeling for circuit level QCA design," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 31–34.

[127] S. Bhanja and S. Sarkar, "Graphical probabilistic inference for ground state and near-ground state computing in QCA circuits," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Nagoya, Japan, Jul. 11–15, 2005, pp. 290–293.

[128] ——, "Switching error modes of QCA circuits," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 383–386.

[129] O. Neto, M. Pacheco, and C. Hall Barbosa, "Neural network simulation and evolutionary synthesis of QCA circuits," *IEEE Trans. Comput.*, vol. 56, no. 2, pp. 191–201, Feb. 2007.

[130] R. Tang, F. Zhang, and Y.-B. Kim, "Design metal-dot based QCA circuits using SPICE model," *Microelectron. J.*, vol. 37, no. 8, pp. 821–827, Aug. 2006.

[131] ——, "QCA-based nano circuits design [adder design example]," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3, Kobe, Japan, May 23–26, 2005, pp. 2527–2530.

[132] ——, "Quantum-dot cellular automata spice macro model," in *Proc. ACM Great Lakes Symp. VLSI*, Chicago, IL, USA, Apr. 17–19, 2005, pp. 108–111.

[133] J. C. Lusth and B. Dixon, "A characterization of important algorithms for quantum-dot cellular automata," *Inf. Sci.*, vol. 113, no. 3–4, pp. 193–204, Feb. 1999.

[134] V. Vankamamidi, M. Ottavi, and F. Lombardi, "A serial memory by quantum-dot cellular automata (QCA)," *IEEE Trans. Comput.*, vol. 57, no. 5, pp. 606–618, May 2008.

[135] ——, "Tile-based design of a serial memory in QCA," in *Proc. ACM Great Lakes Symp. VLSI*, Chicago, IL, USA, Apr. 17–19, 2005, pp. 201–206.

[136] D. Berzon and T. Fountain, "A memory design in QCA using the SQUARES formalism," in *Proc. ACM Great Lakes Symp. VLSI*, Ann Arbor, MI, USA, Mar. 4–6, 1999, pp. 166–169.

[137] M. Ottavi, S. Pontarelli, V. Vankamamidi, A. Salsano, and F. Lombardi, "QCA memory with parallel read/serial write: Design and analysis," *IEE Proc. Circuits Devices Syst.*, vol. 153, no. 3, pp. 199–206, Jun. 2006.

[138] M. Ottavi, V. Vankamamidi, F. Lombardi, S. Pontarelli, and A. Salsano, "Design of a QCA memory with parallel read/serial write," in *Proc. IEEE Computer Society Annu. Symp. VLSI*, Tampa, FL, USA, May 11–12, 2005, pp. 292–294.

[139] M. Ottavi, V. Vankamamidi, F. Lombardi, and S. Pontarelli, "Novel memory designs for QCA implementation," in *Proc. IEEE Conf. Nanotechnology*, vol. 2, Nagoya, Japan, Jul. 11–15, 2005, pp. 545–548.

[140] S. Frost, A. Rodrigues, A. Janiszewski, R. Raush, and P. Kogge, "Memory in motion: A study of storage structures in QCA," in *Proc. Workshop Non-Silicon Computation, Int. Symp. High Performance Computer Architecture*, Boston, MA, USA, Feb. 3, 2002.

[141] S. Frost, A. Rodrigues, C. Giefer, and P. Kogge, "Bouncing threads: Merging a new execution model into a nanotechnology memory," in *Proc. IEEE Computer Society Annu. Symp. VLSI*, Tampa, FL, USA, Feb. 20–21, 2003, pp. 19–25.

[142] S. Frost, "Memory architecture for quantum-dot cellular automata," Master's thesis, Univ. Notre Dame, IN, USA, 2005.

[143] B. Taskin and B. Hong, "Dual-phase line-based QCA memory design," in *Proc. IEEE Conf. Nanotechnology*, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 302–305.

[144] V. Vankamamidi, M. Ottavi, and F. Lombardi, "A line-based parallel memory for QCA implementation," *IEEE Trans. Nanotechnol.*, vol. 4, no. 6, pp. 690–698, Nov. 2005.

[145] M. Ottavi, L. Schiano, S. Pontarelli, V. Vankamamidi, and F. Lombardi, "Timing verification of QCA memory architectures," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 391–394.

[146] X. S. Hu, M. Crocker, M. Niemier, M. Yan, and G. Bernstein, "PLAs in quantum-dot cellular automata," in *Proc. IEEE Computer Society Annu. Symp. Emerging VLSI Technologies and Architectures*, Karlsruhe, Germany, Mar. 2–3, 2006.

[147] M. Niemier and P. Kogge, "The "4-diamond circuit" - a minimally complex nano-scale computational building block in QCA," in *Proc. IEEE Computer Society Annu. Symp. VLSI*, Lafayette, LA, USA, Feb. 19–20, 2004, pp. 3–10.

[148] M. Niemier, A. Rodrigues, and P. Kogge, "A potentially implementable FPGA for quantum dot cellular automata," in *Proc. Workshop Non-Silicon Computation, Int. Symp. High Performance Computer Architecture*, Boston, MA, USA, Feb. 3, 2002.

[149] T. Lantz and E. Peskin, "A QCA implementation of a configurable logic block for an FPGA," in *Proc. IEEE Int. Conf. Reconfigurable Computing and FPGA's*, San Luis Potosi, Mexico, Sep. 20–22 2006, pp. 1–10.

[150] W. Rao, A. Orailoglu, and R. Karri, "Logic level fault tolerance approaches targeting nanoelectronics PLAs," in *Proc. Design, Automation & Test in Europe Conf.*, Nice, France, Apr. 16–20, 2007, pp. 1–5.

[151] J. Janulis, P. Tougaw, S. Henderson, and E. Johnson, "Serial bit-stream analysis using quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 158–164, Mar. 2004.

[152] S. Henderson, E. Johnson, J. Janulis, and P. Tougaw, "Incorporating standard CMOS design process methodologies into the QCA logic design process," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 2–9, Mar. 2004.

[153] S. Sarkar and S. Bhanja, "Synthesizing energy minimizing quantum-dot cellular automata circuits for vision computing," in *Proc. IEEE Conf. Nanotechnology*, Nagoya, Japan, Jul. 11–15, 2005, pp. 541–544.

[154] T. Fountain, "The design of highly-parallel image processing systems using nanoelectronic devices," in *Proc. IEEE Int. Workshop Computer Architecture for Machine Perception*, Boston, MA, USA, Oct. 20–22, 1997, pp. 210–219.

[155] A. Fijany, B. Toomarian, and M. Spotnitz, "Novel highly parallel and systolic architectures using quantum-dot based hardware," in *Proc. Parallel Computing*, Delft, the Netherlands, Aug. 17–20, 1999.

[156] N. Yamamoto, H. Fujisaka, K. Haeiwa, and T. Kamio, "Nanoelectronic circuits for stochastic computing," in *Proc. IEEE Conf. Nanotechnology*, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 306–309.

[157] M. Niemier and P. Kogge, "Logic in wire: Using quantum dots to implement a microprocessor," in *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, vol. 3, Pafos, Cyprus, Sep. 5–8, 1999, pp. 1211–1215.

[158] ——, "Exploring and exploiting wire-level pipelining in emerging technologies," in *Proc. Annu. Int. Symp. Computer Architecture*, Göteborg, Sweden, Jun. 30–Jul. 4, 2001, pp. 166–177.

[159] ——, "Problems in designing with QCAs: Layout = timing," *Int. J. Circuit Theory Appl.*, vol. 29, no. 1, pp. 49–62, Jan. 2001.

[160] K. Walus, M. Mazur, G. Schulhof, and G. Jullien, "Simple 4-bit processor based on quantum-dot cellular automata (QCA)," in *Proc. IEEE Int. Conf. Application-Specific Systems, Architectures and Processors*, Samos, Greece, Jul. 23–25, 2005, pp. 288–293.

[161] M. Ottavi, L. Schiano, F. Lombardi, and D. Tougaw, "HDLQ: A HDL environment for QCA design," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 2, no. 4, pp. 243–261, Oct. 2006.

[162] J. Huang, M. Momenzadeh, and F. Lombardi, "Design of sequential circuits by quantum-dot cellular automata," *Microelectr. J.*, vol. 38, no. 4–5, pp. 525–537, Apr.–May 2007.

[163] M. Momenzadeh, J. Huang, and F. Lombardi, "Defect characterization and tolerance of QCA sequential devices and circuits," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Monterey, CA, USA, Oct. 3–5, 2005, pp. 199–207.

[164] J. Huang, M. Momenzadeh, and F. Lombardi, "Analysis of missing and additional cell defects in sequential quantum-dot cellular automata," *Integration, VLSI J.*, vol. 40, no. 4, pp. 503–515, Jul. 2007.

[165] M. Choi, Z. Patitz, B. Jin, F. Tao, N. Park, and M. Choi, "Designing layout-timing independent quantum-dot cellular automata (QCA) circuits by global asynchrony," *J. Syst. Archit.*, vol. 53, no. 9, pp. 551–567, Sep. 2007.

[166] M. Choi, M. Choi, Z. Patitz, and N. Park, "Efficient and robust delay-insensitive QCA (quantum-dot cellular automata) design," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Arlington, VA, USA, Oct. 4–6, 2006, pp. 80–88.

[167] C. Minsu and P. Nohpill, "Locally synchronous, globally asynchronous design for quantum-dot cellular automata (LSGA QCA)," in *Proc. IEEE Conf. Nanotechnology*, vol. 1, Nagoya, Japan, Jul. 11–15, 2005, pp. 121–124.

[168] M. Bonyadi, S. Azghadi, N. Rad, K. Navi, and E. Afjei, "Logic optimization for majority gate-based nanoelectronic circuits based on genetic algorithm," in *Int. Conf. Electr. Eng.*, Lahore, Pakistan, Apr. 11–12 2007, pp. 1–5.

[169] H. Rahaman, B. Sikdar, and D. Das, "Synthesis of symmetric functions using quantum cellular automata," in *Int. Conf. Design and Test of Integrated Systems in Nanoscale Technology*, La Marsa, Tunisia, Sep. 5–7, 2006, pp. 119–124.

[170] F. Ciontu, C. Cucu, and B. Courtois, "Application-specific architecture for quantum cellular automata," in *Proc. IEEE Conf. Nanotechnology*, Washington, DC, USA, Aug. 26–28, 2002, pp. 351–354.

[171] O. Neto, L. Masiero, M. Pacheco, and C. Barbosa, "Evolvable hardware applied to nanotechnology," in *NASA/ESA Conf. Adaptive Hardware and Systems*, Istanbul, Turkey, Jun. 15–18, 2006, pp. 88–96.

[172] N. Gergel, S. Craft, and J. Lach, "Modeling QCA for area minimization in logic synthesis," in *Proc. ACM Great Lakes Symp. VLSI*, Washington, DC, USA, Apr. 28–29, 2003, pp. 60–63.

[173] R. Zhang and N. K. Jha, "Threshold/majority logic synthesis and concurrent error detection targeting nanoelectronic implementations," in *Proc. ACM Great Lakes Symp. VLSI*, Philadelphia, PA, USA, Apr. 30–May 2, 2006, pp. 8–13.

[174] R. Zhang, P. Gupta, and N. Jha, "Synthesis of majority and minority networks and its applications to QCA, TPL and SET based nanotechnologies," in *Int. Conf. VLSI Design*, Kolkata, India, Jan. 3–7, 2005, pp. 229–234.

[175] S. K. Lim, R. Ravichandran, and M. Niemier, "Partitioning and placement for buildable QCA circuits," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 1, no. 1, pp. 50–72, Mar. 2005.

[176] R. Ravichandran, M. Niemier, and S. K. Lim, "Partitioning and placement for buildable QCA circuits," in *Proc. Asia and South Pacific Design Automation Conf.*, vol. 1, Shanghai, China, Jan. 18–21, 2005, pp. 424–427.

[177] D. Antonelli, D. Chen, T. Dysart, X. Hu, A. Kahng, P. Kogge, R. Murphy, and M. Niemier, "Quantum-dot cellular automata (QCA) circuit partitioning: Problem modeling and solutions," in *Proc. ACM/IEEE Design Automation Conf.*, San Diego, CA, USA, Jun. 7–11, 2004, pp. 363–368.

[178] R. Ravichandran, N. Ladiwala, J. Nguyen, M. Niemier, and S. K. Lim, "Automatic cell placement for quantum-dot cellular automata," in *Proc. ACM Great Lakes Symp. VLSI*, Boston, MA, USA, Apr. 26–28, 2004, pp. 332–337.

[179] M. Tahoori, J. Huang, M. Momenzadeh, and F. Lombardi, "Testing of quantum cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 4, pp. 432–442, Dec. 2004.

[180] M. Tahoori, M. Momenzadeh, J. Huang, and F. Lombardi, "Defects and faults in quantum cellular automata at nano scale," in *Proc. IEEE VLSI Test Symp.*, Napa, CA, USA, Apr. 25–29, 2004, pp. 291–296.

[181] M. Momenzadeh, M. Tahoori, J. Huang, and F. Lombard, "Quantum cellular automata: New defects and faults for new devices," in *Proc. IEEE Int. Symp. Parallel & Distributed Processing*, Sante Fe, NM, USA, Apr. 26–30, 2004, pp. 207–214.

[182] M. Momenzadeh, M. Ottavi, and F. Lombardi, "Modeling QCA defects at molecular-level in combinational circuits," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Monterey, CA, USA, Oct. 3–5, 2005, pp. 208–216.

[183] M. Momenzadeh, J. Huang, M. Tahoori, and F. Lombardi, "On the evaluation of scaling of QCA devices in the presence of defects at manufacturing," *IEEE Trans. Nanotechnol.*, vol. 4, no. 6, pp. 740–743, Nov. 2005.

[184] M. Khatun, T. Barclay, I. Sturzu, and P. Tougaw, "Fault tolerance properties in quantum-dot cellular automata devices," *J. Phys. D: Appl. Phys.*, vol. 39, no. 8, pp. 1489–1494, Apr. 2006.

[185] F. Karim, M. Ottavi, H. Hashempour, V. Vankamamidi, K. Walus, A. Ivanov, and F. Lombardi, "Modeling and evaluating errors due to random clock shifts in quantum-dot cellular automata circuits," *J. Electron. Test.*, vol. 25, no. 1, pp. 55–66, Feb. 2009.

[186] M. Ottavi, H. Hashempour, V. Vankamamidi, F. Karim, K. Walus, and A. Ivanov, "On the error effects of random clock shifts in quantum-dot cellular automata circuits," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Rome, Italy, Sep. 26–28, 2007, pp. 487–498.

[187] F. Karim and K. Walus, "Characterization of the displacement tolerance of QCA interconnects," in *IEEE Int. Workshop Design and Test of Nano Devices, Circuits and Systems*, Cambridge, MA, USA, Sep. 29–30 2008, pp. 49–52.

[188] D. Milosavljevic and S. Cotofana, "A method to analyze the fault tolerance of molecular quantum-dot cellular automata systems," in *Proc. Int. Semiconductor Conference*, Sinaia, Romania, Sep. 27–29, 2006, pp. 399–402.

[189] B. K. Sikdar, "Study of N-detectability in QCA designs," in *IEEE Asian Test Symp.*, Fukuoka, Japan, Nov. 20–23, 2006, pp. 183–188.

[190] X. Ma, J. Huang, C. Metra, and F. Lombardi, "Reversible gates and testability of one dimensional arrays of molecular QCA," *J. Electron. Test.*, vol. 24, no. 1–3, pp. 297–311, Jun. 2008.

[191] ——, "Reversible and testable circuits for molecular QCA design," in *Emerg. Nanotechnol.*, ser. Frontiers in Electron. Test., M. Tehranipoor, Ed. New York, NY, USA: Springer US, 2008, vol. 37, ch. 6, pp. 157–202.

[192] ——, "Testing reversible 1D arrays for molecular QCA," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Arlington, VA, USA, Oct. 4–6, 2006, pp. 71–79.

[193] P. Gupta, N. Jha, and L. Lingappan, "A test generation framework for quantum cellular automata circuits," *IEEE Trans. VLSI Syst.*, vol. 15, no. 1, pp. 24–36, Jan. 2007.

[194] ——, "Test generation for combinational quantum cellular automata (QCA) circuits," in *Proc. Design, Automation & Test in Europe Conf.*, vol. 1, Munich, Germany, Mar. 6–10, 2006, pp. 1–6.

[195] F. Karim, K. Walus, and A. Ivanov, "Testing of combinational majority and minority logic networks," in *IEEE Int. Mixed-Signals, Sensors, and Systems Test Workshop*, Vancouver, BC, Canada, Jun. 18–20, 2008, pp. 1–6.

[196] T. Wei, K. Wu, R. Karri, and A. Orailoglu, "Fault tolerant quantum cellular array (QCA) design using triple modular redundancy with shifted operands," in *Proc. Asia and South Pacific Design Automation Conf.*, Shanghai, China, Jan. 18–21, 2005, pp. 1192–1195.

[197] I. Koren, *Computer Arithmetic Algorithms*, 2nd ed. Natick, MA, USA: A. K. Peters, Ltd., 2002.

[198] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*. New York, NY, USA: Oxford Univ. Press, 2000.

[199] A. Vetteth, K. Walus, V. Dimitrov, and G. Jullien, "Quantum-dot cellular automata carry-look-ahead adder and barrel shifter," in *Proc. IEEE Conf. Emerging Telecommunications Technologies*, Dallas, TX, USA, Sep. 23–24, 2002.

[200] W. Wang, K. Walus, and G. Jullien, "Quantum-dot cellular automata adders," in *Proc. IEEE Conf. Nanotechnology*, San Francisco, CA, USA, Aug. 11–14, 2003, pp. 461–464.

[201] A. Fijany, N. Toomarian, K. Modarress, and M. Spotnitz, "Bit-serial adder based on quantum dots," NASA's Jet Propulsion Laboratory, Pasadena, CA, USA, Tech. Rep. NPO-20869, Jan. 2003.

[202] K. Walus, G. Jullien, and V. Dimitrow, "Computer arithmetic structures for quantum cellular automata," in *Rec. Asilomar Conf. Signals, Systems and Computers*, Pacific Grove, CA, USA, Nov. 9–12, 2003, pp. 1435–1439.

[203] H. Cho and E. Swartzlander, "Pipelined carry lookahead adder design in quantum-dot cellular automata," in *Rec. Asilomar Conf. Signals, Systems and Computers*, Pacific Grove, CA, USA, Oct. 30–Nov. 2, 2005, pp. 1191–1195.

[204] ——, "Modular design of conditional sum adders using quantum-dot cellular automata," in *Proc. IEEE Conf. Nanotechnology*, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 363–366.

[205] H. Cho, "Adder and multiplier design and analysis in quantum-dot cellular automata," Ph.D. dissertation, Univ. Texas at Austin, Austin, TX, USA, 2006. [Online]. Available: http://hdl.handle.net/2152/2848

[206] O. MacSorley, "High-speed arithmetic in binary computers," *Proc. IRE*, vol. 49, no. 1, pp. 67–91, Jan. 1961.

[207] N. Mehmood, M. Hansson, and A. Alvandpour, "An energy-efficient 32-bit multiplier architecture in 90-nm CMOS," in *Proc. Norchip*, Linköping, Sweden, Nov. 20–21, 2006, pp. 35–38.

[208] M. Sjalander and P. Larsson-Edefors, "High-speed and low-power multipliers using the Baugh-Wooley algorithm and HPM reduction tree," in *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, St. Julian's, Malta, Aug. 31–Sep. 3, 2008, pp. 33–36.

[209] N. A. Touba, "Fault-tolerant design," in *System-on-Chip Test Architectures*, L.-T. Wang, C. E. Stroud, and N. A. Touba, Eds. Burlington, MA, USA: Morgan Kaufmann, 2008, pp. 123–170.

[210] T. Dysart and P. Kogge, "Probabilistic analysis of a molecular quantum-dot cellular automata adder," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Rome, Italy, Sep. 26–28, 2007, pp. 478–486.

[211] S. Krishnaswamy, G. Viamontes, I. Markov, and J. Hayes, "Accurate reliability evaluation and enhancement via probabilistic transfer matrices," in *Proc. Design, Automation & Test in Europe Conf.*, Munich, Germany, Mar. 7–11, 2005, pp. 282–287.

[212] S. Krishnaswamy, I. Markov, and J. Hayes, "Tracking uncertainty with probabilistic logic circuit testing," *IEEE Des. Test. Comput.*, vol. 24, no. 4, pp. 312–321, Apr. 2007.

[213] S. Krishnaswamy, G. Viamontes, I. Markov, and J. Hayes, "Probabilistic transfer matrices in symbolic reliability analysis of logic circuits," *ACM Trans. Design Autom. Electron. Systems*, vol. 13, no. 1, Jan. 2008.

[214] L. Bonci and M. Macucci, "Numerical investigation of energy dissipation in quantum cellular automaton circuits," in *Proc. European Conf. Circuit Theory and Design*, vol. 2, Cork, Ireland, Aug. 29–Sep. 1, 2005, pp. 239–242.

[215] ——, "Analysis of power dissipation in clocked quantum cellular automaton circuits," in *Proc. European Solid State Circuits Conf.*, Montreux, Switzerland, Sep. 19–21, 2006, pp. 58–61.

[216] L. Bond and M. Macucci, "Analysis of power dissipation in clocked quantum cellular automaton circuits," in *Proc. European Solid-State Device Research Conf.*, Montreux, Switzerland, Sep. 19–21, 2006, pp. 57–60.

[217] S. Frost-Murphy, M. Ottavi, M. Frank, and E. DeBenedictis, "On the design of reversible QDCA systems," Sandia Nat. Lab., Albuquerque, NM, and Livermore, CA, USA, Tech. Rep. SAND2006-5990, 2006.

[218] J. Huang, X. Ma, and F. Lombardi, "Energy analysis of QCA circuits for reversible computing," in *Proc. IEEE Conf. Nanotechnology*, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 39–42.

[219] S. Srivastava, S. Sarkar, and S. Bhanja, "Power dissipation bounds and models for quantum-dot cellular automata circuits," in *Proc. IEEE Conf. Nanotechnology*, Cincinnati, OH, USA, Jun. 17–20, 2006, pp. 375–378.

[220] M. Frank, "Introduction to reversible computing: Motivation, progress, and challenges," in *Proc. ACM Int. Conf. Computing Frontiers*, Ischia, Italy, May 4–6, 2005, pp. 385–390.