

Steven Bos

Beyond 0 and 1:

A mixed radix design and verification workflow
for modern ternary computers

**Dissertation for the
degree of Ph.D**
Technology

Faculty of Technology, Natural
Sciences and Maritime Studies

Steven Bos

Beyond 0 and 1:

A mixed radix design and verification workflow for modern ternary computers

A PhD dissertation in
Technology

© Steven Bos, 2024

Faculty of Technology, Natural Sciences and Maritime Studies
University of South-Eastern Norway
Kongsberg

Doctoral dissertations at the University of South-Eastern Norway no. 189

ISSN: 2535-5244 (print)

ISSN: 2535-5252 (online)

ISBN: 978-82-7206-854-6 (print)

ISBN: 978-82-7206-855-3 (online)



This publication is licensed with a Creative Commons license. You may copy and redistribute the material in any medium or format. You must give appropriate credit, provide a link to the license, and indicate if changes were made. Complete license terms at <https://creativecommons.org/licenses/by-nc-sa/4.0/deed.en>

Print: University of South-Eastern Norway

To my heroes Koen & Susan Bos-Theuvenet

Preface

This dissertation is submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy (PhD) in Technology from the Department of Science and Industry Systems at the University of South-Eastern Norway (USN). The doctoral work presented here took place between June 1, 2019 and June 1, 2023. The majority of the research was conducted at USN campus Kongsberg in Norway and with some work done at USN campuses Vestfold and Porsgrunn. The work has been done under the supervision of Assoc. Professor Henning Gundersen (USN) and Professor Nils-Olav Skeie (USN). Further guidance was received from the midterm evaluation committee Professor Philipp D. Häfliger from the University of Oslo (UiO) and Professor Lars M. Johansen (USN).

The PhD work was financially supported by the Norwegian Ministry of Education and Research (Kunnskapsdepartementet) and USN as a 4 year PhD Research Fellowship (KD-stilling) with 25% teaching duties.

The candidate is a member of the IEEE MVL and CAS societies as well as the research school for Training the Next Generation of Micro- and Nanotechnology Researchers in Norway (TNNN).

Acknowledgements

During my four year PhD journey I had the pleasure of meeting wonderful and inspiring people and perhaps a new version of myself. Moving from The Netherlands to Norway with my wife and learning the Norwegian language and rich culture was not possible without a long list of people mentioned below – *mea culpa* if I missed some of you!

Without my supervisors Assoc. Professor Henning Gundersen and Professor Nils-Olav Skeie my adventure in the world of ternary computing would not have started. I am deeply grateful for the time they spend with me.

Research is not one solitary endeavour nor does it start at zero as we always build on the insights made before us. Working closely with my fellow co-authors was a true privilege and I like to thank them again for their contributions. Collaborating with Professor Knut E. Aasmundtveit and Assoc. Professor Avisek Roy was immensely valuable as I was able learn about the possibilities of carbon nanotubes and nanofabrication at USN. Thank you for sharing your knowledge and I look forward to continuing our collaboration.

The knowledge landscape is shaped by giants and discussing my research on multi-state memristors with the inventor of these devices, Professor Leon Chua from UC Berkeley was extraordinary. At the start of my PhD I stumbled upon the work of professor Kris

Campbell from Boise State University on programming memristors. Her detailed work inspired me to investigate the analog switching properties of resistive memory. I also like to thank the many great minds I met both in-person and virtually due to the COVID pandemic.

A huge thanks to Professor Morten Melaaen (dean), Assoc. Professor Elisabet Syverud (head of department), Professor Olaf Hallan Graven (former head of department) and Rune Romnes for providing me with a great work environment and resources. Thanks to the USN PhD committee and PhD coordinators Mariken Kjøhl-Røsand, Per Morten Hansen and Siri Luise Tveitan for guidance, structure and organizing the inspiring PhD forums. I must mention the colleagues I had the pleasure to teach three courses with the past four years: Professor Dag Samuelson, Joakim Bjørk, Richard Thue and dr. Richard Anthony. The teaching experience was both academically rewarding and fun. *Tusen hjertelig takk til* Assoc. Professor Sigmund Gudvangen for teaching me the nuances in bokmål and nynorsk and Karoline Moholth Mcclenaghan for sharing her thrilling stories about the first computers in Norway. I am very grateful to Professor Rigmor Baraas from the department of Optometry to allow me to present my PhD work to the Norwegian prime minister Jonas Gahr Støre.

Starting a new research group together with Assoc. Prof. Henning Gundersen brought structure to my work. Meeting weekly with members of the Ternary Research Group, discussing and disseminating the tiniest results was a true joy. I like to thank all members present and past for input on papers and their open-mindedness to explore an unconventional computing paradigm. I especially enjoyed the boundless energy of dr. Radmila Juric.

I had the opportunity to supervise four bright MSc students Halvor Nybø Risto (now PhD candidate), Julian Breivold Nilsen, Mehtab Singh Virk and Erika Fegri and assisted several BSc groups. Thanks for sharing the long hours in pursuit of the adrenaline rush of inevitable progress. I like to thank post-doc dr. Fahim Ahmed Salim and my fellow PhD candidates for an awesome time: Walter Kibet Yego, Haytham Ali, Rune Andre Haugen, Tommy Langen, Agnieszka Lach, Soheila Taghavi Hosnaroudi and Raghav Sikka.

Sharing my work with my old and new friends, brother Niels and sister Jenna-Fay and the always curious Bos and Theuvenet family was immensely relevant. It forced me to find new metaphors and take different viewpoints. Thank you for your support!

This dissertation would not be possible without the unwavering support and extreme patience of my wife Jessica Stokhof. I am hugely indebted and will start returning the immense favor with this ternary thank you: $3^{\text{thank you}}$

Steven Bos

Kongsberg, 1st November 2023

Abstract

For more than 80 years digital computers use the radix-2 or *binary* computer alphabet as their lowest symbolic and physical representation. This doctrine of computing is presumed in every modern computer. The radix economy theorem derives that radix-3 or *ternary* is however the optimal radix. Ternary is the first radix in the Multiple-Valued Logic (MVL) family that enables symmetrical arithmetic using the balanced ternary notation. The ongoing challenge is to engineer devices, circuits and systems that can physically represent three logic levels with competitive power, performance, area and cost metrics. For flash storage and communication MVL is already the industry standard, but logic remains binary. Ever since Dennard scaling stopped in 2005, binary computing is struggling to overcome the increasing power wall, memory wall and Electronic Design and Automation (EDA) wall. A unified MVL compute paradigm can theoretically address these challenges, making it a prime candidate for the beyond-CMOS era.

This article-based thesis is structured in three parts. In the first part binary computing is discussed. The historical reasoning for this choice as well as the current scaling challenges that impede its future were reviewed. The part concludes with a review of several fundamental and engineering limits that are rarely cited but highly relevant when considering another radix such as Shannon's noisy channel theorem and Rent's rule.

In the second part ternary computing is discussed. A brief overview of radix-3 theory and literature is presented. A novel radix comparison methodology is proposed to improve fairness. Historical efforts to build ternary computers were reviewed which started in the 1950's. A categorization of the main benefits of balanced ternary is presented across 7 application domains. The part concludes with an overview of the critique on radix-3.

In the third part practical aspects of ternary computing are discussed: multi-stable devices and EDA tooling. For devices, non-volatile ternary memory control with commercially available memristors was studied. A novel open source software tool uMemristorToolbox and hardware platform for multi-state memristor programming were developed. The experiments confirm that ternary memory with memristors is both feasible and low-cost.

Lastly, EDA tooling and workflows for ternary logic chips are discussed. The open source software tool Mixed Radix Circuit Synthesizer (MRCS) was developed, the first browser-based EDA tool to design and verify binary, ternary and hybrid (mixed radix) circuits. It features a novel MVL circuit synthesis algorithm with HSPICE and verilog output targeting CMOS and multi-threshold CNTFET. The tool was used to design REBEL-2, a novel balanced ternary CPU with RISC-V-like ISA. Four MRCS designs have been tested on a FPGA and submitted for tape-out using the Openlane ASIC workflow.

Keywords: ternary microprocessor, design automation, integrated circuit synthesis

Contents

Preface	III
Abstract	V
Contents	IX
List of Papers	XI
List of Co-supervised Projects	XIII
List of Figures	XVI
List of Tables	XVII
Nomenclature	XIX
1 Introduction	1
1.1 The computer alphabet	1
1.2 Motivation and scope	2
1.3 Historical evolution of binary computing	6
1.4 Moore's curse: 3 scaling walls	9
1.4.1 Power wall	10
1.4.2 Memory wall	11
1.4.3 EDA wall	14
1.5 The fundamental limits of computing	15
1.5.1 Shannon's limit	16
1.5.2 Landauer's limit	19
1.5.3 Radix economy	20
1.5.4 Rent's rule	23
1.6 Research objective and dissertation structure	24
2 The benefits of ternary	27
2.1 Introduction	27
2.2 Ternary basics	29
2.2.1 Heptavintimal notation	29
2.2.2 The third value	30
2.2.3 Mixed radix	31
2.3 Radix comparison methodology	32
2.4 Historical evolution of ternary computing	34
2.5 The seven C's of ternary	36
2.5.1 Computation	36
2.5.2 Communication	37
2.5.3 Energy Consumption	38
2.5.4 Compression	39

Contents

2.5.5	Comprehension	40
2.5.6	Cyber-Security	40
2.5.7	Design complexity	40
2.6	Critique	41
2.7	Conclusion	43
3	Multi-state RRAM development platform	45
3.1	Introduction	45
3.2	Multi-state programming	46
3.3	uMemristorToolbox: A new tool for experimenting with multi-state RRAM . . .	49
3.3.1	Motivation	49
3.3.2	Architecture	50
3.3.3	Experiments	51
3.3.4	Application: Embedded ternary system	54
3.4	Ternary memory controller circuit	55
3.4.1	Simulation	56
3.4.2	Implementation	56
3.5	Conclusion	57
4	Mixed radix EDA for ternary computers	59
4.1	Introduction	59
4.2	MRCs: A new tool for mixed radix design and verification	60
4.2.1	Motivation	60
4.2.2	Architecture	61
4.2.3	Workflows	63
4.3	Mixed radix synthesis engine	65
4.3.1	Introduction	65
4.3.2	Related work	66
4.3.3	Mixed radix synthesis algorithm	67
4.3.4	Binary coded ternary RTL	69
4.4	REBEL-2 Balanced Ternary CPU	71
4.4.1	Motivation	72
4.4.2	Balanced Ternary Instruction Set Architecture	73
4.4.3	Implementation	76
4.5	Radix conversion	77
4.5.1	Binary to Ternary	77
4.5.2	Ternary to Binary	78
4.6	Conclusion	79
5	Discussion	81
5.1	Towards a ternary technology stack	81
5.2	Open questions	82

6 Conclusion	83
Bibliography	87
A uMemristorToolbox: Open source framework to control memristors	113
B Automated synthesis of ternary logic functions in CNTFET circuits	121
C Post-Binary Robotics: Using memristors with ternary states	127
D Ternary computing; The future of IoT?	135
E High speed bi-directional binary-ternary interface with CNTFETS	143
F Ternary and mixed radix CNTFET circuit design, simulation and verification	151
G Additional material	159
G.1 Continuous-time and discrete-time signals	160
G.2 Rebuttal of Buchholz' 9 arguments	161
G.3 A radix compatible form of the CMOS power equation	162
G.4 Using the radix economy argument	164
G.5 Overhead calculation	166
G.6 Comparing baselines to 58.5% or to 63.1%	167
G.7 58.5% is an information limit, not a system limit	168
G.8 Ternary computers architectures from 2004-2022	169
G.9 Experimental 2-trit memristor results using uMemristorToolbox	171
G.10 Multi-state RRAM development platform prototype	173
G.11 Getting started with MRCS	174
G.12 MRCS Limitations	176
G.13 Ternary algebra	179
G.14 Towards a ternary standard cell library	181
G.15 Combinatorial and sequential building blocks	185
G.16 Subcomponents	194
G.17 Online radix conversion tool	197
H TNNN 2023: Ternary VLSI with CMOS using MRCS	199
I Tape-outs	203

List of Papers

Article 1

S. Bos, H. Gundersen and F. Sanfilippo, "uMemristorToolbox: Open source framework to control memristors in Unity for ternary applications", *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)*, Virtual Conference, Japan, 2020, pp. 212-217, doi: 10.1109/ISMVL49045.2020.000-3.

Article 2

H. N. Risto, **S. Bos** and H. Gundersen, "Automated synthesis of netlists for ternary-valued n-ary logic functions in CNTFET circuits", *2020 Proceedings of the 61st Conference on Simulation and Modelling (SIMS)*, Virtual Conference, Finland, 2020, pp. 483-485, doi: 10.3384/ecp20176483

Article 3

S. Bos, J. B. Nilsen and H. Gundersen, "Post-Binary Robotics: Using Memristors With Ternary States for Robotics Control", *2020 IEEE 8th Electronics System-Integration Technology Conference (ESTC)*, Virtual Conference, Norway, 2020, pp. 1-6, doi: 10.1109/ESTC48849.2020.9229820.

Article 4

H. Gundersen and **S. Bos**, "Ternary Computing; The future of IoT?", *2021 25th Proceedings of the Society for Design and Process Science (SDPS)*, Virtual Conference, Norway, 2021, pp. 43-47, link: sdpsnet.org/sdps/documents/sdps-2021/SDPS%202021%20Proceedings.pdf

Article 5

S. Bos, H. N. Risto and H. Gundersen, "High speed bi-directional binary-ternary interface with CNTFETS", *2021 25th Proceedings of the Society for Design and Process Science (SDPS)*, Virtual Conference, Norway, 2021, pp. 38-42, link: sdpsnet.org/sdps/documents/sdps-2021/SDPS%202021%20Proceedings.pdf

Article 6

S. Bos, H. N. Risto and H. Gundersen, "Beyond CMOS: Ternary and mixed radix CNTFET circuit design, simulation and verification", *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, Austin, TX, USA, 2022, pp. 80-85, doi: 10.1109/ISCAS48785.2022.9937259.

List of Co-supervised Projects

1. **Halvor Nybø Risto**, "A study of CNTFET implementations for Ternary Logic and Data Radix Conversion", *Master Thesis*, USN, 2020.
2. **Julian Breivold Nilsen**, "Memristor Implementation of a Ternary Storage Circuit", *Master Thesis*, USN, 2020.
3. **Mehtab Singh Virk**, "Memristor Development Platform - Dual Source Control For Implementations of Multi-state Memristive Memory", *Master Thesis*, USN, 2022.
4. **Erika Fegri**, "Design of a Balanced Ternary Tri-directional Loadable Counter Using CNTFETs", *Master Thesis*, USN, 2022.

List of Figures

1.1	Performance, Power and Area: 48 years of CPU innovation	4
1.2	The computer technology stack	5
1.3	The power wall and dark silicon trend	10
1.4	The memory wall	12
1.5	The memory pyramid	13
1.6	The EDA wall	14
1.7	Trade-offs to increase the radix	19
2.1	20 year IEEE Explore trend for search "ternary AND comput*"	28
3.1	The 3 regions of voltage-controlled bi-polar memristance programming . .	48
3.2	Architecture of uMemristorToolbox with memristor development platform	50
3.3	Board check experiment	51
3.4	DC experiment	52
3.5	Random write experiment	53
3.6	Multi-state programming scheme interface	53
3.7	Retention experiment	54
3.8	ADC experiment	55
3.9	ADC experiment log	55
4.1	MRCS architecture and workflows	61
4.2	User interface of the developer version of MRCS	62
4.3	The RTL-to-GDS ASIC flow from OpenLane	64
4.4	Schematic and simulation excerpt of a BCT counter in Vivado	65
4.5	The mixed radix synthesis algorithm	68
4.6	Logic transformation from truth table to CNTFET implementation	69
4.7	BCT implementation of an STI made with MRCS	71
4.8	REBEL-2 balanced ternary CPU and 9-instruction RISC-like ternary ISA .	75
4.9	382T 4-bit signed binary to 4-trit balanced ternary radix converter	77
4.10	Radix converter circuits for signed binary and balanced ternary	79
G.1	Comparison of digital and analog signals	160
G.2	Measurement of nine memristance levels	171
G.3	Simulation of nine memristance levels	172
G.4	A novel multi-state RRAM development platform	173
G.5	First PCB implementation of the multi-state RRAM development platform	173
G.6	Overview of used memristors	173
G.7	Mixed Radix Circuit Synthesizer (MRCS) user experience	174
G.8	Verilog workaround for BCT with ternary d-latch	176

List of Figures

G.9	Heptavintimal implementation in MRCS	178
G.10	28T gated balanced ternary d-latch based on 2:1 MUX	185
G.11	46T gated balanced ternary d-latch based on NMIN	185
G.12	54T rising-edge master-slave configuration balanced ternary d-flip-flop . . .	186
G.13	52T rising-edge master-slave configuration unbalanced ternary d-flip-flop .	186
G.14	76T DDR master-slave configuration balanced ternary d-flip-flop	187
G.15	110T QDR master-slave configuration balanced ternary d-flip-flop	187
G.16	80T balanced ternary register	188
G.17	RAM-3 implementation with three d-flip-flops	188
G.18	Ternary ROM/RAM	189
G.19	110T BTA design with SUM-based CARRY	190
G.20	Balanced ternary ripple counter	191
G.21	2-trit balanced ternary ripple counter	191
G.22	1-trit synchronous balanced ternary program counter	192
G.23	4-trit synchronous balanced ternary program counter	193
G.24	MUX level 2 implementation	194
G.25	MUX level 1 implementation	194
G.26	DEMUX level 2 implementation	194
G.27	DEMUX level 1 implementation	194
G.28	binary XOR-3 implementation	194
G.29	binary HA-3 implementation	194
G.30	Conditional-STI-3 implementation	195
G.31	4-bit unsigned binary to balanced ternary radix converter	195
G.32	3-trit balanced ternary to 4-bit signed binary radix converter	196
G.33	Online radix converter tool	197

List of Tables

1.1	Relation between discrete radices, compactness and ambiguity	3
1.2	Relation between chapters and relevant papers	25
2.1	Radix-2 and radix-3 MAX truth tables	29
2.2	Heptavintimal (radix-27) encoding	30
4.1	Variants of binary coded balanced ternary	70
G.1	Bi-stable subset of ternary unary functions	180
G.2	Overview of useful arity-1 building blocks	182
G.3	Overview of useful arity-2 building blocks	183
G.4	Overview of useful arity-3 building blocks	184

Nomenclature

Symbol	Explanation
3VL	Three-Valued Logic
ADC	Analog Digital Conversion
ALU	Arithmetic Logic Unit
ASIC	Application Specific Integrated Circuit
BCD	Binary Coded Decimal
BCT	Binary Coded Ternary
BTA	Balanced Ternary Adder
CMOS	Complementary Metal-Oxide Semiconductor
CNTFET	Carbon Nanotube Field-Effect Transistor
CPU	Central Processing Unit
DRAM	Dynamic Random Access Memory
EDA	Electronic Design Automation
FPGA	Field-Programmable Gate Array
HRS	High Resistance State
IC	Integrated Circuit
ISA	Instruction Set Architecture
LRS	Low Resistance State
MOSFET	Metal Oxide Silicon Field Effect Transistor
MRCS	Mixed Radix Circuit Synthesizer
MVL	Multiple-Valued Logic
NTI	Negative Ternary Inverter
PCB	Printed Circuit Board
PDP	Power Delay Product
PPA(C)	Performance, Power, Area (and Cost)
PTI	Positive Ternary Inverter
PVT	Process, Voltage, Temperature
RBR	Redundant Binary Representation
REBEL	RISC-V-like Energy efficient Balanced tErnary Logic
RISC	Reduced Instruction Set Computer
RRAM	Resistive Random Access Memory
RTL	Register Transfer Level
SPICE	Simulation Program with Integrated Circuit Emphasis
SRAM	Static Random Access Memory
STI	Standard Ternary Inverter
TCB	Ternary Coded Binary
VLSI	Very Large Scale Integration

—“At an early point in my thinking about digital computers, I looked at the effects of a change in the base of the number system. How would the structure of a computing machine depend on this choice?”

Prof. John V. Atanasoff, Inventor of the binary computer [1]

—“Computer designers have a common goal: to find a language that makes it easy to build the hardware and the compiler while maximizing performance and minimizing cost and energy”

Prof. Patterson and Prof. Hennessy, authors of Computer Organization and Design in [2]

1

Introduction

1.1 The computer alphabet

For more than 80 years digital computers read, write and “think” in zeros and ones. Remarkably, long and complex patterns of just these two symbols control computers to land a spaceship on the moon, forecast the next Aurora Borealis or provide endless digital entertainment. Computers have propelled us from the industrial age to the information age and these two symbols are at the center. In contrast, most people perform mental calculation with 10 symbols, the digits 0-9. English speakers use an alphabet of 26 symbols. Ancient Greek numerals combined the two symbol sets as each number corresponded to a letter in the alphabet. The fact that most scholars associate α (alpha) with *one* or *first* is a direct consequence of this.

The two-letter computer alphabet structures a computers lowest level language and is known as *binary*. In mathematics and computer science the number of unique symbols in a set to represent a number is called the base or radix (Latin: root) and is analog to the amount of letters in an alphabet. Binary is thus radix-2, ternary with three symbols is radix-3 and denary with its ten symbols 0-9 is radix-10. To express a large number with a set of smaller numbers, a system of transformation rules and notation is required. The ancient Greek numeral system was superseded by the the Hindu-Arabic positional numeral system as it had several advantages. It introduced symbols exclusively for numbers and used positions for order of magnitude. For example, the decimal number “9” (nine) uses one position, while the decimal number “10” (ten) uses two. Crucially, the symbol pattern “10” can be interpreted as something other than a number and is the reason why modern computers should be considered symbol or computer language processors rather than arithmetic processors. Patterson and Hennessy wrote [2]:

1 Introduction

”No matter what the instruction set or its size — RISC-V, MIPS, ARM, x86— never forget that bit patterns have no inherent meaning. The same bit pattern may represent a signed integer, unsigned integer, floating-point number, string, instruction, and so on. In stored-program computers, it is the operation on the bit pattern that determines its meaning.”

In *Origins of Language* [3] Tomasello wrote that human language differs from other animal species in two main ways: humans use symbols and grammar. Central in human language are the various writing systems [4], a system for recording and conveying messages such as the alphabet. Man [5] claims the alphabet to be ”*one of humanities greatest ideas*”. In alphabetic writing systems the smallest units for causing a contrast in meaning are called *graphemes* for symbols. For the physical representation of speech as sound patterns the smallest units are called *phonemes*. Both Pae [4] and Crystal [6] write (paraphrased) ”in a perfect regular system there is one grapheme for each phoneme”. Such a system allows compact encoding of the language while at the same time offer great expressive power needed for labelling new concepts [4]. Alphabetic writing system are recognized as the most economic and versatile of all writing systems [4]. The alphabet size of human languages varies from 11 in Rotokas to 74 letters in Khmer [6]. In modern computers graphemes are depicted as the digits 0 and 1 which correlates 1:1 to their physical representation often expressed as a range of voltage levels such as 0 V - V_{DD} for symbol 0 and V_{DD} for symbol 1. Other electrical quantities can be used as physical representation such as resistance as well as other energy domains such as mechanical, thermal and optical.

Power is the number one design constraint for designing computers [7], [8]. Interestingly, modern computers spend 1000x more energy on communication than on computation and this is increasing with every new generation of smaller chips [9], [10]. If language and its structure is so efficient for humans why did the pioneers of electronic computing Atanasoff, von Neumann and others advocate for binary? Does a richer computer alphabet, a higher radix, have the same benefits as it does for humans? If so, then uprooting this foundation of computing is a radical paradigm shift that effects all digital computers.

1.2 Motivation and scope

The question of radix, namely choosing the alphabet size, has been prevalent since the ancient predecessor of the digital computer; the abacus. The Sumerian version of this mechanical counting device used radix-60 around 2700 BC while other cultures adopted radix-10 [11, p. 11]. The choice for radix-10 for addition and subtraction was based on the 10 human fingers [12]. The importance of choosing a radix can be observed in table 1.1, where the numbers 0-10 are encoded in radix-1, radix-2, radix-3, radix-8 and radix-10. The formula for encoding any positive number n in radix form is Eq. 1.1 [13]. The solution is unambiguous only for that radix.

$$n = x_p * r^p + x_{p-1} * r^{p-1} + \dots + x_0 \quad (1.1)$$

Where radix r and position p are non-negative integers and $x_i \in 0..r - 1$

Table 1.1: Relation between discrete radices, compactness and ambiguity

Decimal Number (Radix-10)	Radix			
	Radix-1	Radix-2	Radix-3	Radix-8
0	\emptyset_1	0_2	0_3	0_8
1	0_1	1_2	1_3	1_8
2	00_1	10_2	2_3	2_8
3	000_1	11_2	10_3	3_8
4	0000_1	100_2	11_3	4_8
5	00000_1	101_2	12_3	5_8
6	000000_1	110_2	20_3	6_8
7	0000000_1	111_2	21_3	7_8
8	00000000_1	1000_2	22_3	10_8
9	000000000_1	1001_2	100_3	11_8
10	0000000000_1	1010_2	101_3	12_8

Less positions are needed for the same information when using higher radices (plural of radix). In practice positions need to be physically implemented as a type of switch such as a transistor or memory cell. Each switch uses space, has a transition time, consumes energy and produces heat when switching. It also has design and fabrication costs tied to it. These properties are captured in the industry standard metric Performance, Power, Area and Costs (PPAC). The implications of radix are directly tied to the amount of positions, thus amount of switches needed. If a switch can process more symbols while having the same dimensions information density increases. Fewer switches for the same functionality can be translated into benefits such as smaller chips, reduced power consumption and power dissipation as heat, increased chip performance and decreased costs.

A nearly 50 year trace of performance, power and area of the central processing unit (CPU) is shown in Fig. 1.1. A similar trace for costs can be found in [15], showing 7 decades of decline of cost/transistor in the same period. A more detailed analysis of the cost trends can be found in [16]. The transistor density trend in Fig. 1.3 shows that technically Gordon Moore's observation known as "Moore's law" [17] of doubling transistor density every two years and reducing cost by half in the same period was mostly accurate for this period. Moore's law has become a roadmap for the semiconductor industry although the actual driver was Dennard's scaling law [18]. After this scaling stopped around 2005, advances in performance, frequency and power all stalled. This will be discuss in more depth in the Moore's Curse section below.

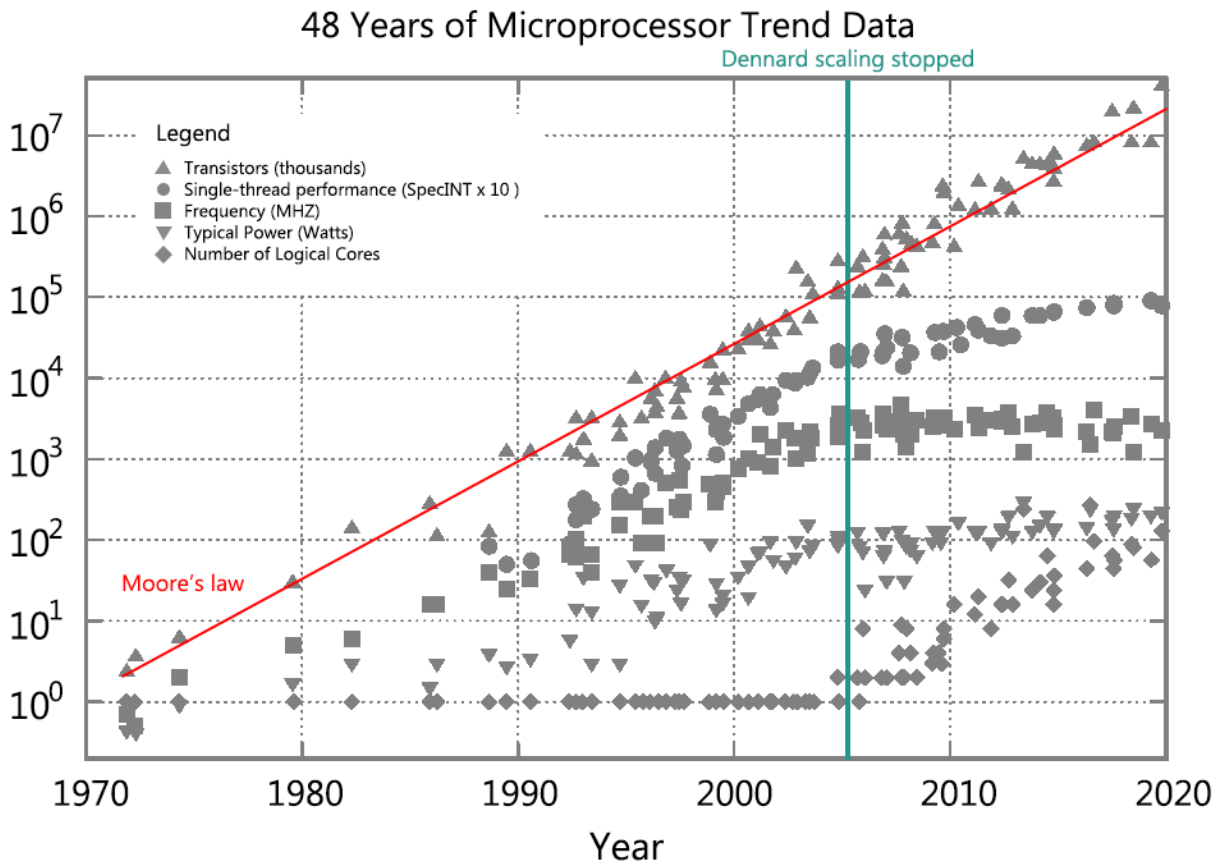


Figure 1.1: Performance, Power and Area: 48 years of CPU innovation. Adapted from [14]

Nature’s solution to information processing, the brain, is both compact and about 5-6 orders of magnitude more efficient than modern computers [10]. Bullock et al. [19] describes the difficulties of modelling the brain as discrete cellular switches with binary action potential (AP), the neuron doctrine:

”In 1959, it was realized that much of the information processing by neurons involves electrical events that are graded in amplitude and decay over distance rather than all-or-nothing electrical spikes that propagate regeneratively”.

The brain-inspired computing paradigm *neuromorphic computing* acknowledges that binary digitization is inefficient and uses analog and binary signals - a mixed signal architecture [20]. This includes multiple-valued logic or mixed radix signals [21]. A paper by Sengupta et al. [22, p. 748] mentions that *”neurons produce a myriad of APs with different shapes and varying heights and width”*. They also show an example of an AP with a similar voltage level but different current levels [22, p. 747] indicating multiple values.

Higher radix computing is studied under the umbrella term multiple-valued logic (MVL). Synonymous terms are m-valued, p -valued, multi-valued, many-valued, multi-level and multi-valent logic. Related terms found in literature to indicate digital technology with more than 2 discrete levels include multi-bit, multi-state, multi-stable, complex states and post-binary. Radix-3 (base-3) or three-valued Logic (3VL) is the first radix in the MVL family and is synonymously called ternary, trinary or tri-stable in literature. Many heavyweights in the computer science field including Shannon [23] and Knuth [24] have recognized the unique arithmetic properties of balanced ternary (symmetrical around zero) for computing. Radix-3 is the first radix with the symmetry property. Ternary is the closest integer to optimum e (≈ 2.71) discussed in the Fundamental limits section below. Higher radices give a higher information density but the economy has a diminishing return. Radix-3 is also the minimum radix to higher-order logic with three truth assessments: True, False, Unknown. Ternary logic stands in contrast to Aristotelian/Boolean logic that can only assign True or False. The classical bivalent view is also called *tertium non datur* or the law of the excluded middle. However, this principle limits reasoning about the world tremendously as no future or unverifiable statements are possible. Some statements might not be true or false *now*, but might be *later*.

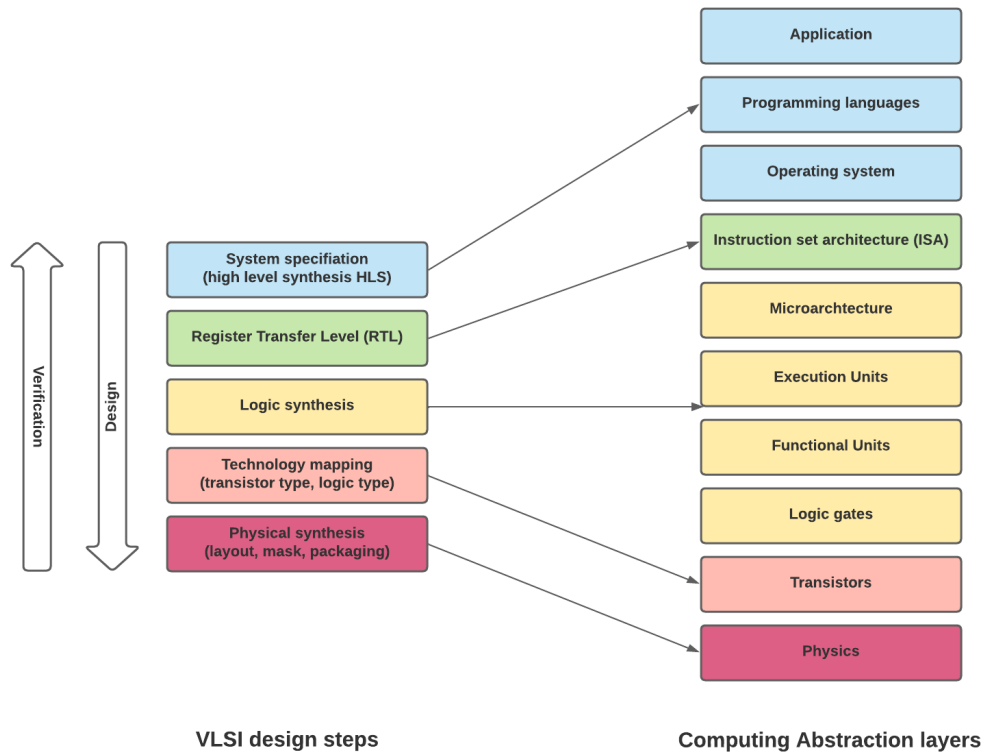


Figure 1.2: The computer technology stack. Layers of abstraction transform atoms into applications. Design and verification steps are supported by EDA tools and flows.

1 Introduction

The choice of radix has effect on every aspect of a computer; the hardware and software. The computer technology stack in Fig. 1.2 shows the many abstraction between the individual atoms and the application that interacts with them. Each abstraction is designed and verified by Electronic Design Automation (EDA) tools and elaborate workflows. In many treatments on computing such as [25], the first concept assumed or explained is the binary radix. This work re-investigates the feasibility of radix-3 computing after the first (and last) commercial ternary computer, the 1958 Russian Setun [26]–[28]. Emphasizes is put on EDA tooling and workflows for ternary Very Large Scale Integration (VLSI). The explored transistor technologies are limited to Carbon Nanotube Field-Effect Transistors (CNTFET) and Metal-Oxide-Field-Effect Transistors (MOSFET) for ternary logic. Complementary MOSFETs are made with a CMOS process and is the industry standard to create high transistor density binary logic chips. CNTFET are a potential candidate to replace MOSFETs [29] and have unique properties to create ternary signals [30]. They have been demonstrated to be compatible with CMOS [31]–[33]. For ternary memory commercially available Resistive-RAM (RRAM or ReRAM) is explored. RRAM is commonly known as memristors [34]. Memristor are a potential candidate for beyond CMOS devices and have desirable properties for MVL such as multiple states per devices [35], [36].

1.3 Historical evolution of binary computing

Early computers were designed to be highly specialised arithmetic machines. They excelled in solving large systems of equations for applications like missile guidance, decryption and atomic bomb calculations [37], [38]. Although the term digital did not exist in the 1940's, the input and output data on digital computers was done in the radix-10 Hindu-Arabic positional numeral system and could be interpreted by human operators without conversion. A brief primer on digital and analog signals in the context of radix is shown in **Appendix G.1**. The transition from radix-10 (denary) to radix-2 (binary) computer alphabets is due to a long chain of events described in detail in [1], [24], [38], [39]. For example the ongoing evolution of the bi-stable transistor from point-contact to MOSFET is the technological cornerstone of binary's success for 75 years [40]. The concept of encoding information in two symbols and performing arithmetic with them has an even longer history and was used in various ancient cultures [41]. A more complete treatment of the discovery and history of binary for computing can be found in [39], [41]–[43]. They include references to events leading up to the invention of the electrical binary computers such as Jacquard's machine using (binary) punched cards to program a loom in 1804, Babbage's mechanical computer in 1837 and George Boole's logic algebra that expanded Aristotlean logic in 1847. In this section five historical papers are discussed that led to the mass adoption of binary in electronic computers. They are treated in chronological order from the 1930's and onwards.

1.3 Historical evolution of binary computing

The first landmark paper was Alan Turing's 1936 "On computable numbers" [44]. This theoretical machine used unary encoding ("tally counting") and laid the mathematical foundation for computing. In the paper Turing described *universal machines*, machines that could do more than arithmetic and are akin to general purpose symbol processors. Turing and von Neumann had met on several occasions and this paper is quoted to have inspired von Neumann [45, p. 10]. Turing also proposed and constructed practical computers, most notable the 1945 ACE [37] which was based on his 1936 paper.

The second influential paper was the Master thesis written by Claude Shannon [46] in 1937. This work laid the foundation for EDA tools describing in detail a link between Boolean algebra and circuit implementation using binary switches. For example, he showed how to synthesize complex functions such as a n -bit full adder with sum and carry signal into a electrical circuit.

The third key paper is a manuscript written by John Vincent Atanasoff [47] who through his pioneering work between 1935 and 1941 is considered by some to be the father of the modern computer [48], [49]. He explicitly investigated the role of radix for building computers [38, p. 307]:

"Considerable thought was given to the design of a computing mechanism that would simultaneously be simple, fast and accurate. After many attempts to devise a conventional computing mechanism with these properties attention was turned to the possibility of changing the base of the numbers in which the computation is carried out. For a short time the base one-hundred was thought to have promise but a calculation of the speed of computation carried out in terms of this base showed it to be so low as to make its use out of the question. However this same calculation showed that the base that theoretically gives the highest speed of calculation is e , the natural base. But the base of numbers must be an integer, and a further calculation indicated that the bases two and three yield number systems with the same and consequently the highest speed of calculation. The choice of the base for a system of numbers to be used for mechanical calculation is a rather different question than if the numbers are to be used in mental calculation."

Together with his PhD student Clifford Berry he build the first electronic binary computer, the Atanasoff-Berry-Computer (ABC). The computer featured many engineering novelties such as electronic switching using vacuum tubes for logic and charge-based storage with capacitors for memory [38]. These two devices were designed to be inherently bi-stable as each device is capable of representing 2 stable states. With them he discovered the devices to implement radix-2 efficiently in computers. It was also very close to the theoretical optimum e . Atanasoff was not aware of Boolean algebra and Shannon's thesis on binary circuit design with Boolean algebra [50].

1 Introduction

The fourth seminal paper was a report by John von Neumann on the working details of a new computer architecture [51]. The foreword by Godfrey mentions that this paper inspired the first generation of computer engineers. Von Neumann joined John W. Mauchly's project as advisor to build the successor to the radix-10 ENIAC, the radix-2 EDVAC. Mauchly visited Atanasoff's lab during ENIAC's development and wrote to Atanasoff he was considering to implement his digital approach. Mauchly denied being inspired by it though. A court ruling ended a patent dispute, finding the ENIAC a derivative of the ABC [1], [48]. More controversially, von Neumann wrote a preliminary report based on the internal discussions of the radix-2 EDVAC computer without referencing Mauchly and others. In the same year he published a report with Goldstine and Burke [52] discussing the role of binary. The main argument favoring binary he gave was based on pragmatics [51]:

"Thus, whether the tubes are used as gates or as triggers, the all-or-none, two equilibrium arrangement are the simplest ones. Since these tube arrangements are to handle numbers by means of their digits, it is natural to use a system of arithmetic in which the digits are also two-valued. This suggest the use of a binary system. The analogs of human neurons are equally all-or-none elements."

It is worth noting that this last sentence was based on the 1943 paper by McCulloch-Pitts [53]. This paper was the first computational model of the human brain and was proven quite early to be far too simplistic and inaccurate [19].

The fifth paper considered to be influential for the mass adoption of radix-2 in modern computers is by Werner Buchholz [12]. The 1955 "Fingers or Fist" paper is one of the few academic works that focussed on radix comparison. Buchholz worked for IBM and collaborated with Gerrit Blaauw on computer architectures [54]. Blaauw is the "inventor" of the 8-bit byte [55] and lead architect of the IBM/360, one of the most successful mass-produced computers. In the Fingers (radix-10) or Fists (radix-2) paper, Buchholz wrote that radix-2 was superior to radix-10 for nine reasons. Buchhold cited von Neumann's EDVAC paper for several of the reasons. Important is that for many of the arguments, radix-10 was implicitly assumed to be implemented with bi-stable devices (and encodings such as binary coded decimals, bi-quinary, etc) as this was the most efficient implementation. A brief analysis of the 9 arguments can be found in the **Appendix G.2**.

Lastly, work with less focus on structured radix comparison from this era was "Arithmetic Operation in Digital Computers" from 1955 by Richards discussing radix-2, radix-3 and radix-10. They mention that the ternary system was seriously considered. An important quote from that work is [56, p. 15]:

"When the difficulties which are encountered in the design of a ternary computer are combined with the dearth of ternary computer components and with the difficulties in adapting the system to applications where the decimal

system is already entrenched, it appears that the disadvantages of the ternary system with positive and negative coefficients substantially outweigh the advantages.”

Shannon's 1950 "A Symmetrical Notation for Numbers" [23] discussed the properties of symmetrical radices around zero (such as radix-3, radix-5, etc) and mentioned that some of the benefits disappear when for instance radix-3 is used with asymmetric arithmetic. This topic will be relevant in Chapter 2 on balanced vs unbalanced ternary. The third honorable mention on radix comparison is the 1950 work by the Engineering Research Associates Staff who briefly mention symmetrical radix-3 (balanced ternary) adder designs [57, p. 287] but do not divulge in a comparison to binary. The first electrical ternary computer, the 1958 Russian Setun [26]–[28] actually used binary coded ternary (2 bits for 1 trit) [58] and will be discussed in Chapter 2. This computer had some commercial success, unique features and piqued interest from the USA [58] but could not influence the path towards binary dominance. Radix-2 was the best option because the radix implementation was done with bi-stable binary logic and memory devices. Implementing other radices with them would make them economically inefficient.

Readers that are interested in historical events after the origins of binary computers are referred to the IEEE Annals of the History of Computing. Central is the focus of miniaturization of bi-stable transistors after the invention of integrated circuits. This became the cornerstone for binary computers.

1.4 Moore's curse: 3 scaling walls

Moore's law has inspired continuous device-centric scaling for over 50 years and counting through a multi-disciplinary effort involving both academia and industry. Revenue in the global semiconductor industry has grown to 700 billion USD in 2023 [59]. The writing on the wall started after Dennard scaling [18] stopped around 2005. Transistors scaled physically afterwards, but other properties such as power consumption did not. The relentless exponential growth by doubling transistor density every two year is unusual compared to other industries [60]. Like any exponential growth curve it is bound to end. Perhaps more importantly, the continued focus on the area metric masks technical debt. Moore's law has become Moore's curse [60]. Performance and power show marginal growth of 3% for nearly 20 years (see Fig. 1.1) and [2, p. 44]. The 2022 Industry Roadmap for Devices and Systems (IRDS), a leading set of frequently updated white papers [61], [62] that discuss the state of semiconductor research in industry and academia, identify many challenges that are deemed critical for further scaling. This includes the suggestion to explore higher radices [62, p. 4]. Three categories of challenges relevant for this thesis are highlighted: power consumption, memory access and EDA tooling and workflows. All three challenges can be considered corollaries of Moore's Law.

1.4.1 Power wall

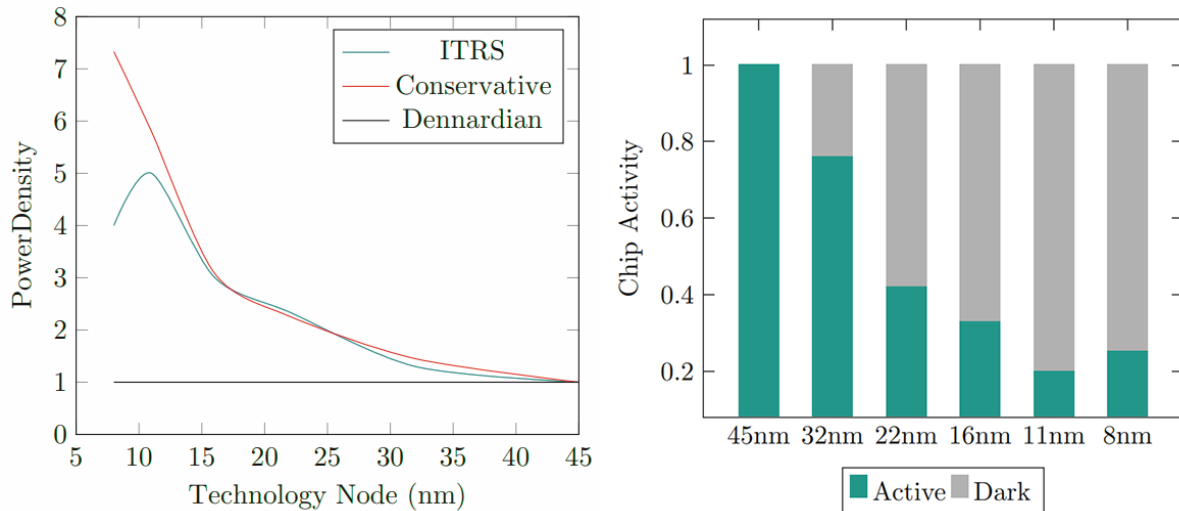


Figure 1.3: (Left) The power wall, a post-Dennard power density trend. (Right) Dark silicon: projected transistor activity over technology nodes. Both plots adapted from [63]

The *power wall* [64] refers to the increasing problem of power delivery and heat dissipation due increased transistor density (see Fig. 1.3). The problem is not limited to billions of fast switching transistors. Each transistor is wired up from its gate, source and drain terminals. The increasingly lengthier interconnects of these transistors also dissipate heat during switching. The power consumption equation is discussed in **Appendix G.3** as it is slightly more complex for higher radix signals. Power is considered the number one design constraint [8] and limits both low-power and high performance computers [64]. The effect of the power wall visible in Fig. 1.1 shows limited single threaded performance increase after 2005. Dennard’s paper [18] observed two major scaling rules. The first was related to scaling interconnects mentioning that “*Scaled interconnects provide roughly constant RC delays because the reduction in line capacitance is offset by an increase in line resistance*” [65]. This means that as interconnects became smaller no speed was gained. A detailed analysis of the interconnect problem or “interconnect wall” in relation to scaling can be found in [66]. The second scaling rule was about transistor scaling, a formula to improve all PPA metrics. The importance of it is perhaps best phrased in Bohr’s retrospective paper on Dennard scaling mentioning that “*as transistors get smaller, they can switch faster and use less power*” [65].

Unfortunately the second scaling rule is no longer feasible which marks the end of Dennard scaling. The rule to double transistor density required a reduction of transistor dimensions by a factor of 0.7 and keep power density constant by reducing supply voltage by the same factor. To reduce the supply voltage while maintaining a good I_{ON}/I_{OFF} ratio at room

temperature the voltage threshold needs to scale down proportionally. For 30 years this could be done by scaling the gate oxide thickness. As gate oxide thickness approached just 5 silicon atomic layers in the early 2000’s, transistors couldn’t switch off properly anymore resulting in constant leakage current [65]. Multiplied with billions of transistors this small leakage became significant - even when doing no computation. This phenomenon is called direct tunneling. Other issues are known as short-channel effects (SCE) [67, p. 496-504] [68] and play an increasing role in sub 100 nm technology nodes [69].

Power density is defined as $D = P_{device}/Area_{device}$. With the ongoing scaling of $Area_{device}$ and without Dennard scaling of P_{device} , power density keeps increasing every technology node (see Fig. 1.3). Transistors operating at 1.5 THz f_{max} have been demonstrated [70]. In practise a fraction of that maximum frequency can be used for switching activity. Operating at lower frequencies to curb power is called dim silicon. The phenomena of disabling transistors is called dark silicon [63] and has been increasing every technology node (Fig. 1.3). Markov also classifies grey silicon, which are additional non-functional but power consuming structures such as repeater gates that are needed because the interconnects have become too short [7]. Repeater insertion, both the amount and placement, is an increasing problem with every node [71]. On 130nm interconnects including repeaters can form 50% of the dynamic power consumption and leakage power [71]. Repeaters are critical for minimizing clock skew in the clock tree network (CTN) and signal integrity. Ideally these repeaters are placed in higher metal layers where there is less interconnect congestion and would reduce the amount and placement problems [72]. Currently, repeater gates share the same physical layer as the other logic gates. Device utilization is considered a main challenge in the 2022 IRDS roadmap [61, p. 3]. If power consumption goes unchecked temporary or permanent thermal related defects arise. Defects include lifetime, performance and reliability.

Various approaches have been discussed in literature [8], [64], [73], [74] to reduce power consumption and curb heat dissipation in hardware. At the system level the most common strategy to mitigate heat dissipation is to use active cooling solutions. Good heat transfer reduces leakage current as the thermal noise floor is lowered [75]. A cool CPU allows higher switching activity α and F_{clk} because electron mobility is negatively affected by temperature. Active cooling require power drawn from the system and can be significant (for instance 40% in supercomputers [75, p. 10]). Recent innovations in the field include MEMS-based active cooling [76]. The rate of progress in heat transfer research cannot cope with the increase in power density [75] due to Moore’s law, necessitating novel approaches at lower abstraction levels.

1.4.2 Memory wall

The *memory wall* [77]–[79] is the latency and bandwidth gap between on-chip data processing and off-chip data retrieval from dynamic RAM (DRAM), see Fig. 1.4. It is also

1 Introduction

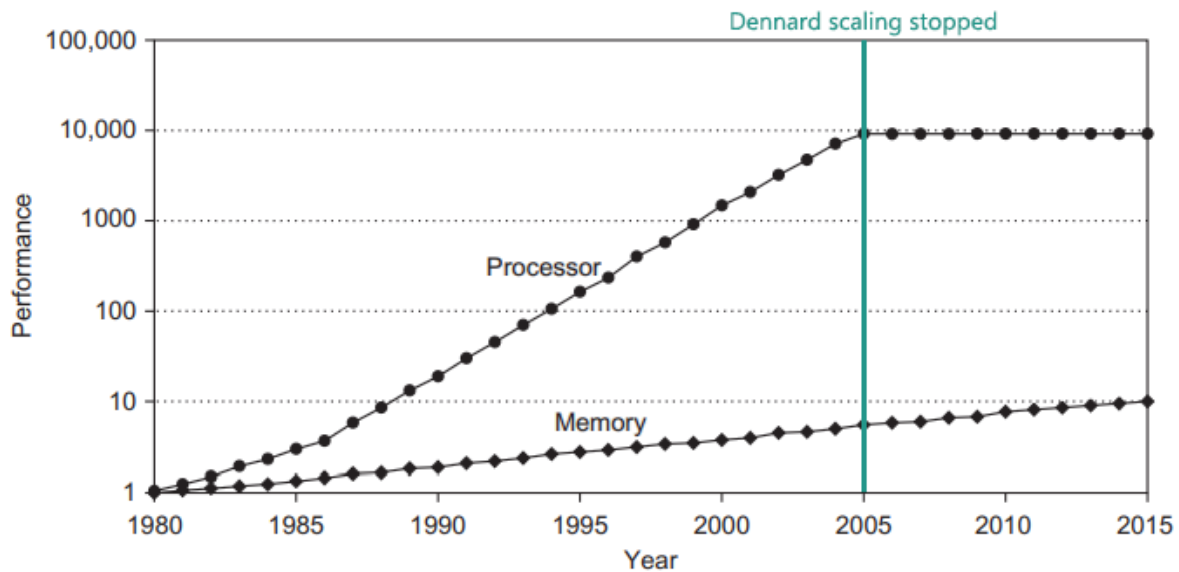


Figure 1.4: The memory wall. The gap between retrieving data and operating on it. Adapted from [78]

called the von Neumann-bottleneck [80]. The gap stabilized as frequency couldn't scale after Dennard scaling stopped. However, in technology nodes after 10 nm Moore's law seems to saturate for both DRAM [81] and static RAM (SRAM) [82] scaling meaning that the gap might widen further again. For DRAM 50% more processing steps are needed to move from technology node $2X$ to $1Z$ and include costly double and quadruple patterning steps [81]. Both logic cells and memory cells suffer from lower supply voltages and increased interconnect resistance as a result of scaling. Worse, for memory cells the decrease in supply voltage and increase in interconnect resistance degrades SRAM and DRAM performance [81], [83] making it harder to scale voltage further. The different technical requirements for logical and memory is part of the reason why DRAM is made on much older technology nodes [81, p. 1386]: *"The low leakage is required to prevent the discharging of the capacitor, and the high ON-current is expected to write the data in a short time"*.

The architecture of a conventional computer is unchanged since Burks, Goldstine and von Neumann's paper [52] and consists of 5 components; datapath, control, input, output and memory. The datapath and control components are together called the processor[2]. Memory physically located on the same die as the processor has the shortest path to the processor and is called *cache* or on-chip memory. This type of memory shares the die area with digital logic blocks and analog blocks. The area SRAM occupies depend on the product but even processor's from 2004 like the Intel Itanium 2 6M can occupy 50% [84] with L2/L3 cache. In terms of power SRAM can consume 25-50% of the total power [78]. The further the memory is from the processor the more power and time it takes to retrieve or access the data, see the memory pyramid in Fig. 1.5.

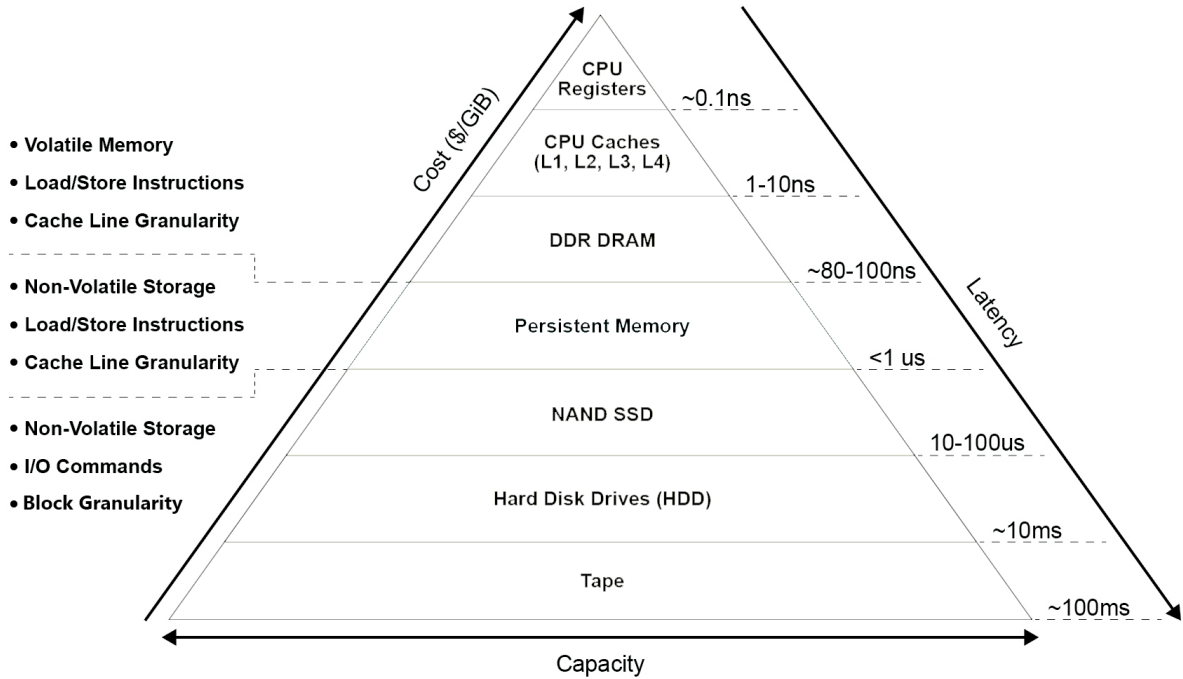


Figure 1.5: The memory hierarchy and latency pyramid. Adapted from [86]

Since 1975 on-chip memory is high density SRAM while the nearest off-chip memory is DRAM [78]. SRAM is typically made with 6 transistors, a 6T-SRAM cell [83]. DRAM is made with 1 transistor and a capacitor, a 1T1C-DRAM cell [81]. To improve latency both SRAM and DRAM cells use pre-charge circuits [85] such that the voltage swings are halved or reduced. The average memory access time, the latency, is data dependent as data in cache doesn't require hundreds of clock cycles to fetch from DRAM [10]. A 32-bits memory load also costs 1300x more energy than a 32-bits ALU operation on a 45nm process [10], a significant increase from 260x at 130nm. Electron transport is more expensive than electron manipulation. The SRAM and DRAM average access time is dependent on cache access time, miss rate and miss penalties [78].

The stored-program concept by von Neumann [51] is the architectural reason for the bottleneck between memory and the processor. Both the Harvard and von Neumann architectures define that computers cannot perform operations directly on memory. Using a memory hierarchy like in Fig. 1.5, data is stored and retrieved from memory and moved to a central processing unit. All conventional computers use this architecture and thus implement some sort of fetch-execute-writeback mechanism. New memory paradigms such as compute-in-memory (CIM) or processing-in-memory (PIM) using novel memory cells such as RRAM and a higher radix can break this paradigm [80], [87].

1.4.3 EDA wall

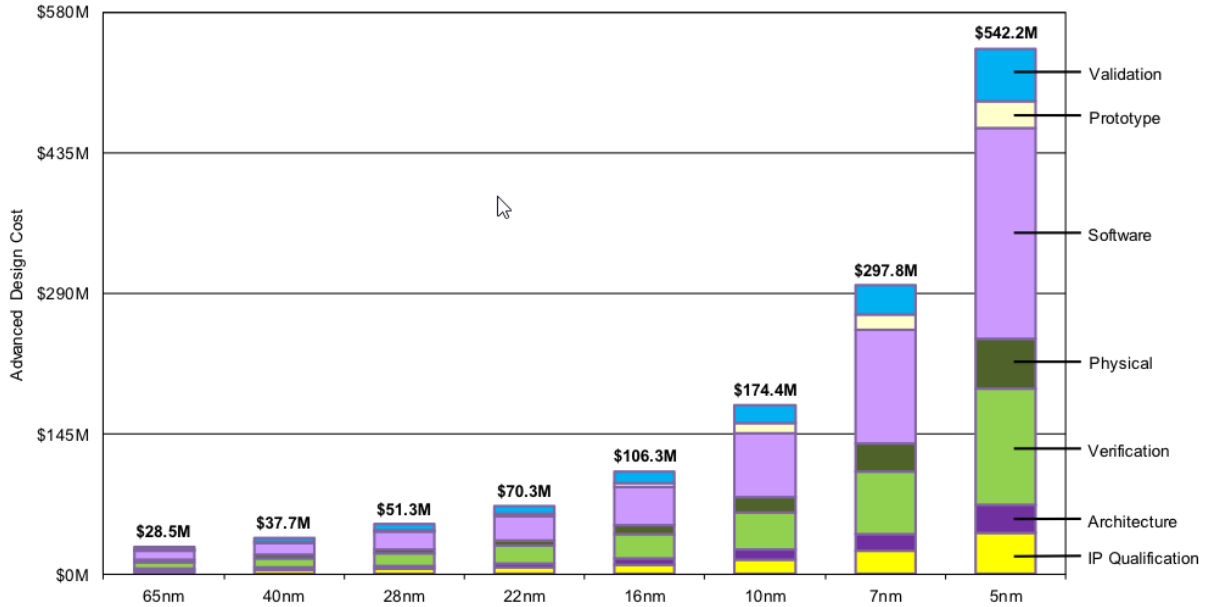


Figure 1.6: The EDA wall. The exponential increase of chip design costs with transistor density. From [88] (original data: IBS)

The exponential increase in electronic design automation (EDA) related cost every technology node [88]–[91] can be considered another wall, the *EDA-wall* and is visible in Fig. 1.6. Moore’s exponential pace of scaling enables more transistors for the same design space pressuring EDA manufacturers to scale design, simulation and verification time at the same pace to keep cost equal [92, p. 219]. The average amount of design rule checking (DRC) operations, a critical early verification process, is exploding [93], [94].

In the early EDA days chips were monolithic and designed with a small team of integrated circuit (IC) design and layout engineers. The first microprocessor, the Intel 4004 from 1971 had just 2300 transistors. The logic design, circuit design, layout design and verification was mostly done by a single person, Federico Faggin [95]. Development of silicon products in those days were done without computer aided design (CAD) tools, a synonym for EDA tools [92], [96]. When complexity grew to several thousands transistors in the mid 1970’s, EDA tools such as SPICE circuit simulators and place and route (PnR) tools became a necessity. With an abundance of tools came also the need for deeper integration and automated workflows. A comprehensive historical overview about the period 1964-2002 can be found in [96]. Despite nearly 70 years of academic interest in higher radix computing few MVL EDA tools exist to accommodate design and verification of MVL circuits [97], [98]. These tools could aid in scaling the EDA wall for both binary and MVL circuits as some Boolean logic problems are better solved in the MVL domain [97], [99].

Transistor density and verification is not the only reason why the EDA wall grows exponentially. The enormous innovation to keep up with Moore's law forces the industry to keep developing new EDA functionality, tooling and workflows. Advances in materials science (such as high-k dielectrics, memristive materials), device evolution (such as planar MOSFET, finFET, GAAFET, CNTFET) and architectures (such as interconnect stackups, superscaler, in-memory compute) all need software to be supported. In 2018 DARPA launched a 100 million open source investment to drastically curb costs of modern silicon and control the EDA wall. This initiative had lead to OpenRoad [100], Openlane [101] and SiliconCompiler [102] which aim to provide complete or partial RTL to GDS flows. Openlane is used in this thesis work. These flows rely on open process design kits (PDK) which are considered the crown jewels of a foundry. Examples of open source PDK's are Skywater foundry's 90nm and 130nm, GlobalFoundries 180nm and ASAP7, a 7nm FINFET PDK [103].

Another popular measure to reduce cost and complexity is reusing battle-tested components, so called intellectual property (IP) blocks. This is a very similar to the reusable component principle in software engineering. According to [91] a modern System-on-Chip (SoC) uses on average over 175 IP blocks with just 20% of the design being custom. Many commercial chips are no longer designed as monolithic. Rather a multi-process or *chiplet* approach is used, merging several dies on a single substrate to form a complete SoC. Gordon Moore already predicted the cost benefits of this strategy in his 1965 paper [17].

The integration of artificial intelligence (AI) might also reduce cost and complexity. The launch of AI accelerator chips to the market is increasing astronomically [104] and influences many industries. The EDA industry has an interesting dependency dynamic with AI accelerators. EDA functionality is being more and more integrated with AI such as CPU design with ChatGPT [105] and IP block placement based on reinforcement learning (RL) [106]. This in turn leads to better AI chip designs, a self-reinforcing loop. Important is that AI for verification (and validation) is still lacking, often because the test patterns depend on the design.

1.5 The fundamental limits of computing

The fundamental limits of computing has historically excited computer researchers [7], [107]–[112]. The discussion if Moore's law has already ended has devolved into semantics. Density scaling is certainly ongoing [15], but the same source shows that transistor costs is not following the same exponential scaling. Renowned chip designer Jim Keller shows that pushing Moore's law has never been a single technology development but rather a series of S-curves [113]. It undeniable though that several parameters depicted in Fig. 1.1 have not seen exponential growth for decades. The availability of more transistors

1 Introduction

made up for the inefficiency of the whole system. This is similar to the abstraction in higher level programming languages. Simplifying programming with natural language increases application development but at the cost of application performance and power consumption [114].

Investigating the influence of radix on computational limits requires an overview of fundamental principles that govern computing. A comprehensive overview is published by Markov [7] and includes several physical, mathematical and practical limits in domains such as material, devices, circuits and software. From these Shannon's limit and Landauer's limit stand out for inspection against the practical limits since it uses radix-2 in its formulation. In addition the mathematical theorem on radix economy is added and Rent's rule, a complexity heuristic that is directly effected by the radix.

1.5.1 Shannon's limit

In 1948 Claude Shannon published "A Mathematical Theory of Communication" in which he extended the theories from Nyquist and Hartley [115]. He was motivated by the various methods that exchanged bandwidth for signal-to-noise ratio (SNR) such as Pulse Code Modulation (PCM). In the paper he proposes a fundamental theory of communication by modelling the effect of noise in the channel and exploiting statistical structure of the source message. His error-free noisy-channel coding theorem also known as Shannon's limit introduces the concept of coding to approach this channel capacity limit. The theory predicts an upper bound, the channel limit, which is measured in a radix of choice such as bits/sec. The channel limit is a mathematical limit on how much information per unit time can be send through a noisy channel with arbitrary low confusion (in other words error-free) at the receiving end. In this section the focus is on the application of the theorem, known as the Shannon-Hartley theorem, to analog (continuous signal) communication through interconnects. Interested readers are encouraged to read the complete presentation of the theorem which includes the nature of information (entropy) and discrete and continuous noisy and noiseless channels in [115]. A less mathematical presentation of the paper with many applications beyond Shannon's work is written by John R. Pierce [116].

The equation of the Shannon noisy-channel theorem for continuous channels is shown in Eq. 1.2 [115, p. 43].

$$C = W * \log\left(\frac{P+N}{N}\right) \quad (1.2)$$

Where C is the channel capacity, W is the bandwidth (in Hz) and log the logarithm in some radix of choice. P is the average signal power and N is average noise power.

When additive white Gaussian noise (AWGN) is assumed [116, p. 173] and information is measured in radix-2 (bits) then the equation becomes the Shannon-Hartley theorem Eq. 1.3 [115, p. 45]

$$C = W * \log_2(1 + \frac{P}{N}), \text{ in bits/sec} \quad (1.3)$$

For given parameters bandwidth W and SNR P/N a higher radix than 2 can be chosen. For example, choose $W = 1.585$ and $P/N = 1$. Then $C = 1.585$ bits /sec. If radix-3 is chosen with the same parameters, then $C \approx 1$ trit /sec. Measuring in a higher radix does not mean that additional information is gained, only that the information is now more compactly encoded. The example shows that information is encoded in 1 symbol in ternary versus 1.585 symbols in binary. Equation 1.3 also shows two obvious limits, W and SNR. Bandwidth is discussed in the next subsection. At or below the noise limit, where SNR approaches 0 or is below it, binary is the only option [116, p. 176]. In that scenario one or multiple binary symbols needs to be send to be decoded as a single noise-free bit.

An alternative form of Eq. 1.3 is derived from Hartley and Nyquist's work by expressing channel capacity as $n * \log * r$, the maximum number of independent pulses n that can be transmitted per second using r levels (such as voltage amplitude levels, the radix). Using Nyquist's theorem n can be replaced by $2 * W$ (twice the bandwidth). The equation can be rewritten such that an expression for the channel capacity is obtained with both W and radix, measured in bits (\log_2 , Eq. 1.4) or in trits (\log_3 , Eq. 1.5) :

$$C = W * \log_2(r^2), \text{ in bits/sec} \quad (1.4)$$

$$C = W * \log_3(r^2), \text{ in trits/sec} \quad (1.5)$$

Where r is expressed as $\sqrt{1 + \frac{P}{N}}$

In these forms it can be seen that to send for example ternary signals noise free for some capacity C , a minimum SNR is needed of $r = 3 = \sqrt{1 + \frac{P}{N}} = 9$. This means that SNR ≥ 8 to be received noise-free. At this lower limit, the average transmission rate requires block coding [116, p. 177]. Transmission without block coding is only possible if the transmitter uses a brief but powerful pulse [116, p. 177]. This gives a good argument for pulse-based signaling architectures in some scenario's such as (ternary) spiking neural networks [117].

Shannon's error-free noisy-channel coding theorem shows through mathematical rigor [115] that binary is fundamentally the simplest and in some extreme cases the only communication signal to obtain the channel limit. In most practical scenarios higher radices

1 Introduction

can be used in such a way that the same channel capacity or higher is obtained. This will be discussed in more detail in Chapter 2. Approaching the limit to an arbitrary low error-rate has both hardware and software consequences.

Approaching the limit with software

The noisy-channel coding theorem introduces the principles now found in data compression theory [118, p. 6] using the concept of adding and removing redundancy to increase information entropy (the average amount of information per symbol) and robustness to noise. This conclusion is best summarized in a quote by Pierce [116, p. 164]:

”Indeed, the whole problem of efficient error-free communication turns out to be that of removing from messages the somewhat inefficient redundancy which they have and then adding redundancy of the right sort in order to allow correction of errors made in transmission.”

In compression theory both the modelling and coding phase depend on the radix [118, p. 6]. In the modelling phase a data source is modelled. In some situations modelling with a discrete symbol alphabet in mind can be beneficial, for example a data source model with an alphabet of 27 symbols can be efficiently encoded in radix-3 with 3 trits/symbol in the coding phase. With a richer alphabet more patterns can be made such that they better model reality [118, p. 23].

Approaching the limit with hardware

The Shannon-Hartley equation shown in Eq. 1.3 models a continuous signal communication channel. In the context of computers such a signal is an analog electrical signal and flows through the dense and lengthy network of interconnects. The channel capacity C is the amount of bits/sec that can flow through a single interconnect. The interconnects are bandwidth-limited, not power limited as the SNR is ≥ 1 . The SNR is such that error likelihood is extremely small during transmission. For example the bit error rate (BER) of the USB 4.2 spec using ternary signalling is $1E^{-19}$ (coded) and $1E^{-8}$ (uncoded) or 1 error in 10^{19} bits send [119]. For high-speed communication block coded signals are common, but for memory cells and logic gates uncoded signals are used.

The bandwidth for computing is the highest frequency possible measured in Hertz such as the clock signal. For interconnects the properties of the conductive material and surrounding insulator attenuate the signal increasingly with higher frequencies. The effect is that a square input pulse becomes a flat, spread out signal [116, p. 26-38]. The degraded signals effects are called attenuated peak-to-peak voltage swings and inter-symbol interference (ISI) [120]. To compensate for these effects higher power is needed which makes higher frequencies after 25GHz energy inefficient in CMOS circuits [121]. The practical

limits with electrical interconnects [122] and optical interconnects [123] have long been theorized and can be summarized as:

- Bandwidth limited. Higher frequencies require exponentially more energy to drive them. Physical material limits (resistive loss) due to high frequency have been known, including to mitigate it with a higher radix [122] (discussed below).
- Area bounded. The physical limit is the wafer size and reticle limit. Wafer sizes have increased, but not much. Wafers are divided in dies with the max size being the reticle limit. Each die has pins around the edges. Compared to other radices, binary has the lowest possible pin density (radix-1 is excluded) and most interconnects.
- Cost limited. Continuously adding more metal layers to mitigate interconnect density issues results in higher design, verification and material costs and decreased wafers-per-hour output. A higher radix can more efficiently use the interconnect infrastructure at lower frequency.
- Temperature limited [116, p. 188]. With the thermal limit as the absolute limit. The closer to the thermal limit, the larger the noise part in SNR becomes.

Trade-off can be made using Eq. 1.3 such that a desired channel capacity is reached [116, p. 178]. For example W can be reduced if SNR is increased for the same C . More variations are shown in Fig. 1.7.

Strategy	Frequency (W)	Signal-To-Noise Ratio (SNR)	Effect	Action
1	F ↓	SNR ↑	Radix ↑	Reduce frequency and increase SNR
2	F =	N ↓	Radix ↑	Reduce signal noise to improve SNR
3	F =	S ↑	Radix ↑	Increase signal power to improve SNR
4	F =	S ↓, SNR >= 8	Radix ↑	Reduce signal power and improve device/wire materials, block encoding
5	F =	SNR =	Radix ↑	Decrease BER requirement, shift error correction from hardware to software

Figure 1.7: Trade-offs to increase the radix based on Shannon-Hartley’s theorem for a fixed capacity C

1.5.2 Landauer’s limit

The Landauer limit [108], [124] in Eq. 1.6 is attributed to Rolf Landauer. He used a computing viewpoint to analyze the minimum amount of energy needed for an ideal bi-stable transistor to switch state.

$$S = k * T * \ln(2) \approx 0.693kT \text{ joule/bit} \tag{1.6}$$

1 Introduction

Where S is the initial state of a closed system in joule, k is Boltzmann constant, T is temperature in Kelvin. The initial state (thermal equilibrium) is equal to zero entropy and since it is a closed system cannot decrease. The entropy must increase with a minimum of $k * T * \ln(2)$ to transition from an unknown binary state to a known binary state.

An arguably better treatment is given by John R. Pierce in [116, p. 201] from a communication viewpoint resulting in the same limit. In the book Pierce transforms Shannon's channel encoding theorem into Eq. 1.6 explaining for example that thermal noise (also known as Johnson–Nyquist noise) works with the white noise assumption in Eq. 1.3. Landauer uses the limit to derive the heat generated to erase a bit, while Pierce and Shannon uses the limit to mean the power needed to transmit one bit per second.

The importance of Landauer's limit is becoming more relevant as heat extraction from a dense integrated circuit is difficult and without cost-effective outlooks to improve it [75]. About this Landauer wrote [108, p. 187]: "*Naturally the amount of heat generation involved is many orders of magnitude smaller than the heat dissipation in any practically conceivable device*". The work by Ruch et al. [10, p. 4] shows that despite much progress, the energy efficiency of computation (performance/watt) is still orders of magnitude away from biological efficiency. Heat is a form of noise and is detrimental to transistors. Thermal noise triggers false bit-flips due to compromised noise margins [109].

Landauer used the Boltzmann-Planck equation (the statistical mechanical definition for entropy) $S = k * \ln(W)$ where W is the amount of microstates to represent a macrostate [108, p. 187]. Landauer's limit in Eq. 1.6 has a $\ln(2)$ term to indicate that the logarithmic base uses binary digits (bits). This means that 2 microstates form a bit, which is the definition of a binary digit. To express the Landauer limit in radix-3 with 3 symbols, the minimum amount of energy needed to switch a tri-stable transistor becomes Eq. 1.7:

$$W = k * T * \ln(3) \approx 1.099kT \text{ joule/trit} \quad (1.7)$$

Unsurprisingly, the increase of 58.5% as given by $\ln(2) * 1.585 = \ln(3)$ is exactly identical to the information increase given by $\frac{\ln(3)}{\ln(2)} = \log_2(3) \approx 1.585$.

1.5.3 Radix economy

The radix economy models which radix is the most economical. The term economy assumes some cost function to represent a number in a radix. The radix economy is not a physical limit but a mathematical theory about the optimal balance between space and representation complexity. The relation between a number and its radix can be seen in Eq. 1.8. The variables n and m can be elements of N , the set of discrete natural numbers or P , the set of continuous irrational numbers.

$$s = 2^n = 3^m \quad (1.8)$$

In Eq. 1.1 it was shown that in a positional numbering system a number can be uniquely represented with 1 or more terms. The equation has two variables to associate costs with: the number of terms or positions and the number of different symbols in the radix. If the amount of positions is more expensive than the amount of symbols in a position, then the optimal radix is obviously the largest radix. A nice example can be found in Hayes [125]. The cost function used in most literature assign equal importance to the number of positions (sometimes called width or w) and the number of distinguishable symbols in a radix (often abbreviated to r). This assumption is discussed in **Paper D**. The result is a cost function $r*w$. It describes a tension between r and w in a positional numbering system to represent a number.

When the radix is allowed to be continuous and the positions are limited to discrete values then this cost function results in a balance between r and w with an optimum in radix- e . The derivation can be found in [126]. Important is the relationship to the number N . For example, $N=3$ (which includes $N=0$, $N=1$ and $N=2$) requires 2 discrete positions for both radix-2 and radix-3. The cost function results in $2*2 < 3*2$, showing that binary has a lower cost for $N=3$ than ternary. Only for a small set of N are other radices equal or superior to the discrete optimum radix-3 [125, p. 491].

In the mathematical domain the radix economy can be used with both continuous and discrete radices and positions. In the engineering domain the radix economy is used with discrete radices and discrete positions when designing digital electronics. This is the focus of work and is addressed below. The computational interpretation of the radix economy is a theorem dating to at least to the 1930's when Atanasoff researched the influence of radix on computing [38, p. 307]. Several papers have mentioned and discussed the radix economy throughout the past 70 years, such as the the 1950 report "high-speed computing devices" [57] and papers from 1980 by Armstrong [127], 1984 by Hurst, [126] 2001 by Hayes [125].

Critical discussion on the merits of the radix economy for computing can be found in literature [128] and social media [129]. The computational interpretation of the radix economy is perhaps best understood by treating number representation (such as in memory, interconnects, pins) separately from number transformation (such as in arithmetic, logic).

Radix economy for number representation

To represent a number in a physical system using a positional numbering system requires Eq. 1.1 to be implemented in hardware. This translation is proven to be quite feasible as many physical phenomena can be exploited and controlled to represent symbols/states. For example a capacitor can hold an variable number of electrons which can be counted

1 Introduction

individually or approximately. A capacitor is an example of a multi-state device as a single device can hold multiple states. Another device that can be used to distinguish states is a transistor. These devices are bi-stable where the stable operation regions *saturation* and *cut-off* determines the states. The biggest leap in information storage happened when the numbers were no longer represented with bi-stable devices but with multi-stable devices [111, p. 229].

Radix economy for number transformation

While the radix economy for number representation has demonstrated merits in guiding storage product roadmaps, this is less the case for number transformation (switching/processing) such as arithmetic and logic. VLSI compatible ("wafer scale") tri-stable logic devices are non-existent but have been constructed from bi-stable devices [130], [131]. It can be argued that a bi-stable device with shifted V_{th} such as a single CNTFET [30] behaves like a tri-stable device as it is also stable at $\frac{V_{DD}}{2}$.

Transistors are multi-purpose devices and using it for memory purposes is different than for logic. For memory the function is to encode information while for logic it is to transform and sometimes "drive" information. A collection of transistors for logic encodes a logic function such as a truth table. A truth table is an input to output state mapping in a certain radix. While the truth table is unique, the circuit implementation it actually represents is not. This means that there is redundancy in the encoded information. It might be possible to merge or tweak some transistors or wire them differently without changing the logic function. Of course some other property might change such as delay or power consumption. How well that redundancy can be removed signifies the efficiency of the design. Willard Van Orman Quine showed in his 1952 seminal paper "The problem of simplifying truth function" [132] that redundancy can be removed using a mechanical procedure to an optimal form. He notes that this optimal form is not necessarily the optimal circuit [132, p. 522]: *"So the problem which I shall examine is that of converting any normal formula into a simplest normal equivalent. This is not the most general form of the simplification problem from the point of view of engineering, since it can happen that some short non-normal formula represents a still cheaper electric circuit than any normal equivalent"*.

This remark can be directly seen in CMOS circuits. The incredible benefit of CMOS logic compared to other logic families is that there is near zero static power consumption. This comes at a cost of using two devices to encode binary signals; actively pull the signal down to V_{SS}/GND with an NMOS (logical 0) and actively pull it up to V_{DD} with a PMOS (logical 1). This sub-optimal encoding of 1 device per state (radix-1) pays off as designs scale to millions of transistors. The radix economy argument for number transformations should be used with care when forming conclusions and is discussed in **Appendix G.4**.

1.5.4 Rent's rule

Rent's rule is not a fundamental limit but an empirical observation of a power-law relation between the number of pins and the number of logic gates (blocks) in a circuit design [9], [10], [133]–[135]. Rent's rule is formulated in Eq. 1.9:

$$P = K * B^r \quad (1.9)$$

Where P is the average number of pins, K is the average number of pins per block, B is the number of blocks in a design and r is Rent's exponent. K and r are both constants. Rent's exponent has a range $0 < r < 1$.

The value of $r = 1$ equals random block placement and no optimization [10, p. 3]. This upper bound prioritizes communication over longer wires [134, p. 640] and in parallel [133, p. 1473] while the other extreme prioritized shorter wires and serial communication. The lower bound is called the intrinsic Rent exponent and depends on the wire topology. Two version of Rent's exponent exist. The first is called external (package level) Rent's exponent and has a typical value of 0.36 [134, p. 640]. The other is the internal (gate level) Rent's exponent and has various values depending on the function (such as $0.47 < r < 0.75$ [135, p. 149]).

Information travels over wires which are commonly called interconnects. A three terminal transistor has 3 interconnects, while a two terminal memristor, resistor, capacitor, inductor or diode has two. Adding a single device thus adds multiple wires. As transistors per unit area grows, the amount of interconnects per unit area grows even quicker. With increased interconnect density a myriad of engineering issues occur such as optimal block placement, routing interconnects, design time, signal integrity (inter-symbol interference, crosstalk, signal loss, power supply noise, etc) and various fabrication issues. The increase of interconnect density and the steady rise of additional metal layers to address some of the issues have been attributed to Rent's rule [9, p. 82].

The importance of Rent's rule is discussed in the memory wall. Communication of information has become the biggest bottleneck for performance [10]. Data is serialized at the cost of performance since pin space is insufficient [10]. Off-chip data retrieval costs several orders of magnitude more power than ALU computation [9]. The increased interconnect density and interconnect shrinking result in decreased bandwidth and high RC delays [136]. As interconnects shrink further extra power consuming transistors (repeaters) are needed [9] to break up interconnects and reduce delay. This works because the gate delay of the transistor has far exceeded the interconnect performance after scaling beyond 100 nm [136]. The device-centric scaling effort has ignored that there is a natural performance optimum. This optimum can be shifted to some extent with new material insights [137]. At data-centre scale, power consumption is dominated by poor interconnects scaling, limiting target performance improvement by at least 10X according

1 Introduction

to Hoefflinger [111, p. 424]. The insight that with a higher radix less blocks and pins per block are theoretically needed can directly be plugged in Rent's rule. This results in fewer interconnects but assumes that logic gates exist in a higher radix that are more efficient to create logic functions than with binary for the same area.

1.6 Research objective and dissertation structure

The central question of this work was to revisit Atanasoff's and von Neumann's 80 year old conclusion and determine if radix-2 (binary) is still the better radix choice for computing given new device and material advancements. Their conclusion was grounded in number theoretical arguments (such as being close to the optimal radix), assumptions (such as the neuron is binary) and practical arguments (such as simple to fabricate bi-stable devices for memory and logic). To refute their conclusion required a similar approach through a modern lens.

This article-based thesis is structured in three parts. In the first part, Chapter 1, binary computing and its fundamental and practical limits are discussed. In the second part, Chapter 2, ternary computing and its merits and demerits are discussed. In the third part which covers chapters 3 and 4, practical aspects of ternary computing are discussed. Chapter 3 explores how to build and simulate low-cost multi-state RRAM controllers on a breadboard and PCB. In Chapter 4 ternary logic synthesis, mixed radix EDA tooling, radix conversion and a RISC-V-like ternary CPU design are discussed. The appendices contain additional material on each chapter for readers seeking deeper understanding.

Research contributions were made by publishing six peer-reviewed articles and open sourcing all code and data. Four designs were submitted for tape-out. Several journal articles based on these papers and content found in this thesis are in the planned/submit stage:

Table 1.2: Relation between chapters and relevant papers

	Chapter	Relevant papers
2	The benefits of ternary	
	2.5 The seven C's of ternary	D
3	Multi-state RRAM development platform	
	3.3 uMemristorToolbox: A new tool for experimenting with multi-state RRAM	A
	3.4 Implementation of a multi-state RRAM controller for breadboard and PCB	A,C, planned paper
4	Mixed radix EDA workflow for ternary computers	
	4.2 MRCS: A new EDA tool and workflow for mixed radix design & verification	F, planned paper
	4.3 Mixed radix synthesis engine	B
	4.4 REBEL-2: A novel balanced ternary CPU and ISA	F, planned paper
	4.5 Radix conversion	E,F

—“One of the most important advantages of multiple-valued logic can be found at the system level”

Prof. Kameyama, proving MVL feasibility [138]

—“Symmetrical notation offers attractive possibilities for general purpose computing machines [...] The use of a symmetrical notation simplifies many of the circuits required to take care of signs in addition and subtraction, and to properly round off numbers”

Prof. Claude E. Shannon, father of information theory [23]

2

The benefits of ternary

2.1 Introduction

The pioneers of digital computing Atanasoff [38, p. 307], von Neumann [12, p. 4] and Shannon [23] all considered the 3-symbol computer alphabet a candidate for computing. Feynman in [25, p. 20] said that binary is a practical decision due to easy electronic representation. With regards to the computer alphabet Feynman said [25, p. 2]:

”[...] we can choose our basic set of elements with a lot of freedom, and all this choice really affects is the efficiency of our language, and hence the sizes of our books.”

This quote forms the central idea in this chapter: ternary is preferred in situations when efficiency is more important than simplicity. The popularity of ternary computing in literature decreased in the 1990’s but is increasing steadily in the last 20 years. It is currently at an all time high in the IEEE Explore database (see Fig. 2.1).

Ternary, trinary, radix-3, base-3 or 3VL computing is studied under the umbrella of MVL [139]. An introduction or complete overview of the field of MVL is out of the scope of this work. Suggested are several introductory resources and survey papers on radix-3 and MVL from the last 15 years:

- **Gaudet (2016)** [140]. Overview of the state of MVL. It provides an introduction to MVL terminology and concepts and presents recent developments, challenges and future directions of the field.
- **Miller and Thornton (2008)** [141]. Overview of MVL algebra, logic and representations.

2 The benefits of ternary

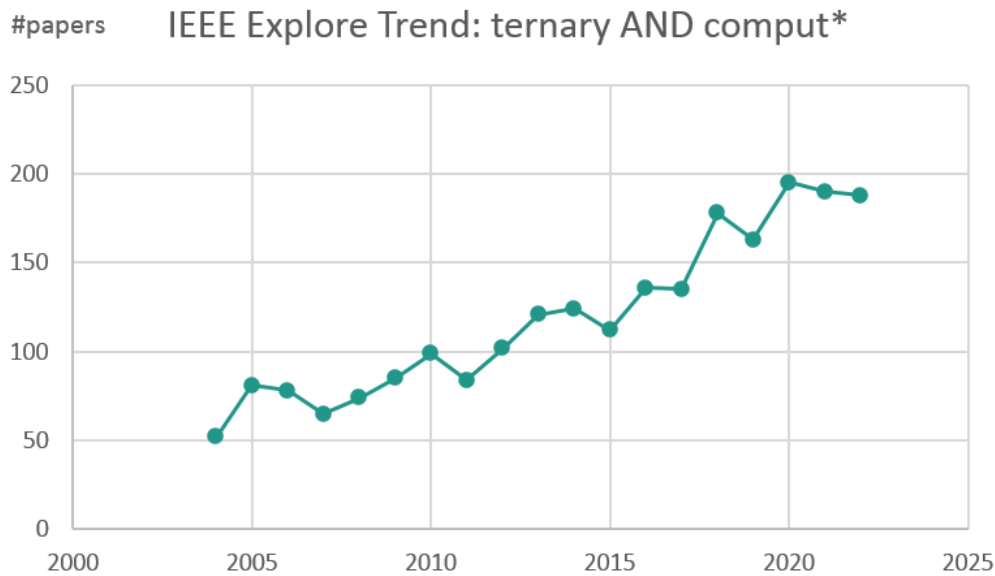


Figure 2.1: 20 year IEEE Explore trend for search "ternary AND comput*"

- **Dhande et al. (2014)** [142]. Overview of ternary computer concepts such as logic minimization methods and building blocks like the STI, ALU and full adder.
- **Nemati et al. (2023)** [143]. Overview of ternary full adders. It also includes a comprehensive demography of ternary researchers and where they published. Several key design parameters (mode, logic type, transistor technology, etc) are explained.
- **Andreev et al. (2022)** [144]. Overview of materials for MVL including tri-stable and multi-stable materials. Work by Jo et al. (2021) [145] and Kim et al. (2020) [146] also presents such an overview.
- **Sandhie et al. (2021)** [147]. Overview of MVL device technologies. It includes ternary FINFET, CNTFET and memristor based devices. Other recommended readings are by: Karmakar (2022) [148] on multi-channel FET, Moaiyeri et al. (2021) [149] on ternary Negative Capacitance CNTFET (NC-CTNFET), Son et al. (2022) [130] on CMOS-compatible ternary feedback FET (FBFET) and Yousefi et al. (2022) [150] on FINFET with ternary RRAM.

The chapter is organized in four sections. In the first section the heptavintimal notation used in this work and a limited set of ternary concepts are presented. In the second section the history of ternary computing is discussed. Thirdly, **Paper D** on the benefits of ternary is discussed. Seven application domains are considered where ternary can have a competitive advantage to binary. Lastly, the main arguments found in literature against ternary are presented.

2.2 Ternary basics

2.2.1 Heptavintimal notation

In binary all possible logic functions with one or two inputs and one output are given a unique name to indicate its behavior such as AND, OR, INVERT. This behavior is also captured in another representation such as a truth table (TT), shown in Fig. 2.1. Each binary or higher radix logic function has exactly one TT, a property called *canonical*. Other canonical logic representations exist such as sum of products (SoP), product of sums (PoS), AND/OR Multiple-valued decision diagrams (AOMDD) [151] and others.

Table 2.1: **Left.** Truth table of the radix-2 MAX (OR) gate **Right.** Truth table of the radix-3 MAX gate. Binary is a subset (in green) of ternary when all binary 1's are replaced with ternary 2's.

A	B	MAX
0	0	0
0	1	1
1	0	1
1	1	1

A	B	MAX
0	0	0
0	1	1
0	2	2
1	0	1
1	1	1
1	2	2
2	0	2
2	1	2
2	2	2

The 2-dimensional TT from Table 2.1 can be written as a 1-dimensional bit pattern 0111 or 0b0111 or 0111₂. These four digits represent a value between 0 and 15 and can be compressed into 1 symbol using a radix with an alphabet of at least 16 symbols. Typically powers of 2 are used, so hexadecimal (radix-16) or radix-64 are appropriate. In case of hexadecimal the OR gate gets the name 0x7. This is quite useful for large TT's and where many functions are not yet named but need to be canonically referred to. In this work the ternary encoding scheme based on radix-27 by Douglas W. Jones [152] is used. This scheme is based on the symbols 0-9A-Z but ignores some symbols that might be confusing such O (the letter o) since it is similar to the 0 (number zero). Since binary truth tables are a subset of ternary, assigning a heptavintimal index to binary gates is possible. For example the binary OR-gate becomes ZxP₂₇ or 0tZxP, where x (lower case!) is a *don't care* value. This work uses the 0t-notation as it implies the special heptavintimal encoding. This follows the 0x-notation for hexadecimal.

2 The benefits of ternary

Table 2.2: Douglas W. Jones' heptavintimal (radix-27) notation in [152]

Radix-10	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Radix-3	000	001	002	010	011	012	020	021	022	100	101	102	110	111
Radix-27	0	1	2	3	4	5	6	7	8	9	A	B	C	D
Radix-10	14	15	16	17	18	19	20	21	22	23	24	25	26	
Radix-3	112	120	121	122	200	201	202	210	211	212	220	221	222	
Radix-27	E	F	G	H	K	M	N	P	R	T	V	X	Z	

2.2.2 The third value

The ternary alphabet has an odd amount of symbols. Unique for odd radices is the ability to choose between two symbol sets, unsigned and signed. Unsigned numbers are an extension of the binary set $\{0,1,2\}$. In literature $\{0, \frac{1}{2}, 1\}$ is also found. The term used throughout this thesis is *unbalanced ternary* with the set $\{0,1,2\}$. Unbalanced ternary values are denoted with a $_3$ postfix. Signed numbers include negative numbers and in binary are often made with the 2's complement convention. For ternary we can easily convert the unbalanced set by a logical shift left resulting in $\{-1,0,1\}$. This set is balanced around zero and is called balanced ternary. In literature $\{\bar{1},0,1\}$ is also found. The set used in this work is $\{-,0,+ \}$ such that all symbols are one character. Balanced ternary values are denoted with a $_3$ postfix and uses an overlined 3. The third value in balanced and unbalanced sets are different logically but can be the same physically. This is an important abstraction. Hamacher and Vranesic [153] explored the difference between unbalanced ternary, balanced ternary and binary for multiplier arithmetic. They show that balanced ternary results in a reduced gate count in comparison to binary and unbalanced ternary at the expense of logic delay. This benefit becomes larger as the architecture increases and is discussed in [126, p. 1166].

Implementing the the three logical symbols is traditionally done in three ways: with one voltage source, two voltage sources or with current sources. Current sources are fast, need fewer transistors, have lower noise sensitivity and several other benefits [154]. However they scale badly as they have high static power consumption making it unsuitable for dense logic chips. Voltage sources are slightly more complex in transistor count but can scale better. Single power supply solutions require voltage division to create the middle voltage state and is thus also affected by static power consumption issues. Several techniques exist to reduce static power consumption such as low V_{DD} or subthreshold switching [131], dynamic logic switching [149], feedback [130], body effect [155]. Often a middle voltage state is achieved at the cost of lower drive strength or reduced frequency. Another strategy is to avoid/minimize the middle state as much as possible by remapping the associated logic state [143]. With two voltage sources the middle value can be created directly at the cost of a more complex power delivery network (PND). Examples are

V_{DD} and $\frac{V_{DD}}{2}$ or V_{DD} and $-V_{DD}$ sources. Note that it does not matter if balanced or unbalanced sets are used, a common mistake in literature. Dual V_{DD} (sometimes called multi V_{DD}) solutions account for about 40% of the adders in the survey by Nemati et al. [143]. The argument that a dual power supply negates the interconnect length reduction benefit compared to single power supply as used in binary requires further investigation, especially to assess its effects on the system level.

The philosophical meaning of the third value in logic processing was explored before the dawn of electrical computing by Łukasiewicz in 1920 [156, p. 87-88]. Statements about the future like "the glass will be empty" can be evaluated at a later time by extending to three truth states {True, False, Unknown}. The philosophical side of three valued logic has also been explored by Putnam, mentioning the importance of a different language when using ternary [157, p. 102]: *"using three-valued logic means adopting a systematic way of using the logical words which agrees in certain respects with the usual way of using them, but which also disagrees in certain cases, in particular the cases in which truth values are unknown"*. Furthermore, Putnam argues [157, p. 105]: "One can abandon two-valued logic without changing the meaning of 'true' and 'false' in a silly way.". Some logical functions are not universally accepted for ternary as the middle value is treated differently. For example three different truth table interpretations of logical implication are shown in [139, p. 88].

2.2.3 Mixed radix

Electrical signals are either analog, digital or mixed signal (**Appendix G.1**). Digital signals can be further subdivided into the amount of discrete levels that can be distinguished such as $r=2$ (binary), $r=3$ (ternary), $r=4$ (quaternary), etc or a mixture; mixed radix. Mixed radix signals are sometimes called mixed logic or mixed mode logic signals [158]. Humans use mixed radix number systems daily as currencies are often a mixture of coins and notes with different radices [56, p. 22]. The date and time are other examples of common mixed radix structures with day-month-year having radices 31-12-10000 and hour-minute-second having radices 24-60-60 [24, p. 208-209].

Mixed radix systems for computing is discussed in literature since the dawn of electrical computing. For example the 1955 work by Richards [56, p. 22-25] considers both pure (also called fixed or uniform) radix systems and mixed radix systems. He notes that historically most mixed radix systems have switched to pure radix systems and use radix conversion when computation are needed in another radix. Only in application specific contexts or subsystem context (such as ternary ALU's see [139, p. 94]) do mixed radix systems make sense according to Richards. Again the limitation of "available computer components" [56, p. 25] is the key argument why pure radix systems should be chosen.

There are information theoretical arguments to consider mixed radix system with binary and ternary. At the logic gate and functional block level input signals can be inherently

2 The benefits of ternary

mixed radix. For example a 2:1 multiplexer of ternary signals has a binary select and ternary data signal. Using pure ternary signals would waste a state per signal or require encoding logic. Another example is the ternary register discussed in Chapter 4.4. A typical ternary register contains binary signals for read, write, clock while ternary signals are used for data. A mixed radix system is theoretically the closest to the optimum e as the average can approximate this irrational number with discrete components. For example 3 trits and 1 bit as shown in Eq. 2.1:

$$BT TT = \frac{2 + 3 + 3 + 3}{4} = 2.75 \approx e \quad (2.1)$$

Where B is binary digit and T is ternary digit.

The advantage of mixed radix systems with binary and ternary with CNTFET was shown in 2009 [155] and in 2020 [155]. The same CNTFET architecture and devices are used in the mixed radix synthesis algorithm discussed in Chapter 4.2.

2.3 Radix comparison methodology

Radix comparison is not trivial but is necessary to understand its usefulness compared to the status quo. The radix economy has been discussed in Chapter 1 and **Paper D**. Additional relevant context is given in **Appendices G.4-G.7**. Several articles have been written about comparison between binary and ternary, most notable by Etienne and Israël [159]. The IC landscape has changed a lot since 1988 with CMOS and voltage-mode circuits being the clear champions. The VLSI criteria have not changed and PPAC is still the primary criteria. In [160] Etienne outlines his methodology to compare binary and ternary. The often cited information limit of 1.585 (58.5%) is used to compare logic gates as a heuristic. As argued in **Appendix G.4 and G.7**, using this limit for logic gate comparison might lead to wrong conclusions. A system level benchmark should be developed to replace the heuristic. Miller and Thornton [141, p. 115] and Etienne [161] have been advocating for more benchmark effort such that the true benefits of any claimed MVL advantage can be tested and analyzed. This MVL-to-binary VLSI benchmark should include common program listings and algorithms such that the input and output patterns are identical yet encoded in another radix.

The following properties are proposed for fair comparison between binary and ternary:

- **Feature parity.** For example, a binary full adder cannot process negative numbers and need additional gates and pins for a 2's complement implementation such that it can be compared to a balanced ternary full adder.

- **Optimal architecture.** The overhead inefficiency as discussed in **Appendix G.5** in another base is large, especially when scaled to double digit architectures. The architecture should be a multiple of the radix and not the nearest to a binary equivalent like 32-bit.
- **Optimal pin efficiency.** A pin should be fully utilized. For example a (3,2) binary full adder is efficient: the 3 (individual) inputs can count from 0 to 3 while the output with 2 bits can do the same. For balanced ternary a (4,2) compressor is efficient: the 4 (individual) inputs can count from -4 to +4 while the output with 2 trits can do the same. A (3,2) ternary full adder would not be efficient when scaled as some output stats are unused.
- **Remap states.** If some state transitions are inefficient for example a transition to $\frac{V_{dd}}{2}$, then remapping of states can be considered such that they are statistically avoided or minimized. This has been mentioned in [143, p. 8].
- **Exploit inherent properties.** For ternary many efficient arithmetic properties are available if the balanced ternary encoding is used [23].
- **Uniform static noise margins.** For example, a standard ternary inverter has four "eyes" which need to be equally spaced and symmetrical [162].

A proper comparison should include:

- **Process-Voltage-Temperature (PVT) variation analysis.** With beyond-CMOS and other exotic devices it is uncertain how they behave after fabrication at different process conditions ("corners").
- **Average Power-Delay-Product (PDP) metrics.** This should include performance at various frequencies and with load. For smaller circuits, performance per transition would provide great insight.
- **Area metrics.** The transistor density per unit area including shared input inverters and buffers.
- **Open source.** Replication of designs is often tedious as device model, SPICE simulation model and results are often summarized in papers and not complete.

2.4 Historical evolution of ternary computing

The appreciation of radix-3 as a useful numbering system goes back to at least 750 AD by the Persians [24]. The first recorded ternary calculating device was the mechanical balanced ternary calculator build by Thomas Fowler in 1840 [24], [163]. Fowler was a banker and this device was constructed to aid in calculations with a pre-decimal British currency. A century later, in 1950 Claude Shannon wrote a small paper on the benefits of a balanced ternary arithmetic with formal proofs [23].

Both Atanasoff [38, p. 307] and von Neumann [12] seriously considered ternary for their digital computers. The first reported interest in developing a ternary computer and listing its advantages over binary was in 1952 by Grosch [164]. This relatively unknown memo can be considered one of the seed points to which ternary computing became a reality. Grosch advocated for a balanced ternary ALU in the Whirlwind II computer project [139, p. 94]. While that computer never materialized, in 1958 Nikolai P. Brusentsov did succeed in making the first electrical (balanced) ternary computer [26]–[28]. The project was done with a small team of academics headed by Sergei Sobolev. Brusentsov independently learned about the prospects of ternary computers and learned about the work of Grosch [27, p. 60]. The ternary computer was named Setun after a small nearby river and was constructed at the Moscow State University. The asynchronous computer was based on balanced ternary encoding and featured 18-trit words [28]. The machine garnered the attention of the USA which is detailed in [27]. Willis H. Ware noted in his report that the implementation of the elementary unit, the trit, was done with two magnetic cores. This encoding is called binary encoded ternary (BET) or binary coded ternary (BCT). This makes the implementation inefficient [58]:

”It was explained that the choice of base-3 was made because it can be shown that in some sense a base of 3 provides the most efficient utilization of equipment. Since a base-3 electronic technique is not available, they decided to construct a base-4 machine and to utilize only 3 of the 4 possible states. The unused 4th state in each case is available for some form of checking”.

Despite this implementation inefficiency Brusentsov mentioned that the Setun was 2.5x cheaper in 1965 [26], had greater numerical resolution (18-trit vs 8-bit), needed less power and was more reliable than the competitors, such as the PDP-8 [27, p. 166-175]. About 50 Setun computers were produced, but plans for mass production and export were stopped due to political reasons [26], [27]. An improved version was made for the 100th birthday of Lenin in 1970, the Setun 70. This 24 instruction computer was mostly used as an educational tool [26].

Ternary computers were actively considered in the 1950-60’s [165]–[170]. Alexander wrote a short paper called *”the ternary computer”* [171] discussing the relevant aspects and

2.4 Historical evolution of ternary computing

motivation to construct a ternary computer. Mechanical tri-stable devices were developed by NASA [165] and ternary algebra and switching theory was well developed [166].

In the 1970's an emulation was made of a balanced ternary computer on binary hardware. The TERNAC [172], [173] was designed to compare binary to ternary (implemented as BCT) on a conceptual level [172, p. 86]:

”As there was no attempt to utilize in any optimal way the binary hardware on which we are emulating, there is no point whatsoever to do any speed comparisons between the ternary and binary computers. The final evaluation will consist of monitoring the types of operations performed, the number of instructions used in each category, memory utilization, etc. These numbers will then be compared to binary machines performing the same type of work.”

The result was remarkable [173, p. 87]: “[...] *ternary structures are not “native” to a binary machine, their emulation is as efficient as the binary structures*”. The initial version was already competitive in both cost and speed to binary [139, p. 95]

In 1977 David C. Rine edited a compilation of MVL papers [139] on switching theory, mixed radix systems, computing and applications. For example the paper by Michalski concluded that MVL is very useful for machine learning with problems that are multi-modal and multi-class [139, p. 532]. Another relevant paper was by Hamacher and Vranesic with a direct comparison of binary to unbalanced and balanced ternary for high-speed parallel multipliers. Balanced ternary scaled several factors better than binary in simulations. In 1988 Kameyama et al. demonstrated this practically with a 32x32 bit signed quaternary multiplier CMOS chip. It outperformed binary 2x in area, power consumption, transistors, 7x in amount of interconnects and with similar delay [138].

In 1974 Mouftah and Jordan [174] showed ternary circuits implementations with CMOS such as the AndOrInvert (AOI) gate. Their 1977 paper [175] reported on other important circuits for a ternary computer such as a D-flip-flop, counter, MIN/MAX/STI gate. In the same year Etiemble and Israel [176] presented an overview of ternary circuits for various logic families including CMOS and circuits such as a 16T4R comparator.

Probably the most cited paper for ternary computer researchers is the 1984 *Multiple-Valued Logic—its Status and its Future* paper by Hurst [126]. This introductory paper contains a wealth of knowledge on MVL concepts and technologies from that era, some which are still relevant today. For example MVL system architectures are discussed that use binary logic but with MVL interconnect infrastructure. Such architectures can be found in modern high-end compute products such as the Nvidia RTX 4090 video card using Micron's GDDR6x [121] and USB 4.2 products [119].

Another well cited paper is the 2001 *Third Base* paper by Hayes [125]. This paper visualizes the radix economy well. It also mentions several unique benefits for algorithms and software when using radix-3 data structures regardless of hardware implementation.

2 The benefits of ternary

An honorable mention is the 90-slide presentation of Charles Bay at the 2015 C++Now conference which covers the ternary computer from a software engineer/programmer perspective while constantly comparing to binary [177].

Ternary is considered in many novel compute paradigms in the last two decades, including neuromorphic computing [178], stochastic computing [179], optical computing [180], quantum computing [181] and optical quantum computers [182]. Several conventional ternary computer paradigms have also been published, with only a few in peer-reviewed channels. They are listed in **Appendix G.8**.

2.5 The seven C's of ternary

Paper D is a short position paper on the many application domains where ternary signals and logic can be more advantageous than binary. The paper is relevant as the advantages of ternary are scattered in literature, dated or explained with scarce information. This is acknowledged by the technical program for the 2015 IEEE International Solid-State Circuits Conference (ISSCC) which mention that "*MVL is used "under the hood" in a variety of contexts, albeit under varying names*" [183]. Prior efforts can be found in for instance [140] (especially table 4), [125], [184, p. 32], [126, p. 1160, 1174–1175], [139, p. 475–548]. The work by K.C. Smith [185, p. 630–633] titled "The Prospects for Multivalued Logic: A Technology and Applications View" has a similar approach as **Paper D** with 5 of 7 C's being very similar. The missing two were cyber-security and energy consumption. Cyber-security was probably not as important in 1981 as it is today due to the internet. Energy consumption was not yet a first-order design constraint which became apparent after Dennard scaling stopped [8].

2.5.1 Computation

Claude Shannon recognized the theoretical potential of balanced ternary for faster mechanized processing and general purpose computation [23]. Although no balanced ternary computer system exist demonstrating binary parity, practical demonstrations of faster computation with ternary do exist both historically [138] and recently [186].

The classical CPU performance equation [2, p. 36] shown in Eq. 2.2 is radix independent:

$$CPU_{total\ execution\ time} = \frac{Seconds}{Program} = \frac{Instructions}{Program} * \frac{Clockcycles}{Instruction} * \frac{Seconds}{Clockcycle} \quad (2.2)$$

This equation has three terms, instruction count, cycles per instruction (CPI) and clock rate.

Instruction count

The instruction count can be reduced significantly with balanced ternary [187] which is partly due to information compression. Ternary assembly is stored in ternary memory cells. Further reduction is possible when allowing ternary data paths using a 3-way branching which for binary needs at least 2 instructions.

Cycles per instruction

The data transfer instructions like load/store word are one of the most used instructions (for instance 35% of all instructions on average in a CPU benchmark [2, p. 174]). As discussed in Chapter 1.5.4 and in [9], [10], data transfer has increasingly become problematic causing the CPU to idle. This is especially impeding memory-intensive programs like AI applications [79]. Superscalar architectures that enable instruction-level parallelism to decrease the CPI are heavily burdened by the long delays needed for data transfer. A higher radix increases the information density and thus brings more data closer to the ALU. Perhaps surprisingly, Reichenbach et al. [188] report that an inefficient implementation of two bits for 1 trit (binary coded ternary or BCT) shows 2.3x speed-up with arithmetic operations including add/subtract. A similar speed-up of 2.7x is reported in [189] with ternary neural networks. The usefulness of BCT is known for a long time in ternary content-addressable memories (TCAM). By using ternary encoding with regular binary SRAM cells, *"low-latency search is done in an energy-efficient manner"* at the cost of density (the unused fourth state) [140, p. 7].

Clock rate

Transistors have been demonstrated to operate at 1.5 THz [70] which is nearly 1000x faster than modern CPU's. Interconnects do not scale as well as transistors as discussed in Chapter 1.4.1 and increasingly limit the clock rate. The long global interconnects due to more metal layers became the weakest link. Transistor density is clearly prioritized over single thread performance as shown in Fig. 1.1. Rather than trying to increase the maximum frequency marginally, larger benefits can be achieved by utilizing the interconnect infrastructure with a higher radix and lower frequency. This strategy has recently been used by Micron for high speed communication with GDDR6x (graphics) memory [121], [190] as increasing the clock rate further became impractical with CMOS.

An interesting experiment would be to encode global interconnects with binary encoded ternary or binary encoded quaternary as a significant part of the available transistors are used as repeaters (grey silicon [7]) and could be repurposed in a higher radix architecture with less metal layers. With shorter global interconnects and assuming no thermal issues, clock rate could potentially increase.

2.5.2 Communication

The benefits of ternary (and higher radices) for communication have been discussed in early ternary literature from 1961 [166]:

2 The benefits of ternary

”In digital communications systems various workers have shown that, to realize much higher accuracy in data transmission the radix 2 must be abandoned and high-redundancy codes developed. For this application, ternary circuits and codes are only the beginning of multi-valued elements and logic.”

In the modern era, the domain of high speed communication has only recently adopted higher radix communication. Before 2017 inter-chip and peripheral signaling standards for USB, I2C/SPI, Thunderbolt and PCI express have mostly been binary at the physical level [191]–[193]. The notable exception was Ethernet, WIFI and Bluetooth which were already using a higher radix but with less speed per lane [191]. Currently, the latest editions of these standards are all either binary encoded ternary (I3C [193]), PAM-3 (USB 4.2 [119], Thunderbolt 5 [194]) or PAM-4(PCI Express 6 [192]). WIFI 7 (IEEE 802.11be) uses QAM-4096 modulation (radix-4096, 12 bits/symbol) signals [195]. Bluetooth Classic uses differential phase shift modulation (8-DPSK, radix-8) for its fastest data transfer mode. Interestingly, Bluetooth Low Energy (BLE) uses the binary version (GFSK) [196]. BLE’s lowest power mode send multiple packages for a single bit thus exchanging range and speed for lower power. In [120] Ransom Stephens explains why binary communication is considered impractical by signal integrity engineers when higher bandwidth are needed.

2.5.3 Energy Consumption

The power equation in **Appendix G.3** has a data dependent component; dynamic power. With balanced ternary less switching activity is needed:

- For inversion only logical values -1 and 1 need inversions compared to 2’s or 3’s complement based inversion [197]
- Carry signals happens less often (only $\frac{2}{9}$ states) in a 2-input balanced ternary full adder and the ripples are much shorter due to the larger number capacity [184, p. 32].

Much power is dissipated in the long interconnects and due to many metal layers. Repeaters can form 50% of the dynamic power consumption and leakage power [71]. With ternary signals less static and dynamic power is dissipated as full swing switching events are fewer. Repeater gates, interconnects and devices are also fewer as the information density increases. Alemendar et al. (2017) [189] report that ternary neural network with BCT signals result in 3.1x better energy efficiency. The current state-of-the art video cards Nvidia RTX 3090 and RTX 4090 both feature Micron’s GDDR6X DRAM memory [121], [190]. This memory is using PAM-4 rather than NRZ (radix-2). The authors report 57% more bandwidth, 50% lower clock frequency and reduced circuit complexity: *”including complete removal of the PLL and reduced gate fan-out design, both of which combine to lower active power while simultaneously relaxing the on-die timing budget”* [121].

Recently the automotive industry adopted the MIPI A-PHY standard to prepare for the next generation autonomous vehicles. These cars can be considered energy efficient mobile compute platforms with extreme resilience, reliability and bandwidth requirements. Two profiles have been designed with binary for the "*simple and low cost*" profile and higher radices for the "*high bandwidth, EMC immunity and lower packet error rate profile*" [198]. Similarly, the IoT industry adopted I3C, the successor to I2C and SPI. This standard uses binary encoded ternary for its fastest transfer mode and is designed for extreme low power consumption. In a white paper reduced power consumption is reported of $> 4x$ compared to I2C for the fastest transfer modes [199].

2.5.4 Compression

Compression benefits using radix-3 compared to radix-2 can come in many forms. The transition to higher radix storage is cited as a "*quantum jump in density well beyond the roadmap*" [111, p. 229] and is illustrated with a timeline. Commercial storage products include multi-level cell (MLC) solid state drives (SSD's) and Micro Secure Digital (SD) cards commonly found in digital camera's. Samsung announced that the new GDDR7 memory chips for video cards will use radix-3 signalling (PAM-3) and are quoted to be both faster and consume less power per pin [200].

Similarly to memory cells, the amount of input-output (I/O) pins are reduced as each pin has a higher information density. Die space is costly and scarce and interface pins are generally large. Pin efficiency is a necessity for high bandwidth products as increasing frequency after 25GHz becomes inefficient in CMOS circuits [121]. Fewer pins and fewer devices like memory cells also reduce the amount of interconnects. Interconnect reduction is probably the most cited benefit in relation to higher radices [126, p. 1161] and [201]. Four terminal logic devices like transistors have 3 interconnects (gate, source, drain) with the fourth being connected to the base. Reducing a single transistor thus reduces the amount of interconnect with 3x. It should be noted that not all interconnects are equally problematic. In general long interconnects are troublesome [9]. Reduction of transistor count with balanced ternary logic has been demonstrated with semi-floating gate transistors by Gundersen and Berg [202]. Their balanced ternary full adder chip consisted of 32T compared to the common binary 28T design. In addition, balanced ternary handles negative numbers which further increases transistor count for the binary case when doing fair comparison.

The instruction set (ISA) of a RISC processor can also be reduced with ternary signals [187]. The 40 RISC-V instructions in the RV32i minimal Instruction Set Architecture (ISA) was reduced to just 24 with full compatibility. In addition the executable was smaller in size. This is possible because some instructions become redundant with balanced ternary such as unsigned operations. Opcodes, function codes and operands can be compressed due higher information density making the instruction format denser.

2.5.5 Comprehension

The persistence of binary has led to a binary-mindset [27]. Software engineers learn data structures such as binary trees and Boolean control flows with *if-else* constructions. Ternary data structures such as ternary search trees can exceed binary in performance [125], [203].

Like naturally binary signals such as {true, false}, naturally ternary signals are found everywhere. In the process control industry actuators can be controlled with {rotate clockwise, rotate counter-clockwise, stop rotating} [166], magnetic field sensors that measure {attract, repel, out-of-range} or polarized light sensors that measure {x-polarized, y-polarized, no light}. In physics subatomic particles such as protons, neutrons and electrons hold {positive electrical charge, no charge or negative charge} respectively. In communication the widely used Morse code consist of three symbols {dot, dash, space}. Modelling them is optimal with the ternary computer alphabet but also more understandable as there is no need to split states over multiple variables.

2.5.6 Cyber-Security

Synthesis of (static) hazard-free logic gates have already been mentioned in the early '60 [165]. Side channel attacks by reading power traces are more difficult with either dual-rail balanced ternary or binary encoded ternary. The middle value can be cancelled out and less state transitions are needed compared to the same information in binary. Power balancing is discussed in [204, p. 21]. Binary encoded ternary using the set {01,10} and a single spacer state {00} makes *"power consumption of synchronous circuits independent from data processed"* [205].

The work by Hossain et al. discusses the advantages and pitfall of MVL for hardware security [98] and is a great introduction to the topic in general. Cambou et al. addresses the benefits of ternary computing for information assurance [206], Physically Unclonable Functions (PUF) with ternary RRAM [207], using mixed radices for security [208] and how ternary can enhance key distribution protocols [209].

Safer systems can also be created with ternary memory from a software point of view. The third state can be used as the uninitialized state of a Boolean variable. Safety-first programming languages like Ada can catch initialization errors at compile time or can gracefully handle raised third state usage exceptions during run-time.

2.5.7 Design complexity

Chapter 1.4.3 discussed the EDA wall, the increasing cost of designing chips as transistor density increased to 100+ million per mm^2 . As discussed in **Paper D**, complexity exist

on many abstraction levels in the VLSI design and verification stack. In general balanced ternary reduces complexity with:

- **Negative numbers.** They can be naturally encoded with 3 states, thus removing the need for 2's/3's complement and signed arithmetic. The sign is the most significant non-zero trit while in binary this is a dedicated bit.
- **Rounding to nearest integer.** Rounding is identical to truncation [23], [24]. For truncated multiplication, balanced ternary outperform both radix-2 and other radices [210]
- **Semantically three-valued signals.** An example is {rotate clockwise, rotate counter-clockwise, stop rotating}. In binary this signal would need 2 bits to be encoded.

The interconnect routing problem is greatly simplified with a higher radix as fewer interconnects are needed. Interconnect routing is an exponential problem with newer technology nodes and of the reason why more metal layers are needed [9]. For the same channel capacity, the frequency can be reduced using Shannon-Hartley's equation. Lower frequencies make design simpler as signal integrity becomes exponentially harder with higher frequencies. Hollis et al. report[121]:

"In contrast, while these alternative solutions demonstrate clock reception at 9 and 12 GHz, respectively, the maximum clock frequency of the GDDR6X interface is only 5.5 GHz when supporting a 22-Gb/s external data rate. This fact enables circuit simplification, including complete removal of the PLL and reduced gate fan-out design, both of which combine to lower active power while simultaneously relaxing the on-die timing budget."

Others, like Reichenbach et al. [188, p. 43698] suggest that it might be possible for binary CPU's with ternary data paths to replace the floating point units with ternary integer arithmetic. Zhu et al. (2023) [211] suggest that quantized AI accelerators based on convolutional neural networks (CNN) using ternary weights can replace multiplication operations with addition operations. This results in a better balance between accuracy and speed. The USB-IF Electrical Workgroup noted that the radix-3 design for USB 4.2 has a slightly simpler architecture than radix-4 (despite not being a multiple of 2) and that test equipment is available [119].

2.6 Critique

Radix-2 logic devices are the simplest to build, verify, control and understand and have been crafted to near perfection over a period of 70 years. The amount of investments and long roadmaps have led to enormous inertia. There are many applications where older

2 The benefits of ternary

technology nodes such as 45nm or the 20 year old 130nm provide all the needed die area, bandwidth and density. With the chiplet strategy older nodes can be compactly combined with new nodes which is more cost effective than a monolithic process. Other strategies such as 3D integration with complementary FET (CFET) [212] push Moore's law even further, although without addressing dark, dim or grey silicon.

Like radix-2 technology, radix-3 technology has its challenges. Arguably the largest disadvantage of ternary is practical rather than fundamental. Constructing efficient ternary logic circuits is hard due to missing tri-stable or ternary compatible switching devices [213]. The cycling gate (also called increment or shift) for example is a unary gate that increment the input with one state such that $-1 \Rightarrow 0$, $0 \Rightarrow 1$ and $1 \Rightarrow -1$. This fundamental and common function needs several devices to be constructed. Two cycling gates and a buffer gate are found in a ternary full adder which makes them inefficient compared to binary. As mentioned earlier, despite the lack of ternary compatible devices, BCT or ternary signals can still generate benefits at the system level.

At present none of the reported ternary technologies satisfy the 5 requirements for good ternary compatible transistors at VLSI scale: *"concatenability, non-linearity, feedback elimination, gain and a functionally complete set of gates"* [214, p. 166]. To compete with CMOS it should also have *"zero static power dissipation in all stable states, similar output impedance in all states and have no need for passive devices such as resistors"* [126, p. 1168]. With multi- V_{th} CNTFET some ternary logic functions can be simulated efficiently, especially when they are part of mixed radix functions (discussed in Chapter 4). Multi- V_{th} CNTFET for ternary logic cannot construct cycling gates efficiently and has not been demonstrated in production. The lack of ternary logic at wafer scale and in particular with CNTFET technology is a common critique [215], [216] although it is certainly researched [31], [131], [217]. Skywater Technologies Foundry is offering experimental fixed V_{th} CNTFET wafers with multiple CNTFET layers and RRAM [218].

Another disadvantage of ternary (and higher radices) is its reduced noise margin [219]. The noise-margin is halved for ternary, which might cause erroneous switching events and other effects [220]. Crosstalk with ternary logic and CNTFET have been studied in [221]. Other challenges include sharp switching points ("the 4-eyes") with uniform distances and self-restoring levels [162], [222]. With ternary logic less interconnects are needed which allow slightly larger spacing between dense interconnects regions and lower frequency which improves signal integrity. System level verification is needed to evaluate the trade-offs of reduced noise margins compared to the advances in reducing noise with new technology nodes.

The lack of MVL EDA tooling hinders faster research and development [97], [98]. A valid critique on ternary are the many knowledge gaps and limited experimental data. For example do ternary clock signals with 2 rising edges and 2 falling edges per cycle reduce CPU design complexity? Is a dual power supply power delivery network needed to unlock some ternary benefits at the system level? MVL EDA tooling is discussed in Chapter 4.

The most outspoken critic of MVL for computing is Prof. Etiemble who has contributed to the ternary and MVL community since the 1970's [176]. A selection of his main critique can be found in [128], [159], [161], [215], [216], [223]–[227]. Some arguments have proven to be outdated [161] (for instance NVlink, PCI express and USB are now radix-3 or radix-4) or based on absence of recent VLSI evidence rather than fundamental reasoning.

2.7 Conclusion

Herb Sutter poetically wrote "*The Free Lunch Is Over*" in 2005 [228] about the impending scaling problems discussed in Chapter 1.4. The semiconductor industry is forced to pivot and now reconsiders higher radices for all aspects of computing including logic [62, p. 4]. In this chapter the benefits of radix-3 are discussed as it is the optimum radix according to the radix economy. Ternary or BCT signals for computing are after period of decreased interest, at an all-time-high in terms of papers and research groups. The last 20 years also saw a broad adoption of MVL in standards and commercial products including consumer, automotive and IoT industries.

A small history on ternary computing with the most relevant literature between 1950's to 2000 was presented. Initiatives to design modern ternary computers in the last 20 years are listed in **Appendix G.8**. Comparison to binary is notoriously hard as for fair comparison identical features and capabilities should be compared which requires good knowledge of the inherent properties of both radices. The symmetry property of balanced ternary result in reduced switching activity and thus power consumption and should be the encoding of choice. Balanced ternary encoding should be chosen over unbalanced ternary regardless of the implementation with one or two power supplies. Arithmetic with numbers is more efficiently done in balanced ternary. When functionality contains both naturally binary and ternary signals then a mixed radix design is the optimal solution as the least amount of states are discarded. Examples of such designs are shown in Chapter 4.

The 7C's of ternary discussed in **Paper D** are seven applications domains where ternary has demonstrated advantages to some degree over binary. MVL has grown into a multi-billion industry and is certainly not a niche anymore [223]. The largest remaining binary frontier is logic which relies on bi-stable devices and extremely optimized binary signal architectures. Many initiatives exist to address this frontier with emerging devices such as multi-threshold CNTFET and RRAM but none have proven to be a competitor at wafer scale yet. Importantly, as discussed in Chapter 1.5, no fundamental principles prevent ternary logic to reach binary logic efficiency as measured by VLSI metrics. The largest barrier is an engineering and economical challenge to develop novel ternary compatible devices and cost-effective fabrication processes. With binary being the *status quo* much work is needed in education, EDA tool development and experimentation to unlock the true potential of ternary.

—“If it is pinched, it is a memristor”

Prof. Leon O. Chua, Inventor of the memristor [229]

—“Absence of evidence is not evidence of absence”

Prof. Carl Sagan, Astrophysicist and science communicator

3

Multi-state RRAM development platform

3.1 Introduction

The construction of an economical ternary computer requires hardware components that are compatible with the computing paradigm. The building blocks of any modern digital computer are the nano-sized elements that store and transform digital signals; memory and logic devices. How efficient, performant and cost-effective these devices can encode and process the radix determines the feasibility of an alternative, post-binary computing paradigm. The radix economy presented in Chapter 1.5.3 argues that the cost function $radix * width$ models devices for number representation well when a single device can encode multiple states. Mass-produced multi-state memory cells have been dominating the consumer mass-storage, high bandwidth, extreme low power and high-reliability industries. The notable exception is SRAM and DRAM, the high performance categories where bi-stable devices still rule. Conventional multi-state memory cells trap charges which are then related to a macro state but there are certainly more options. Many emerging multi-state memory cells exploit memristive switching effects [230]. Examples are spin-transfer torque magnetoresistive RAM (STT-MRAM), phase-change memory (PCM) and Resistive RAM (ReRAM or RRAM). RRAM, also called memristors is researched in this thesis work for storing ternary values. A great overview of the memory landscape can be found in [231] showing that RRAM has several desirable properties such as femtojoule and nanosecond switching and a large ratio between the lowest and highest memristance state. It is also together with PCM the most mature technology [230] with several foundries offering RRAM devices [232].

Grosch proposed a ternary computer in 1952 and considered using either tri-stable ferromagnetic or ferroelectric memory devices [164]. The plots in the report show a pinched

3 Multi-state RRAM development platform

hysteresis loop, the hallmark of RRAM according to Chua in [229]. Leon Chua constructed a theoretical framework in the 1971 seminal paper called "Memristor - The missing circuit element" [34]. It took more than 50 years until the process to create memristors suitable for ternary computers was discovered by Hewlett-Packard(HP). In the abtly named paper "The missing memristor found" Strukov et al. [233, p. 81] show great control up to four stable memristive states.

Like other emerging devices RRAM tries to address the holy grail of memory: picosecond read/write performance, high information density with nanometer dimensions, both reliable and durable and most importantly cheap to fabricate. Song et al. [230] reported in 2023 that: *"To date, no memristive device has been identified as a clear winner to replace CMOS-based electronics."* Several challenges have been identified in literature, most notably consistent and lower device-to-device and cycle-to-cycle variation during fabrication and endurance [231] as well as state drift during operation [234]. State drift is especially problematic with multiple states due to their inherently lower noise margin.

Memristors are thus still considered emerging memory technology with few commercially available products. Examples are the Fujitsu MB85AS12MT chips with memory controller [235], Knowm M+SDC series without memory controller and Crossbar ReRAM IP for custom integration. The Fujitsu product line can be purchased on breakout boards for around \$15,- for 4 MB non-volatile memory [236]. Discrete memristors in individual or crossbars configuration can be bought for \$120, respectively \$385,- dollar [237]. These devices are ideal for experimentation such as multi-state control and have similar read/write specifications [238] as devices that can be fabricated in commercial foundries [239].

This chapter is organised in two sections. In the first section **Paper A** is discussed which presents uMemristorToolbox, a software framework to control multiple states and experiment with physical memristors. The paper was the first in this PhD project. Since its publication the software has been extended for use in two MSc theses [240], [241] to allow more elaborate programming schemes and experiments. In the second section memristor simulation and multi-state RRAM circuits are discussed. These topics are central in **Paper C** which explores the feasibility of three-valued memristor applications using commercially available components.

3.2 Multi-state programming

The paper by Strukov et al. [35] demonstrated for the first time how Chua's theoretical memristors can be fabricated. They show several figures of merits including how to program multiple states in a single device with voltage and current. In [233, p. 34] they explain the discovery process and the relation between material and number of states. In [242] Pickett, Strukov and others conclude that *"the switching effect is primarily due to an effective tunneling distance modulation"*.

Campbell in [243] proposed the self-directed channel (SDC) memristor which can be purchased from Knowm. These devices are used in **Paper A** and **Paper C**. In the paper she discusses device fabrication, the forming procedure of new devices and various programming schemes relevant for continuous memristance programming. Stathopoulos et al. in [244] show the relationship between pulse duration and amplitude with several combinations resulting in the same state transition. They also demonstrate 4-bit (16 states) state density without cherry picking across multiple devices. Geler-Kremer et al. in [245] shows several additional figures of merit that are useful for demonstrating multi-state control and device behavior. Rao et al. in [36] proposes a 3-level control algorithm using only short forward and erase voltage pulses with adaptive width and height to increase state density to 11-bits (2048 states) in a single device. This was done on a CMOS circuit from a commercial foundry. They also investigated the material physics why their denoising process gives these remarkable state density results in both simulation and measurements.

Memristor or memristance programming can be divided in four quadrants with axes analog vs digital and voltage vs current. Analog signals are waveforms such as sawtooth, sine and custom shaped continuous signals. Digital signals are square wave signals. The amplitude, direction (sign) and amount of pulses are relevant in both analog and digital signals. The memristance can be programmed by controlling either the voltage or current through the memristor for both read and write actions. The memristance is unaffected if the voltage or current is low enough allowing indestructive reads actions. The range of the memristance is between the High Resistance State (HRS), which is typically hundreds of $k\Omega$ and the Low Resistance State (LRS), typically a few $k\Omega$. The hallmark I-V plot showing these boundary states is the pinched hysteresis loop through the origin, shown in Fig. 3.1. The figure also shows the two control directions for multi-state programming given a random memristance state; towards LRS using positive voltage or towards HRS using negative voltage. In both cases it must be above (or below) a critical voltage threshold for the programming effect to occur.

A diverse range of programming schemes can be employed to program multiple-states. For example an *erase-set* scheme puts the memristor in HRS and then set its to a state between HRS and LRS using a forward pulse. The reverse, a *set-erase* scheme is also possible, setting the memristor in LRS and then set it to a state between LRS and HRS using a reverse pulse. Both schemes require 2 pulses but allow programming from a known initial state in a normal situation and thus prevents the need for a prior read action. Another example is the *delta program* scheme where a memristor is nudged into a state using small positive or negative write actions (above the read threshold). The *delta-set* scheme or *delta-erase* scheme requires 1 pulse to transition between states but possibly requires a read action before writing since the initial state might be unknown. A memory controller can aid in choosing the appropriate scheme and parameters, assert that the writing action result is in the desired state, balance memristor utilization and/or flag erroneous memristors. The memory controller in **Paper A** adjusts the appropriate

3 Multi-state RRAM development platform

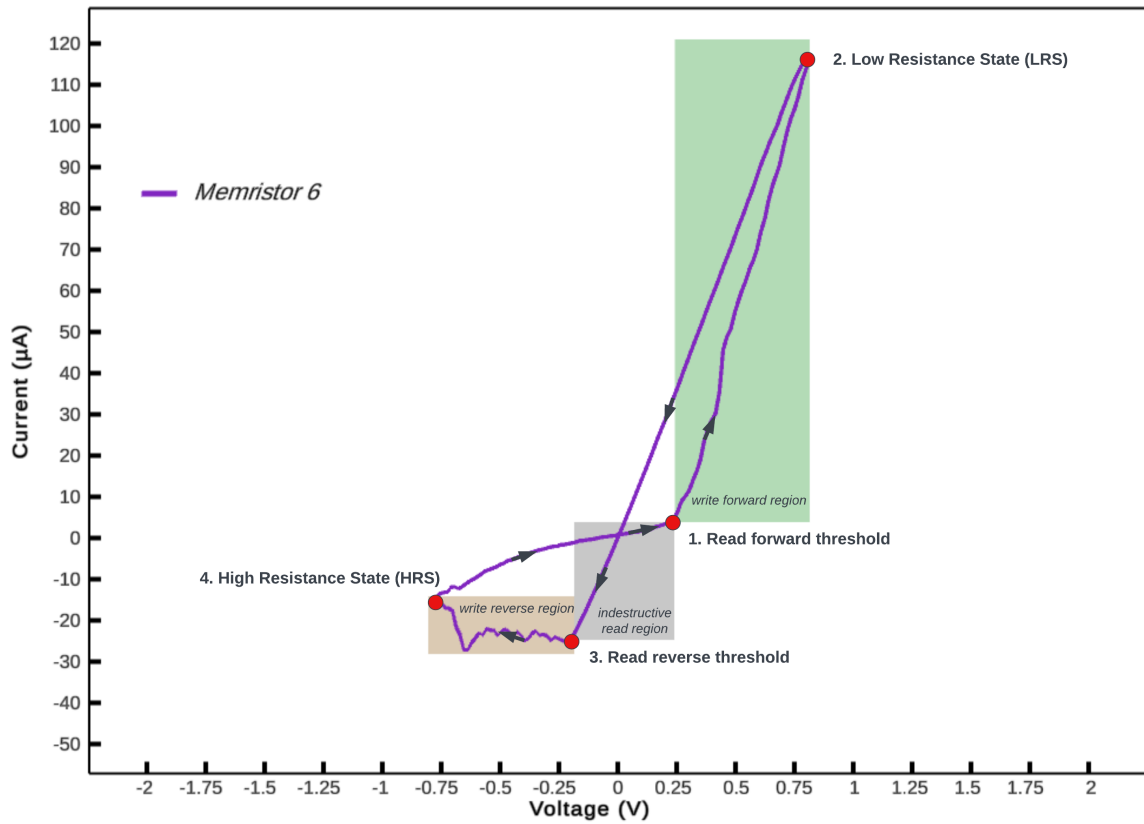


Figure 3.1: The three regions of voltage-controlled bi-polar memristance programming. Visible is that the forward and reverse regions are not symmetrical. The indestructive read region has two thresholds that can be used for measuring the memristance without influencing it.

scheme and validates a writing action.

The MSc thesis by Virk [241] shows several examples of multi-state programming schemes using uMemristorToolbox and the same memristors and setup as in **Paper A**. He also shows it being used for 7 uniformly separated memristance bands. 2-trits (9-states) is possible for a period of time in a cherry-picked instance as is shown in Fig. G.2 from **Appendix G.9**.

3.3 uMemristorToolbox: A new tool for experimenting with multi-state RRAM

3.3.1 Motivation

uMemristorToolbox was proposed in **Paper A** and presented at the IEEE ISMVL 2020 conference in virtual Miyazaki (Japan) due to the COVID pandemic. The first letter stands for Unity the development environment and μ , the die space and programming time dimensions of the first devices under test. uMemristorToolbox is a software framework to control multiple states and experiment with physical memristors. The tool addresses the 15% implementation versus 85% simulation knowledge gap identified by Taherinejad and Radakovits [246]. The tool was conceived after working with the Knownm memristors and open source tool "Memristor Discovery" written in the Java computer language [247]. This application has a general purpose design and lacks several experiments relevant for multi-state programming research. The initial design requirements can be summarized to:

- **A wide array of multi-state control signals.** The target audience are ternary researchers and computer engineering students that seek hands-on MVL experience with physical memristive devices. Memristors can be programmed with various schemes using unipolar or bi-polar pulse trains (digital) or custom waveforms (analog). The tool should process at least 2 decades of ohmic range from $1k\Omega$ to $200k\Omega$ resistance levels, -3 V to +3 V programming voltages and 1 Hz to 1 MHz ($1\ \mu\text{s}$ period) switching speeds.
- **Memristor characterisation.** Memristors come in various packages and with different materials. Any experiment must start with a sanity check to see if the I-V plot is a pinched hysteresis loop and how the response to frequency is. New memristors must be formed first which defines their sensitivity and resistance range.
- **Ternary memory controller and benchmark.** For memristors to be suitable for ternary applications a controller is needed to program ternary states regardless of the previous state. This is notoriously difficult as memristors are non-linear, non-deterministic devices with small variations between devices, production runs and drive histories. Benchmarks are needed to validate state retention with multiple reads and validate state confusion with random writes.
- **Rapid data-driven prototyping environment.** The Unity engine is designed for real-time 2D/3D data visualisation and uses C#, one the most popular high level programming languages.

3.3.2 Architecture

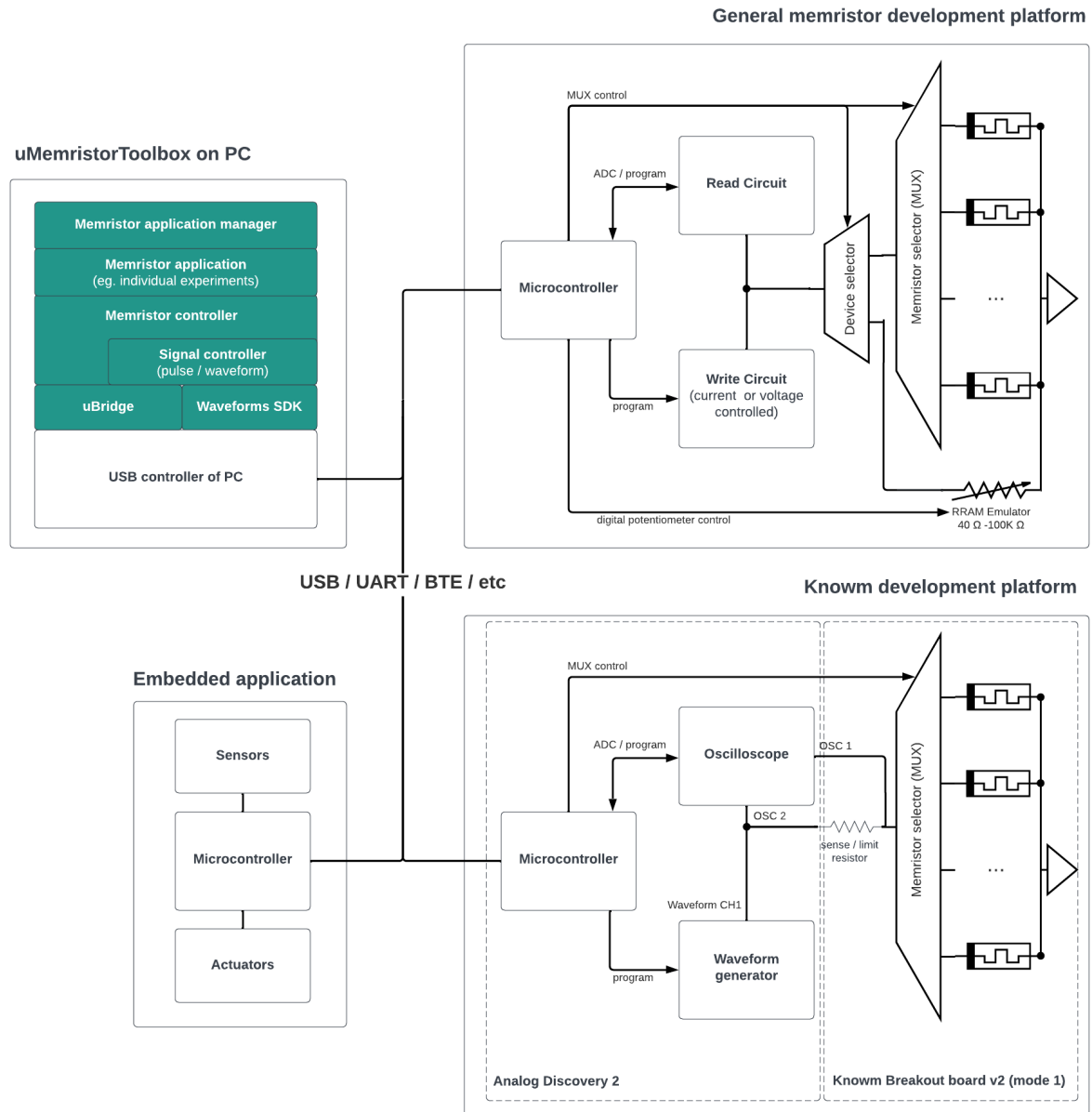


Figure 3.2: Architecture of uMemristorToolbox with memristor development platform

The architecture of uMemristorToolbox can be seen in Fig. 3.2 which adds a generic memristor development platform option and refactored software architecture developed after **Paper A**. The software changes can be found in the commit history of the Github repository [248]. Three implementations of the memristor development platform can be found: **Paper C**, [241] and [249].

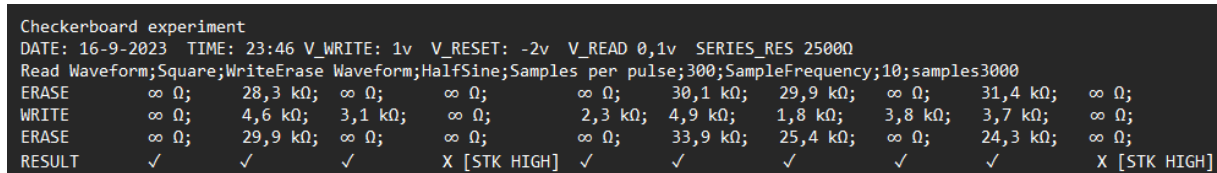
3.3 uMemristorToolbox: A new tool for experimenting with multi-state RRAM

uMemristorToolbox started as a direct port of the open source Memristor-Discovery application by Knowm with the exception of the jSpice simulator. This simulator was used to estimate resistance values based on measurements at low currents. uMemristorToolbox reports only raw measurement values. The Waveforms SDK for C# was integrated to communicate with the Analog Discovery (AD) 2. Bi-directional communication with microcontrollers such as the AT2560 used in the Arduino Mega is made possible through a serial port communication layer 'uBridge'. This enables scenario's such as remotely programming a memristor via uMemristorToolbox and reading memory values back. Other scenario's include controlling the memristor development board with microcontrollers and using uMemristorToolbox for central data logging and experiment control.

3.3.3 Experiments

Four of the six original Knowm experiments have been implemented either 1:1 ("board check"), combined into one experiment ("DC" and "hysteresis") or under a different name ("pulse experiment") to better reflect the goal. The missing two ("synapse12" and "classify12") work on differential pairs of memristors. Pairs emulate synapses of a neural network and are not the focus of this tool. The experiments available in uMemristorToolbox are listed below. The figures contain new measurements with the same setup as described in **Paper A**. Not listed are the *EraseWriteRead* and *ReadAfterDisconnect* experiments which were used to investigate the role of a full system power down and MUX switching on the programmed state.

Board check experiment



```
Checkerboard experiment
DATE: 16-9-2023 TIME: 23:46 V_WRITE: 1v V_RESET: -2v V_READ 0,1v SERIES_RES 2500Ω
Read Waveform;Square;WriteErase Waveform;HalfSine;Samples per pulse;300;SampleFrequency;10;samples3000
ERASE ∞ Ω; 28,3 kΩ; ∞ Ω; ∞ Ω; ∞ Ω; 30,1 kΩ; 29,9 kΩ; ∞ Ω; 31,4 kΩ; ∞ Ω;
WRITE ∞ Ω; 4,6 kΩ; 3,1 kΩ; ∞ Ω; 2,3 kΩ; 4,9 kΩ; 1,8 kΩ; 3,8 kΩ; 3,7 kΩ; ∞ Ω;
ERASE ∞ Ω; 29,9 kΩ; ∞ Ω; ∞ Ω; ∞ Ω; 33,9 kΩ; 25,4 kΩ; ∞ Ω; 24,3 kΩ; ∞ Ω;
RESULT ✓ ✓ ✓ X [STK HIGH] ✓ ✓ ✓ ✓ ✓ X [STK HIGH]
```

Figure 3.3: The board check experiment. Shown are 9 out of 16 memristors. The first column is a test pattern with the MUX not selecting any memristor.

The board check experiment is shown in Fig. 3.3. This experiment is typically done when working with a new memristor or when starting a session after not using the memristors for some time. Visible are the HRS and LRS states with a common programming scheme. This experiment can be repeated to "form" a new memristor or to try to recover memristors that are "stuck" in one of the states as fairly high voltages are used. The experiment performs a erase-read-write-read-erase-read pattern. In between each action is a 25 ms delay. Default settings are -2 V and +1 V 10 Hz halfsine wave for writing and +0.1 V 10 Hz square wave for reading.

DC experiment

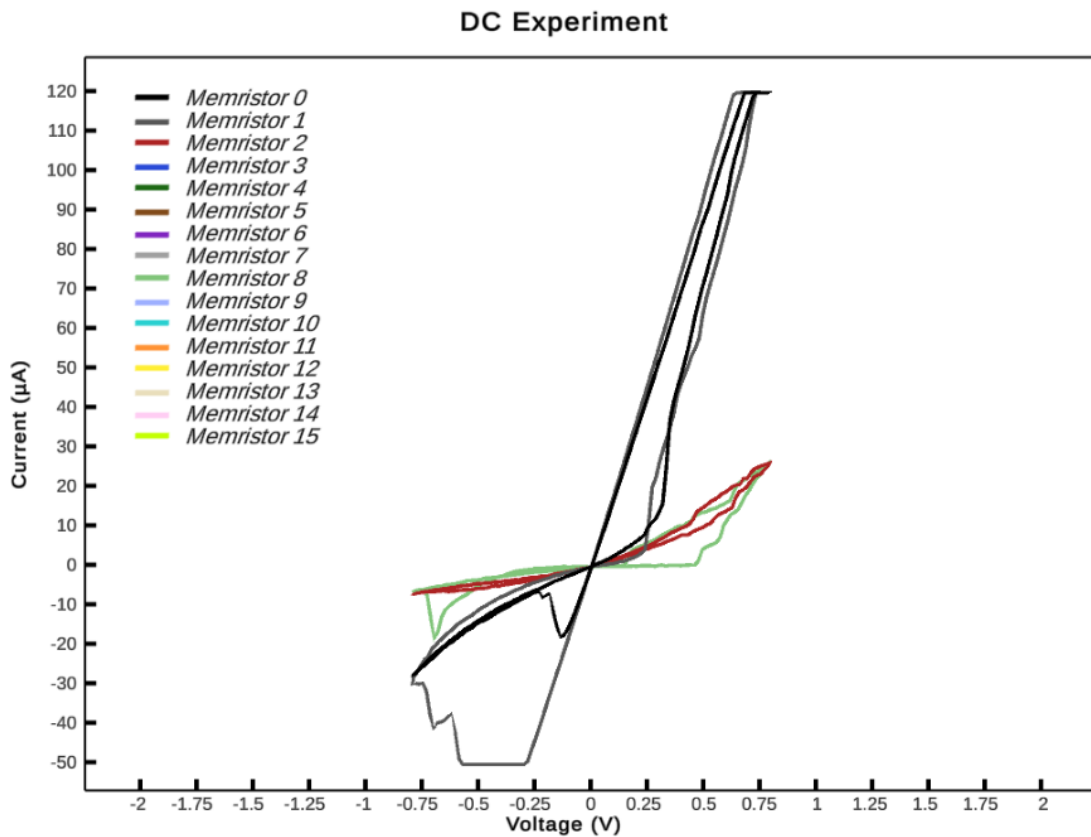


Figure 3.4: The DC experiment. Shown is the average of 5 pinched hysteresis loops through the origin from 4 out of 16 memristors. The plot clips measurements outside the viewport.

The DC experiment shown in Fig. 3.4 is a little misnomer. It contains both DC and AC experiments by showing one or more I-V plots at a certain frequency. This experiment is typically run after the board check experiment as it shows the signature pinched hysteresis plot of the memristor if the memristor is in a good condition. It is an important sanity check before and sometimes in between experiments. By changing the frequency, faster writes are possible but at the same time the hysteresis loops collapse such that the gap between HRS and LRS narrows. This is detrimental for multi-state programming. Default settings are single erase before doing the experiment and one 2 Hz cycle of a triangle-updown 1.6 V peak-to-peak waveform. Individual or all memristors can be selected. Multiple cycles can be averaged or overlaid.

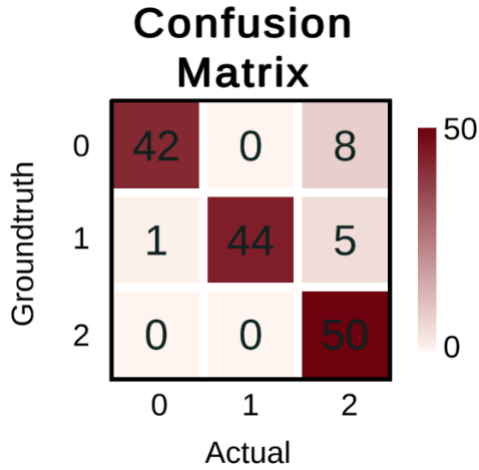


Figure 3.5: The random write experiment. Visible is a confusion matrix of $N=50 * 3$ random states. The error rate is $14/150 \approx 9.3\%$

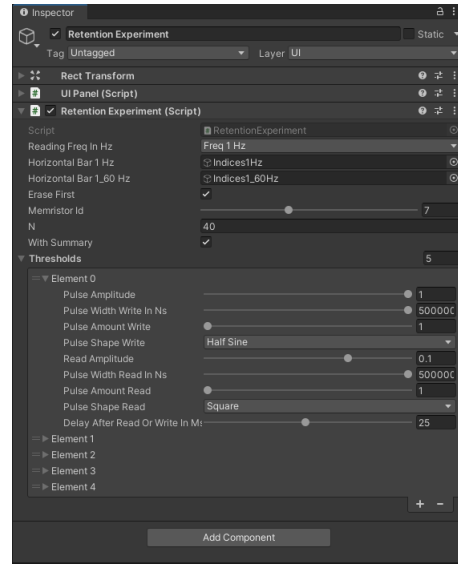


Figure 3.6: Multi-state programming scheme interface used in the retention experiment. Shown is a wide array of control for each write and read action.

Random write experiment

The random write experiment is shown in Fig. 3.5. It uses the memristor memory controller for ternary states described in **Paper A**. The confusion matrix plot show the desired state and the actual state after programming. The experiment starts with two erase actions followed by the creation of N number of states of each of the 3 logic values. These values are then randomized and send to the memory controller to be scheduled. The experiment validates the need for a memory controller as the log shows that often more than one write cycle is needed to get the desired state.

Retention experiment

The retention experiment is shown in Fig. 3.7. This experiment tests the non-volatile property of the memristors by performing a single write followed by a periodic read operation. It also allows for testing which read operations (such as voltage amplitude) are non-destructive. The experiment can optionally start by performing an erase operation. The experiment uses the programming scheme interface shown in Fig. 3.6 which allow customs programming schemes. The experiment can either show each write-once-read-multiple action as a discrete lines as shown in Fig. 3 and 4 of **Paper A** or as a continuous line as shown in Fig. 3.7. The retention experiment does not use the memristor memory controller for controlling the desired digital state. This allows for experimentation of all programming schemes including delta and erase-set programming schemes and in the direction from HRS to LRS or from LRS to HRS. Write operations without memory controller makes multi-state control naturally less predictable as state density increases.

3 Multi-state RRAM development platform

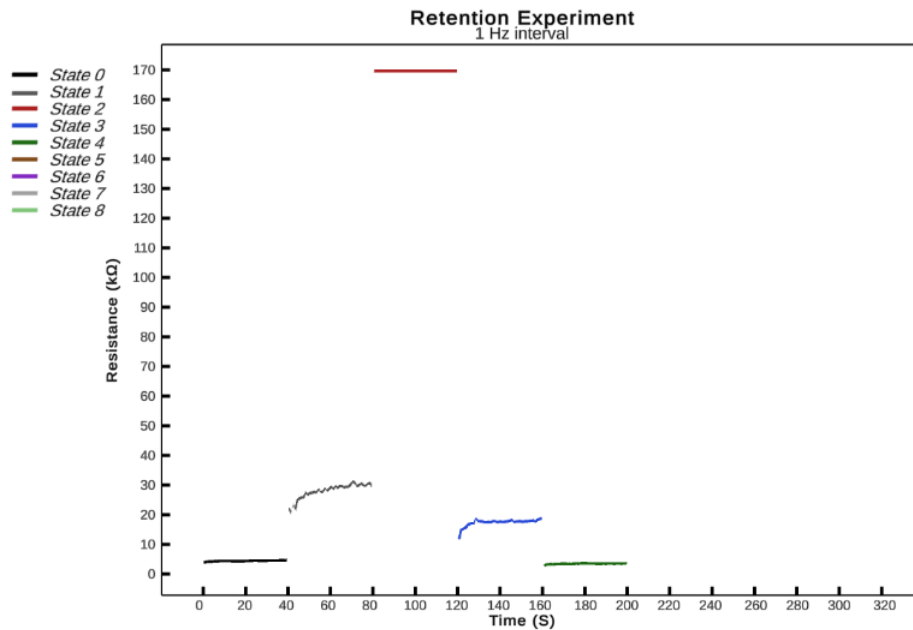


Figure 3.7: The retention experiment. Shown is a 1-trit pyramid shape delta-programming scheme with very similar parameters as in **Paper A**. The only difference is that instead of 0.4 V forward voltage for the middle state now 0.35 V or -0.35 V was used. The red bar is clipped to the ceiling as the values are several hundred k Ω .

3.3.4 Application: Embedded ternary system

Paper A discusses an embedded ternary system using a (binary) Arduino Mega board and serial interface to uMemristorToolbox. This proof-of-concept system had a 4x4 keypad such that each of the 16 memristors had a unique physical button. The logical state in the memristor increased (modulo 3) after pressing a button. A read operation was performed after a write state such that the embedded system could show the measured ternary state on a 8x8 LED matrix. The LEDs, grouped in squares of 2x2 were either all off, 2-on-2-off or all on. The workstation with uMemristorToolbox installed ran the Analog-Digital-Conversion (ADC) experiment as the continuous memristance state is digitized in a discrete logical state. In the experiment uMemristorToolbox served three purposes: logging memristor writes/reads (see Fig. 3.9), relaying the write instruction to the memristor from the embedded system and controlling the experiment by showing the input (=groundtruth) and output. The UI, shown in Fig. 3.8, allows the user to emulate the embedded system input and output in case no embedded system is available. The proof-of-concept application shows that multiple multi-state memristors can easily be integrated with embedded controllers using uMemristorToolbox.

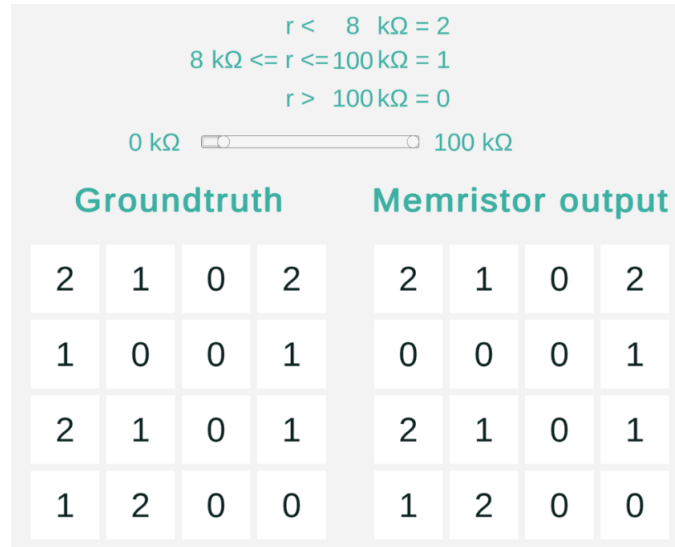


Figure 3.8: The ADC experiment. Visible are two slide bars to adjust the threshold to classify a logical 2 or 0 with the middle region being classified as state 1. The bottom left is memristor with Id 1. Id 2 is on the same row next to Id 1 while Id 5 is above Id 1. During this session Id 9 was requested to become logical 1 but was stuck at logical 0.

```

One trit adc experiment
DATE: 17-9-2023 TIME: 01:05 V_WRITE: 1v V_RESET: -2v V_READ 0,1v SERIES_RES 25000
MID;1;STATUS;OK;ACTION;WriteState1;ACTUAL;1;GROUNDTRUTH;1;K_OHMS;34,22271;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;26,22271;D_ToUpperThreshold;65,77729;TRY;0
MID;2;STATUS;FAIL;ACTION;WriteState1;ACTUAL;2;GROUNDTRUTH;1;K_OHMS;5,19324;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;2,80676;D_ToUpperThreshold;94,80676;TRY;0
MID;2;STATUS;FAIL;ACTION;WriteState1;ACTUAL;0;GROUNDTRUTH;1;K_OHMS;oneindig;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;oneindig;D_ToUpperThreshold;oneindig;TRY;1
MID;2;STATUS;FAIL;ACTION;WriteState1;ACTUAL;2;GROUNDTRUTH;1;K_OHMS;5,175447;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;2,824553;D_ToUpperThreshold;94,824553;TRY;2
MID;2;STATUS;FAIL;ACTION;WriteState1;ACTUAL;0;GROUNDTRUTH;1;K_OHMS;oneindig;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;oneindig;D_ToUpperThreshold;oneindig;TRY;3
MID;2;STATUS;OK;ACTION;WriteState2;ACTUAL;2;GROUNDTRUTH;2;K_OHMS;2,63809;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;5,36191;D_ToUpperThreshold;97,36191;TRY;0
MID;8;STATUS;OK;ACTION;WriteState1;ACTUAL;1;GROUNDTRUTH;1;K_OHMS;11,09907;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;3,099066;D_ToUpperThreshold;88,90993;TRY;0
MID;6;STATUS;FAIL;ACTION;WriteState1;ACTUAL;2;GROUNDTRUTH;1;K_OHMS;6,894299;LowerTHRESHOLD;8;UpperTHRESHOLD;100;D_ToLowerThreshold;1,105701;D_ToUpperThreshold;93,1057;TRY;0
  
```

Figure 3.9: Partial log of the ADC experiment of Fig. 3.8. Visible is that the memristor memory controller often needs several actions to program the correct state.

3.4 Ternary memory controller circuit

To get started with RRAM experimentation requires memristors, a 1 MHz+ waveform generator and an oscilloscope. Most electronic labs have a waveform generator and oscilloscope. Eight bare memristors without memory controller and breakout board can be bought for \$ 120 [237]. This setup is not really suited for classroom settings and without a multiplexer is also limited to manually controlling individual memristors. Many universities use Digilent Analog Discovery (AD) kits for electrical engineering education. These portable USB devices contain a waveform generator and oscilloscope. An AD compatible breakout board, 16 memristors, control software and instruction manual can be bought for \$ 365 [237]. These breakout boards use -1 pC charge injection multiplexers (Vishay DG445DY) as 11 pC charge injection multiplexers (ADG512) were proven to be too large and thus change the memristance state when selecting another memristor [249, p. 19].

3 Multi-state RRAM development platform

While 6x more costly, the AD + breakout board setup is great for first-time RRAM exploration. The control software, either Knowm’s Memristor Discovery or uMemristorToolbox removes or minimizes the risk of damaging the device when conducting experiments.

The AD and breakout board are too costly and large for embedded applications. The AD is designed for general purpose analog experimentation. A novel RRAM memory controller circuit designed for ternary operations was proposed in **Paper C**. The multi-state RRAM controller costs a fraction, has a smaller footprint and lower energy consumption. With an embedded microcontroller the memory controller circuit can function independent of uMemristorToolbox. The paper was presented at the IEEE ESTC 2020 conference in Vestfold, Norway. Central in the paper are multi-state RRAM simulation and implementation on a breadboard.

3.4.1 Simulation

A wide range of memristive (RRAM) device model exist in literature and can be divided in physics-based simulators and behavior-based simulators. Physics-based simulators [36], [242], [250] analyse the material properties and dynamics to asses the effects on resistive switching. These ideal models can be improved by aligning them using for instance electrical, physical and thermal measurements to approximate realistic device behavior. Physics-based simulation provides the most accurate data but also cost the most time to simulate. Depending on the complexity of the circuit, non-ideal behavior-based models can be considered. Examples of such models are the current-controlled TEAM model [251], voltage-controlled VTEAM model [252]. A updated version of the two models was found to be quick and reliable while simulating a network of 200.000 memristors in HSPICE [253]. For the simulation of memristor-based logic gates the IMPLY model [254] and MAGIC model [255] were proposed. The MAGIC model can construct a functionally complete set with NOR and NAND gates. Compared to IMPLY no additional control circuitry is required except at the initialization phase and is the closest to the operation of a conventional CMOS logic gate.

In **Paper C** Knowm’s Mean Metastable Switch model (MMS) [256] is used in the LTspice simulator. MMS is a behavior-based model. The same model is used in the MSc work by Virk [241]. In that work 2-trit (9 memristive states) were simulated using a delta-programming scheme on a novel multi-state RRAM development platform (see Fig. G.3 in **Appendix G.9**).

3.4.2 Implementation

The implementation in **Paper C** was a proof-of-concept to validate prototyping a multi-state memristor memory controller with short jumper wires and a breadboard. The

starting point was replicating the binary memristor controller design by Radakovits and Taherinejad [257]. The next step was to design, simulate and prototype a new write and read circuit to increase the amount of stable states from 2 to 3. The write circuit for the middle state was realized by adding another voltage bridge. The read circuit was made by adding a second opamp and reference resistor to make a 3-level window comparator. The output was biased with a voltage divider such that all three voltage levels were positive and could be read by a microcontroller ADC. The paper reports that the relative amount of extra discrete components needed was lower than the information limit of 58.5% when going from radix-2 to radix-3. This means that radix-3 at the macro scale has a slight advantage in terms of components count. This advantage becomes significant with larger PCB designs that address many memristors.

The design worked moderately well on a breadboard. The application that was explored in **Paper C** was *post-binary robotics*. In a *Async Random Write Once then Async Read Many* experiment with in total 81 random writes and 257 random reads a total of 64 reads were erroneous (25%). The random async experiment is very similar to the retention experiment which uses fixed interval writes and reads. The found 25% read error is significantly more than the 9% in Fig. 3.5 or 11% found in **Paper A**. All three result are not nearly low enough for commercial integration, but do set the stage for further experimentation.

Several improvements were identified to reduce costs, increase safe operation or functionality. For example, instead of two costly opamps, one opamp and two diodes can be used for a 3-level window comparator [258, p. 112]. The bipolar transistors can be replaced with MOSFETS, reducing component variation. An input MUX could be added to prevent multiple voltage bridges being accidentally enabled by software glitches, potentially creating a short-circuit. The breadboard implementation was the basis for a PCB-based multi-state RRAM development platform prototype shown in **Appendix G.10** (see Fig. G.4 and Fig. G.5). This prototype features programmable voltage and current pulses, a variable resistor to emulate memristors and a DIP socket. It is described in more detail in [241]. The memristors tested with this prototype PCB as well as the ones used in **Paper A** and **C** are shown in Fig. G.6 of **Appendix G.10**.

3.5 Conclusion

In this chapter **Paper A** was discussed which proposed uMemristorToolbox, an open source framework to experiment with physical multi-state memristive devices. The tool aids in the characterization of these devices with plots on hysteresis, non-volatility, read/write speed, random writes and state density. Multi-state storage is an important step towards higher-radix computing. A proof-of-concept 16-trit application was integrated with an embedded system to process multi-state input and output. The 16 trits were read from

3 Multi-state RRAM development platform

and written to a commercial 16x1 SDC memristors. This demonstrated the feasibility of a low-cost memristor memory controller circuit for ternary operation and validates the radix economy for number representation as discussed in Chapter 1.5.3.

A multi-state RRAM development platform was developed in **Paper C** to replace the most expensive component of the memristive circuit, the Analog Discovery 2. The platform was demonstrated on a breadboard with good alignment to the LTspice simulation. Both autonomous operation of the platform via a microcontroller or operation via uMemristorToolbox are possible. This allows a smooth transition from memristor experiments to standalone memristor applications.

A dual voltage/current-source PCB implementation based on **Paper C** is currently being developed which features custom pulse trains using various shapes at high resolution intervals. Simulations of the PCB in LTspice show that 2-trit (9-states) state density should be feasible. Measurements with the Knowm development board and uMemristorToolbox shows close proximity to this amount. The work by Rao et al. [36] on denoising could potentially increase the density to even higher trits. Further research is needed to explore implementation of the multi-state memristor controller at the IC level. Work at the IC level has become increasingly affordable with the introduction of openPDK's such as the 130 nm openPDK of Skywater Foundry [232]. This PDK offers RRAM devices that are functional with a binary memristor controller for machine learning applications [239]. Open source PDK's enables exciting possibilities for low-cost, large volume multi-state memristor research.

—“The theory of switching circuits may be divided into two major divisions, analysis and synthesis. [...] finding a circuit satisfying certain given operating conditions, and in particular the best circuit is, in general, more difficult and more important from the practical standpoint.”

Claude E. Shannon [259]

—“All attempts to recreate some kind of ternary machine have been unsuccessful. [...] it appears that people having been drilled with binary logic can't think in ternary terms.”

Nikolai P. Brusentsov, inventor of Setun [27]

4

Mixed radix EDA for ternary computers

4.1 Introduction

Every computing paradigm needs its own Electronic Design Automation (EDA) tools and workflows to design, verify and fabricate with. A ternary computer is no exception. Designing a ternary computer, especially a balanced ternary one, requires different design principles and a different mind set. It requires EDA tools, user interfaces and workflows that facilitate *ternary thinking* rather than enforce patterns of the asymmetrical binary computing paradigm. To foster ternary thinking designers should engage with ternary logic using existing, mature CMOS technology as well as future devices that are inherently ternary. Special attention should be given to design and verification automation as designs become more information dense. To be successful, such a tool must support industry standard verification tools and ways to compare to binary. Few MVL and in particular ternary EDA tools and synthesis algorithms exist [97], [98]. No evidence was found that an EDA tool exists that is designed for large scale integration of ternary and mixed radix logic circuits.

In this chapter **Paper B**, **E** and **F** are discussed. In section 4.2 **Paper F** is discussed which introduces Mixed Radix Circuit Synthesizer (MRCS), the first open source and browser-based EDA tool to design and verify binary, ternary and hybrid (mixed radix) circuits. MRCS has been used in one master thesis [260] and several tape-outs. In section 4.3 MRCS's mixed radix synthesis engine and **Paper B** are discussed. In section 4.4 REBEL-2 is discussed, a RISC-V-like balanced ternary CPU with a novel ternary instruction set architecture (ISA). Finally in section 4.5 **Paper E** is discussed and shows how binary to ternary radix conversion and the inverse is done theoretically and with MRCS. These circuits allow existing binary chips to interface with novel ternary chips. Readers that are

interested in using MRCS to design and verify their own binary, ternary or mixed radix circuits are referred to **Appendices G.11-G.16**. There practical information on the inner workings, installation and limitations of MRCS can be found. Ternary designers can find monadic, diadic and triadic tables with important combinatorial and sequential building blocks as well as larger building blocks such as latches, flip-flops, adders, ROM/RAM and program counters.

4.2 MRCS: A new tool for mixed radix design and verification

4.2.1 Motivation

Mixed Radix Circuit Synthesizer (MRCS, pronounced / 'm ɜ:rsi/) was introduced at IEEE ISCAS 2022 in Austin, Texas, USA (**Paper F**). It is a open source graphical EDA tool to aid with design and verification activities of radix-2, radix-3 and mixed radix logic. An overview of the features can be found in Paper F and demonstration video [261].

The motivation to develop the tool grew out of frustration to design non-trivial ternary circuits. Most publications on ternary computing described small functional units like the full adder, flip-flop or 1-trit ALU. Ternary VLSI, circuits with thousands of transistors making up a multi-trit ALU, CPU or whole microcontrollers are rare. The initial design requirements for the open source tool can be summarized to three ideas:

- **Flexible user-centered workflows.** The target audience are ternary researchers and computer engineering students. The tool must have a graphical user interface as scripting has a steeper learning curve. It should be browser based such that no installation is required. Verification of desired logical behavior should be possible within the tool. The architecture must abstract logic design from circuit implementation such that users can try different synthesis methods and device technologies. Logic design must support hierarchical networks of truth tables which are arguably the simplest logic representation to teach and understand.
- **Industry-standard data format.** The tool must support data formats such as HSPICE netlist and verilog such that ternary designs can be verified with external tools.
- **Mixed radix.** The tool must support both binary, ternary and mixed radix circuits. Many signals are binary in nature such as a select signal of a 2:1 MUX or the binary clock signal in synchronous components. Encoding and processing these in ternary would require more complexity and transistors. It also makes comparing to binary easier as the same tool and synthesis engine is used.

4.2.2 Architecture

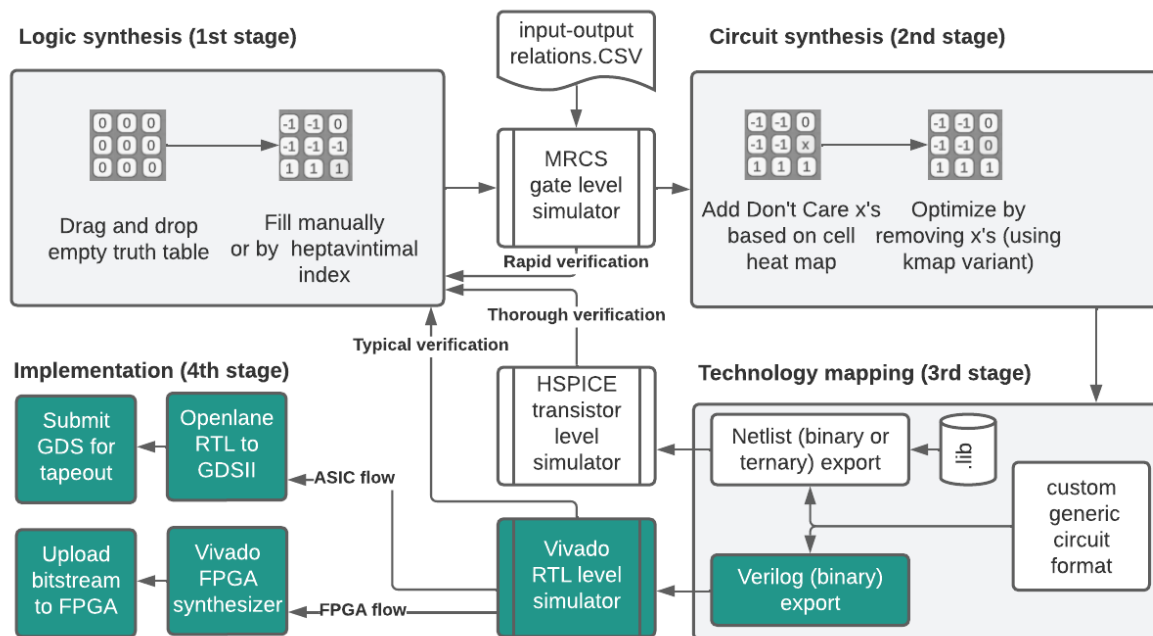


Figure 4.1: MRCS architecture and workflow. In green are features developed after **Paper F**

The architecture of MRCS can be seen in Fig. 4.1 with two new workflows (ASIC and FPGA) developed after **Paper F** in green. These are discussed in section 4.2.3. The four stages shown in Fig. 4.1 are structured around two core activities; design and verification. A typical chip development process would involve several design-verify iterations. The tool is made in Unity, a 3D game engine that support high level programming in C#, low level programming in C++ and web programming with WebGL and javascript. Unity supports multiple platforms with the same behavioral code base. About 98% of the MRCS code base is identical for the developer platform, standalone Windows platform and WebGL platform. The WebGL version runs in the browser while the standalone version is an executable that works offline. For developers a developer version is available which requires installing Unity version 2023.1 (or later) with WebGL plugin and Visual Studio (see Fig. 4.2). The standalone version is generally the fastest. Most code is written in C# with the synthesis engine made in C++ and C# (discussed in section 4.3). The tool is made open source with a permissive GPL 3.0 license to be used for commercial and patent use, but comes with no warranty and requires source changes to be disclosed. The Github repository can be found in [261]. The online viewer can be found at ternaryresearch.com [262].

4 Mixed radix EDA for ternary computers

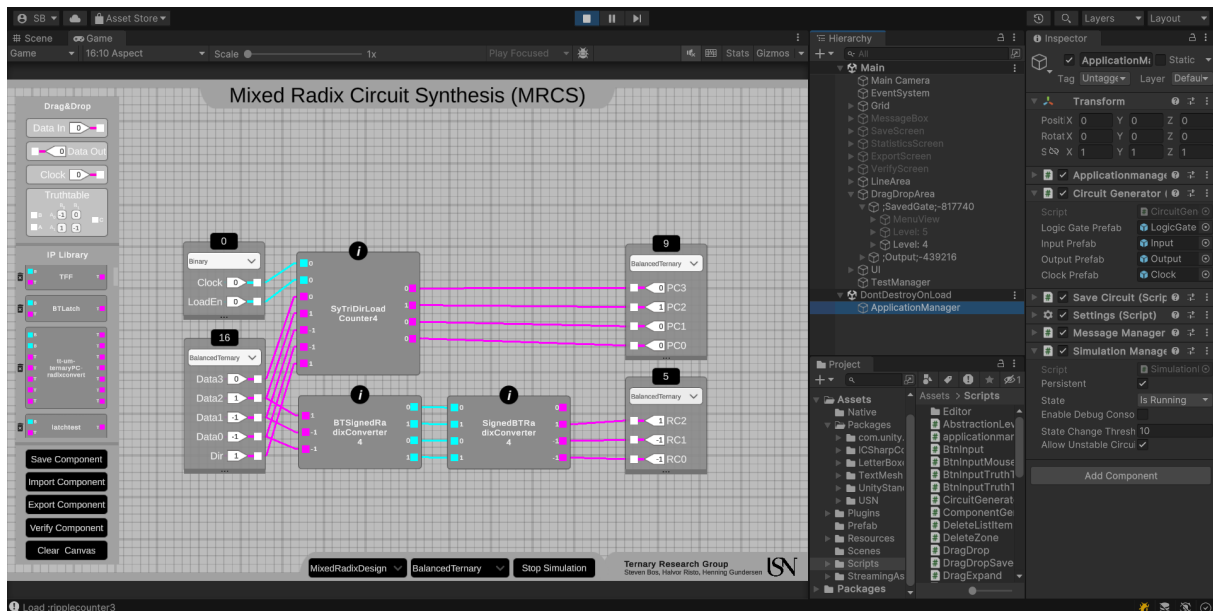


Figure 4.2: User interface of the developer version of MRCS

Design

In MRCS circuit design is abstracted from logic design. For example, the balanced ternary adder truth table described in literature from the 60's [169] uniquely captures all transitions for addition and subtraction. Many circuit implementations of this essential logic function exist [143]. The abstraction allows circuit designers to build on a history of ternary logic designs. It also makes designs such as adders future proof as better optimizations, synthesis algorithms, logic and memory devices are invented.

Verification

MRCS promotes a top-down hierarchical verification approach with a networks of truth tables. This approach is discussed in [263, p. 427] and allows each sub circuit to be quickly verified in isolation. Various tools are used in different scenarios:

- **Manual gate-level verification.** MRCS's gate-level, cycle-accurate simulator propagates manual input changes through the connected network of truth tables. It is the fastest type of verification but also the least realistic of the three verification options. The simulator updates the truth table cells background color to indicate usage ("heat map"). The active truth table cell is highlighted which allows tracing input to output. The CNTFET schematic view shows the active transistor path of a logic gate based on the inputs and greys-out transistors operating in the cut-off region. MRCS's simulator can work with binary and ternary signals in various encodings.
- **Automated gate-level verification.** MRCS's simulator can also be used with

automated input changes by uploading a .csv with input-output relations (the "Verify Component" button). This verification option is identical as the first option in terms of speed and realism but simplifies verification of larger circuits. Any input-output mismatch compared to the provided groundtruth is logged and reported. The same feature can also be used to program a Read-Only-Memory (ROM) component with assembly instructions in machine code.

- **Analog and mixed-signal (AMS) verification.** Synopsys's HSPICE simulator is the industry standard verification tool and gives both the slowest and most realistic verification. The generated main.sp netlist requires manual input patterns and simulation settings such as the type of simulation and duration before running a simulation. The resulting .tr0 file can be inspected with Synopsys's Custom WaveView. HSPICE can work with binary and ternary signals.
- **RTL verification.** AMD's Vivado suite is another industry standard verification tool and gives a balance between speed and realism. Vivado cannot work with ternary signals which limits designs to use binary coded ternary (BCT) signals. The generated verilog can run on the RTL simulator which allows behavioral, post-synthesis and post-implementation simulations in increasing realism.

4.2.3 Workflows

MRCS has been designed and verified for three mixed radix workflows: HSPICE, ASIC and FPGA (see Fig. 4.1). The design phase is identical for all three flows and regardless of the used radix and briefly explained in the HSPICE flow. When binary hardware is targeted such as FPGA's or CMOS-based ASIC then any ternary signal is automatically converted to BCT signals.

HSPICE flow

The first stage of a design-verify cycle starts with dragging truth tables and/or reusable components on the workspace, wire them up and label the input and outputs to give them meaning (see Fig. G.7 in **Appendix G.11**). The MRCS gate-level, cycle-accurate simulator is running by default such that a traceable flow from input to output is visible. The desired functionality can be achieved by modifying truth table cells. In the second stage, unused truth table cell (colored white) can be marked with a *don't care*-symbol x . The markings are used by the synthesis engine to reduce the transistor count for that logic gate. In the third stage designs are saved as reusable components. Hardware mappings are generated of the new design and possible references to earlier designed or imported reusable components are integrated. Several design rules are checked (DRC) such as all inputs/outputs being wired-up and the component having a valid name. The generated netlist and verilog files are input for HSPICE and Register-Transfer-Level (RTL) tools. The HSPICE flow uses multi-threshold CNTFET devices based on the Stanford CNTFET model [264]. HSPICE netlist generation algorithm is discussed in section 4.3.3.

ASIC flow

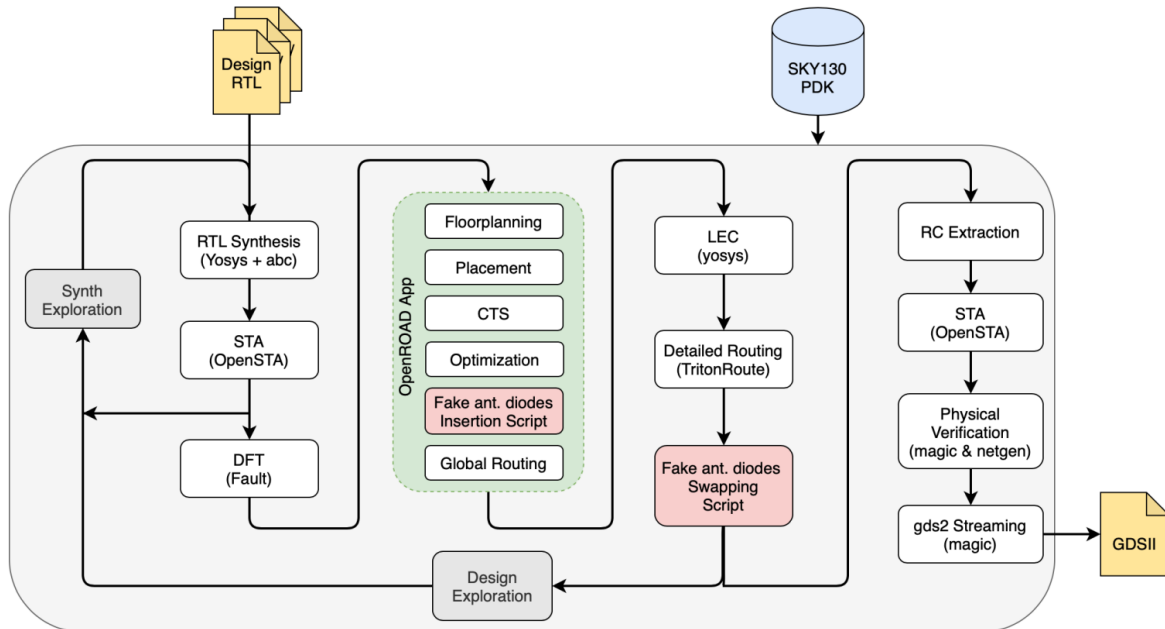


Figure 4.3: The RTL-to-GDS ASIC flow from OpenLane [101].

The RTL-to-GDS flow by OpenLane [101] (see Fig. 4.3) transforms MRCS’s verilog into GDSII, the industry standard database format accepted by foundries. The OpenLane flow automates all the relevant activities such as floorplanning, clock tree synthesis (CTS), static timing analysis (STA) and routing. As input RTL and a supported open PDK such as the Skywater 130nm OpenPDK or the GlobalFoundries 180nm is expected. The OpenLane flow can be further automated with the tapeout service TinyTapeout [265]. By uploading the verilog design to a cloned github repository the entire openlane flow is executed using Github actions. The result is integrated in a large multi-project-wafer (MPW). The costs for a physical chip mounted on a PCB breakout are (\approx \$100) for $160 \times 100 \mu\text{m}$. It is also possible to purchase an entire wafer for \approx \$10,000 with eFables.

FPGA flow

Popular FPGA’s such as the Digilent Basys-3 are common in academia. With the free standard edition of AMD’s Vivado ML 2022.2+ suite MRCS’s verilog files can be transformed to bitstreams. The bitstreams can be uploaded to FPGA’s. This RTL-to-Bitstream FPGA flow has various intermediate steps to verify correct behavior incl. post-synthesis and post-layout simulations. The MRCS github repository contains various examples that include a verilog testbench and constraint file (XDC) for the Basys-3.

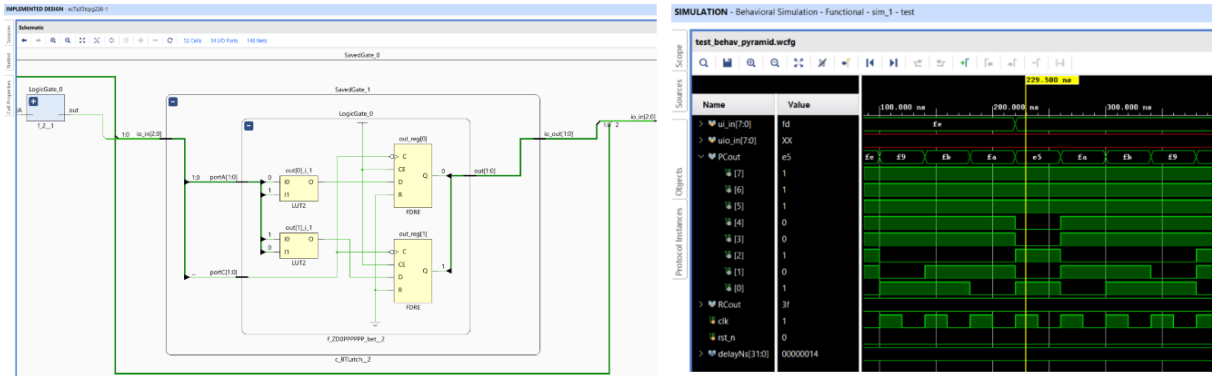


Figure 4.4: Schematic and simulation excerpt of a BCT counter in Vivado as part of the RTL-To-Bitstream flow. **Left.** Schematic of the ternary d-latch from **Paper F** that is synthesized as 2 registers with BCT encoding. **Right.** The simulator shows a count up and down pyramid pattern of the tri-directional loadable ternary counter from [266].

4.3 Mixed radix synthesis engine

4.3.1 Introduction

Logic synthesis is a set of procedures to convert a description of a finite-state machine (FSM) to an optimized circuit implementation with discrete switching devices [96, p. 62]. The procedures are algebraic operations while the description is a data structure or representation of the logic function. Logic synthesis consists of two phases: the optimization or algebraic phase (see **Appendix G.13**) and the technology mapping phase. In the optimization phase the potentially huge logic representation needs to be optimized with some objective such as minimizing PPA or improving noise immunity, routability, etc. while keeping functional equivalence/satisfiability (SAT). In the technology mapping phase the optimized (network of) logic functions is mapped to a small selection of logic gates called standard cells. Standard cells provide an abstraction between the logic domain and the physical domain. By using only a small set of reusable standard cells layout engineers can hyper optimize these cells for a variety of PPA and process, voltage, temperature variation (PVT) requirements. The final result is a gate-level netlist that can be verified with simulation and converted into a circuit layout by placing and routing the found network of standard cells.

Logic synthesis is an active research field [139], [267]. Synthesis methods need to adapt to the challenges Moore's law bring. For example Brayton [263, p. 413] mentioned that communication and interconnects are becoming such a problem that perhaps synthesis methods should reverse their approach: make a rough interconnect infrastructure first and do logic synthesis second. In the same paper many ideas were proposed to improve synthesis in the future. MVL synthesis was positioned as a prime candidate [263, p. 404].

Historical perspectives and literature on MVL synthesis until 2001 can be found in [99]. Importantly, MVL devices were not required to improve synthesis and regular CMOS would suffice as mentioned by Dubrova [99]:

”Conceptually, the best way to benefit from multiple-valued logic is to describe a system using some multiple-valued representation and to manipulate such a representation directly. When no further optimization is possible in purely multiple-valued form, a suitable encoding can be applied to transform the multiple-valued representation into a Boolean one. However, this approach is not widely used mainly due to the lack of mature packages for representation and manipulation of multiple-valued functions, as well as for multiple-valued synthesis, optimization and verification.”

Dubrova mentions that the optimized MVL representation can (optionally) be transformed to Boolean encodings. The need for Boolean transformation depends on the second synthesis phase, the hardware mapping. If only Boolean switching devices are available, such as in CMOS and multi-threshold CNTFET, then the first synthesis phase must convert MVL input to Boolean output. Regardless of the usage of Boolean switching devices, the output signal can still be either three amplitude levels (ternary signalling) or BCT using two amplitude levels (binary signalling). For example a single bi-stable device can switch between $\frac{V_{DD}}{2}$ and GND or V_{DD} and GND. Another example is the usage of two bi-stable devices to make a voltage divider.

4.3.2 Related work

There is limited literature on ternary synthesis. The 1963 paper by Yoeli and Rosenfeld [168] is important as it provides a full theoretical treatment of logic representation, chain-based Post algebra (see **Appendix G.13**) and optimization with the Quine-McCluskey’s method extended to the ternary case. In [142] several ternary logic minimization methods including Quine-McCluskey’s method are discussed. Historically the Berkeley logic synthesis group has been leading in practical MVL synthesis tooling [268]. They developed the two-level synthesis tool ESPRESSO-MV for MVL [269]. Two level synthesis uses MIN blocks at the first level and MAX blocks at the second. They also developed MIS-MV [270] and MVSIS [271], a multi-level synthesis tools for MVL. It is interesting to note that the research group pivoted development to a binary-only tool called ABC [272]. ABC is perhaps the most popular open source synthesis tool and is used in tool-chains such as Openlane. The ABC paper by Brayton and Mishchenko [273] discussed the transition from MIS->SIS->MVSIS->ABC. They mention two reasons: the discovery of And-Inverter Graphs (AIG) as compact logic representation with useful synthesis properties and Boolean satisfiability (SAT) for logic verification of binary circuits. The second reason was more practical in nature as *”fast AIG-based optimizations could be run with*

different parameter settings, resulting in the discovery of better solutions while costing much less memory and runtime.”

Recent ternary synthesis literature often use multi-threshold CNTFET as the device of choice [155], [274]. According to the survey paper by Nemati et al. [143] nearly $\frac{2}{3}$ of the 125 papers use them, with CMOS and memristors trailing second and third with 15% and <7%. The algebra is often not discussed (for example [155], [275], [276]). Very often a variant of the chain-based Post algebra is implicitly used with some set of the Post-literals like the *cycle/mod₃* operator.

A multitude of synthesized circuit architectures are discussed in [126] that are still relevant. Unfortunately many architectures are mislabelled after rediscovery decades later. For example, in [143], [275] T-gate architectures are not discussed but ternary MUX architectures are despite being identical for the 3:1 case. Another example are quasi-binary MVL architectures [126], [277] which use positive and negative synthesizing networks. For ternary these are pull-Up and pull-Down networks for full-swing and half-swing such as in [155]. Four network to encode 3 states is not a very efficient coding for ternary. Since the mixed radix synthesis algorithm discussed in paper B also uses CNTFET and a quasi-binary MVL architecture a brief timeline is needed to trace the contributions.

- **2005:** Multi-threshold CNTFET for ternary are first reported [30].
- **2007:** Improved and open sourced SPICE model [264].
- **2009-2011:** Improved circuits by replacing resistors with CNTFET [274], [278].
- **2018-2021:** Rediscovery Quine-McCluskey optimization for ternary [279] (earlier [142], [165], [168]) and usage of quasi-binary MVL architecture with multi-threshold CNTFET switching table [155], [280].
- **2020-2022: Papers B and F.** Reimplementation of [280] and open sourced. Generation of HSPICE simulatable netlists and BCT verilog. Synthesis of binary, ternary and mixed radix circuits with multi-threshold CNTFET and CMOS devices. Integration in a no-installation graphical environment supporting three design and verification workflows.

4.3.3 Mixed radix synthesis algorithm

The mixed radix synthesis algorithm with single voltage source (0.9 V) and multi-threshold CNTFET described in paper B is based on Kim et al. [155], [280]. The work is unfortunately not open sourced, though an undocumented executable is available. The re-implemented algorithm shown in Fig. 4.5 has several similarities and differences compared to Figure 7 in [280]). The method of splitting a truth table into a quasi-binary

4 Mixed radix EDA for ternary computers

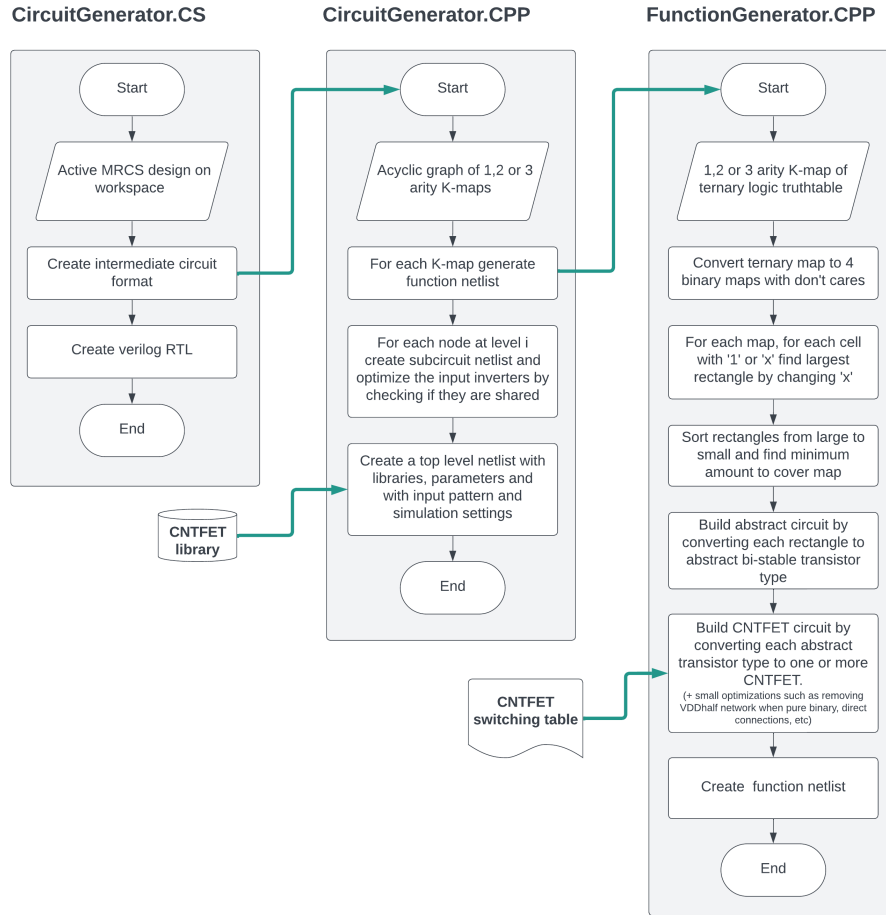


Figure 4.5: The mixed radix synthesis algorithm from **Paper B** as implemented in MRCS

MVL architecture with four networks (Figure 6 in [280]) is identical. The same switching table and placement of the diodes is used (Figure 3 and 4 in [280]) is used. Large differences are that the SoP representation and Quine-McCluskey method are not used. Rather, the optimization step uses a Karnaugh-map [281] brute-force covering method **Paper B** and [282]. The solution space is limited to 3-input logic functions making a brute-force approach viable. Several small optimizations are also done such as removing the unused half-swing networks if pure binary functions are detected. The synthesis algorithm is extended to form hierarchical networks of truth tables. This enables the creation of non-trivial circuits composed of multiple truth-tables such as CPU's (see section 4.4). Lastly, the synthesis algorithm generates netlists and verilog for direct verification in HSPICE, Vivado and deployment on FPGA and ASIC. The proposed algorithm significantly reduces the time from design to verification and allows various practical workflows described earlier. The effects of synthesis algorithm is shown in Fig. 4.6 where a truth table is transformed into a circuit implementation with multi-threshold CNTFET.

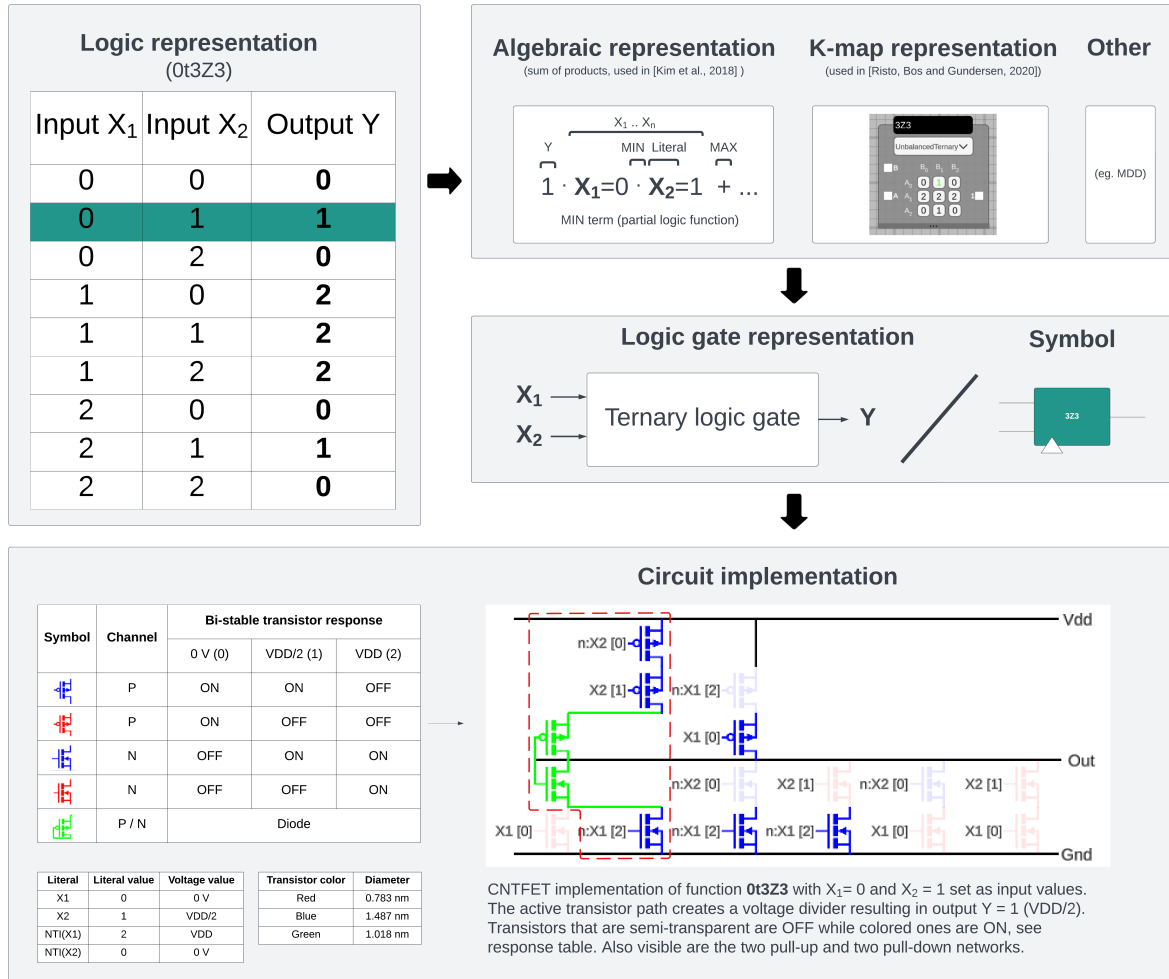


Figure 4.6: Logic transformation from truth table to CNTFET implementation using MRCS’s synthesis algorithm described in Fig. 4.5 and **Paper B**

4.3.4 Binary coded ternary RTL

Ternary synthesis with ternary signals using CNTFET has not been demonstrated outside simulation. Ternary synthesis with BCT signals on the other hand is very feasible and is demonstrated to occasionally outperform binary [188], [189]. BCT allows experimenting with ternary logic design and verification with modern CMOS. Ternary logic with CMOS has been proven feasible as early as 1974 by Mouftah and Jordan [174]. In that paper a ternary And-Or-Invert (AOI) 6T2R CMOS circuit was presented. Replacing the 2R with 2T would make the 8T circuit competitive in transistor count to a standard 6T binary AOI CMOS circuit. The 5 functions the ternary AOI can perform are STI, PTI, NTI, NMAX and NMIN and would make a great standard cell.

4 Mixed radix EDA for ternary computers

In literature the term BCT is sometimes called *bit pairing* [263, p. 411] or *redundant binary representation* (RBR). Two bit for one trit implementations result in one degree of freedom. This has resulted in non-standard encodings for both unbalanced and balanced ternary (see Table 4.1).

Table 4.1: Variants of binary coded balanced ternary

Bit 1	Bit 0	BSD-SV	BSD-SUM	BSD-PN	BSD-PNX
0	0	0	-1	0	x (illegal)
0	1	1	0	-1	-1
1	0	0	0	1	1
1	1	-1	1	0	0

Following the naming scheme used in [188], Binary Signed Digit Positive-Negative-Exclusive (BSD-PNX) is proposed with 0b00 as illegal state for five reasons:

- Only one bit needs to change regardless of transition, reducing power consumption.
- The fourth state is an illegal/faulty/uninitialized state. Using it as a redundant logic state would invalidate that only one bit needs to change. It also removes ambiguity.
- With BSD-PN(X) encoding the heavily used standard ternary inverter (STI) becomes a simple cross wiring binary implementation, possibly with a buffer cell (see Fig. 4.7).
- Implementation with a single differential opamp allows easy conversion to ternary signals.
- Literature [188] seems to indicate that BSD-PN coding has arithmetic advantages resulting in reduced power consumption and increased performance.

The BCT verilog syntax is rather verbose as no ternary constructs can be used. Brayton and Khatri [283] created a verilog extension for MVL that might improve this. A recent specification of TernaryVerilog was also found, but the status is unknown and the source is not in the public domain [284]. Without ternary-oriented verilog BCT truth tables explode in size, especially for 3-ary truth tables such as the 2:1 ternary MUX with index *0tZD0DDDPPP*. The process of writing BCT verilog in MRCS is automated and is visible in Fig. 4.7. Special attention should be paid to the limitations written in **Appendix G.12** regarding the binary or BCT verilog conversion of sequential logic such as latches.

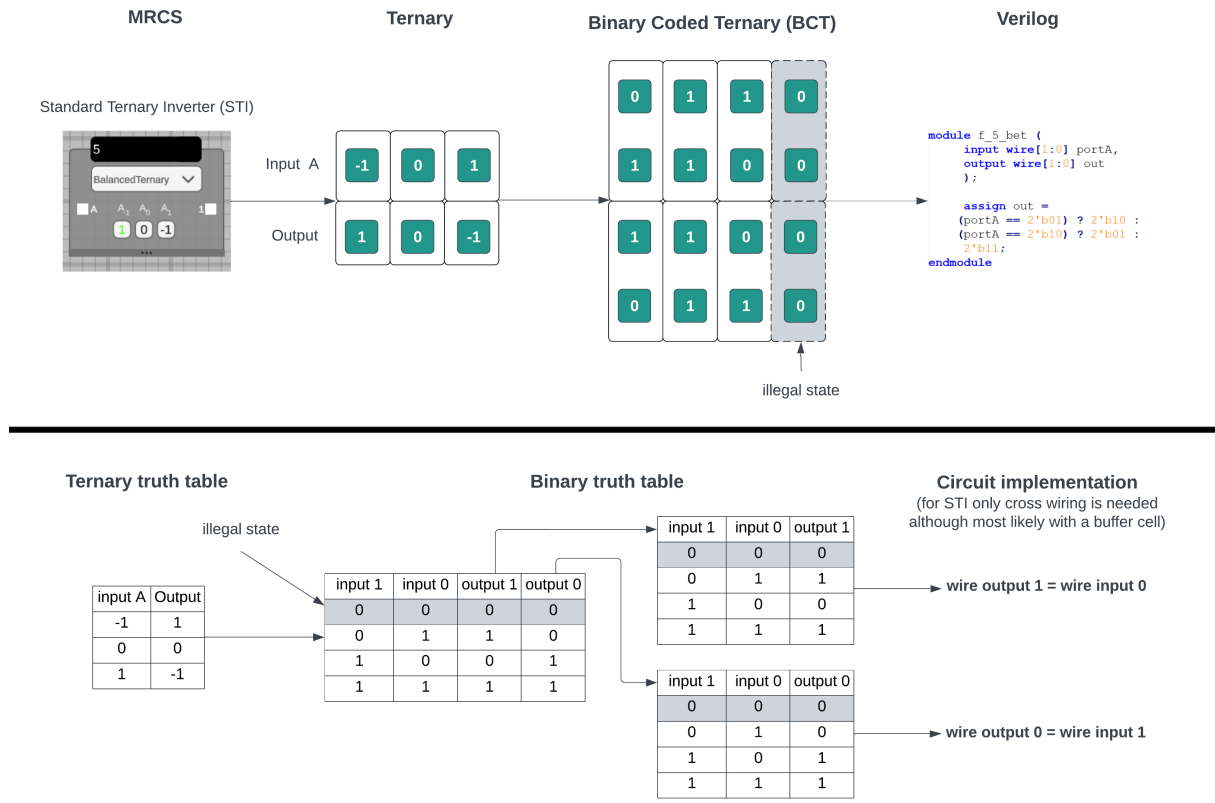


Figure 4.7: Binary Coded Ternary (BCT) implementation of a standard ternary inverter (STI) in MRCS. For unbalanced ternary just add 1 to the balanced ternary numbers. **Top.** From MRCS component to MRCS generated verilog. **Bottom.** From ternary truth table to two binary truth tables that can be implemented with CMOS

4.4 REBEL-2 Balanced Ternary CPU

In this section REBEL-2 is presented, a novel 2-trit balanced ternary CPU made with MRCS (see Fig. 8. REBEL-2 is an acronym for RISC-V-like Energy efficient Balanced tErnary Logic CPU. The postfix denotes the 2-trit wide memory address bus. A new ternary computer should inherit the same spirit as Brousentsov when he designed the last ternary computer in 1970, the improved Setun [28]. Brousentsov's ternary principle states that a ternary computer must have [197]:

- Ternary logic
- Ternary memory
- Balanced ternary encoding

With MRCS a ternary computer can be designed that implements these principles using the building blocks shown in **Appendix G.14 and G.15**. The designer can be oblivious to the hardware implementation and use only ternary logic gates, memory and pins.

4.4.1 Motivation

A binary CNTFET CPU has been made as a proof of concept by Shulaker et al. [285] in 2013 and at a commercial foundry by Hills et al. [29] in 2019. The discovery of multi-threshold CNTFET for ternary logic was made by Raychowdhury and Roy [30] in 2005. They tuned V_{th} by changing the diameter of the nanotube while Wang et al. [286] in 2013 achieved the same effect with doping. No literature was found demonstrating a ternary CNTFET CPU outside simulations. Related work like Kam et al. (2022) [187] discuss a novel balanced ternary CPU design with ternary assembly. Their instruction set architecture (ISA) occasionally generates larger assemblies (for instance the BLT instruction expands to 3 instructions), use only two operands and does not fully exploit the benefits balanced ternary offers to reduce the instruction set even further (such as three-way branching). The paper demonstrated great benefits in various benchmarks compared to binary RISC-V designs. More recently Gadgil et al. demonstrated a ternary CNTFET CPU but with unbalanced ternary encoding [287].

The REBEL-2 is the first modern ternary computer designed with the unique qualities of radix-3 in minds. It is a Harvard-style CPU with instructions in ROM and data in RAM. Compared to von Neumann-style CPU's Harvard-style design requires more area as instructions and data are not stored in the same physical memory. The benefit is that both ROM and RAM can be accessed in the same clock cycle. Arguably, the design is simpler despite additional bus interconnects and control logic. The small memory address bus and simplicity-oriented design make REBEL-2 more suitable for educational purposes than real-world applications. The architecture is heavily inspired by the book "Computer Organization and Design - RISC-V edition" by Patterson & Hennessy [2]. Reduced Instruction Set Computer (RISC) is a computer architecture that in essence shifts complexity from hardware to software allowing computer designers to optimize hardware at the cost of writing more software. Through software abstraction layers this burden is mostly hidden from the programmer. RISC uses elementary instructions while Complex Instruction Set Computer (CISC) uses higher level instructions that are decoded into simpler instructions on the hardware. Examples of RISC architectures are ARM, Atmel AVR, MIPS, SPARC, PowerPC, RISC-V. The traditional PC market uses x86, x86-64 CISC architectures for server and workstations. According to Patterson [288] CISC products have been reduced to a one percent market share while RISC make up the rest.

The REBEL-2 architecture targets low-power applications as this property is often associated with RISC architectures. The high level requirements were:

- RISC-V-like implementation based on RV32I
- Native ternary instructions (such as ADD/SUB adder, 3-way branching and compare)
- 2-trit registers to be non-trivial gates
- No external memory with Harvard style separation of program and data
- Single cycle instructions
- Reading at rising edge, writing at falling edge
- Compact, uniform instruction format

4.4.2 Balanced Ternary Instruction Set Architecture

Central in the design of a CPU is the organisation of the instructions, the instruction set architecture (ISA). This critical abstraction layer is the contract that allow chip design engineers to focus on the hardware and software engineers on the software. The ISA dictates the functionality of a CPU, with each instruction being implemented with one or more electrical data paths. The implementation of these paths determines the actual functionality and performance. Traditionally instruction sets have had little innovation and were considered a stable contract between the software and hardware domain. Few companies developed ISA's (examples include Intel, AMD, ARM). The ISA's were licensed and often non-customizable. This changed with the arrival of RISC-V in 2015 which allowed customization of the ISA and usage royalty free. Ternary computers require new ISA's or extensions to existing ISA's since their capabilities are not well captured in the currently available sets. For example, instruction formats can be more dense with 3-valued fields and more simplified without unsigned instructions. Smaller ISA's lead to simpler CPU designs. REBEL-2 has just nine instructions. Scaling this architecture to a wider memory address bus, such as a tryte ¹ is possible. This would necessitate a redesign of the ISA since opcodes and function fields require much less scaling compared to the operand fields.

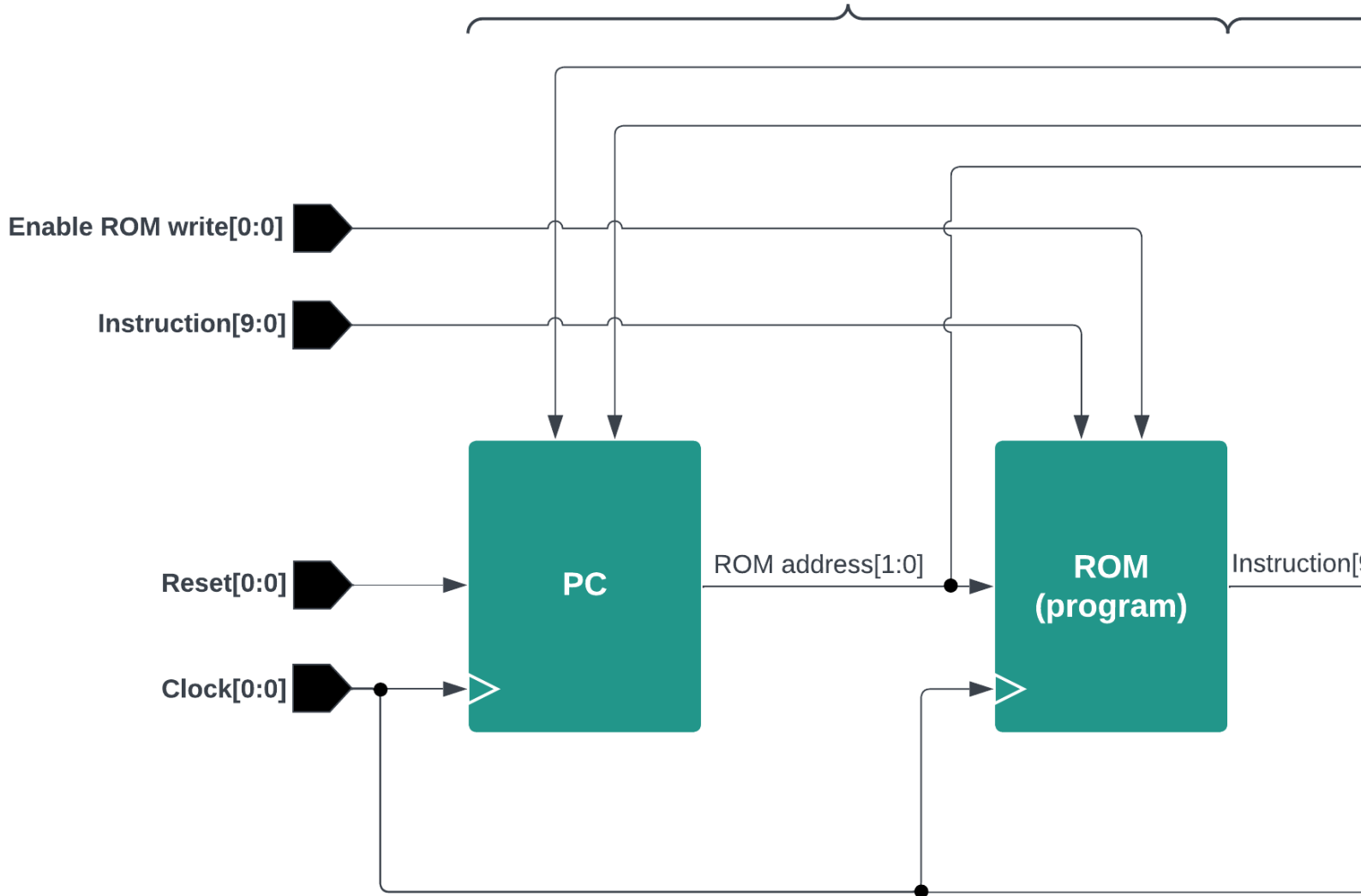
The memory address bus is 2-trit wide, resulting in the limited addressing of only nine 2-trit registers. Each register has 2-trit storage capacity. The most-significant-trit (MST) is the left-most trit. To fully use the nine addresses 10-trit wide instructions are used with four 2-trit operands such that most of the 40 RV32I instructions can be represented in the ISA. The opcodes are also 2-trit fields. Several instructions that do not use the

¹There is currently no standard as to how large a tryte, the equivalent of a byte, should be in ternary. A 5-trit tryte would have slightly less resolution than the 8-bit byte (243 vs 256 states) but the a 6-trit tryte would have almost 3 times more (729 vs 256) states. A 9-trit tryte is the closest to the 8-bit byte for trit-wise operations such as bit-masking.

REBEL-2 Balanced Te

4 operands, 1 instruction format, 2-trit memory bus, 1

Fetch



MST (most significant trit)

Instruction format

Mnemonic	Opcode ₃ [1:0]	Rs1[1:0]	Rs2[1:0]	Rd1[1:0]	Rd2[1:0]/Func	Description	Subs
ADD	--	A	B	Y	(x0)	if rd2=x0 then R[rd1]=R[rs1]+R[rs2] else R[rd1]=R[rs1]+R[rs2]	S
ADDi	-0	A	imm	Y	imm	R[rd1 + imm]=R[rs1] + imm	SUP NOP
ADDi2	-+	imm	imm	Y	Z	R[rd1]= imm and R[rd2]= imm	
MUDI	0-	A	B	Y	(x0)	if rd2=x0 then R[rd1]=R[rs1] * R[rs2] else R[rd1]=R[rs1] / R[rs2]	M
MIMA	00	A	B	Y	(fs) f=function, s=scope	case rd2=-- : R[rd1]=MIN _{word-wise} (R[rs1],R[rs2]) rd2=-0 : R[rd1]=MIN _{trit-wise} (R[rs1],R[rs2]) rd2=+- : R[rd1]=MAX _{word-wise} (R[rs1],R[rs2]) rd2=+0 : R[rd1]=MAX _{trit-wise} (R[rs1],R[rs2])	A
SHi	0+	A	pp p=position	Y	dt d=direction, t=inserted trit	R[rd1]=SHIFT(R[rs1], pp , d , t) d : - is left, 0 is cyclic, + is right pp : 0+ is 1 position, pos +- is 2 positions	SLLi
COMP	+-	A	B	Y	(x0)	if rd2=x0 then R[rd1]=COMP _{word-wise} (R[rs1],R[rs1]) else COMP _{trit-wise} (R[rs1],R[rs1])	
BCEG	+0	A	B	Y (x0)	Z (x0)	case R[rs1]=R[rs2] goto R[rd1] R[rs1]>R[rs2] goto R[rd2] R[rs1]<R[rs2] no jump if rdi = x0 then goto R[rdi] is disabled	BGE
PCO	++	A	imm	Y	ff	case rd2=0- : R[rd1] = PC + imm no jump rd2=00: R[rd1] = PC + 1 goto R[rs1 + imm] rd2=0+: R[rd1] = PC + 1 goto PC + imm	JAL,

ternary CPU architecture

Harvard RISC-V-like Instruction Set Architecture (ISA)

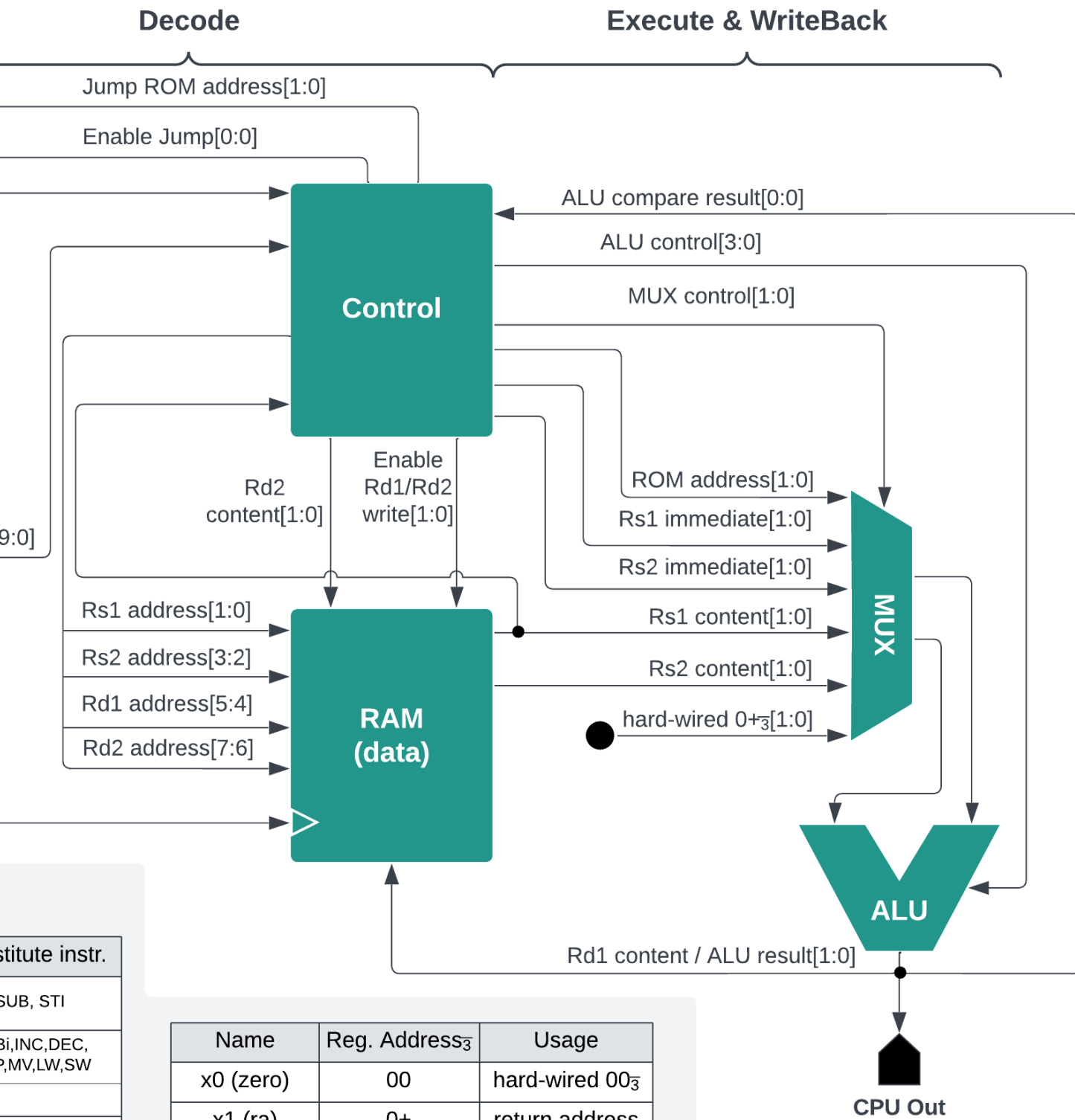


Figure 4.8: RISC-V-like Energy efficient Balanced Ternary Logic 2-trit memory bus (REBEL-2) CPU with novel ternary ISA

- stitute instr.
- SUB, STI
- Bi, INC, DEC, P, MV, LW, SW
- MUL, DIV
- AND, OR
- SRLi, SRAi, SLAi
- SLT
- E, BEQ, BLT
- JALR, AUIPC

Implementation notes:

- x0 is used in BCEG to mask (ignore) a compare condition and cannot be used to jump to address 00₃. In JAL x0 can be used as Rs1 to jump to address 00₃ (=reset PC). With mask BCEG becomes a classic 2-way branch.
- Bring-up instructions: First initialize PC to 00₃: set Reset=HIGH, do full clock cycle and then Reset=LOW. Next, set ROM_write = HIGH and provide a ROM instruction every full clock cycle. After a maximum of nine clock cycles set ROW_write = LOW. The CPU is now ready to execute the program at the next clock cycle. During ROM programming ROM output = LOW or tri-stated.
- Read operations occur at falling-edge, write operations at rising-edge. All instructions are single cycle.
- Two 2-trit adders are hidden. One is in the PC block to compute the next instruction address with +1. The other is in the Control block to compute Rd1 address, Rd2 content or jump ROM address for ADDi, ADDi2 and PCO instructions
- Unsigned RV32i instructions are irrelevant with balanced ternary and are omitted. Half word RV32i instructions are also irrelevant with the full word instruction format and are omitted.
- The usage of x1 for return address purposes is recommended and not mandatory.
- The ISA is not RV32i compatible as BNE, SRA, SLA, SLL, SRL, SLTi, XOR, XORi, Ori, ANDi and ECALL/EBREAK are missing. These can be implemented with multiple instructions except for ECALL/BREAK.
- This simple balanced ternary ISA is designed for educational purposes. No external memory is required.
- The PCO (program counter operation) instructions use the hardwired 0+₃

”Register Destination 2” (Rd2) field have an additional 2-trit function field to allow more functionality. Example functionality include selecting the scope between trit-wise (1-trit) or word-wide (2-trit) operation for some logic functions such as COMP. Novel functionality includes two store operations in one cycle and selection of shift left, right or cycling operation. The 32-bit RV32I base instruction set has 40 instructions [289, p. 130] of which 38 are essential [289, p. 13]. The REBEL-2 ISA has full word load and store operations such that instructions like load half word, load byte or load upper immediate can be ignored. Unsigned instructions can be ignored as only signed instruction can be optimally used with balanced ternary. Branch instruction like BEQ and BLT can be compressed to just one instruction using a ternary comparator and delegating operands ordering to the compiler. Using the function field several instructions can be merged such as AUIPC/JAL/JALR, SLLi/SRLi/SRAi/SLAi and ADD/SUB. In total 11/38 RV32i instructions are missing from the REBEL-2 ISA which are BNE, SRA, SLA, SLL, SRL, SLTi, XOR, XORi, ORi, ANDi and ECALL/EBREAK. The majority of these are branch, shift and logic instructions which can be resolved with 2 or more instructions. The only instruction that cannot be resolved is ECALL/EBREAK to transfer control to a debugger/Operating System from an interrupt. All missing instructions could technically be fitted in unused function fields but it would obfuscate the ISA tremendously. For example, the MIMA instruction could be merged with the MUDI and XOR instruction since the format is identical and the 2-trit operand plus 2-trit functional field are large enough.

4.4.3 Implementation

Validation and performance benchmarks with comparison to binary and tapeout of the whole CPU is planned. Several key building blocks of the REBEL-2 CPU have been simulated, tested on FPGA and submitted for tape-out including the multi-trit adder, multiplier, multiplexer, RAM/ROM and program counter. They can be found in **Appendix G.15**.

The 2-trit balanced ternary ALU has eight 2-ary functions: ADD/SUB, STI, COMP, MIN, MAX, SHIFT, MULTIPLY and DIVIDE. The ninth function could be the ternary XOR function [290] such that the entire RV32i set is covered. Most of the logic functions operate on whole words but some (such as COMP) allow trit-wise operation. Demonstration video’s of a 1-trit balanced ternary ALU with 5 functions and a 2-trit balanced ternary calculator [291] can be found in [292], [293].

4.5 Radix conversion

In Chapter 1.3 the history of binary computers was briefly reviewed. Computers such as the ENIAC, UNIVAC I and IBM 650 were radix-10. At the implementation level radix-10 numbers were actually made with bi-stable devices and BCD (or similar) encodings [12]. Switching to radix-2 at the architecture level meant costly radix conversion to allow human readable input and output (I/O). It is interesting to note that even today, BCD arithmetic is present in the Intel 64 and IA-32 ISA [294]. Example instructions are *Load Binary Coded Decimal* (FBLD), *Store BCD Integer and Pop* (FBSTP) and *ASCII Adjust After Addition* (AAA). As these instructions are rarely used, the AMD 64 ISA considers these instructions legacy and are not available when operating in 64-bit mode. The non-hardware accelerated radix conversion method by Samet [295] from 1971 is still relevant for general software radix conversion. Online software radix conversion with a custom developed tool is discussed **Appendix G.17**.

All modern digital electronics use binary logic. Ignoring billions of existing chips when pivoting from binary to ternary would result in an unprecedented amount of e-waste and is against UN Sustainable Development Goal 12: Responsible production and consumption. Rather, the chiplet approach discussed in Chapter 1 should be considered with either dedicated radix conversion chips or integrated conversion circuitry in ternary chips. In this section the results from **Paper E** on radix conversion are discussed which are made with MRCS from **Paper F**. Shown is how signed binary to balanced ternary conversion and the inverse is possible from theory to implementation with CNTFET and CMOS. The BCT to unary coded ternary and inverse radix converter chip is detailed in [296]). The signed binary to balanced ternary radix converter and inverse radix converter chip is discussed in **Paper E** and open sourced in [266].

4.5.1 Binary to Ternary

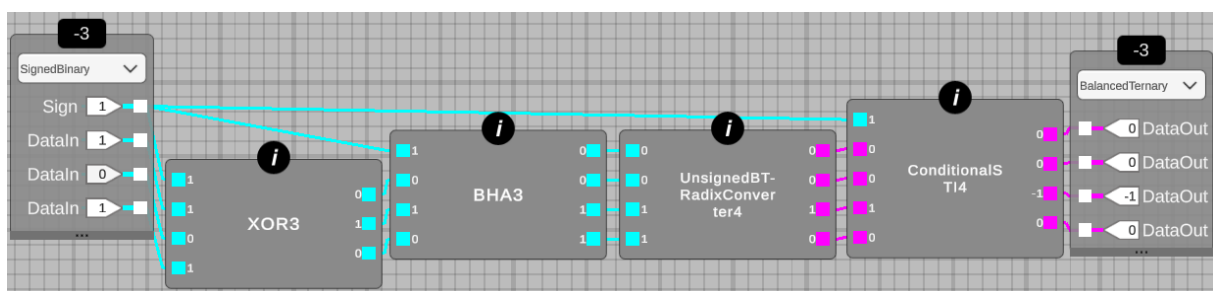


Figure 4.9: A 382T 4-bit 2's complement signed binary to 4-trit balanced ternary radix converter. This design reuses the 170T proposed design from **Paper E**. Components can be found in Appendix G.16

There is surprisingly little written on binary to ternary radix conversion in hardware (see a small survey in [282]). In particular radix conversion with signed/balanced arithmetic is rare. In **Paper E** the method by Li et al. [297] was extended to 15 bits and 10 trits. In addition the inverse matrices, balanced ternary to 2's complement signed binary, were . Our inverse matrices are also extendable and have a remarkable property: generating output in 2's complement regardless of the bit width comes straight from the matrix. The original paper by Li uses an implementation with SQUID gates for an unsigned binary to balanced ternary converter. In **Paper E** an implementation is given with CNTFET gates generated by MRCS for the same converter. Two ways were presented to implement the conversion matrices for unsigned binary to balanced ternary. The trivial approach is to replace each term with a balanced ternary full adder (BTA). The other approach optimizes each BTA to the essential circuitry, for example when a carry signal is not needed. The full procedure is described in **Paper E**. The proposed 4-bit unsigned to 4-trit balanced ternary radix converter (Fig. G.31 in **Appendix G.16** had a PDP (worst measured delay * average power consumption) of 7.824e-16 joule. The state-of-the-art for 4-bit unbalanced binary to unbalanced ternary using CNTFET [298] had a PDP of 3.024e-15 joule, an improvement of 74%. Both implementations have no capacitive load connected. The result is achieved despite it being an unfair comparison. Balanced ternary needs additional transistors for positive and negative carry signals while unbalanced ternary only needs a positive carry signal. For feature parity unbalanced ternary needs to add additional circuitry to implement 3's complement to handle negative numbers. **Paper E** reports that 176T are needed but the latest version of the synthesis engine in MRCS optimized it to 170T.

Paper E also discusses reusing the the proposed converter to make a 2's complement signed binary to balanced ternary converter by adding additional circuitry and is shown in Fig. 4.9. The implementation of the additional circuits in Fig. 4.9 can be found in **Appendix G.16**. An improved version of this radix converter would use a smaller unsigned binary to balanced ternary converter since the binary input range of -8 to +7 can be achieved with 3 trits instead of 4. Three trits have a range of -13 to +13. The overhead mentioned in **Appendix G.5** is large as even with 3 trits half of the states are unused.

4.5.2 Ternary to Binary

In **Paper E** the inverse matrices are also provided but an implementation was not included. A novel 139T 3-trit balanced ternary to 4-bit signed binary radix converter implementation made with MRCS is shown in Fig. G.32 of **Appendix G.16**. The method to construct this implementation is similar to the converter shown in Fig. 4.9 except now table 5 from **Paper E** is used.

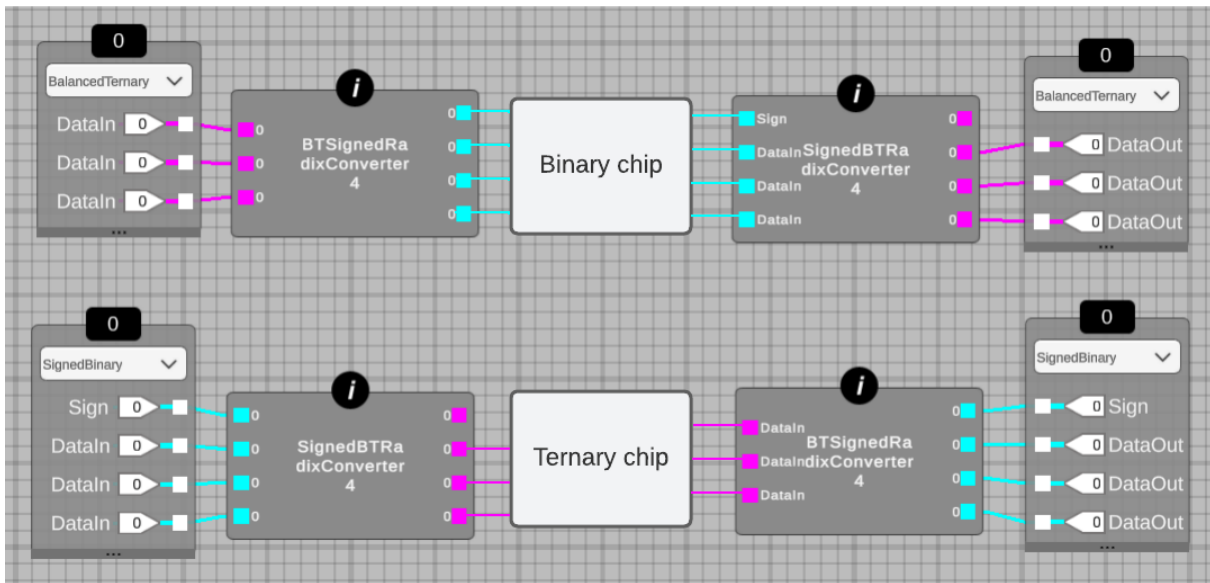


Figure 4.10: The radix converter circuits from Fig. 4.9 of **Paper E** and Fig. G.32 of **Appendix G.16** used in two configurations. The top configuration is used for signed binary chips and the bottom for balanced ternary chips.

The radix converter circuits in Fig. 4.9 and Fig. G.32 are simple IP blocks that allow interfacing binary logic circuits with ternary logic circuits. As a final test, both circuits have been chained back-to-back (Fig. 4.10). The two possible configurations are useful for common scenarios where it is desired to either integrate binary chips in a ternary system or integrate ternary chips in a binary system. The back-to-back configuration has been simulated in Vivado, verified on Basys-3 FPGA and submitted for tape-out with TinyTapeout 3.5 [266]. A simulated die shot can be found in **Appendix I**. All design, test bench and constraint files for the Basys-3 FPGA are open sourced and can be found in [266].

4.6 Conclusion

In this chapter **Papers B, E and F** were discussed. Central was MRCS, the first online ternary synthesis tool to design and verify ternary logic chips. The verilog generated for BCT signals allow experimenting with ternary logic using modern binary CMOS technology. The verilog can be deployed on FPGA's, work with simulations tools like Vivado and can be hardened for tape-out as ASIC with OpenLane. The HSPICE netlist generated for ternary signals allows mixed signal simulation of ternary circuits based on emerging multi-threshold CNTFET.

4 Mixed radix EDA for ternary computers

Several verified MRCS designs have been discussed in this chapter. A comprehensive overview of standard binary and ternary building blocks is presented for both combinatorial and sequential logic in **Appendix G.14 and G.15**. To interface with binary chips, binary to ternary and ternary to binary radix conversion circuits are presented. REBEL-2, a 2-trit RISC-like balanced ternary CPU architecture is presented with a novel balanced ternary ISA. Four MRCS designs have been submitted for tape-out using the TinyTapeout service on 130nm.

The online EDA tool lays the foundation for a larger vision to design ternary VLSI and mixed radix VLSI with CNTFET, RRAM and CMOS. Future versions of MRCS will explore integration with ALIGN for analog layout [299] and integration with an RRAM compiler [300]. This enables integrating the multi-state RRAM controller from Chapter 3 in MRCS. Lastly, large language models (LLM) for high level synthesis [105] with ternary verilog in MRCS will be explored to speed up the design process.

—*“The only phrase I’ve ever disliked is, ‘Why, we’ve always done it that way.’ I always tell young people, ‘Go ahead and do it. You can always apologize later.”*

Adm. Grace Murray Hopper, pioneer in computer programming

—*“The physical world remains stubbornly analog .. they must ultimately rely on analog interfaces to digitize real-world information”*

Prof. Murmann and Prof. Hoefflinger in [301]

5

Discussion

5.1 Towards a ternary technology stack

Binary computing has enormous inertia. It is a multi-billion ecosystem, with fabrication processes, software algorithms and education being optimized for it. Disrupting this ecosystem with radix-3 logic to overcome the three scaling walls is unlikely to happen in the next decade. Technology roadmaps are planned well in advance and new technologies are introduced in small numbers initially. The slow transition path might be similar to the MVL adoption paths of the memory and communication industry. Fortunately, binary is a subset of ternary so the transition path can be smooth. The same ternary logic design can be implemented with ternary signals or with BCT signals, as demonstrated with MRCS.

There are several steep challenges and milestones to beat for radix-3 logic to be competitive in practice. Perhaps the largest challenge is to make a power, performance, and area (PPA) competitive balanced ternary full adders. They are ubiquitous logic gates in CPU designs. This would require novel switching devices that can be fabricated with a CMOS-compatible process. Another steep challenge is to design production-ready ternary logic chips and demonstrate the theoretical merits at the system level. An example could be the low-power IoT industry with a 5-trit ternary microcontroller and compare it to an 8-bit equivalent. The goal for radix-3 is to be competitive on both cost and specs at mass-production scale for products that value efficiency more than simplicity. A major milestone to overcome is the binary RISC-V microcontrollers by WinChipHead ((WCH) for \$0.10 per chip [302]. Another milestone is the simplicity of designing microcontrollers with hardware description languages (HDL) like Silice which can describe a binary dual-core RISC-V CPU with just 120 lines of code [303]. Despite these steep challenges

and milestones the transition to radix-3 for logic seems to be a question of *when* rather than *if*. The IRDS 2022 roadmap projects the use of beyond-CMOS devices and focus on efficiency to be a necessity after 2028 [61].

The memory wall shows that there is a need for a better balance between communication and computation. Perhaps the greatest inspiration for modern IC's is Nature's balanced performance/watt solution: the brain [112]. In the long term a return to analog computing or a best-of-both-worlds [304] would make sense, with radix-3 being the first step towards a higher radix compute paradigm. A higher radix would benefit various emerging computing fields such as probabilistic, neuromorphic, optical and quantum computing. The mechanisms for computing in these fields are not tied to binary signals or encoding. A full embrace of the ternary compute paradigm requires reinventing/optimizing the technology stack and EDA flows presented in Fig. 1.2. This tremendous exercise touches all aspects of ternary computing: From programming language constructs and ternary-aware compilers, data structures, algorithms and operating systems to ISA, ternary VLSI and physical devices that work optimally with ternary signals.

5.2 Open questions

In 1977 Hamacher and Vranesic stated that ternary research needs to satisfy 3 factors to be a serious alternative to binary [153]: intellectual challenge, physical feasibility and applicability. This thesis showed it was worthwhile to view computing ideas and theories pioneered in the 1950's through a modern lens. There is a clear demand for more compute power in smaller, lower power form factors. There is nothing fundamentally blocking ternary computers and no computing limits are reached. New devices, circuits and systems are certainly waiting to be discovered. Five large open questions have been identified to continue this research:

1. What is needed to enrich hardware description languages (HDL) and high level synthesis (HLS) tools to describe and control the new capabilities radix-3 offers?
2. Does radix-3 reduce design complexity of the asynchronous compute paradigm as synchronization mechanisms such as the {read lock, no lock, write lock} can be captured in a single trit?
3. How can one or more layers of multi-state memristors, a memory controller and multi-threshold CNTFET logic be integrated in a 3D SoC workflow [305]?
4. What are the practical challenges and benefits of synchronous designs with a ternary clock, especially with respect to power consumption?
5. How does BCT compare to binary at the system level when considering a functionally equivalent CPU design on older process nodes using VLSI metrics?

—“The key message for the last 50 years was that ‘there is a lot of room at the bottom’. the message for the next 50 years is ‘there is a lot of room between devices’ or **‘there is a lot of slack in wires’**”

P. Ruch et al., IBM Research [10]

—“An early mentor told me to run towards problems, because that’s where you’ll find opportunity”

Lisa Su, CEO of AMD

6

Conclusion

The aim of this PhD project was to revisit Atanasoff’s and von Neumann’s 80 year old conclusion and determine if radix-2 (binary) is still the better radix choice for computing given new device and material advancements. The full answer is presented over four chapters in this thesis and is summarized below.

The blunt conclusion is that radix-2 can no longer be considered the most efficient radix for computing. It remains the simplest radix and is therefore unlikely to become obsolete. The nuanced conclusion is that logic is still radix-2 and that memory and communication have shifted to MVL in the last 20 years for efficiency reasons. Post-binary examples include the latest SSD’s, SD-cards, GDDR7 DRAM, embedded RRAM and communication standards like USB, I3C, Bluetooth EDR, WIFI and Ethernet. The fragmented situation of logic being binary and memory and communication being non-binary requires continuous and costly signal conversions. The heart of the problem is the continued bi-stable device-centric scaling which became inefficient after Dennard-scaling stopped in 2005. Under utilization of transistors (dark silicon) and the increasing energy and delay gap between computation on the ALU and communication with the ALU shows the need for more efficient computation using a higher radix rather than faster computation with radix-2. Fundamental and practical limits like Shannon’s limit, Rent’s rule, material and area limits at high frequencies further support a higher radix approach.

The radix economy theorem shows mathematically that radix-3 (ternary) is the closest discrete radix to the optimum e , but only when considering that the number of devices and the number of symbols per device are equally costly. This model has been shown to be accurate with memory devices which can hold multiple states (symbols) per device. For logic this cost model seems wrong as encoding circuit representations with CMOS is done in radix-1 with PMOS and NMOS devices each encoding a state. Ternary is the first

6 Conclusion

member of the MVL family and also the first odd radix. Shannon demonstrated in the 1950's that balanced arithmetic around zero with odd radices is incredibly efficient and elegant for computation. Several benefits of ternary have been discussed in this thesis which were categorized around seven application domains (7 C's).

Central in this thesis is addressing the simulation, implementation and verification of radix-3 circuits. Measurements and experiments with multi-stable devices are still under reported in literature. The absence of higher radix EDA tools and modern MVL synthesis algorithms has consistently been mentioned as a knowledge gap. Another lacuna is fair radix comparison and system level benchmarking. Several recommendations were proposed in the thesis including feature parity and optimal pin utilization. The first tool developed was uMemristorToolbox, an open source software framework to experiment with multi-state memristive devices. Using the tool a non-volatile ternary memory controller was made and a multi-state RRAM development platform. Initial experiments confirm that radix-3 memory with memristors is both feasible and low-cost. More work is needed at larger scale with more device, chip and batch variance under wider operating conditions.

The second tool developed was Mixed Radix Circuit Synthesizer (MRCS), the first browser-based EDA tool to design and verify binary, ternary and hybrid (mixed radix) circuits. It features a novel MVL circuit synthesis algorithm with HSPICE and verilog output targeting CMOS and multi-threshold CNTFET. Ternary circuits are automatically converted to BCT when targeting verilog such that designers can focus on functionality. Designs can be converted to ASIC with Openlane's RTL-to-GDS toolchain, HSPICE Simulator or to FPGA with Vivado's RTL-To-Bitstream toolchain. Binary-to-ternary and inverse hardware converters made with MRCS are discussed in the thesis which enable seamless interfacing between the two radices. The tool was used to design REBEL-2, a novel balanced ternary CPU with RISC-V-like ISA. The REBEL-2 ISA can resolve $\frac{27}{38}$ RV32i instructions with only 9 opcodes and is great for educational purposes. Four MRCS designs have been submitted for tape-out using TinyTapeout, a new and affordable tape-out service. Initial verification of a TinyTapeout 2 sample show correct functionality.

This thesis started with the question if a richer computer alphabet has the same benefits as it does for humans. Patterson and Hennessy [2, p. 68] answered this splendidly:

”Computer designers have a common goal: to find a language that makes it easy to build the hardware and the compiler while maximizing performance and minimizing cost and energy.”

The REBEL-2 ISA and industry developments that use ternary or BCT signals show that a richer computer alphabet empowers the computer language to scale today's power, memory and EDA walls. It is time to pivot away from device-centric scaling and engage in efficiency-centric scaling using a higher radix. Radix-3 is the optimal radix and would be the prime choice for this new compute paradigm.

Bibliography

- [1] J. V. Atanasoff, 'Advent of electronic digital computing,' *Annals of the History of Computing*, vol. 6, no. 3, pp. 229–282, 1984. DOI: 10.1109/MAHC.1984.10028.
- [2] D. A. Patterson and J. L. Hennessy, *Computer Organization and Design RISC-V Edition: The Hardware Software Interface*. Morgan Kaufmann Publishers Inc., 2020, ISBN: 9780128203316.
- [3] M. Tomasello, *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press, 2003, ISBN: 9780674010307. [Online]. Available: <http://www.jstor.org/stable/j.ctv26070v8>.
- [4] H. K. Pae, 'The alphabet,' in *Script Effects as the Hidden Drive of the Mind, Cognition, and Culture*. Springer, 2020. DOI: 10.1007/978-3-030-55152-0_4.
- [5] J. Man, 'Alpha beta,' in *How 26 Letters Shaped The Western World*. John Wiley & Sons, 2001, ISBN: 978-0-471-41574-9.
- [6] D. Crystal, 'The Cambridge encyclopedia of language,' in Cambridge University Press, 2010, vol. 3, ISBN: 978-0-521-73650-3.
- [7] I. L. Markov, 'Limits on fundamental limits to computation,' *Nature*, vol. 512, no. 7513, pp. 147–154, Aug. 2014. DOI: 10.1038/nature13570.
- [8] T. Mudge, 'Power: A first-class architectural design constraint,' *Computer*, vol. 34, no. 4, pp. 52–58, 2001. DOI: 10.1109/2.917539.
- [9] S. Moore and D. Greenfield, 'The next resource war: Computation vs. Communication,' in *Proceedings International Workshop on System Level Interconnect Prediction*, Association for Computing Machinery, 2008, pp. 81–86. DOI: 10.1145/1353610.1353627.
- [10] P. Ruch, T. Brunschweiler, W. Escher, S. Paredes and B. Michel, 'Toward five-dimensional scaling: How density improves efficiency in future computers,' *IBM Journal of Research and Development*, vol. 55, no. 5, pp. 1–13, 2011. DOI: 10.1147/JRD.2011.2165677.
- [11] G. Ifrah, *The universal history of computing: from the abacus to the quantum computer*. Wiley, 2001, ISBN: 978-0471441472.
- [12] W. Buchholz, 'Fingers or fists? (the choice of decimal or binary representation),' *Communications of the ACM*, vol. 2, no. 12, pp. 3–11, 1959. DOI: 10.1145/368518.368529.
- [13] N. H. McCoy, 'Introduction to modern algebra,' in Allyn and Bacon, 1968. DOI: 10.1145/368518.368529.
- [14] K. Rupp, *48 years of microprocessor trend data*, 2020. [Online]. Available: 10.5281/zenodo.3947823.

Bibliography

- [15] S. K. Moore and D. Schneider, *The state of the transistor in 3 charts*, 2022. [Online]. Available: <https://spectrum.ieee.org/transistor-density>.
- [16] K. Flamm, *Measuring Moore's law: Evidence from price, cost, and quality indexes*, 2018. [Online]. Available: [10.3386/w24553](https://doi.org/10.3386/w24553).
- [17] G. E. Moore, 'Cramming more components onto integrated circuits,' *IEEE Solid-State Circuits Society Newsletter*, vol. 11, no. 3, pp. 33–35, 2006. DOI: [10.1109/N-SSC.2006.4785860](https://doi.org/10.1109/N-SSC.2006.4785860).
- [18] R. Dennard, F. Gaensslen, H.-N. Yu, V. Rideout, E. Bassous and A. LeBlanc, 'Design of ion-implanted MOSFET's with very small physical dimensions,' *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, 1974. DOI: [10.1109/JSSC.1974.1050511](https://doi.org/10.1109/JSSC.1974.1050511).
- [19] T. H. Bullock, M. V. L. Bennett, D. Johnston, R. Josephson, E. Marder and R. D. Fields, 'The neuron doctrine, redux,' *Science*, vol. 310, no. 5749, pp. 791–793, 2005. DOI: [10.1126/science.1114394](https://doi.org/10.1126/science.1114394).
- [20] A. Mehonic and A. J. Kenyon, 'Brain-inspired computing needs a master plan,' *Nature*, vol. 604, no. 7905, pp. 255–260, Apr. 2022. DOI: [10.1038/s41586-021-04362-w](https://doi.org/10.1038/s41586-021-04362-w).
- [21] L. Baudry, I. Lukyanchuk and V. M. Vinokur, 'Ferroelectric symmetry-protected multibit memory cell,' *Scientific Reports*, vol. 7, no. 1, p. 42196, Feb. 2017. DOI: [10.1038/srep42196](https://doi.org/10.1038/srep42196).
- [22] B. Sengupta and M. B. Stemmler, 'Power consumption during neuronal computation,' *Proceedings of the IEEE*, vol. 102, no. 5, pp. 738–750, 2014. DOI: [10.1109/JPROC.2014.2307755](https://doi.org/10.1109/JPROC.2014.2307755).
- [23] C. E. Shannon, 'A symmetrical notation for numbers,' *The American Mathematical Monthly*, vol. 57, no. 2, pp. 90–93, 1950. DOI: [10.1080/00029890.1950.11999490](https://doi.org/10.1080/00029890.1950.11999490).
- [24] D. E. Knuth, 'Positional number systems,' in *The Art of Computer Programming: Vol. 2 Seminumerical Algorithms*, Addison-Wesley, 1969, ch. 4.1, pp. 207–209, ISBN: 0-201-03802-1.
- [25] R. P. Feynman, *Feynman Lectures on Computation (Frontiers in Physics)*, T. Hey and A. R. W., Eds. CRC Press, 2000, vol. 1, ISBN: 978-0-738-20296-9.
- [26] N. P. Brousentsov, S. P. Maslov, J. Ramil Alvarez and E. A. Zhogolev, 'Development of ternary computers at Moscow State University,' *Russian Virtual Computer Museum*, 2002. [Online]. Available: <https://www.computer-museum.ru/english/setun.htm>.
- [27] F. Hunger, *Setun: An inquiry into the Soviet Ternary Computer*. Institut für Buchkunst Leipzig, 2007.

- [28] J. R. Brusentsov N. P.; Alvarez, ‘Ternary computers: The Setun and the Setun 70,’ *Perspectives on Soviet and Russian Computing. SoRuCom 2006. IFIP Advances in Information and Communication Technology*, vol. 357, pp. 74–80, 2011. DOI: 10.1007/978-3-642-22816-2_10.
- [29] G. Hills, C. Lau, A. Wright *et al.*, ‘Modern microprocessor built from complementary carbon nanotube transistors,’ *Nature*, vol. 572, no. 7771, pp. 595–602, Aug. 2019. DOI: 10.1038/s41586-019-1493-8.
- [30] A. Raychowdhury and K. Roy, ‘Carbon-nanotube-based voltage-mode multiple-valued logic design,’ *IEEE Transactions on Nanotechnology*, vol. 4, no. 2, pp. 168–179, 2005. DOI: 10.1109/TNANO.2004.842068.
- [31] M. D. Bishop, G. Hills, T. Srimani *et al.*, ‘Fabrication of carbon nanotube field-effect transistors in commercial silicon manufacturing facilities,’ *Nature Electronics*, vol. 3, no. 8, pp. 492–501, Aug. 2020. DOI: 10.1038/s41928-020-0419-7.
- [32] K. E. Aasmundtveit, A. Roy and B. Q. Ta, ‘Direct integration of carbon nanotubes in CMOS, towards an industrially feasible process: A review,’ *IEEE Transactions on Nanotechnology*, vol. 19, pp. 113–122, 2020. DOI: 10.1109/TNANO.2019.2961415.
- [33] K. E. Aasmundtveit, B. Q. Ta, L. Lin, E. Halvorsen and N. Hoivik, ‘Direct integration of carbon nanotubes in Si microstructures,’ *Journal of Micromechanics and Microengineering*, vol. 22, no. 7, Jun. 2012. DOI: 10.1088/0960-1317/22/7/074006.
- [34] L. Chua, ‘Memristor: The missing circuit element,’ *IEEE Transactions on Circuit Theory*, vol. 18, no. 5, pp. 507–519, 1971. DOI: 10.1109/TCT.1971.1083337.
- [35] D. B. Strukov, G. S. Snider, D. R. Stewart and R. S. Williams, ‘The missing memristor found,’ *Nature*, vol. 453, no. 7191, pp. 80–83, May 2008. DOI: 10.1038/nature06932.
- [36] M. Rao, H. Tang, J. Wu *et al.*, ‘Thousands of conductance levels in memristors integrated on CMOS,’ *Nature*, vol. 615, no. 7954, pp. 823–829, Mar. 2023. DOI: 10.1038/s41586-023-05759-5.
- [37] B. E. Carpenber, ‘Turing and ACE: Lessons from a 1946 computer design,’ *1992 CERN School of Computing*, 1991. DOI: 10.5170/CERN-1993-003.230.
- [38] W. Phillips, J. V. Atanasoff, D. Michie, J. W. Mauchly, H. H. Goldstine and A. Goldstine, *The Advent of Electronic Computers*, B. Randell, Ed. Springer, 1973, pp. 287–347. DOI: 10.1007/978-3-642-96145-8_7.
- [39] W. Aspray and A. Glaser, ‘History of binary and other nondecimal numeration,’ 1981, ISBN: 978-0938228004.
- [40] H. Iwai, ‘History of transistor invention: 75th anniversary,’ in *2022 IEEE 16th International Conference on Solid-State & Integrated Circuit Technology (ICSICT)*, 2022, pp. 1–10. DOI: 10.1109/ICSICT55466.2022.9963262.

Bibliography

- [41] P. Ball, ‘Polynesian people used binary numbers 600 years ago,’ *Nature*, Dec. 2013. DOI: 10.1038/nature.2013.14380.
- [42] J. Ares, J. Lara, D. Lizcano and M. A. Martínez, ‘Who discovered the binary system and arithmetic? Did Leibniz plagiarize Caramuel?’ *Science and Engineering Ethics*, vol. 24, no. 1, pp. 173–188, Feb. 2018. DOI: 10.1007/s11948-017-9890-6.
- [43] J. W. Shirley, ‘Binary Numeration before Leibniz,’ *American Journal of Physics*, vol. 19, no. 8, pp. 452–454, Nov. 1951. DOI: 10.1119/1.1933042.
- [44] A. M. Turing, ‘On computable numbers, with an application to the entscheidungsproblem,’ *Proceedings of the London Mathematical Society*, vol. s2-42, no. 1, pp. 230–265, 1937. DOI: 10.1112/plms/s2-42.1.230.
- [45] B. Randell, ‘On Alan Turing and the origins of digital computers,’ 1972, pp. 1–20. [Online]. Available: <https://www.computerhistory.org/collections/catalog/102724643>.
- [46] C. E. Shannon, ‘A symbolic analysis of relay and switching circuits,’ *Transactions of the American Institute of Electrical Engineers*, vol. 57, no. 12, pp. 713–723, 1938. DOI: 10.1109/T-AIEE.1938.5057767.
- [47] J. Gilbey, ‘Biography: The ABC of computing,’ *Nature*, vol. 468, no. 7325, pp. 760–761, Dec. 2010. DOI: 10.1038/468760a.
- [48] V. Getov, ‘Insights into the origins of the IEEE computer society and the invention of electronic digital computing,’ *Computer*, vol. 54, pp. 13–18, Aug. 2021. DOI: 10.1109/MC.2021.3084067.
- [49] J. Smile, *The man who invented the computer: the biography of John Atanasoff, digital pioneer*. Doubleday, 2010, ISBN: 0385527136.
- [50] I. H. Anellis, ‘John Vincent Atanasoff: His place in the history of computer logic and technology,’ *Modern Logic*, vol. 7, no. 1, pp. 1–24, 1997. [Online]. Available: <https://projecteuclid.org/journals/modern-logic/volume-7/issue-1/John-Vincent-Atanasoff---his-place-in-the-history/rml/1204900339.full/>.
- [51] J. von Neumann, ‘First draft of a report on the EDVAC,’ *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993. DOI: 10.1109/85.238389.
- [52] A. W. Burks, H. H. Goldstine and J. von Neumann, ‘Preliminary discussion of the logical design of an electronic computing instrument,’ in *Papers of John von Neumann on Computing and Computing Theory*, W. Aspray and A. W. Burks, Eds., MIT Press, 1987, pp. 97–142. DOI: <https://dl.acm.org/doi/10.5555/98326.98337>.
- [53] W. S. McCulloch and W. Pitts, ‘A logical calculus of the ideas immanent in nervous activity,’ *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. DOI: 10.1007/BF02478259.

- [54] F. P. Brooks, G. A. Blaauw and W. Buchholz, ‘Processing data in bits and pieces,’ *IRE Transactions on Electronic Computers*, vol. EC-8, no. 2, pp. 118–124, 1959. DOI: 10.1109/TEC.1959.5219512.
- [55] L. Shustek, ‘An interview with Fred Brooks,’ *Communications of the ACM*, vol. 58, no. 11, pp. 36–40, Oct. 2015, ISSN: 0001-0782. DOI: 10.1145/2822519.
- [56] R. Richards, *Arithmetic operations in digital computers*. D. van Nostrand Company Inc., 1955. [Online]. Available: https://archive.org/details/arithmetic_operations_in_digital_computers.
- [57] Engineering Research Associates Staff, *High-Speed Computing Devices*, C. Tompkins, J. Wakelin and W. Stifler Jr., Eds. McGraw-Hill, 1950, ISBN: 9780262050289.
- [58] W. H. Ware, ‘Soviet computer technology–1959,’ *Communications of the ACM*, 1960. DOI: 10.1145/367149.1047530.
- [59] Gartner, *Gartner forecasts worldwide semiconductor revenue to grow 13.6% in 2022*, 2022. [Online]. Available: <https://www.gartner.com/en/newsroom/press-releases/2022-04-26-gartner-forecasts-worldwide-semiconductor-revenue-to-grow-13-6-percent-in-2022>.
- [60] V. Smil, *Moore’s curse. There is a dark side to the revolution in electronics: Unjustified technological expectations*, 2015. [Online]. Available: <https://spectrum.ieee.org/moores-curse>.
- [61] More Moore team, *International Roadmap for Devices and Systems: 2022 update More Moore*, 2022. [Online]. Available: https://irds.ieee.org/images/files/pdf/2022/2022IRDS_MM.pdf.
- [62] Beyond CMOS team, *International Roadmap for Devices and Systems: 2022 update Beyond CMOS and emerging materials integration*, 2022. [Online]. Available: https://irds.ieee.org/images/files/pdf/2022/2022IRDS_BC.pdf.
- [63] A. Kanduri, A. M. Rahmani, P. Liljeberg, A. Hemani, A. Jantsch and H. Tenhunen, *A Perspective on Dark Silicon*, A. M. Rahmani, P. Liljeberg, A. Hemani, A. Jantsch and H. Tenhunen, Eds. Cham: Springer, 2017, pp. 3–20. DOI: 10.1007/978-3-319-31596-6_1.
- [64] P. Bose, ‘Power wall,’ in *Encyclopedia of Parallel Computing*, D. Padua, Ed. Boston, MA: Springer, 2011, pp. 1593–1608. DOI: 10.1007/978-0-387-09766-4_499.
- [65] M. Bohr, ‘A 30 year retrospective on Dennard’s MOSFET scaling paper,’ *IEEE Solid-State Circuits Society Newsletter*, vol. 12, no. 1, pp. 11–13, 2007. DOI: 10.1109/N-SSC.2007.4785534.
- [66] R. Ho, K. Mai and M. Horowitz, ‘The future of wires,’ *Proceedings of the IEEE*, vol. 89, no. 4, pp. 490–504, 2001. DOI: 10.1109/5.920580.
- [67] S. M. Sze, *Setun, An inquiry into the Soviet Ternary Computer*. Wiley and Sons, 1981, ISBN: 978-0471056614.

Bibliography

- [68] A. Agarwal, P. C. Pradhan and B. P. Swain, 'From FET to SET: A review,' in *Advances in Electronics, Communication and Computing*, A. Kalam, S. Das and K. Sharma, Eds., Springer, 2018, pp. 199–209, ISBN: 978-981-10-4765-7.
- [69] L. Qin, C. Li, Y. Wei *et al.*, 'Recent developments in negative capacitance gate-all-around field effect transistors: A review,' *IEEE Access*, vol. 11, pp. 14 028–14 042, 2023. DOI: 10.1109/ACCESS.2023.3243697.
- [70] W. R. Deal, K. Leong, W. Yoshida, A. Zamora and X. B. Mei, 'InP HEMT integrated circuits operating above 1,000 GHz,' in *IEEE International Electron Devices Meeting (IEDM)*, 2016, pp. 29.1.1–29.1.4. DOI: 10.1109/IEDM.2016.7838502.
- [71] D. C. Sekar, A. Naeemi, R. Sarvari, J. A. Davis and J. D. Meindl, 'Intsim: A cad tool for optimization of multilevel interconnect networks,' in *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, 2007, pp. 560–567. DOI: 10.1109/ICCAD.2007.4397324.
- [72] Z. Tokei, *Scaling the back end of line – a toolbox filled with new processes, boosters and conductors*, 2019. [Online]. Available: <https://www.imec-int.com%7C/en/imec-magazine/imec-magazine-september-2019/scaling-the-beol-a-toolbox-filled-with-new-processes-boosters-and-conductors>.
- [73] D. Brooks, P. Bose, S. Schuster *et al.*, 'Power-aware microarchitecture: Design and modeling challenges for next-generation microprocessors,' *IEEE Micro*, vol. 20, no. 6, pp. 26–44, 2000. DOI: 10.1109/40.888701.
- [74] A. Ranjan, *Micro-architectural exploration for low power design*, 2015. [Online]. Available: <https://semiengineering.com/micro-architectural-exploration-for-low-power-design/>.
- [75] W. Nakayama, 'Heat in computers: Applied heat transfer in information technology,' *Journal of Heat Transfer*, vol. 136, Jan. 2014. DOI: 10.1115/1.4025377.
- [76] G. M. Gilson, S. J. Pickering, D. B. Hann and C. Gerada, 'Piezoelectric fan cooling: A novel high reliability electric machine thermal management solution,' *IEEE Transactions on Industrial Electronics*, vol. 60, no. 11, pp. 4841–4851, 2013. DOI: 10.1109/TIE.2012.2224081.
- [77] W. Wulf and S. McKee, 'Hitting the memory wall: Implications of the obvious,' *Computer Architecture News*, vol. 23, Jan. 1996.
- [78] J. L. Hennessy and D. A. Patterson, *Computer Architecture, A Quantitative Approach*. Morgan Kaufmann, 2019, ISBN: 9780128119051.
- [79] M. Jung, S. A. McKee, C. Sudarshan, C. Dropmann, C. Weis and N. Wehn, 'Driving into the Memory Wall: The role of memory for advanced driver assistance systems and autonomous driving,' in *Proceedings of the International Symposium on Memory Systems*, Association for Computing Machinery, 2018, pp. 377–386. DOI: 10.1145/3240302.3240322.

- [80] X. Zou, S. Xu, X. Chen, L. Yan and Y. Han, ‘Breaking the von Neumann bottleneck: Architecture-level processing-in-memory technology,’ *Science China Information Sciences*, vol. 64, no. 6, p. 160 404, Apr. 2021. DOI: 10.1007/s11432-020-3227-1.
- [81] A. Spessot and H. Oh, ‘1T-1C Dynamic Random Access Memory status, challenges, and prospects,’ *IEEE Transactions on Electron Devices*, vol. 67, no. 4, pp. 1382–1393, 2020. DOI: 10.1109/TED.2020.2963911.
- [82] T. Hiramoto, ‘Five nanometre CMOS technology,’ *Nature Electronics*, vol. 2, no. 12, pp. 557–558, Dec. 2019. DOI: 10.1038/s41928-019-0343-x.
- [83] J. Ryckaert, P. Weckx and S. M. Salahuddin, ‘3 - SRAM technology status and perspectives,’ in *Semiconductor Memories and Systems*, A. Redaelli and F. Pellizzer, Eds., Woodhead Publishing, 2022, pp. 55–86. DOI: 10.1016/B978-0-12-820758-1.00010-8.
- [84] S. Rusu, H. Muljono and B. Cherkauer, ‘Itanium 2 processor 6M: Higher frequency and larger L3 cache,’ *Micro, IEEE*, vol. 24, pp. 10–18, Apr. 2004. DOI: 10.1109/MM.2004.1289279.
- [85] H. M. D. Kabir and M. Chan, ‘SRAM precharge system for reducing write power,’ *HKIE Transactions*, vol. 22, no. 1, pp. 1–8, 2015. DOI: 10.1080/1023697X.2014.970761.
- [86] Intel, *Intel optane persistent memory*, 2020. [Online]. Available: <https://www.intel.com/content/www/us/en/developer/articles/technical/aerospike-pmdk-intel-optane-persistent-memory.html>.
- [87] W. Wan, R. Kubendran, C. Schaefer *et al.*, ‘A compute-in-memory chip based on Resistive Random-Access Memory,’ *Nature*, vol. 608, no. 7923, pp. 504–512, Aug. 2022. DOI: 10.1038/s41586-022-04992-8.
- [88] M. Lapedus, *Big trouble at 3nm*, 2018. [Online]. Available: <https://semiengineering.com/big-trouble-at-3nm>.
- [89] O. Burkacky, M. de Jong and J. Dragon, *Strategies to lead in the semiconductor world*, 2022. [Online]. Available: <https://semiengineering.com/micro-architectural-exploration-for-low-power-design/>.
- [90] T. Ajayi, D. Blaauw, T.-B. Chan *et al.*, ‘Openroad: Toward a self-driving, open-source digital layout implementation tool chain,’ in *Proc. Government Microcircuit Applications and Critical Technology Conference*, 2019, pp. 1105–1110.
- [91] O. Andreas, *Intelligent design of electronic assets (IDEA) & posh open source hardware (POSH)*, 2017. [Online]. Available: https://www.darpa.mil/attachments/eri_design_proposers_day.pdf.
- [92] S. M. Sze, *A Short History of Circuits and Systems, An inquiry into the Soviet Ternary Computer*, 1th, F. Maloberti and A. C. Davies, Eds. River, 2016, ISBN: 978-8793379718.

Bibliography

- [93] D. Abercrombie and M. White, *IC design: Preparing for the next node*, 2019. [Online]. Available: <https://resources.sw.siemens.com/en-US/white-paper-ic-design-preparing-for-the-next-node#disw-fulfillment-form>.
- [94] E. Sperling, *Design rule complexity rising*, 2018. [Online]. Available: <https://semiengineering.com/design-rule-complexity-rising/>.
- [95] F. Faggin, ‘The making of the first microprocessor,’ *IEEE Solid-State Circuits Magazine*, vol. 1, no. 1, pp. 8–21, 2009. DOI: 10.1109/MSSC.2008.930938.
- [96] A. Sangiovanni-Vincentelli, ‘The tides of EDA,’ *IEEE Design & Test of Computers*, vol. 20, no. 6, pp. 59–75, 2003. DOI: 10.1109/MDT.2003.1246165.
- [97] E. Dubrova, ‘Multiple-Valued Logic in VLSI: Challenges and opportunities,’ *Proceedings of NORCHIP’99*, Nov. 1999.
- [98] T. Hossain, S. M. M. Ahsan and T. Hoque, ‘Potential and pitfalls of Multi-Valued Logic circuits for hardware security,’ in *Dallas Circuits and Systems Conference (DCAS)*, 2023, pp. 1–6. DOI: 10.1109/DCAS57389.2023.10130261.
- [99] E. Dubrova, ‘Multiple-Valued Logic synthesis and optimization,’ in *Logic Synthesis and Verification*, S. Hassoun and T. Sasao, Eds. Boston, MA: Springer, 2002, pp. 89–114. DOI: 10.1007/978-1-4615-0817-5_4.
- [100] T. Ajayi, V. A. Chhabria, M. Fogaça *et al.*, ‘Toward an open-source digital flow: First learnings from the OpenROAD project,’ in *Proceedings Annual Design Automation Conference 2019*, New York, NY, USA: Association for Computing Machinery, 2019. DOI: 10.1145/3316781.3326334.
- [101] M. Shalan and T. Edwards, ‘Building OpenLANE: A 130nm OpenROAD-based tapeout- proven flow,’ in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–6, ISBN: 978-1-6654-2324-3.
- [102] A. Olofsson, W. Ransohoff and N. Moroze, ‘A distributed approach to silicon compilation,’ in *Proceedings ACM/IEEE Design Automation Conference*, New York, NY, USA: Association for Computing Machinery, 2022, pp. 1343–1346. DOI: 10.1145/3489517.3530673.
- [103] ‘ASAP7: A 7-nm finFET predictive process design kit,’ *Microelectronics Journal*, vol. 53, pp. 105–115, 2016. DOI: 10.1016/j.mejo.2016.04.006.
- [104] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi and J. Kepner, ‘AI and ML accelerator survey and trends,’ in *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, 2022, pp. 1–10. DOI: 10.1109/HPEC55821.2022.9926331.
- [105] J. Blocklove, S. Garg, R. Karri and H. Pearce, ‘Chip-chat: Challenges and opportunities in conversational hardware design,’ in *Workshop on Machine Learning for CAD (MLCAD)*, 2023, pp. 1–6. DOI: 10.1109/MLCAD58807.2023.10299874.

- [106] A. Mirhoseini, A. Goldie, M. Yazgan *et al.*, ‘A graph placement methodology for fast chip design,’ *Nature*, vol. 594, no. 7862, pp. 207–212, Jun. 2021. DOI: 10.1038/s41586-021-03544-w.
- [107] S. Lloyd, ‘Ultimate physical limits to computation,’ *Nature*, vol. 406, no. 6799, pp. 1047–1054, Aug. 2000. DOI: 10.1038/35023282.
- [108] R. Landauer, ‘Irreversibility and heat generation in the computing process,’ *IBM Journal of Research and Development*, vol. 5, no. 3, pp. 183–191, 1961. DOI: 10.1147/rd.53.0183.
- [109] L. B. Kish, ‘End of Moore’s law: Thermal (noise) death of integration in micro and nano electronics,’ *Physics Letters A*, vol. 305, no. 3, pp. 144–149, 2002. DOI: 10.1016/S0375-9601(02)01365-8.
- [110] V. Zhirnov, R. Cavin, J. Hutchby and G. Bourianoff, ‘Limits to binary logic switch scaling: A gedanken model,’ *Proceedings of the IEEE*, vol. 91, no. 11, pp. 1934–1939, 2003. DOI: 10.1109/JPROC.2003.818324.
- [111] ‘Towards Terabit memories,’ in *Chips 2020: A Guide to the Future of Nanoelectronics*, B. Hoefflinger, Ed. Springer, 2011, pp. 255–249. DOI: 10.1007/978-3-642-23096-7.
- [112] V. Beiu, ‘Grand challenges of nanoelectronics and possible architectural solutions: What do Shannon, von Neumann, Kolmogorov, and Feynman have to do with Moore,’ in *International Symposium on Multiple-Valued Logic (ISMVL’07)*, 2007. DOI: 10.1109/ISMVL.2007.27.
- [113] J. Keller, *Moore’s law is not dead*, 2019. [Online]. Available: <https://www.youtube.com/live/oIG9ztQw2Gc?feature=share>.
- [114] R. Pereira, M. Couto, F. Ribeiro *et al.*, ‘Energy efficiency across programming languages: How do energy, time, and memory relate?’ In *Proceedings ACM SIGPLAN International Conference on Software Language Engineering*, Association for Computing Machinery, 2017, pp. 256–267. DOI: 10.1145/3136014.3136031.
- [115] C. E. Shannon, ‘A mathematical theory of communication,’ *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.
- [116] J. R. Pierce, *An Introduction to Information Theory: Symbols, Signals and Noise*. Dover publications, 1980, ISBN: 978-0-486-24061-9.
- [117] C. Sun, Q. Chen, Y. Fu and L. Li, ‘Deep spiking neural network with ternary spikes,’ in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2022, pp. 251–254. DOI: 10.1109/BioCAS54905.2022.9948581.
- [118] K. Sayood, *Introduction to Data Compression*. Elsevier, 2006, ISBN: 978-0-12-620862-7.

Bibliography

- [119] USB-IF Electrical Workgroup, *USB 80G PHY layer analysis*, 2023. [Online]. Available: <https://www.usb.org/document-library/usb-80g-phy-layer-analysis>.
- [120] R. Stephens, *PAM4 for better and worse; is PAM4 worth the hassle*, 2019. [Online]. Available: <https://www.signalintegrityjournal.com/articles/1151-pam4-for-better-and-worse>.
- [121] T. M. Hollis, R. Schneider, M. Brox *et al.*, ‘An 8-Gb GDDR6X DRAM achieving 22 Gb/s/pin with single-ended PAM-4 signaling,’ *IEEE Journal of Solid-State Circuits*, vol. 57, no. 1, pp. 224–235, 2022. DOI: 10.1109/JSSC.2021.3104093.
- [122] D. Miller and H. Ozaktas, ‘Limit to the bit-rate capacity of electrical interconnects from the aspect ratio of the system architecture,’ *Journal of Parallel and Distributed Computing*, vol. 41, no. 1, pp. 42–52, 1997. DOI: 10.1006/jpdc.1996.1285.
- [123] D. A. B. Miller, ‘Attojoule optoelectronics for low-energy information processing and communications,’ *Journal of Lightwave Technology*, vol. 35, no. 3, pp. 346–396, 2017. DOI: 10.1109/JLT.2017.2647779.
- [124] C. S. Lent, ‘Information and entropy in physical systems,’ in *Energy Limits in Computation: A Review of Landauer’s Principle, Theory and Experiments*, C. S. Lent, A. O. Orlov, W. Porod and G. L. Snider, Eds. Cham: Springer, 2019, pp. 1–63. DOI: 10.1007/978-3-319-93458-7_1.
- [125] B. Hayes, ‘Computing science: Third base,’ *American Scientist*, vol. 89, no. 6, pp. 490–494, 2001. DOI: 10.1511/2001.40.490.
- [126] Hurst, ‘Multiple-Valued Logic: Its Status and its Future,’ *IEEE Transactions on Computers*, vol. C-33, no. 12, pp. 1160–1179, 1984. DOI: 10.1109/TC.1984.1676392.
- [127] Armstrong, ‘The complexity of computational circuits versus radix,’ *IEEE Transactions on Computers*, vol. C-29, no. 10, pp. 937–941, 1980. DOI: 10.1109/TC.1980.1675480.
- [128] D. Etiemble, *Ternary circuits: Why $r=3$ is not the optimal radix for computation*, 2019. arXiv: 1908.06841 [cs.AR].
- [129] A. Howe, *What is the radix economy*, 2022. [Online]. Available: https://www.youtube.com/watch?v=ad5r_-NkZE.
- [130] J. Son, K. Cho and S. Kim, ‘New ternary inverter with memory function using silicon feedback field-effect transistors,’ *Scientific Reports*, vol. 12, no. 1, p. 12907, Jul. 2022. DOI: 10.1038/s41598-022-17035-z.
- [131] J. W. Jeong, Y.-E. Choi, W.-S. Kim *et al.*, ‘Tunnelling-based ternary metal-oxide-semiconductor technology,’ *Nature Electronics*, vol. 2, no. 7, pp. 307–312, Jul. 2019. DOI: 10.1038/s41928-019-0272-8.

- [132] W. V. Quine, ‘The problem of simplifying truth functions,’ *The American Mathematical Monthly*, vol. 59, no. 8, pp. 521–531, 1952. DOI: 10.1080/00029890.1952.11988183.
- [133] B. Landman and R. Russo, ‘On a pin versus block relationship for partitions of logic graphs,’ *IEEE Transactions on Computers*, vol. C-20, no. 12, pp. 1469–1479, 1971. DOI: 10.1109/T-C.1971.223159.
- [134] P. Christie and D. Stroobandt, ‘The interpretation and application of Rent’s rule,’ *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 6, pp. 639–648, 2000. DOI: 10.1109/92.902258.
- [135] R. L. Russo, ‘On the tradeoff between logic performance and circuit-to-pin ratio for LSI,’ *IEEE Transactions on Computers*, vol. C-21, no. 2, pp. 147–153, 1972. DOI: 10.1109/TC.1972.5008919.
- [136] J. P. Gambino, S. A. Adderly and J. U. Knickerbocker, ‘An overview of Through-Silicon-Via technology and manufacturing challenges,’ *Microelectron. Eng.*, vol. 135, no. C, pp. 73–106, Mar. 2015. DOI: 10.1016/j.mee.2014.10.019.
- [137] Y. Nakata, S. Ozaki and H. Kudo, ‘Multilevel interconnect technology for 45-nm node CMOS LSIs,’ *Fujitsu Scientific & Technical Journal*, vol. 46, Jan. 2010. [Online]. Available: <https://www.fujitsu.com/global/about/resources/publications/fstj/archives/vol46-1.html>.
- [138] M. Kameyama, S. Kawahito and T. Higuchi, ‘A multiplier chip with multiple-valued bidirectional current-mode logic circuits,’ *Computer*, vol. 21, pp. 43–56, 1988. DOI: 10.1109/2.50.
- [139] D. C. Rine, Ed., *Computer Science and Multiple-Valued Logic: Theory and Applications*. North-Holland, 1977, ISBN: 978-1483249926.
- [140] V. Gaudet, ‘A survey and tutorial on contemporary aspects of multiple-valued logic and its application to microelectronic circuits,’ *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 1, pp. 5–12, 2016. DOI: 10.1109/JETCAS.2016.2528041.
- [141] M. D. Miller and M. A. Thornton, ‘MVL concepts and algebra,’ in *Multiple Valued Logic: Concepts and Representations*. Cham: Springer, 2008, pp. 21–42, ISBN: 978-3-031-79779-8. DOI: 10.1007/978-3-031-79779-8_2.
- [142] A. Dhande, V. Ingole and V. Ghiye, *Ternary Digital System: Concepts and Applications*. SM Online Publishers LLC, Oct. 2014, ISBN: 9780996274500.
- [143] S. Nemati, M. Haghi Kashani and R. Faghieh Mirzaee, ‘Comprehensive survey of Ternary Full Adders: Statistics, corrections, and assessments,’ *IET Circuits, Devices & Systems*, vol. 17, no. 3, pp. 111–134, 2023. DOI: 10.1049/cds2.12152.
- [144] M. Andreev, S. Seo, K.-S. Jung and J.-H. Park, ‘Looking beyond 0 and 1: Principles and technology of Multi-Valued Logic devices,’ *Advanced Materials*, vol. 34, no. 51, 2022. DOI: 10.1002/adma.202108830.

Bibliography

- [145] S. B. Jo, J. Kang and J. H. Cho, ‘Recent advances on Multivalued Logic gates: A materials perspective,’ *Advanced Science*, vol. 8, no. 8, 2021. DOI: 10.1002/advs.202004216.
- [146] K. R. Kim, J. W. Jeong, Y.-E. Choi, W.-S. Kim and J. & Chang, ‘Multi-Valued Logic device technology; overview, status, and its future for Peta-scale information density,’ *Journal of Semiconductor Engineering*, vol. 1, pp. 57–63, 2020. DOI: 10.22895/JSE.2020.0007.
- [147] Z. T. Sandhie, J. A. Patel, F. U. Ahmed and M. H. Chowdhury, ‘Investigation of Multiple-Valued Logic technologies for beyond-binary era,’ *ACM Computing Surveys*, vol. 54, no. 1, Jan. 2021. DOI: 10.1145/3431230.
- [148] S. Karmakar, ‘Multivalued Logic inverter using Multiple Channel Field Effect Transistor (mcfet),’ *Silicon*, vol. 14, no. 14, pp. 9041–9049, Sep. 2022. DOI: 10.1007/s12633-021-01503-8.
- [149] M. H. Moaiyeri, M. K. Q. Jooq, A. Al-Shidaifat and H. Song, ‘Breaking the limits in ternary logic: An ultra-efficient auto-backup/restore nonvolatile ternary flip-flop using negative capacitance CNTFET technology,’ *IEEE Access*, vol. 9, pp. 132 641–132 651, 2021. DOI: 10.1109/ACCESS.2021.3114408.
- [150] A. Yousefi, N. Eslami and M. H. Moaiyeri, ‘A reliable and energy-efficient non-volatile ternary memory based on hybrid finfet/rram technology,’ *IEEE Access*, vol. 10, pp. 105 040–105 051, 2022. DOI: 10.1109/ACCESS.2022.3211562.
- [151] R. Mateescu and R. Dechter, ‘AND/OR Multi-Valued Decision Diagrams (AOM-DDs) for weighted graphical models,’ *Proceedings Conference on Uncertainty in Artificial Intelligence (UAI) 2007*, Jun. 2012. DOI: 10.1613/jair.2605.
- [152] D. W. Jones, *The Ternary Manifesto · Heptavintimal encoding of ternary values*, 2012. [Online]. Available: <http://homepage.divms.uiowa.edu/~jones/ternary/hept.shtml>.
- [153] V. C. Hamacher and Z. G. Vranesic, ‘Multi-Valued Logic in arithmetic units,’ in Dec. 1977, pp. 485–505. DOI: 10.1016/B978-0-7204-0406-7.50021-1.
- [154] M. Moradi, R. F. Mirzaee and K. Navi, ‘New current-mode integrated ternary MIN/MAX circuits without constant independent current sources,’ *Journal of Electrical and Computer Engineering*, vol. 2015, 2015. DOI: <https://dl.acm.org/doi/abs/10.1155/2015/782089>.
- [155] S. Kim, S.-Y. Lee, S. Park, K. R. Kim and S. Kang, ‘A logic synthesis methodology for low-power ternary logic circuits,’ *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 9, pp. 3138–3151, 2020. DOI: 10.1109/TCSI.2020.2990748.
- [156] J. Lukasiewicz and J. Łukasiewicz, *Selected works / Jan Łukasiewicz ; edited by L. Borkowski* (Studies in logic and the foundations of mathematics), eng. Amsterdam: North-Holland, 1970, ISBN: 0-7204-2252-3.

- [157] H. Putnam, ‘Three-valued logic,’ in *The Logico-Algebraic Approach to Quantum Mechanics: Volume I: Historical Evolution*, C. A. Hooker, Ed. Dordrecht: Springer Netherlands, 1975, pp. 99–107. DOI: 10.1007/978-94-010-1795-4_5.
- [158] K. Degawa, T. Aoki, T. Higuchi, H. Inokawa and A. Takahashi, ‘A single-electron-transistor logic gate family and its application - part i: Basic components for binary, multiple-valued and mixed-mode logic,’ in *Proceedings International Symposium on Multiple-Valued Logic*, 2004, pp. 262–268. DOI: 10.1109/ISMVL.2004.1319952.
- [159] D. Etiemble and M. Israel, ‘Comparison of binary and multivalued ICs according to VLSI criteria,’ *Computer*, vol. 21, no. 4, pp. 28–42, 1988. DOI: 10.1109/2.49.
- [160] D. Etiemble, ‘Comments on “high-performance and energy-efficient CNFET-based designs for ternary logic circuits”,’ *IEEE Access*, vol. 8, pp. 220 015–220 016, 2020. DOI: 10.1109/ACCESS.2020.3041531.
- [161] D. Etiemble, *Evolution of technologies and multivalued circuits*, 2019. arXiv: 1907.01451 [cs.ET].
- [162] M. Takbiri, R. Faghih Mirzaee and K. Navi, ‘Analytical review of noise margin in MVL: Clarification of a deceptive matter,’ *Circuits, Systems, and Signal Processing*, vol. 38, no. 9, pp. 4280–4301, Sep. 2019. DOI: 10.1007/s00034-019-01063-8.
- [163] M. Glusker, *The ternary calculating machine of Thomas Fowler*, 2005. [Online]. Available: <https://www.mortati.com/glusker/fowler/>.
- [164] H. J. R. Grosch, ‘Signed ternary arithmetic,’ *Digital Computer Lab. Memo. M-1496. MIT*, 1952. [Online]. Available: http://www.bitsavers.org/pdf/mit/whirlwind/M-series/M-1496_Signed_Ternary_Arithmetic_May52.pdf.
- [165] R. D. Merrill, ‘A tabular minimization procedure for ternary switching functions,’ *IEEE Transactions on Electronic Computers*, vol. EC-15, no. 4, pp. 578–585, 1966. DOI: 10.1109/PGEC.1966.264380.
- [166] R. Hallworth and F. Heath, ‘Semiconductor circuits for ternary logic,’ *Proceedings of the IEE - Part C: Monographs*, vol. 109, 219–225(6), 15 Mar. 1962. DOI: 10.1049/pi-c.1962.0028.
- [167] R. D. Berlin, ‘Synthesis of n-valued switching circuits,’ *IRE Transactions on Electronic Computers*, vol. EC-7, no. 1, pp. 52–56, 1958. DOI: 10.1109/TEC.1958.5222096.
- [168] M. Yoeli and G. Rosenfeld, ‘Logical design of ternary switching circuits,’ *IEEE Transactions on Electronic Computers*, vol. EC-14, no. 1, pp. 19–29, 1965. DOI: 10.1109/PGEC.1965.264050.
- [169] I. Halpern and M. Yoeli, ‘Ternary arithmetic unit,’ *Proceedings of the Institution of Electrical Engineers*, vol. 115, 1385–1388(3), 10 Oct. 1968. DOI: 10.1049/piee.1968.0246.

Bibliography

- [170] D. Porat, ‘Three-valued digital systems,’ *Proceedings of the Institution of Electrical Engineers*, vol. 116, 947–954(7), 6 Jun. 1969. DOI: 10.1049/piee.1969.0177.
- [171] W. Alexander, ‘The ternary computer,’ *Electronics and Power*, vol. 10, no. 2, pp. 36–39, 1964. DOI: 10.1049/ep.1964.0037.
- [172] G. Frieder, ‘Ternary computers: Part i: Motivation for ternary computers,’ in *Conference Record of the 5th Annual Workshop on Microprogramming*, New York, NY, USA: Association for Computing Machinery, 1972, pp. 83–86. DOI: 10.1145/776378.776392.
- [173] G. Frieder and C. Luk, ‘Ternary computers: Part 2: Emulation of a ternary computer,’ in *Conference Record of the 5th Annual Workshop on Microprogramming*, New York, NY, USA: Association for Computing Machinery, 1972, pp. 86–89. DOI: 10.1145/776378.776393.
- [174] H. Mouftah and I. Jordan, ‘Implementation of 3-valued logic with C.O.S.M.O.S. Integrated Circuits,’ *Electronics Letters*, vol. 10, 441–442(1), 21 Oct. 1974. DOI: 10.1049/e1:19740350.
- [175] H. Mouftah and I. Jordan, ‘Design of ternary COS/MOS memory and sequential circuits,’ *IEEE Transactions on Computers*, vol. C-26, no. 3, pp. 281–288, 1977. DOI: 10.1109/TC.1977.1674821.
- [176] D. Etiemble and M. Isreal, ‘Implementation of ternary circuits with-binary Integrated Circuits,’ *IEEE Transactions on Computers*, vol. C-26, no. 12, pp. 1222–1233, 1977. DOI: 10.1109/TC.1977.1674783.
- [177] C. Bay, *Your CPU is binary*, 2015. [Online]. Available: <https://www.youtube.com/watch?v=gLJr0TFw6J0>.
- [178] Z. Li, P.-Y. Chen, H. Xu and S. Yu, ‘Design of ternary neural network with 3-D vertical RRAM array,’ *IEEE Transactions on Electron Devices*, vol. 64, no. 6, pp. 2721–2727, 2017. DOI: 10.1109/TED.2017.2697361.
- [179] A. Alaghi, W. Qian and J. P. Hayes, ‘The promise and challenge of stochastic computing,’ *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 8, pp. 1515–1531, 2018. DOI: 10.1109/TCAD.2017.2778107.
- [180] H. Wang, S. Ouyang, Y. Shen and X. Chen, ‘Ternary optical computer: An overview and recent developments,’ in *International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, 2021, pp. 82–87. DOI: 10.1109/PAAP54281.2021.9720446.
- [181] M. Ilyas, S. Cui and M. Perkowski, ‘Ternary logic design in topological quantum computing,’ *Journal of Physics A: Mathematical and Theoretical*, vol. 55, no. 30, p. 305302, Jul. 2022. DOI: 10.1088/1751-8121/ac7b55.

- [182] S. Johnsen, ‘Towards optical quantum computing,’ Ph.D. dissertation, Dec. 2002. DOI: 10.13140/RG.2.2.26778.03526.
- [183] V. Gaudet, J. T. Butler, R. Wille and N. Homma, ‘Guest editorial emerging topics in Multiple-Valued Logic and its applications,’ *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 1, pp. 1–4, 2016. DOI: 10.1109/JETCAS.2016.2529507.
- [184] H. Gundersen, ‘Aspects of balanced ternary arithmetics implemented using CMOS recharged semi-floating gate,’ Ph.D. dissertation, University of Oslo, 2008. [Online]. Available: <https://api.semanticscholar.org/CorpusID:62288680>.
- [185] Smith, ‘The prospects for multivalued logic: A technology and applications view,’ *IEEE Transactions on Computers*, vol. C-30, no. 9, pp. 619–634, 1981. DOI: 10.1109/TC.1981.1675860.
- [186] J. Yoon, S. Baek, S. Kim and S. Kang, ‘Optimizing Ternary Multiplier design with Fast Ternary Adder,’ *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 2, pp. 766–770, 2023. DOI: 10.1109/TCSII.2022.3210282.
- [187] D. Kam, J. G. Min, J. Yoon, S. Kim, S. Kang and Y. Lee, ‘Design and evaluation frameworks for advanced RISC-based ternary processor,’ in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022, pp. 1077–1082. DOI: 10.23919/DATE54114.2022.9774584.
- [188] M. Reichenbach, J. Knödtel, S. Rachuj and D. Fey, ‘RISC-V3: A RISC-V compatible CPU with a data path based on redundant number systems,’ *IEEE Access*, vol. 9, pp. 43 684–43 700, 2021. DOI: 10.1109/ACCESS.2021.3063238.
- [189] H. Alemdar, V. Leroy, A. Prost-Boucle and F. Pétrot, ‘Ternary neural networks for resource-efficient AI applications,’ in *International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 2547–2554. DOI: 10.1109/IJCNN.2017.7966166.
- [190] T. M. Hollis, R. Schneider, M. Brox *et al.*, ‘25.3 an 8Gb GDDR6X DRAM achieving 22Gb/s/pin with single-ended PAM4 signaling,’ in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 348–350. DOI: 10.1109/ISSCC42613.2021.9365925.
- [191] S. Goyal, P. Agarwal and S. Gupta, ‘Demonstration of a single-lane 80 Gbps PAM-4 full-duplex serial link,’ in *2019 IEEE Symposium on High-Performance Interconnects (HOTI)*, 2019, pp. 32–35. DOI: 10.1109/HOTI.2019.00021.
- [192] Wikipedia, *PCI express comparison table*, 2023. [Online]. Available: https://en.wikipedia.org/wiki/PCI_Express#Comparison_table.
- [193] Mipi.Org, *I3C and I3C basic frequently asked questions*, 2023. [Online]. Available: <https://www.mipi.org/resources/i3c-frequently-asked-questions>.
- [194] Intel, *Intel introduces thunderbolt 5 connectivity standard*, 2023. [Online]. Available: <https://www.intel.com/content/www/us/en/newsroom/news/intel-introduces-thunderbolt-5-standard.html>.

Bibliography

- [195] E. Khorov, I. Levitsky and I. F. Akyildiz, ‘Current status and directions of IEEE 802.11be, the future WIFI 7,’ *IEEE Access*, vol. 8, 2020. DOI: 10.1109/ACCESS.2020.2993448.
- [196] Bluetooth.com, *Bluetooth technology overview*, 2023. [Online]. Available: <https://www.bluetooth.com/learn-about-bluetooth/tech-overview/>.
- [197] A. Stakhov, ‘Brousentsov’s Ternary Principle, Bergman’s Number System and Ternary Mirror-symmetrical Arithmetic,’ *The Computer Journal*, vol. 45, no. 2, pp. 221–236, Jan. 2002. DOI: 10.1093/comjnl/45.2.221.
- [198] H. Saar, *A deep dive into MIPI A-PHY and its benefits for automotive*, 2021. [Online]. Available: <https://www.mipi.org/blog/a-deep-dive-into-mipi-a-phy-and-its-benefits-for-automotive>.
- [199] M. Scarlatella, *White paper outlines breakthrough IoT power efficiencies available with MIPI I3C/I3C Basic*, 2023. [Online]. Available: <https://www.mipi.org/blog/white-paper-outlines-breakthrough-iot-power-efficiencies-available-with-mipi-i3c/i3c-basic>.
- [200] S. P. Release, *Samsung develops industry’s first GDDR7 DRAM to unlock the next generation of graphics performance*, 2023. [Online]. Available: <https://news.samsung.com/global/samsung-develops-industrys-first-gddr7-dram-to-unlock-the-next-generation-of-graphics-performance>.
- [201] K. Kim, S. Kim, Y. Lee *et al.*, ‘Extreme low power technology using ternary arithmetic logic circuits via drastic interconnect length reduction,’ in *International Symposium on Multiple-Valued Logic (ISMVL)*, 2020, pp. 155–158. DOI: 10.1109/ISMVL49045.2020.00-13.
- [202] H. Gundersen and Y. Berg, ‘Fast addition using balanced ternary counters designed with CMOS semi-floating gate devices,’ in *International Symposium on Multiple-Valued Logic (ISMVL)*, 2007, pp. 30–30. DOI: 10.1109/ISMVL.2007.23.
- [203] D. E. Siegel, ‘All searches are divided into three parts: String searches using ternary trees,’ in *Proceedings of the APL98 Conference on Array Processing Languages*, ser. APL ’98, Rome, Italy: Association for Computing Machinery, 1998, pp. 57–68. DOI: 10.1145/327559.327618.
- [204] P. Kocher, J. Jaffe, B. Jun and P. Rohatgi, ‘Introduction to differential power analysis,’ *Journal of Cryptographic Engineering*, vol. 1, no. 1, pp. 5–27, Apr. 2011. DOI: 10.1007/s13389-011-0006-y.
- [205] D. Sokolov, J. Murphy, A. Bystrov and A. Yakovlev, ‘Improving the security of dual-rail circuits,’ in *Cryptographic Hardware and Embedded Systems - CHES 2004*, Springer, 2004, pp. 282–297.
- [206] B. Cambou, P. G. Flikkema, J. Palmer, D. Telesca and C. Philabaum, ‘Can ternary computing improve information assurance?’ *Cryptography*, vol. 2, no. 1, 2018. DOI: 10.3390/cryptography2010006.

- [207] B. Cambou and M. Orłowski, ‘PUF designed with Resistive RAM and ternary states,’ in *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, Association for Computing Machinery, 2016. DOI: 10.1145/2897795.2897808.
- [208] P. G. Flikkema, J. Palmer, T. Yalcin and B. Cambou, ‘Dynamic computational diversity with multi-radix logic and memory,’ in *2020 IEEE High Performance Extreme Computing Conference (HPEC)*, 2020, pp. 1–6. DOI: 10.1109/HPEC43674.2020.9286255.
- [209] S. Assiri, B. Cambou, D. D. Booher, D. Ghanai Miandoab and M. Mohammadinodoushan, ‘Key exchange using ternary system to enhance security,’ in *Annual Computing and Communication Workshop and Conference (CCWC)*, 2019, pp. 0488–0492. DOI: 10.1109/CCWC.2019.8666511.
- [210] B. Parhami, ‘Truncated ternary multipliers,’ *IET Computers & Digital Techniques*, vol. 9, no. 2, pp. 101–105, 2015. DOI: 10.1049/iet-cdt.2013.0133.
- [211] S. Zhu, L. H. K. Duong, H. Chen, D. Liu and W. Liu, ‘FAT: An in-memory accelerator with fast addition for ternary weight Neural Networks,’ *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 3, pp. 781–794, Mar. 2023. DOI: 10.1109/TCAD.2022.3184276.
- [212] P. Schuddinck, F. M. Bufler, Y. Xiang *et al.*, ‘Ppac of sheet-based cfet configurations for 4 track design with 16nm metal pitch,’ in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2022, pp. 365–366. DOI: 10.1109/VLSITechnologyandCir46769.2022.9830492.
- [213] Z. G. Vranesic and K. C. Smith, ‘Engineering aspects of multi-valued logic systems,’ *Computer*, vol. 7, no. 9, pp. 34–41, 1974. DOI: 10.1109/MC.1974.6323306.
- [214] B. Behin-Aein, D. Datta, S. Salahuddin and S. Datta, ‘Proposal for an all-spin logic device with built-in memory,’ *Nature Nanotechnology*, vol. 5, no. 4, pp. 266–270, Apr. 2010. DOI: 10.1038/nnano.2010.31.
- [215] D. Etiemble, ‘Common fallacies about multivalued circuits,’ *Asian Journal of Research in Computer Science*, vol. 12, no. 4, pp. 67–83, Dec. 2021. DOI: 10.9734/ajrcos/2021/v12i430295.
- [216] D. Etiemble, *Technologies and computing paradigms: Beyond moore’s law?* 2022. arXiv: 2206.03201 [cs.ET].
- [217] S. Shin, E. Jang, J. W. Jeong and K. R. Kim, ‘CMOS-compatible ternary device platform for physical synthesis of Multi-Valued Logic circuits,’ in *2017 IEEE 47th International Symposium on Multiple-Valued Logic (ISMVL)*, 2017, pp. 284–289. DOI: 10.1109/ISMVL.2017.48.
- [218] Skywater Technologies, *Carbon Nanotube SoCs: High-density, stackable SoCs with Carbon Nanotube CMOS FETs + ReRAM*, 2020. [Online]. Available: <https://www.skywatertechnology.com/carbon-nanotubes/>.

Bibliography

- [219] M. Forghani and B. Razavi, 'Circuit bandwidth requirements for NRZ and PAM4 signals,' in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 990–994. DOI: 10.1109/ISCAS48785.2022.9937588.
- [220] B. D. Madhuri and S. Sunithamani, 'Crosstalk noise analysis of on-chip interconnects for ternary logic applications using FDTD,' *Microelectronics Journal*, vol. 93, 2019. DOI: 10.1016/j.mejo.2019.104633.
- [221] S. Pathania, S. Kumar and R. Sharma, 'Crosstalk analysis for rough copper interconnects considering ternary logic,' in *IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*, 2018, pp. 1–3. DOI: 10.1109/EDAPS.2018.8680906.
- [222] M. Takbiri, K. Navi and R. F. Mirzaee, 'Noise margin calculation in Multiple-Valued Logic,' in *International Conference on Computer and Knowledge Engineering (ICCKE)*, 2020, pp. 250–255. DOI: 10.1109/ICCKE50421.2020.9303638.
- [223] D. Etiemble, 'Why M-valued circuits are restricted to a small niche,' *J. Multiple Valued Log. Soft Comput.*, vol. 9, no. 1, pp. 109–123, 2003. [Online]. Available: <https://www.oldcitypublishing.com/journals/mvlsc-home/mvlsc-issue-contents/mvlsc-volume-9-number-1-2003/mvlsc-9-1-p-109-123/>.
- [224] D. Etiemble and R. A. Jaber, 'Design of (3,2) and (4,2) CNTFET ternary counters for multipliers,' vol. 16, pp. 103–118, Jul. 2023. DOI: 10.9734/ajrcos/2023/v16i3349.
- [225] D. Etiemble, 'Post algebras and ternary adders,' *Journal of Electrical Systems and Information Technology*, vol. 10, no. 1, p. 20, Mar. 2023. DOI: 10.1186/s43067-023-00088-z.
- [226] D. Etiemble, *Multivalued circuits and interconnect issues*, 2020. arXiv: 2012.01267 [cs.AR].
- [227] D. Etiemble, 'On the performance of Multivalued Integrated Circuits: Past, present and future,' in *Proceedings International Symposium on Multiple-Valued Logic*, 1992, pp. 156–164. DOI: 10.1109/ISMVL.1992.186790.
- [228] H. Sutter, *The free lunch is over; a fundamental turn toward concurrency in software*, 2005. [Online]. Available: <http://www.gotw.ca/publications/concurrency-ddj.htm>.
- [229] L. O. Chua, *The chua lectures - part 1 : Once over lightly*, 2015. [Online]. Available: <https://youtu.be/B9Z2Ktacd4s&t=665>.
- [230] M.-K. Song, J.-H. Kang, X. Zhang *et al.*, 'Recent advances and future prospects for memristive materials, devices, and systems,' *ACS Nano*, vol. 17, no. 13, pp. 11 994–12 039, Jul. 2023, ISSN: 1936-0851. DOI: 10.1021/acsnano.3c03505.
- [231] M. Lanza, A. Sebastian, W. D. Lu *et al.*, 'Memristive technologies for data storage, computation, encryption, and radio-frequency communication,' *Science*, vol. 376, no. 6597, eabj9979, 2022. DOI: 10.1126/science.abj9979.

- [232] S. Technologies, *Skywater open source pdk*, 2020. [Online]. Available: <https://github.com/google/skywater-pdk>.
- [233] R. S. Williams, ‘How we found the missing memristor,’ *IEEE Spectrum*, vol. 45, no. 12, pp. 28–35, 2008. DOI: 10.1109/MSPEC.2008.4687366.
- [234] W. Shim, J. Meng, X. Peng, J. Seo and S. Yu, ‘Impact of multilevel retention characteristics on RRAM based DNN inference engine,’ in *IEEE International Reliability Physics Symposium, IRPS 2021 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Mar. 2021. DOI: 10.1109/IRPS46558.2021.9405210.
- [235] Fujitsu Semiconductor Memory Solution Limited, *ReRAM overview (white paper)*, 2023. [Online]. Available: https://www.fujitsu.com/jp/group/fsm/en/products/reram/ReRAM_whitepaper_2023e.pdf.
- [236] Fujitsu Semiconductor Memory Solution Limited, *MIKROE-3641 ReRAM click development board*, 2023. [Online]. Available: <https://www.mikroe.com/reram-click>.
- [237] Knowm Inc., *Knowm Shop*, 2019. [Online]. Available: <https://knowm.com/collections/all>.
- [238] Knowm Inc., *Knowm SDC datasheet*, 2019. [Online]. Available: https://knowm.org/downloads/Knowm_Memristors.pdf.
- [239] S. K. Kingra, V. Parmar, D. Verma *et al.*, ‘Fully binarized, parallel, RRAM-based computing primitive for in-memory similarity search,’ *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 1, pp. 46–50, 2023. DOI: 10.1109/TCSII.2022.3207378.
- [240] J. B. Nilsen, ‘Memristor Implementation of a Ternary Storage Circuit,’ M.S. thesis, USN, Norway, 2020.
- [241] M. S. Virk, ‘Memristor Development Platform - Dual Source Control For Implementations of Multistate Memristive Memory,’ M.S. thesis, USN, Norway, 2022.
- [242] M. D. Pickett, D. B. Strukov, J. L. Borghetti *et al.*, ‘Switching dynamics in titanium dioxide memristive devices,’ *Journal of Applied Physics*, vol. 106, no. 7, Oct. 2009. DOI: 10.1063/1.3236506.
- [243] K. A. Campbell, ‘Self-Directed Channel memristor for high temperature operation,’ *Microelectronics journal*, vol. 59, pp. 10–14, 2017. DOI: 10.1016/j.mejo.2016.11.006.
- [244] S. Stathopoulos, A. Khiat, M. Trapatseli *et al.*, ‘Multibit memory operation of metal-oxide bi-layer memristors,’ *Nature Scientific Reports*, vol. 7, p. 17532, 2017. DOI: 10.1038/s41598-017-17785-1.
- [245] J. Geler-Kremer, F. Eltes, P. Stark *et al.*, ‘A ferroelectric multilevel non-volatile photonic phase shifter,’ *Nature Photonics*, vol. 16, no. 7, pp. 491–497, Jul. 2022. DOI: 10.1038/s41566-022-01003-0.

Bibliography

- [246] N. TaheriNejad and D. Radakovits, ‘From behavioral design of memristive circuits and systems to physical implementations,’ *IEEE Circuits and Systems Magazine*, vol. 19, no. 4, pp. 6–18, 2019. DOI: 10.1109/MCAS.2019.2945209.
- [247] Knowm Inc., *Memristor discovery github project*, 2019. [Online]. Available: <https://github.com/knowm/memristor-discovery> (visited on 01/03/2020).
- [248] S. Bos, *uMemristorToolbox github repository*, 2022. [Online]. Available: <https://github.com/aiunderstand/uMemristorToolbox>.
- [249] A. Nugent, *Knowm memristor discovery manual*, 2019.
- [250] A. Jagath, C. Leong, N. Thulasiraman and H. Almurib, ‘An insight into physics based RRAM models – a review,’ *The Journal of Engineering*, vol. 2019, May 2019. DOI: 10.1049/joe.2018.5234.
- [251] S. Kvatinsky, E. G. Friedman, A. Kolodny and U. C. Weiser, ‘Team: Threshold adaptive memristor model,’ *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 1, pp. 211–221, 2013. DOI: 10.1109/TCSI.2012.2215714.
- [252] S. Kvatinsky, M. Ramadan, E. G. Friedman and A. Kolodny, ‘VTEAM: A general model for voltage-controlled memristors,’ *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 786–790, 2015. DOI: 10.1109/TCSII.2015.2433536.
- [253] D. Bielek, Z. Kolka, V. Biolková, Z. Bielek and S. Kvatinsky, ‘(V)TEAM for SPICE simulation of memristive devices with improved numerical performance,’ *IEEE Access*, vol. 9, pp. 30 242–30 255, 2021. DOI: 10.1109/ACCESS.2021.3059241.
- [254] S. Kvatinsky, G. Satat, N. Wald, E. G. Friedman, A. Kolodny and U. C. Weiser, ‘Memristor-Based Material Implication (IMPLY) logic: Design principles and methodologies,’ *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 10, pp. 2054–2066, 2014. DOI: 10.1109/TVLSI.2013.2282132.
- [255] S. Kvatinsky, D. Belousov, S. Liman *et al.*, ‘MAGIC—Memristor-Aided Logic,’ *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 11, pp. 895–899, 2014. DOI: 10.1109/TCSII.2014.2357292.
- [256] T. Molter, *Knowm SDC datasheet*, 2017. [Online]. Available: <https://knowm.org/memristor-models-in-ltspice/>.
- [257] D. Radakovits and N. TaheriNejad, ‘Implementation and characterization of a memristive memory system,’ in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, 2019, pp. 1–4. DOI: 10.1109/CCECE.2019.8861788.
- [258] G. Tobey, J. Graeme and L. Huelsman, *Burr-Brown Operational Amplifiers Design and Applications*. McGraw Hill, 1971, ISBN: 978-0070649170.

- [259] C. E. Shannon, ‘The synthesis of two-terminal switching circuits,’ *The Bell System Technical Journal*, vol. 28, no. 1, pp. 59–98, 1949. DOI: 10.1002/j.1538-7305.1949.tb03624.x.
- [260] E. Fegri, ‘Design of a Balanced Ternary Tridirectional Loadable Counter Using CNTFETs,’ M.S. thesis, USN, Norway, 2022.
- [261] S. Bos, *Mixed Radix Circuit Synthesizer (MRCS) github repository*, 2022. [Online]. Available: <https://github.com/aiunderstand/MixedRadixCircuitSynthesis>.
- [262] S. Bos, *Mixed Radix Circuit Synthesis website*, 2022. [Online]. Available: <https://ternaryresearch.com/webgl/mixedradixcircuitsynthesizer/>.
- [263] R. K. Brayton, ‘The future of logic synthesis and verification,’ in *Logic Synthesis and Verification*. Boston, MA: Springer, 2002, pp. 403–434. DOI: 10.1007/978-1-4615-0817-5_15.
- [264] J. Deng and H.-S. P. Wong, ‘A compact SPICE model for Carbon-Nanotube Field-Effect Transistors including nonidealities and its application—part i: Model of the intrinsic channel region,’ *IEEE Transactions on Electron Devices*, vol. 54, no. 12, pp. 3186–3194, 2007. DOI: 10.1109/TED.2007.909030.
- [265] M. Venn, *Tinytapeout.com*. [Online]. Available: <https://tinytapeout.com/>.
- [266] S. Bos, *Tapeout 4: Balanced ternary counter and signed binary radix converter*, 2023. [Online]. Available: https://github.com/aiunderstand/tt03p5-4-trit-balanced-ternary-counter-bt_signb_bt-radix-converto.
- [267] S. Hassoun and T. Sasao, Eds., *Logic Synthesis and Verification*. Springer New York, 2001, ISBN: 978-0-7923-7606-4.
- [268] Berkeley MVSIS research group, *MVSIS: Logic synthesis and verification*, 2002. [Online]. Available: ptolemy.berkeley.edu/projects/embedded/mvsis/mvlogic.html.
- [269] R. Rudell and A. Sangiovanni-Vincentelli, ‘Exact minimization of Multiple-Valued Functions for PLA optimization,’ in *The Best of ICCAD: 20 Years of Excellence in Computer-Aided Design*, A. Kuehlmann, Ed. Springer, 2003, pp. 205–216. DOI: 10.1007/978-1-4615-0292-0_16.
- [270] L. Lavagno, S. Malik, R. Brayton and A. Sangiovanni-Vincentelli, ‘MIS-MV: Optimization of Multi-Level Logic with Multiple-Values Inputs,’ in *IEEE International Conference on Computer-Aided Design. Digest of Technical Papers*, 1990, pp. 560–563. DOI: 10.1109/ICCAD.1990.129981.
- [271] Y. Jiang and R. Brayton, ‘Logic optimization and code generation for embedded control applications,’ in *International Symposium on Hardware/Software Codesign*, 2001, pp. 225–229. DOI: 10.1109/HSC.2001.924680.

Bibliography

- [272] Berkeley Logic Synthesis and Verification Group, *ABC: A system for sequential synthesis and verification*, 2005. [Online]. Available: <https://people.eecs.berkeley.edu/~alanmi/abc>.
- [273] R. Brayton and A. Mishchenko, ‘ABC: An academic industrial-strength verification tool,’ in *Computer Aided Verification*, T. Touili, B. Cook and P. Jackson, Eds., Springer, 2010, pp. 24–40, ISBN: 978-3-642-14295-6.
- [274] S. Lin, Y.-B. Kim and F. Lombardi, ‘CNTFET-based design of ternary logic gates and arithmetic circuits,’ *IEEE Transactions on Nanotechnology*, vol. 10, no. 2, pp. 217–225, 2011. DOI: 10.1109/TNANO.2009.2036845.
- [275] C. Vudadha, A. Surya, S. Agrawal and M. B. Srinivas, ‘Synthesis of ternary logic circuits using 2:1 multiplexers,’ *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 12, pp. 4313–4325, 2018. DOI: 10.1109/TCSI.2018.2838258.
- [276] M. Lin, Q. Han, W. Luo, X. Wang, J. Chen and W. Lyu, ‘A ternary memristor full adder based on literal operation and module operation,’ *International Journal of Circuit Theory and Applications*, vol. 50, no. 8, pp. 2932–2940, 2022. DOI: 10.1002/cta.3287.
- [277] A. Paugh, ‘Application of binary devices and Boolean algebra to the realisation of 3-valued logic circuits,’ *Proceedings of the Institution of Electrical Engineers*, vol. 114, 335–338(3), 3 Mar. 1967. DOI: 10.1049/piee.1967.0072.
- [278] S. Lin, Y.-B. Kim and F. Lombardi, ‘A novel CNTFET-based ternary logic gate design,’ in *IEEE International Midwest Symposium on Circuits and Systems*, 2009, pp. 435–438. DOI: 10.1109/MWSCAS.2009.5236063.
- [279] S.-Y. Lee, S. Kim and S. Kang, ‘Ternary logic synthesis with modified Quine-McCluskey algorithm,’ in *International Symposium on Multiple-Valued Logic (ISMVL)*, 2019, pp. 158–163. DOI: 10.1109/ISMVL.2019.00035.
- [280] S. Kim, T. Lim and S. Kang, ‘An optimal gate design for the synthesis of ternary logic circuits,’ in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2018, pp. 476–481. DOI: 10.1109/ASPDAC.2018.8297369.
- [281] M. Karnaugh, ‘The map method for synthesis of combinational logic circuits,’ *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 72, no. 5, pp. 593–599, 1953. DOI: 10.1109/TCE.1953.6371932.
- [282] H. N. Risto, ‘A study of CNTFET implementations for ternary logic and data radix conversion,’ M.S. thesis, USN, Norway, 2020.
- [283] R. Brayton and S. Khatri, ‘Multi-Valued Logic synthesis,’ in *Proceedings International Conference on VLSI Design*, 1999, pp. 196–205. DOI: 10.1109/ICVD.1999.745148.

- [284] L. Duret-Robert, *TernaryVerilog : A custom hardware description language*, 2020. [Online]. Available: <https://louis-dr.github.io/ternaryverilog.html>.
- [285] M. M. Shulaker, G. Hills, N. Patil *et al.*, ‘Carbon Nanotube Computer,’ *Nature*, vol. 501, no. 7468, pp. 526–530, Sep. 2013. DOI: 10.1038/nature12502.
- [286] H. Wang, P. Wei, Y. Li *et al.*, ‘Tuning the threshold voltage of carbon nanotube transistors by n-type molecular doping for robust and flexible complementary circuits,’ *Proceedings of the National Academy of Sciences*, vol. 111, no. 13, pp. 4776–4781, 2014. DOI: 10.1073/pnas.1320045111.
- [287] S. Gadgil, G. N. Sandesh and chetan Kumar V, ‘Design and Implementation of a CNTFET-Based Ternary Logic Processor,’ Mar. 2023. DOI: 10.36227/techrxiv.22259437.v1.
- [288] D. Patterson, ‘Reduced instruction set computers then and now,’ *Computer*, vol. 50, no. 12, pp. 10–12, Dec. 2017. DOI: 10.1109/MC.2017.4451206.
- [289] A. Waterman and K. Asanovi, *The RISC-V instruction set manual volume i: Unprivileged ISA*, 2019. [Online]. Available: <https://riscv.org/wp-content/uploads/2019/06/riscv-spec.pdf>.
- [290] F. Pelletier and A. Hartline, ‘Ternary Exclusive OR,’ *Logic Journal of the IGPL*, vol. 16, no. 1, pp. 75–83, 2008. DOI: 10.1093/jigpal/jzm027.
- [291] S. Bos, *Tapeout 3: Balanced ternary calculator*, 2023. [Online]. Available: <https://github.com/aiunderstand/tt03-balanced-ternary-calculator>.
- [292] S. Bos, *1 trit balanced ternary ALU with MIN, MAX, STI, ADD, MULTIPLY gates*, 2022. [Online]. Available: <https://youtu.be/ApnVnEOL-ng>.
- [293] S. Bos, *How to build a balanced ternary calculator chip using MRCS*, 2023. [Online]. Available: <https://youtu.be/-DzVKAxmSQ0>.
- [294] Intel Corporation, *Intel 64 and IA-32 architectures software developer’s manual: Instruction set reference, a-z*, 2023. [Online]. Available: <https://cdrdv2-public.intel.com/774492/325383-sdm-vol-2abcd.pdf>.
- [295] P. A. Samet, ‘A note on radix conversion for integers,’ *Software: Practice and Experience*, vol. 1, no. 1, pp. 93–96, 1971. DOI: 10.1002/spe.4380010109.
- [296] S. Bos, *Tapeout 2: Binary encoded ternary to unary encoded ternary radix converter and comparator*, 2022. [Online]. Available: <https://github.com/aiunderstand/tt02-async-binary-ternary-convert-compare>.
- [297] Fu-Qiang Li, M. Morisue and T. Ogata, ‘A proposal of josephson binary-to-ternary converter,’ *IEEE Transactions on Applied Superconductivity*, vol. 5, no. 2, pp. 2632–2635, Jun. 1995. DOI: 10.1109/77.403130.

Bibliography

- [298] M. Shahangian, S. A. Hosseini and S. H. Pishgar Komleh, ‘Design of a multi-digit binary-to-ternary converter based on CNTFETs,’ *Circuits, Systems, and Signal Processing*, vol. 38, no. 6, pp. 2544–2563, Jun. 2019. DOI: 10.1007/s00034-018-0977-3.
- [299] T. Dhar, K. Kunal, Y. Li *et al.*, ‘ALIGN: A system for automating analog layout,’ *IEEE Design & Test*, vol. 38, no. 2, pp. 8–18, 2021. DOI: 10.1109/MDAT.2020.3042177.
- [300] D. Antoniadis, A. Mifsud, P. Feng and T. G. Constandinou, ‘An open-source RRAM compiler,’ in *2022 20th IEEE Interregional NEWCAS Conference (NEW-CAS)*, 2022, pp. 465–469. DOI: 10.1109/NEWCAS52662.2022.9842222.
- [301] B. Murmann and B. Hoefflinger, ‘The thirties,’ in *NANO-CHIPS 2030: On-Chip AI for an Efficient Data-Driven World*, B. Murmann and B. Hoefflinger, Eds. Cham: Springer International Publishing, 2020, pp. 577–583. DOI: 10.1007/978-3-030-18338-7_30.
- [302] WinChipHead, *32-bit general-purpose RISC-V MCU-CH32V003*, 2022. [Online]. Available: <https://github.com/openwch/ch32v003>.
- [303] S. Lefebvre, *ICE-V Dual*, 2021. [Online]. Available: <https://github.com/sylefeb/Silice/blob/master/projects/ice-v/IceVDual.md>.
- [304] G. Cowan, R. Melville and Y. Tsvividis, ‘A VLSI analog computer/digital computer accelerator,’ *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 42–53, 2006. DOI: 10.1109/JSSC.2005.858618.
- [305] M. M. Sabry Aly, T. F. Wu, A. Bartolo *et al.*, ‘The n3xt approach to energy-efficient abundant-data computing,’ *Proceedings of the IEEE*, vol. 107, no. 1, pp. 19–48, 2019. DOI: 10.1109/JPROC.2018.2882603.
- [306] A. Shabarshin, *3-nity alpha: 9-trit ternary computer architecture*, 2004. [Online]. Available: <http://ternary.info/wiki/index.php?n=Alpha>About>.
- [307] A. Shabarshin, *Shared silicon: Binary, ternary, quaternary logic tapeout*, 2015. [Online]. Available: <https://hackaday.io/project/11779-shared-silicon>.
- [308] J. Connelly, *Ternary computing testbed 3 trit computer architecture*, 2008. [Online]. Available: <http://xyzyzy.freeshell.org/trinary/CPE%20Report%20-%20Ternary%20Computing%20Testbed%20-%20RC6a.pdf>.
- [309] V. Lofgren, *Tunguska*, 2008. [Online]. Available: <https://tunguska.sourceforge.net/about.html>.
- [310] T. Leathers, *Simple balanced ternary computer virtual machine*, 2016. [Online]. Available: <https://github.com/SBTCVM/SBTCVM-Gen2-9>.
- [311] D. Jones, *Trillium*, 2016. [Online]. Available: <https://homepage.cs.uiowa.edu/~dwjones/ternary/trillium.shtml>.

- [312] D. Jones, *Binary Coded Ternary and its inverse*, 2016. [Online]. Available: <https://homepage.cs.uiowa.edu/~dwjones/ternary/bct.shtml>.
- [313] D. V. Sokolov, *Triador: 3-trit balanced ternary computer architecture*, 2017. [Online]. Available: <https://hackaday.io/project/28579-homebrew-ternary-computer>.
- [314] L. Duret-Robert, *Simple as possible (SAP) one 9-trit*, 2019. [Online]. Available: <https://louis-dr.github.io/sap1.html>.
- [315] L. Duret-Robert, *CMOS implementation and analysis of Ternary Arithmetic Logic Unit*, 2019. [Online]. Available: <https://louis-dr.github.io/ternalu3.html>.
- [316] C. D. Mauro, *3isa: 20-trit ternary computer architecture*, 2019. [Online]. Available: <https://www.youtube.com/watch?v=ohhkcVuTL5k>.
- [317] C. L. Rosa, *24-trit computer project 5500fp*, 2019. [Online]. Available: <https://www.facebook.com/profile.php?id=100065548687080>.
- [318] S. Bos, *Base-base pair bruteforce solver*, 2020. [Online]. Available: <https://github.com/aiunderstand/Base-BaseIntApprox-PairSolver>.
- [319] N. J. A. Sloane and R. K. Guy, *Denominators of convergents to $\log_2 3$ (formerly m1428)*, 1995. [Online]. Available: <https://oeis.org/A005664>.
- [320] D. Wust, D. Fey and J. Knödtel, ‘A programmable ternary CPU using hybrid CMOS/memristor circuits,’ *International Journal of Parallel, Emergent and Distributed Systems*, vol. 33, no. 4, pp. 387–407, 2018. DOI: 10.1080/17445760.2017.1422251.
- [321] Z. Liu, T. Pan, S. Jia and U. Wang, ‘Design of a novel ternary SRAM sense amplifier using CNFET,’ in *IEEE International Conference on ASIC (ASICON)*, 2017, pp. 207–210. DOI: 10.1109/ASICON.2017.8252448.
- [322] Y. Miyasaka, A. Mishchenko, J. Wawrzynek and N. J. Fraser, ‘Synthesizing a class of practical Boolean functions using truth tables,’ *Proceedings of the International Workshop on Logic & Synthesis, IWLS 2022*, 2022. [Online]. Available: https://people.eecs.berkeley.edu/~alanmi/publications/2022/iwls22_reo.pdf.
- [323] J. Ko, K. Park, S. Yong, T. Jeong, T. H. Kim and T. Song, ‘An optimal design methodology of ternary logic in Iso-device ternary CMOS,’ in *IEEE International Symposium on Multiple-Valued Logic (ISMVL)*, 2021, pp. 189–194. DOI: 10.1109/ISMVL51352.2021.00040.
- [324] A. C. Seabaugh and M. A. Reed, ‘Chapter 11 - resonant-tunneling transistors,’ in *Heterostructures and Quantum Devices*, vol. 24, Elsevier, 1994, pp. 351–383. DOI: 10.1016/B978-0-12-234124-3.50016-1.
- [325] M. D. Miller and M. A. Thornton, ‘Functional representations,’ in *Multiple Valued Logic: Concepts and Representations*. Cham: Springer, 2008, pp. 43–67. DOI: 10.1007/978-3-031-79779-8_3.

Bibliography

- [326] J. McClellan, R. Schafer and M. Yoder, *Signal Processing First*. Pearson Prentice Hall, 2003, ISBN: 0-13-120265-0.
- [327] J. K. Saini, A. Srinivasulu and R. Kumawat, ‘Fast and energy efficient full adder circuit using 14 CNFETs,’ *Solid State Electronics Letters*, vol. 2, pp. 67–78, 2020. DOI: 10.1016/j.ssel.2020.09.002.
- [328] T.-D. Ene and J. E. Stine, ‘Point-targeted sparseness and ling transforms on parallel prefix adder trees,’ in *IEEE Symposium on Computer Arithmetic (ARITH)*, 2022, pp. 68–75. DOI: 10.1109/ARITH54963.2022.00021.
- [329] R. Tocci, N. Widmer and G. Moss, *Digital Systems: Principles and Applications*. Pearson, 2017, ISBN: 1292162007.
- [330] S. Kim, S.-Y. Lee, S. Park and S. Kang, ‘Design of quad-edge-triggered sequential logic circuits for ternary logic,’ in *IEEE International Symposium on Multiple-Valued Logic (ISMVL)*, 2019, pp. 37–42. DOI: 10.1109/ISMVL.2019.00015.
- [331] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th. USA: Addison-Wesley Publishing Company, 2010, ISBN: 0321547748.
- [332] R. F. Mirzaee and N. Farahani, *Design of a Ternary Edge-Triggered D Flip-Flip-Flop for Multiple-Valued Sequential Logic*, 2016. arXiv: 1609.03897.
- [333] S. Bos, *Mixed Radix Converter app*, 2022. [Online]. Available: <https://github.com/aiunderstand/ternary-workbench-unity/tree/master/MixedRadixConverter>.
- [334] S. Bos, *Tapeout 1: 4-bit tristate loadable binary counter*, 2022. [Online]. Available: <https://github.com/aiunderstand/tt02-4bit-tristate-loadable-counter>.
- [335] Z. Han, ‘The Power-Delay Product and its implication to CMOS inverter,’ *Journal of Physics: Conference Series*, vol. 1754, no. 1, p. 012 131, Feb. 2021. DOI: 10.1088/1742-6596/1754/1/012131.
- [336] M. Flynn, P. Hung and K. Rudd, ‘Deep submicron microprocessor design issues,’ *IEEE Micro*, vol. 19, no. 4, pp. 11–22, 1999. DOI: 10.1109/40.782563.
- [337] C. M. University, *Cmos power consumption*, 2003. [Online]. Available: <https://course.ece.cmu.edu/~ece322/LECTURES/Lecture13/Lecture13.03.pdf>.
- [338] J. Rabaey and R. Amirtharajah, *CMOS power dissipation and trends*, 2008. [Online]. Available: https://www.ece.ucdavis.edu/~ramirtha/EEC216/W08/lecture1_updated.pdf.
- [339] Unity, *Unity API: Application.persistentdatapath*, 2022. [Online]. Available: <https://docs.unity3d.com/ScriptReference/Application-persistentDataPath.html>.
- [340] S. Bos, *How to build a balanced ternary calculator chip using MRCS*, 2023. [Online]. Available: <https://www.youtube.com/watch?v=-DzVKAxmSQ0>.



**uMemristorToolbox: Open source
framework to control memristors in Unity
for ternary applications**

**Not available online due to publisher
restrictions**

B

Automated synthesis of netlists for ternary valued n-ary logic functions in CNTFET circuits

Automated synthesis of netlists for ternary-valued n-ary logic functions in CNTFET circuits

Halvor Nybø Risto Steven Bos Henning Gundersen

Ternary Research Group, Department of Science and Industry Systems, University of South-Eastern Norway, Norway, {henning.gundersen}@usn.no

Abstract

This paper is an investigation of automated netlist synthesis for ternary-valued n-ary logic functions, based on a static ternary gate design methodology. We present an open-source C++ implementation, which outputs a ready-to-simulate SPICE subcircuit netlist file for ternary-valued n-ary function circuits. A circuit schematic of the 3-operand carry is demonstrated as synthesized by the netlist generator. We investigate a holistic (non-compound) approach to designing balanced full-adders by using 3-operand functions as compared to a traditional 2-operand compound design methodology. Three gate-level design approaches (compound, non-compound and hybrid) for the balanced full-adder have been simulated in HSPICE and are compared to each other and the state-of-the-art with simulation results. Furthermore, we propose to standardize the ternary functions by indexing them. This indexing system allows for the convenience of referencing any possible logic function with no ambiguity. This indexing is necessary as most ternary functions do not have semantic names (e.g. AND, OR) and the amount of unique 3-valued functions grows exponentially with higher arity.

Keywords: ternary, netlist, synthesis, simulation

1 Introduction

In recent years, along with developments of the carbon nanotube field-effect transistor (CNTFET), there have been a handful of papers on the designs and synthesis of ternary or 3-valued logic gates implemented in simulations of CNTFET circuits. One paper in particular (Kim et al., 2018) proposes a design method for ternary logic gates, with the use of pull-up and pull-down networks constructed from a truth table for the circuit. In ternary logic, with only two operands, there are 19683 possible logic gates. With three operands, 7.6e12 logic gates are possible. The process of designing the circuit and writing the netlists of these circuits can be a tedious process, especially for circuits with more than two operands. Therefore, an open-source netlist synthesizer is of much use. The study (Lee et al., 2019) reports to have automated this process, however their code is not open-source.

2 Function Indexing

To unambiguously refer to any of the many logic functions, we propose a simple indexing system.

2.1 Range of index in arities

The number of possible functions for a specific arity and radix can be calculated as in Equation 1, where R is the radix and A is the arity.

$$F_{range} = R^{R^A} \quad (1)$$

Table 1. Range of functions in arities and radices

Arity	Radix 2	Radix 3
1	$2^{2^1} = 4$	$3^{3^1} = 27$
2	$2^{2^2} = 16$	$3^{3^2} = 19683$
3	$2^{2^3} = 256$	$3^{3^3} = 7,625,597,484,987$

While in binary, there are few enough functions that naming the useful functions (AND, OR, XOR, etc.) has been a feasible practice, for ternary logic the quantity of possible functions is more unwieldy, as can be seen in Table 1. Therefore, an indexing system is proposed to refer to specific ternary-radix functions.

The indexing system maps every truth table to an index by counting up from 0 to the function range, along with the values of the truth table. As an example, a function always outputting the low value would be the 0th index. Then, the first row of the truth table acts as the three lowest-significance trits of the index, and so on. With a truth table listed vertically, the output values for a specific function can be read in ternary as the function index.

2.2 Heptavintimal index encoding

We adopt the usage of the base-27 heptavintimal notation for ternary values (Jones, 2012), as it conveniently covers three ternary digits (trits) per symbol, as shown in Table 2. As one operand can have one of three values, the truth table of a function is three trits long in each dimension. Therefore, a logic function index can conveniently be encoded with the heptavintimal notation.

Table 2. The heptavintimal notation

Weight(Decimal)	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Ternary	000	001	002	010	011	012	020	021	022	100	101	102	110	111
Heptavintimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D

Weight(Decimal)	14	15	16	17	18	19	20	21	22	23	24	25	26	
Ternary	112	120	121	122	200	201	202	210	211	212	220	221	222	
Heptavintimal	E	F	G	H	K	M	N	P	R	T	V	X	Z	

3 Methodology

Based on the static ternary gate design methodology of (Kim et al., 2018), we have implemented an algorithm in C++, which produces a ready-to-simulate SPICE subcircuit netlist file from a circuit truth table.

The program takes an n-dimensional truth table, and constructs the four truth tables for the pull-up and pull-down networks. Then, for each network, a set of n-dimensional rectangular groupings are found. Each of these groupings will provide a transistor path to the output within each pull-up and pull-down network.

3.1 Usage

To use the program, compile the open-source code in a C++ compiler. The netlists will be generated in the same file directory as the compiled program. When the program starts, it asks for the function arity, and the values of each element in the three-by-three n-dimensional truth table, with values low(0), middle(1), high(2), don't care(x). The filename will be generated as the function index of the specific function. The transistor parameters, as well as which CNTFET model is being used, can be specified with the string variables p0, p1, p2, n0, n1, n2.

The program will produce a subcircuit which must be connected externally to a 0.9V voltage supply, and the operand inputs. The circuits rely on external 2-transistor Positive Ternary Inverters (PTI) and Negative Ternary Inverters (NTI) to achieve the four different transistor operations detailed in (Kim et al., 2018).

3.2 Logic minimization algorithm

The logic minimization done to produce an optimized circuit is similar to karnaugh-mapping. The grouping algorithm takes the truth tables for each transistor network and draws n-dimensional rectangular groupings which covers every '1' on the truth table for each network, with as few groupings as possible. These groupings represent the transistor-paths in the circuit towards the output within each of the four transistor networks. Each transistor in series narrows down the throughput, until the logical rectangle of a grouping is achieved.

4 Circuit schematics

The common procedure for constructing functions with more than two operands is to combine smaller functions

to create bigger compound functions. However, circuits with more operands can also be generated with our program. Therefore we investigate the usage of 3-operand functions in a 1-trit balanced full-adder circuit, in the form of a non-compound and a hybrid gate architecture. These architectures are a more holistic view of the function as a relation between input and output. Figure 1 shows three different balanced full-adder circuit design approaches.

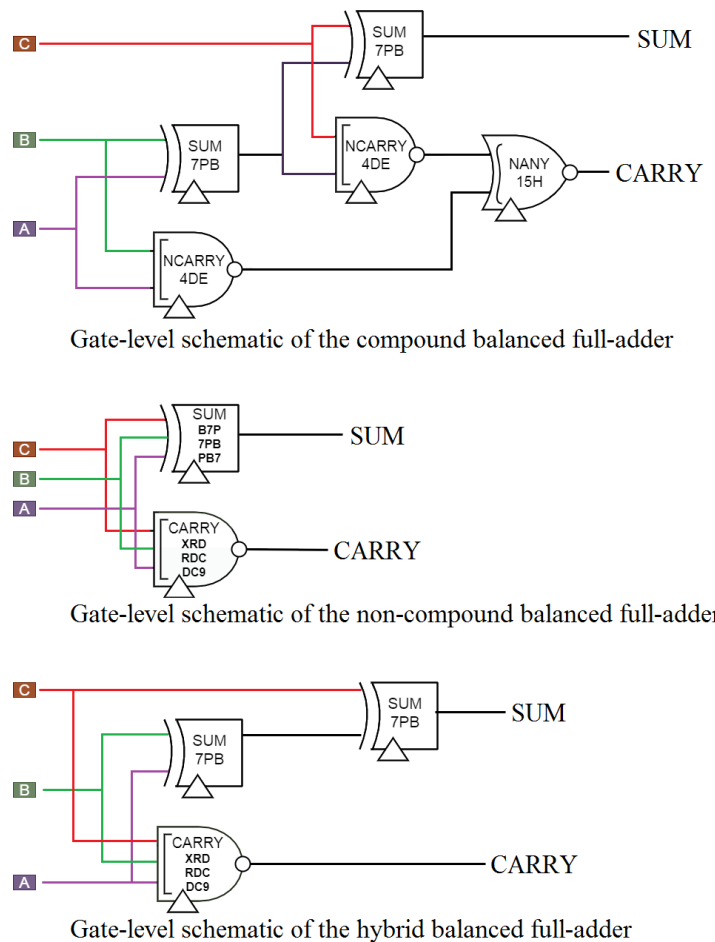


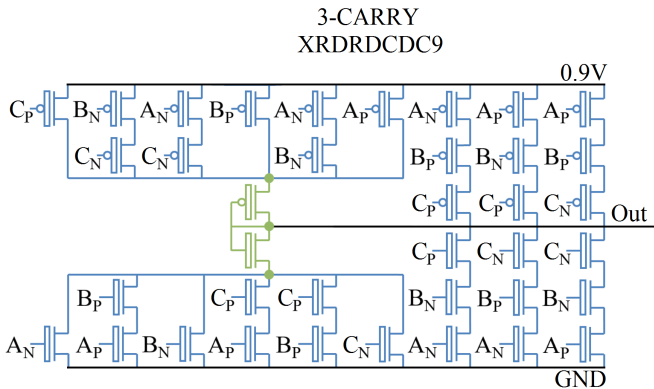
Figure 1. Three approaches for a balanced full-adder

Figure 2 shows the circuit schematic for the 3-carry circuit used in the hybrid 1-trit full-adder, with diameters 1.487 nm and 1.018 nm being depicted as blue and green respectively.

Table 3. Simulation results with 2fF load capacitor

Circuit	Transistors	Avg. power 500MHz	Avg. power 50MHz	Worst Delay	PDP 500MHz	PDP 50MHz
2-sum (7PB)	40 (32)	0.61 μ W	0.35 μ W	530ps	0.327e-15 J	0.185e-15 J
2-nary (4DE)	10	0.80 μ W	0.80 μ W	20ps	0.016e-15 J	0.016e-15 J
2-nary (15H)	18	0.33 μ W	0.32 μ W	40ps	0.013e-15 J	0.013e-15 J
3-sum (B7P7BPB7)	150 (138)	0.56 μ W	0.34 μ W	1530ps	0.856e-15 J	0.526e-15 J
3-carry (XRDRDCDC9)	50 (38)	0.87 μ W	0.82 μ W	30ps	0.026e-15 J	0.024e-15 J
(Lee et al., 2019) Unbalanced FA	106	(no data reported)	0.47 μ W	0.89ns	(no data reported)	0.421e-15 J
(Vudadha et al., 2018) Unbalanced FA	98	(no data reported)	0.43 μW*	1.57ns*	(no data reported)	0.667e-15 J*
Proposed Compound Balanced FA	118 (102)	2.73 μ W	2.29 μ W	0.55ns	1.44e-15 J	1.262e-15 J
Proposed Non-compound Balanced FA	188 (176)	1.67 μW	1.18 μ W	1.53ns	2.55e-15 J	1.805e-15 J
Proposed Hybrid Balanced FA	118 (102)	1.96 μ W	1.50 μ W	0.56ns	1.10e-15 J	0.840e-15 J

* see (Lee et al., 2019)

**Figure 2.** The balanced 3-operand carry function

5 Simulation results

The simulations were done in HSPICE, with the standard 32nm CNFET model technology from Stanford University. (Deng and Wong, 2007)

For the sake of these simulations, voltages below 200mV is considered "low", 250mV to 650mV is "middle", and above 700mV is "high".

Table 3 shows simulation results for some balanced functions, and compares three different circuit concepts of a balanced full-adder. The average current is measured at 500MHz and 50MHz. All measurements include the external PTI and NTI inverters of 2 transistors each where they are required. The transistor count is shown with and without the external inverters. A capacitive load of 2fF was put on the output to ground.

6 Discussion

The runtime performance of the synthesizer can be further optimized for > 7 arity. Under that condition circuit solutions can be found in reasonable time on standard hardware. Due to the sheer number of possible functions, only a minority of the circuit solutions were tested. However, all the tests produced the correct output values.

It should be possible to optimize the circuit solutions even further as we found by manual inspection. This is especially true for circuits with high arity functions. It is interesting to see that the performance of a 1-trit balanced ternary full-adder compared is comparable to an unbalanced version, commonly found in literature.

7 Conclusion

For up to 7 operands, a circuit of any 1-output ternary-valued function can be produced. We show that 3-operand functions can be implemented in circuits such as a 1-trit balanced full-adder, and may in some cases outperform a traditional 2-operand design strategy, as the hybrid full-adder was shown to outperform the compound full-adder in terms of power-delay-product (PDP) performance.

This paper has provided three contributions:

1. An open-source implementation for synthesis of n-ary ternary-valued CNTFET circuits (Risto, 2020).
2. An indexing system has been proposed which allows for any possible ternary-valued logic function to be referenced unambiguously.
3. A novel 3 operand, classical 2 operand, and a hybrid 1-trit balanced full-adder circuits have been simulated and compared with simulation results.

References

- J. Deng and H. . P. Wong. A compact spice model for carbon-nanotube field-effect transistors including nonidealities and its application—part i: Model of the intrinsic channel region. *IEEE Transactions on Electron Devices*, 54(12):3186–3194, 2007.
- Douglas W. Jones. The Ternary Manifesto · Heptavintimal encoding of ternary values, 2012. URL <http://homepage.divms.uiowa.edu/~jones/ternary/hept.shtml>.
- S. Kim, T. Lim, and S. Kang. An optimal gate design for the synthesis of ternary logic circuits. In *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 476–481, 2018.
- S. Lee, S. Kim, and S. Kang. Ternary logic synthesis with modified quine-mccluskey algorithm. In *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL)*, pages 158–163, 2019.
- Halvor Nybø Risto. Automated synthesis of netlists for ternary-valued n-ary logic functions in cntfet circuits, September 2020. URL <https://doi.org/10.5281/zenodo.4015574>.
- C. Vudadha, A. Surya, S. Agrawal, and M. B. Srinivas. Synthesis of ternary logic circuits using 2:1 multiplexers. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(12):4313–4325, 2018.



Post-Binary Robotics: Using memristors with ternary states for robotics control

**Not available online due to publisher
restrictions**

D

Ternary computing; The future of IoT?

TERNARY COMPUTING; THE FUTURE OF IOT?

Henning Gundersen, Steven Bos
 Department of Science and Industry systems,
 University of South-Eastern Norway
henning.gundersen@usn.no steven.bos@usn.no

ABSTRACT

Ternary logic, in which the number of discrete logic levels are restricted to three, has been a subject to excessive research over several years. In this position paper we discuss advantages and consider the future impact for IoT devices in seven categories: computational power, communication, compression, comprehension, cyber-security, design complexity and energy consumption.

INTRODUCTION

The use of CMOS-transistors has forced both hardware and software developers to use binary solutions as these transistors inherently have two stable states; "on" or "off". Between these two stable states is an undesired analog state of behavior which is designed to be as short as possible. Other bases have been used for computation as well such as the 1958 ternary computer Setun (Brousentsov et al. 2002). In practice this computer could not make 3 stable states, but used a multiple of 2, namely 4 stable states and discarded one state for computation (Ware, 1960). By using balanced ternary numbers (-1,0,+1) it is possible to add both negative and positive numbers without using a sign bit. Note that this is purely a logical encoding with implications for logic gate design. At the signal level three positive voltage levels are still possible. Balanced signals around ground are also possible but require a dual power supply design.

Donald Knuth, a computer scientist known for his seminal work *The Art of Computer Programming* stated (Knuth, 1981):

"If it would have been possible to build reliable ternary architecture, everybody would be using it."

Recent advances on memristors (Chua, 1971) and CNTFETs (Lin & kim & Lombardi, 2011) now make it possible to design and fabricate reliable ternary hardware (Moaiyeri et al. 2013; Nancy Soliman et al. 2019).

Ternary logic vs. Multiple valued logic

In the last few decades Multiple-Valued logic (MVL) has been proposed as a possible substitute to binary logic. While binary logic is limited to only two states, "true" or "false", multiple-valued logic can replace these with finitely or infinitely numbers of values. A MVL system is defined as a system operating on a higher radix than two (Smith, 1988). A radix- n set has n elements, $\{0, 1, \dots, n-1\}$. The practicality of MVL depends on the accessibility of the devices constructed for MVL operations (Etiemble, 1992). The

engineering challenge is thus to fabricate devices that are able to switch between the different logical levels, and preferably be less complex (eg. with equal or less components, cost of fabrication, die area, power consumption and signal propagation delay) than their binary counterparts. Ternary logic is MVL compliant. However, it only uses three logic states, "0", "1" and "2". Higher radices give more complexity, so is there an optimal radix?

The radix economy

Historically, the radix economy of a number N was proposed as a cost metric to compute the optimal radix. This number is often the computer architecture, the largest number that the processor can handle such as $N = 16 \text{ bits} = 65536$. The metric uses the *rw-product* to estimate the cost or hardware complexity. Here r is the radix or amount of unique symbols. The w is the width of the word or amount of positions needed to encode a random n from N using the symbols available. For instance, the number 5 is encoded in binary as 101_2 , in unbalanced ternary as 12_3 and balanced ternary as $+0-3b$. Therefore, to encode the number 5, binary and balanced ternary needs 3 positions while unbalanced ternary needs just 2. This is a direct consequence of computers using a positional numbering system. The *rw-products* are respectively 6, 9 and 6. If $N=6$ we should compute the *rw-product* of the other possible n and average (assumed uniform) the number to be able to compare the radices. The derivation of the function *rw* gives a minimal point at 2.71828 (Hayes, 2001), where the radix and width are treated as continuous variables. This point is remarkably the napierian base (Appleyard, 1913). The closest discrete number to this minimal point is radix 3, hence it being a more optimal numbering system than base 2 (Hayes, 2001).

Although possibly the best metric we currently have, there are several points of critique one should be aware of. The first is related to r , which can also be interpreted as the amount of devices needed to store (eg. memristors, capacitors, transistors) and process (eg. transistors) all symbols. If we need more than $\log(3)/\log(2) * 100\% \approx 58.5\%$ devices (for a large N such as 2^{16} advantages discussed below might be void. This average number can thus be seen as the theoretical design overhead. Another point is related to the equal proportionality of the *rw-product*. One could interpret the r being more important as binary has a 70 year advantage in fabrication cost and functional efficiency (eg. caching, branch prediction) thus directly influencing the first point. In this work we assume that ternary is more efficient than binary because we expect the fabrication cost to go down and designs be more efficient when adopted. Since the

rw-product gives a sense of the amount of actions or transistor switches required to process n , we argue that the amount of transistors per area compared to binary is a good estimate when designing ternary chips.

TERNARY LOGIC AND IOT

Ternary computing has several advantages compared to binary computing of which are grouped in 7 categories below. As IoT devices have strict requirements with regards to power consumption, heat generation, footprints and costs, this class of devices will especially benefit from the transition to pure ternary or mixed binary-ternary signals.

Computation

Where does the 58.5 % information advantage of ternary over binary come from? To represent or store in a register a number N , less positions and thus storage devices are needed for ternary. When focusing on performance or logic operations on numbers less transistor switches are needed. What does this mean for computation? Let's assume we design an adder logic gate to add and subtract numbers. It belongs to the most fundamental functional units in a CPU and optimizing it results in various improvements across the whole CPU. If a 1 bit binary adder is constructed with 28 transistors, a ternary variant (with the same functionality) can use up to $28 \times 1.585 = 44$ transistors and still be more efficient. Since many of these adders exist in one design, this advantage scales rapidly. A 22 bit binary adder will use $22 \times 28 = 616$ transistors with a resolution of $2^{22} = 4194304$. A 14 trit ternary adder will use $14 \times 44 = 616$ transistors with a resolution of $3^{14} = 4782969$, so it can handle slightly larger numbers than a binary design for the same transistor count. Note that 14 trits is the closest discrete approximation of $2^{22} = 3^n$.

With an efficient synthesis tool (Lin & kim & Lombardi, 2011) it is possible to make optimal ternary designs that will lead to area benefits as shown in Fig. 1 where a larger digit size gives better advantage (S. Kim, 2018). IoT devices will benefit of more computational power on less area. They will also benefit from using the balanced ternary numbering system as the arithmetical computation will be faster. For subtraction operations 2's complement is not needed in a balanced numbering system as the same adder can do both adding and subtracting. In addition, you only need one logic gate (Gundersen, 2006) to perform a comparison for more, less and equal. This can be used in search tree structures to increase the speed for search operations.

Communication

In wireless communication today different modulation methods are used. For instance, in telecommunication and digital cable TV, a common method is Quadrature Amplitude Modulation (QAM) (Lajos L. Hanzo et al. 2004). Digital QAM uses a combination of Phase Shift Keying (PSK) and Amplitude Shift Keying (ASK) for information interchange. Quadrature refers to a 90° phase difference. Each combination of phase and amplitude, the *modulation state* and the *symbol*, represents a combination of two or more bits. This means that a multi-valued encoded signal is already used in communication.

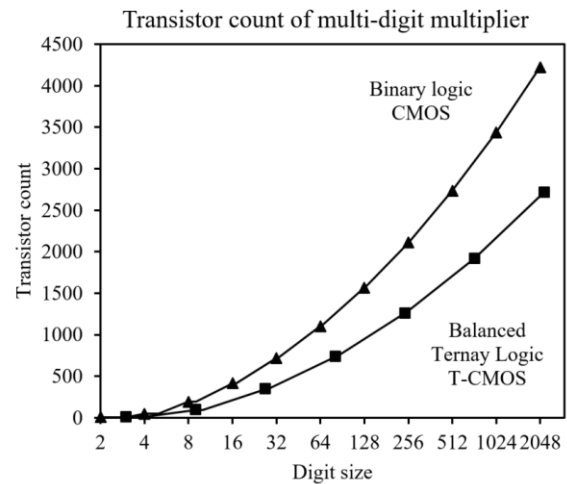


Fig. 1 Number of Transistor vs. digit size of a signed multiplier (S. Kim, 2018)

So why does the sending and receiving hardware have to encode and decode to binary values? For IoT devices less conversion is better as each conversion consumes energy. Instead of sending modulated binary signals, why not send modulated multi-valued signals? A 8 trit word compared to a 8 bit word can carry $3^8/2^8=25.6$ times more information which directly leads to higher communication speed and less delay which is an important issue in communication between servers and for internal network communication between IoT devices.

Consumption

Energy consumption is an issue for IoT devices due to their small footprint, battery reliance and limited cooling options. Often these devices have embedded various sensors which benefit from low power operation. In binary when flipping bits, a signal toggles between 0 and 1, the full length of a signal bandwidth. In balanced ternary the states are -1, 0 and +1. Four out of possible six state changes have half signal lengths, from -1 to 0 or +1 to 1 and reverse. Dynamic power constitutes of the transient power consumption and will thus be smaller in ternary for the same amount of transistors.

With CNTFETs it is possible to make energy efficient MVL-circuits (Ramzi A. Jaber et al. 2019). There are currently not built any ternary circuits using CNTFET-technology, but simulations show a low power-delay product consumption (Sunmean, 2018), which shows the potential of building power efficient ternary designs. Ternary Tunneling CMOS transistors on commercial wafers are able to show these consistent power advantages (Jeong et al. 2019). In 2019 a modern microprocessor built from complementary CNTFETs was built (Gage Hills et. al. 2019). They have overcome major intrinsic CNT fabrication process challenges and demonstrate it is possible to build a complete microprocessor using complementary CNTFETs. The same technology can also be used to build ternary computing hardware.

Compression

While discussing the origin of the information advantage we made a clear distinction between memory and logic, the

two fundamental building blocks of a computer. For memory this means less needed memory devices and can thus be seen as compression of information. For example, a 16 bit number has a resolution of $2^{16}=65536$ while a sixteen-trit number has a resolution of $3^{16}=43046721$. The closest approximation that can encapsulate all values of that resolution with ternary would be an eleven-trit number. This has resolution of $3^{11}=177147$ which means the handling of +170 % larger numbers and an effective reduction in devices of -31 %. In multimedia applications, this can lead to improved audio, photo and video compression algorithms.

For storing data, we showed that off-the-shelf memory resistors or memristors are excellent devices for low power and non-volatile (persistent) MVL data storage (Bos, 2020). They can be made as small as a few nm^2 (S. Pi et al. 2018). Other researchers have demonstrated that a single device can hold up to 6.5 bits of information (96 states) in a single memristor (Stathopoulos et al. 2017). Like in modern data communication, data storage in solid state drives already use non-binary techniques. Using capacitors, a single memory element can store 16 (4 bits) different charge levels/states (Toshiba, 2017). The trend is to cram more stable charge levels into devices, thus further increasing the information density or compression ratio. As binary encoding/decoding (multiples of 2) are used, read and write time will either increasingly take longer in serial or take more silicon space when done in parallel.

Comprehension

The world is analog and modelling it in binary result in known and unknown simplifications, especially in software engineering. The closer we can get to the analog equivalent, the less error prone our software can be. In a great exposé Charley Bey ¹ gives several examples of why writing software is made easier with ternary computers. For example, in low level programming it is important to know your data type when working with integers being either signed or unsigned. Compilers and instructions set require additional code to deal with negative numbers. Most microcontrollers have control registers with sign and sign overflow flags.

Using binary result in classical semantic problems as it must be either true or false. Maybe, partial or unknown are not valid possibilities. This illusion of simplicity effects our code structures and interpretation. For example, in binary if a bit is not 0 it must be the opposite, a 1. This is logically and functionally not always true. An accidental bit flip due to eg. cosmic radiation is hard to prevent. With a logically "unknown" middle state the change for a state inversion can be reduced. Binary control structures like if/else also force the programmer to disregard the option of more options. For humans modelling a state as unknown is common and gives the system (eg. the O/S or hardware) the possibility to notice this unknown, reflect and resolve it at a later time.

Cyber Security

Today we have smart homes and smart cities. We use more and more smart devices connected together in large IoT based networks. This makes us vulnerable for cyber-attacks. What can we do against hacking and other threats? One popular attack vector are the communication channels,

¹ Presentation at the C++Now Conference 2016. Slides and presentation at <https://youtu.be/gLJrOTFw6J0>

especially in an IoT network. Current technology uses binary solutions, but as we have been discussing in earlier sections, ternary has more information density. By making a 128 trit encryption key we can generate $3^{128}=1.18 \times 10^{61}$ different combinations, compared with a 128 bit encryption key which has $2^{128}=3.40 \times 10^{38}$ combinations, hence by using a ternary secure key (Bertrand Cambou et al. 2018) we decrease the possibilities for cracking the encrypted key substantially. In addition, we can use a ternary Physical Unclonable Function (PUF) circuit using CNTFETs (Zhengyang He et al. 2018; Nitish Kumar et al. 2019) and also implement Ternary Addressable PUF Generator (T-APG) utilized as a hardware layer for protection of databases (Mohammadinodoushan et al. 2019). By mixing binary with ternary signals for eg. an encryption scheme, logic states are obscured or lost, especially when reading with binary hardware. A hard problem for cryptographic circuits is hiding its output at the power level. With differential power analysis (DPA) the output of such circuits can be read. One of the solutions is by using "multi-bit data representations" or in other words MVL (Kocher et al. 2011).

Design Complexity

Design complexity exist on many levels. When designing logic gates, we can choose from $2^{(2^2)}=16$ different logic functions with two inputs and one output in binary. Examples are the AND, OR, XOR gates. In ternary we can make $3^{(3^3)}=19683$ logic functions for a similar two-input gate. While in binary each gate has found its usefulness, in ternary many of these logic gates are undiscovered. The amount of expressiveness is much greater, meaning that complex designs in binary can be a single gate solution in ternary such as a half adder gate, carry or data latch. A standard library of these logic gates would mean a reusable way to build larger circuits with more overview due to fewer individual logic gates.

Another level of design complexity is at the transistor geometry level, especially wire layout. These interconnects can take up to 90 % of the dynamic power consumption and often requires manual labor to do correctly (Magen et al. 2004). With smaller technology nodes (currently at 5nm) more transistors can be crammed, and thus more interconnects need to be connected. These interconnects are used in binary fashion while they inherently can handle analog signals. This inefficiency causes the switch to dissipate or charge all power in the wire before a transistor can switch. With ternary a design can save in both amount of interconnects, as there are less transistors needed, and in interconnect usage with on average smaller power transistions.

At the highest design level every chip is designed with power constraints. Cooling solution have not improved since the 2000's meaning that every technology node introduces more dark silicon. This term signifies that modern processors can have more transistors per cm^2 but that less of them can be active or must run at reduced speed at the same time to prevent overheating. It is estimated that barely 10 % of the transistors can be active in modern designs. With a higher utilization of interconnects and with less transistors needed

an increase of efficiency can be realized by switching to a mixed binary/ternary or full ternary design.

For IoT devices a simpler and smaller design mean more possibilities for form factors and faster development cycles. For example, without the need for a sign bit, die space is saved, logic gate functionality is made closer to how humans normally handle negative numbers and programming the gate with assembly becomes simpler.

CONCLUSION

In this position paper we gave an overview of 7 categories where a transition from pure binary to a mixed binary/ternary or pure ternary signal would be beneficial for general computing and IoT devices specifically due to their requirements. This trend is ongoing for a number of years with communication technology being non-binary since inception and solid-state storage solution being non-binary since the last decade. We find that MVL is often disguised in technology or adapted to binary just for the sake of compatibility. Now with new advances in logic processing using CNTFET, memristors and TCMOS, the missing piece is finally addressed. This finally enables the pursuit of post-binary solutions and practical experimentation. Especially the area of reduced noise margins that follow from a higher radix and the state stability at high frequencies is a topic that need more research.

Binary technology is not inferior. The 70 year legacy will require many fabrication and engineering challenges to be solved to bring ternary to the same level such that the assumptions in the radix economy hold. But when it is solved, IoT devices being low power, small and low cost will benefit the most as they reap all benefits discussed.

ACKNOWLEDGMENT

The authors would like to thank the Ternary Research Group and the SMART research group at the University of South-Eastern Norway for supportive and constructive feedback.

REFERENCES

Appleyard, R. (1913) The Direct Measurement of the Napierian Base, *Proc. Phys. Soc. London*, 26, page(s) 178-182.

Bertrand Cambou, Paul G. Flikkema, James Palme, Donald Telesca and Christopher Philabaum (2018) Can Ternary Computing Improve Information Assurance? *Cryptography* 2018, 2(1), 6.

Bos, S.; Gundersen, H.; Sanfilippo, F. (2020) uMemristorToolbox: Open source framework to control memristors in Unity for ternary applications, 2020 *IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL2020)*, pp. 212-217, doi: 10.1109/ISMVL49045.2020.000-3.

Brousentsov, N. P.; Maslov, S.P.; Ramil Alvarez, J.; Zhogolev, E. A. (2002). Development of ternary computers at Moscow State University, *Russian Virtual Computer Museum*.

Chua, L. (1971). Memristor - The missing circuit element, *IEEE Transactions on Circuit Theory Volume: 18*, Issue: 5, Page(s): 507-519, September 1971.

Etiemble, D. (1992) On the Performance of Multiple-valued Integrated Circuits: Past, Present, and Future. *Proceedings of the 22th international symposium on Multiple-valued logic*, page(s) 56-164.

Gage Hills, Christian Lau, Andrew Wright, Samuel Fuller, Mindy D. Bishop, Tathagata Srimani, Pritpal Kanhaiya, Rebecca Ho, Aya Amer, Yosi Stein, Denis Murphy, Arvind, Anantha Chandrakasan and Max M. Shulakerm. (2019) Modern microprocessor built from complementary carbon nanotube transistors, *Nature* 572, page(s) 595-602.

Gundersen H.; Berg, Y. (2006) A novel ternary more, less and equality circuit using recharged semi-floating gate devices 2006 *IEEE International Symposium on Circuits and Systems*, pp. 4 pp.-3172, doi: 10.1109/ISCAS.2006.1693298.

Hayes, B. (2001). Computing Science: Third Base. *American Scientist*, 89(6), page(s) 490-494.

Jeong, J.W., Choi, YE., Kim, WS., Park, JH., Kim, S., Shin, S., Lee, K., Chang, J., Kim, SJ. and Kin, K.R. (2019) Tunnelling-based ternary metal-oxide-semiconductor technology. *2019 Nature Electron* 2, pp. 307-312, doi: 10.1038/s41928-019-0272-8.

Knuth, D. (1981). The Art of Computer Programming, Second edition, Seminumerical Algorithm Vol 2, *Addison-Wesley Publishing Company*, page(s) 190-193.

Kocher, P.; Jaffe, J.; Jun, B. et al. (2011) Introduction to differential power analysis. *J. Cryptogr Eng* 1, 5-27. doi: 10.1007/s13389-011-0006-y

Lajos L. Hanzo; Soon Xin Ng; Thomas Keller; William Webb. (2004) Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo-Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems, *Wiley-IEEE Press*, pp.1-1, doi: 10.1109/9780470010594.part1.

Lin, S.; Kim, Y.; Lombardi, F. (2011). CNTFET-Based Design of Ternary Logic Gates and Arithmetic Circuits, *IEEE Transactions on Nanotechnology*, vol. 10, no. 2, page(s). 217-225, March 2011.

Magen, N.; Kolodny, A.; U. Weiser.; Hamir, N. (2004) Interconnect-power dissipation in a microprocessor, *Proceedings of the 2004 International Workshop on System Level Interconnect Prediction, ser. SLIP '04*. New York, USA: Association for Computing Machinery, pp. 7-13. doi: 10.1145/966747.966750

Moaiyeri, M.H., Mirzaee, R.F., Doostaregan, A., ET AL.: (2013) A universal method for designing low-power carbon nanotube FET-based multiple-valued logic circuits, *IET Comput. Digit. Tech.*, 2013, 7, (4), page(s) 167-181.

Mohammadinodoushan, M.; Cambou, B.; Philabaum, C.; Hely, D.; Booher, D. D. (2019) Implementation of Password Management System Using Ternary Addressable PUF Generator, *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1-8, doi:10.1109/SAHCN.2019.8824792

- Nancy Soliman, Mohammed E. Fouda , Abdullah G. Alharbi, Lobna A. Said, Ahmed H. Madian and Ahmed G. Radwan, (2019) Ternary Functions Design Using Memristive Threshold Logic, *IEEE Access* (Volume: 7,2019).
- Nitish Kumar; Jialuo Chen; Monodeep Kar; Suresh K. Sitaraman; Saibal Mukhopadhyay; Satish Kumar (2019) Multigated Carbon Nanotube Field Effect Transistors-Based Physically Unclonable Functions As Security Keys, *IEEE Internet of Things Journal* Volume: 6, Issue: 1, 2019.
- Ramzi A. Jaber, Abdalla Kassem, Ahmad M. El-Hajj, Lina A. El-Nimri, Ali Massoud Haidar.(2019) High-Performance and Energy-Efficient CNFET-Based Designs for Ternary Logic Circuits ,*IEEE Access PP(99):1-1, July 2019.*
- S. Kim, T. Lim and S. Kang,; Seokhyeong Kang,(2018) An optimal gate design for the synthesis of ternary logic circuits, *2018 IEEE 23rd Asia and South Pacific Design Automation Conference (ASP-DAC).*
- Smith, K. C. (1988) Multiple-Valued logic: A Tutorial and Appreciation. *IEEE Computers Vol. 21*, page(s) 17-27.
- S. Pi, C. Li, H. Jiang, W. Xia, H. Xin, J. J. Yang, and Q. Xia. (2018) Memristor crossbars with 4.5 terabits-per-inch-square density and two nanometer dimensions. *arXiv.org April 2018.*
- Stathopoulos, S.; Khiat, A.; Trapatseli, M. et al. (2017) Multibit memory operation of metal-oxide bi-layer memristors. *Sci Rep* 7, 17532. doi:10.1038/s41598-017-17785-1.
- Toshiba (2017) Toshiba Develops World's First 4-bit Per Cell QLC NAND Flash Memory. *www.techpowerup.com June 28, 2017*, Retrieved 19. December 2021.
- Ware, Willis H. (1960). Soviet Computer Technology— 1959.*Communications of the ACM, March 1960, Vol. 3 No. 3*, Page(s) 131-166.
- Zhengyang He ; Kai Ren ; Jiayan Chen ; Xinyi Dai ; Zhao Pan; Yuejun Zhang (2018) Design of Delayed Ternary PUF Circuit Based on CNFET, *2018 24th Asia-Pacific Conference on Communications (APCC).*

E

**High speed bi-directional binary-ternary
interface with CNTFETS**

HIGH SPEED BI-DIRECTIONAL BINARY-TERNARY INTERFACE WITH CNTFETS

Steven Bos, Halvor Nybø Risto, Henning Gundersen

Department of Science and Industry Systems,

University of South-Eastern Norway

steven.bos@usn.no halvor.n.risto@usn.no henning.gundersen@usn.no

ABSTRACT

The world is built on binary electronics, thus high speed and bi-directional radix conversion is needed to enable billions of binary devices to co-exist with non-binary ones. In this article we discuss a generic method for conversion between binary and ternary. A CNTFET implementation is given using a balanced ternary full adder. The implementation is simulated using HSPICE 2020 and is made open source. We demonstrate that nibble word conversion speeds of over 25 GB/s with power consumption of $97.8 \mu\text{W}$ are achievable with 1760 CNTFETs switching at 5 GHz.

INTRODUCTION

Computers work with binary signals and Boolean logic. Despite analog computers preceding them, discrete signal computers quickly became the standard. In the 1930's John Atanasoff, inventor of the first binary electronic computer, argued for a binary base. In his quest for a faster and more precise computer he realized the hardware to store and process symbols must be low cost, simple and compatible with the rest of the system (Atanasoff, 1940). In that landmark paper he proposed the building blocks of modern computers: capacitors as memory devices, triodes (transistors) as compute devices and implicitly the usage of Boolean algebra.

Modern memory devices no longer store just 2 states. Off-the-shelf solid state drives can have multiple bits stored in a single capacitor, driving down cost, energy consumption and increasing performance and information density (Gulak, 1998). Modern compute devices still use 2-state (binary) signals and Boolean logic. Multiple Valued Logic (MVL) or post-binary alternatives are being actively researched such as ternary CMOS (Jeong et al., 2019), memristor-transistors Zahoor et al., 2021), CNTFETs (Kim, Lim and Kang, 2018) as we approach the physical limits of computing with binary signals (Markov, 2014). Special interest is in computing with ternary signals as they are the closest discrete base to the optimum (Hurst, 1984). Higher bases are still interesting, but the highest average gain for processing large numbers (e.g. 16 bit or higher architectures) can be found when moving from base 2 to 3, which is $\log 3 / \log 2 \approx 58.5\%$.

The transition to MVL computers with increasingly higher bases is not instantaneous and requires interfacing with billions of existing (binary) devices. In this paper we focus on machine-machine interfacing at the signal level.

RELATED WORK

Limited work has been published on radix conversion, especially on binary to ternary and inverse converters with focus on circuit implementation. In (Arjmand et al., 2012) an unsigned ternary to unsigned binary converter for Quantum-dot Cellular Automata (QCA) is proposed. While this could generally be implemented in any ternary logic circuit, this is not a capacity efficient conversion method, as two bits are used to store one trit. In (Shahangian, Hosseini and Komleh, 2019), an algorithm for unsigned binary to unbalanced ternary conversion is proposed, with a simulated circuit implementation using CNTFETs. This method is efficient and scalable. In (Li, Morisue and Ogata, 1995) an unsigned binary to balanced ternary converter based on Josephson junctions was proposed. This specific circuit requires superconductors at low temperatures. They use a digit-relation matrix method. We choose to use this method and implement it using CNTFETS and balanced instead of unbalanced ternary. Balanced ternary encoding allows us to discard the sign bit, a source of complexity in both chip design and understanding. We are also aware of the work of Ashur Rafiev (Rafiev, 2011) using Reed-Muller expansions to synthesize unsigned binary to unbalanced ternary radix conversion circuits. This work uses binary CMOS technology for the hardware mapping and focuses on power balancing. Higher radix encoded signals are created with binary signals and is therefore suboptimal. Similarly, in (Iguchi, Sasao, Matsuura, 2006) binary coded ternary is used. The paper presents a generic method implementable in (binary) FPGA and focuses on unbalanced ternary, while we are interested in native ternary and balanced radix conversion.

PARALLEL ARITHMETIC CONVERSION METHOD

Digital computers use a positional numeral system. The value of number N follows from the value of a digit multiplied by a factor determined by the position of the digit. For example in the decimal system the 10 allowable digit values (known as symbols) are 0..9 while each position is a factor of 10 position. The number 16 can thus also be written as $1*10^1 + 6*10^0$. We can trivially convert this to another radix such as binary. For example, $2^4 = 16$ and is encoded in binary as 10000_2 or $1*2^4 + 0*2^3 + 0*2^2 + 0*2^1 + 0*2^0$ which will yield the same number. The astute reader will recognize that radix conversion can thus be seen as solving a system of linear equations. The system is a special case of the mixed

Table 4 15-bit to 10-trit conversion matrix

$t_0 =$	$+b_{14} - b_{13} + b_{12} - b_{11} + b_{10} - b_9 + b_8 - b_7 + b_6 - b_5 + b_4 - b_3 + b_2 - b_1 + b_0$
$t_1 =$	$+b_{14} + b_{13} - b_{11} - b_{10} + b_8 + b_7 - b_5 - b_4 + b_2 + b_1$
$t_2 =$	$-b_{14} + b_{13} - b_{12} + b_{11} - b_{10} + b_9 + b_8 - b_7 + b_6 + b_5 - b_4 + b_3$
$t_3 =$	$+b_{14} - b_{12} + b_{11} - b_{10} + b_9 - b_7 - b_6 + b_5 + b_4$
$t_4 =$	$+b_{14} - b_{13} + b_{11} + b_{10} - b_7 + b_6$
$t_5 =$	$+b_{14} + b_{13} - b_{12} - b_{11} + b_{10} - b_9 + b_8 + b_7$
$t_6 =$	$+b_{14} - b_{13} + b_{10} + b_9$
$t_7 =$	$+b_{14} + b_{13} - b_{12} + b_{11}$
$t_8 =$	$-b_{14} + b_{13} + b_{12}$
$t_9 =$	$+b_{14}$

Table 5 10-trit to 15-bit conversion matrix

$b_0 =$	$+t_9 + t_8 + t_7 + t_6 + t_5 + t_4 + t_3 + t_2 + t_1 + t_0$
$b_1 =$	$+t_9 + t_7 + t_5 + t_3 + t_1$
$b_2 =$	
$b_3 =$	$+t_7 + t_6 + t_3 + t_2$
$b_4 =$	$+t_6 + t_5 + t_4 + t_3$
$b_5 =$	$+t_9 + t_8 + t_5 + t_4$
$b_6 =$	$+t_9 + t_6 + t_5 + t_4$
$b_7 =$	$+t_9 + t_8 + t_7 + t_6 + t_5$
$b_8 =$	$+t_8$
$b_9 =$	$+t_6$
$b_{10} =$	$+t_9$
$b_{11} =$	$+t_9 + t_8 + t_7$
$b_{12} =$	$+t_8$
$b_{13} =$	
$b_{14} =$	$+t_9$

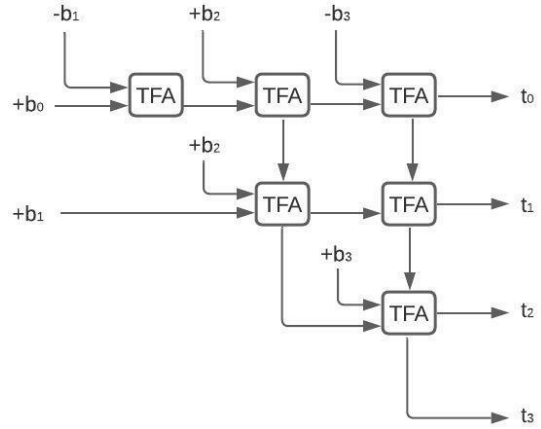


Fig. 1 A naive implementation of the 4-bit to 4-trit conversion matrix found in table 4 using only hybrid Balanced Ternary Full Adders (TFA) from (Risto, Bos, Gundersen, 2020). It covers the first columns and rows from the table

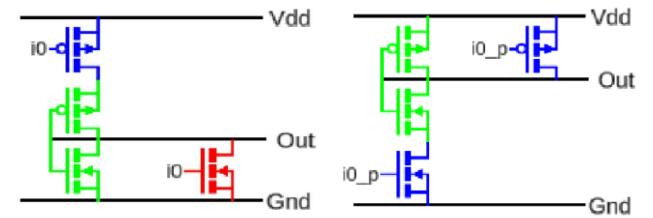


Fig. 2 Input circuitry to make binary signals compatible with balanced ternary. The color of the transistor denotes different CNTFET types with different activation thresholds for the middle value (Kim, Lim and Kang, 2018). (Left) The unary function 4 in heptavintimal notation has 4 transistors inverting every binary 1 to a ternary -1 (ground). (Right) The unary function R in heptavintimal notation has 6 transistors, 4 are shown. 2 more are needed for a Positive Ternary Inverter (PTI) in front of the input $i0_p$.

We present an early attempt to an optimized unsigned binary to balanced ternary radix converter in Fig. 4. The 6 TFA's used above each had 1 triadic (3 input) carry and 2 dyadic (2 input) sum logic gate, resulting in 6 triadic components, and 12 dyadic components. Our proposed radix converter has 2 triadic components and 9 dyadic components and includes input circuitry such as inverters. The total transistor count is 176, a reduction of 76% compared to the naive approach. All components are adaptations of the sum and carry logic gates. The two indices in the names indicate which parts of the conversion matrix they address e.g. index 00 means row 0, column 0. The postscript a or b denotes the 1st or 2nd sum logic gate in the hybrid TFA. Multiple variants are needed as input terms are sometimes negative or carry does not propagate resulting in various heptavintimal indices.

HSPICE SIMULATION

Fig. 3 shows the HSPICE output of the proposed radix converter from Fig 4. Table 6 shows the performance of the proposed design compared to the state-of-the-art. Note that the comparison is not completely fair as the radix converter in (Shahangian, Hosseini and Komleh, 2019) uses unbalanced ternary which generally requires fewer transistors.

Table 6 Simulation results at 5 GHz input frequency

Radix converter	Transistors	Avg. Power	Worst measured delay
4b4t (unbal.) ¹	151	64.41e-06 W	46.95ps
proposed 4b4t (bal.)	176	9.78e-06 W	80ps

¹ (Shahangian, Hosseini and Komleh, 2019)

DISCUSSION

Adders circuits are central in many functional units such as shift-add circuits and arithmetic logic units (ALU's), so further optimizing them is generally worthwhile. Logic gate design is an art, and many more optimizations can be made when studying radix conversion circuit at the transistor level. Our proposed radix converter is capable of high-speed processing and has low power consumption. It is comparable to the state of the art, with nearly matching transistor count and outperforming it in power consumption by -85%. In addition, the circuit can handle add/subtract arithmetic due to its balanced ternary nature. With little circuitry we can assign the most significant bit (MSB) to be the sign bit and automatically convert signed binary to balanced ternary. This makes the proposed radix converter a multi-purpose radix converter. The inverse, a ternary to binary radix converter can be constructed by using the same procedure but now based on Table 5. An optimized version would use ternary input-binary

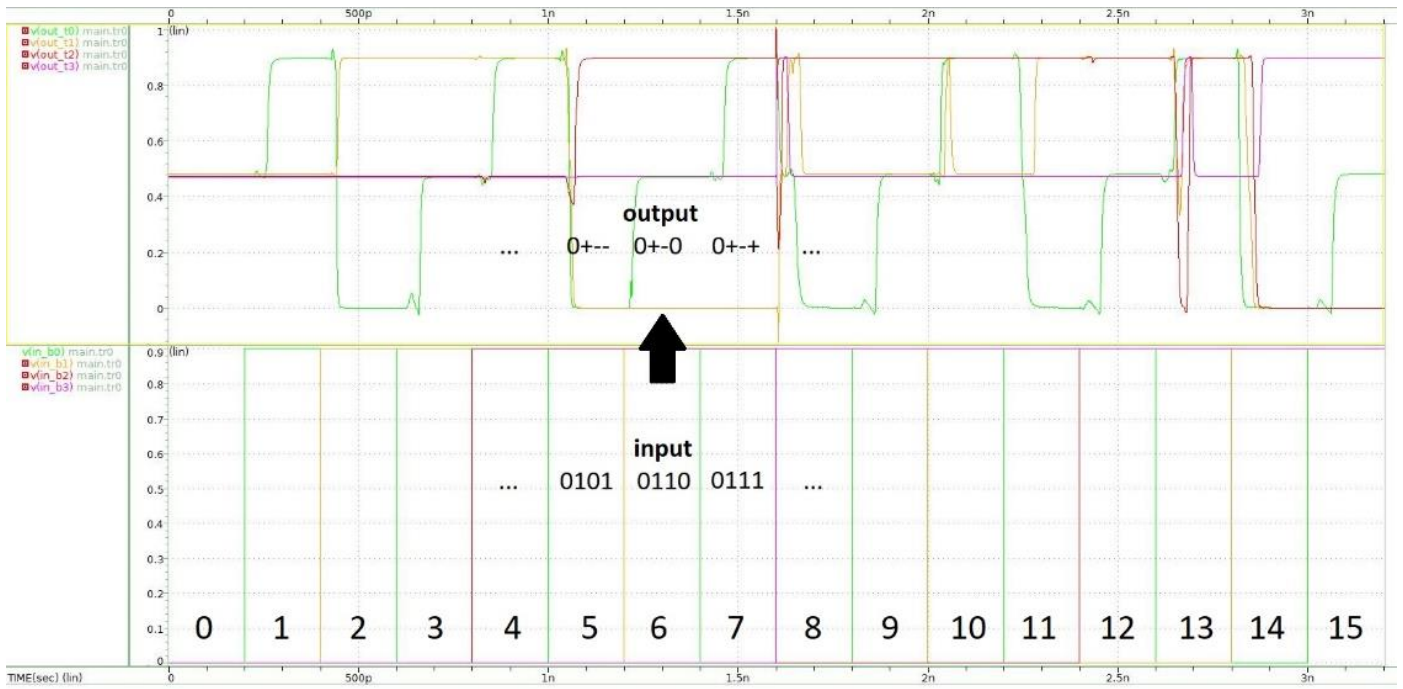


Fig. 3 (Top) Transient simulation of trit signals T0..T3, of the proposed 4b4t radix converter in fig. 4. The transition from decimal 7 to 8 determines the worst case delay of 80 ps (12.5 GHz). **(Bottom)** The binary inputs signals B0..B3 follow a 200 ps (5GHz) cadence going from 0000 to 1111 (decimal 0 to decimal 15).

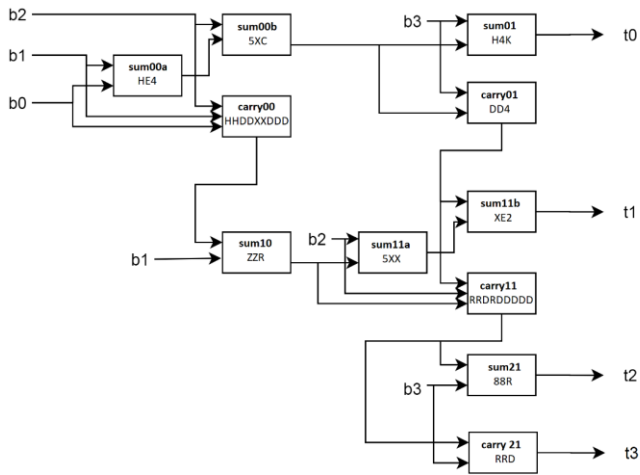


Fig. 4 Block schematic of the proposed unsigned 4 bit to balanced 4 trit radix converter. The 11 CTNfet circuits schematics of the logic gates are available (Bos and Risto, 2021).

output sum logic gates and ternary carry logic gates. In combination with a 1:2 multiplexer a single bi-directional circuit can be constructed where the multiplexer selects which conversion circuit should be active.

CONCLUSION

A generic look-up table (LUT) method to generate and optimize signed binary and balanced ternary radix conversion matrices have been presented. Simulation data

and netlist files of the 11 logic gates are made open source (Bos and Risto, 2021). The method can easily be extended to 64 bits or higher. A 4-bit version has been simulated in HSPICE. This circuit can be connected in parallel if we consider words to be 4 bits (one nibble). This means data can be converted at a rate of $9.78 \mu\text{W}$ and 176 transistors per 2.5 GB/s when the input frequency is 5 GHz. For example a 25GB/s radix converter would have a power consumption of $97.8 \mu\text{W}$ and component count of 1760 transistors, ideal for data heavy but energy efficient multi-radix IoT devices. Higher frequency operation should be possible as the worst case delay is 80 ps, about 12.5 GHz instead of the current 5 GHz.

FUTURE WORK

As CNTFET circuit fabrication is currently only possibly in highly specialized fabs, research is needed to investigate if radix converters can also be realized with standard CMOS technology (Morozov, Pilipko and Korotkov, 2009). With carefully matched off-the-shelf MOSFETs stable middle voltage levels can be realized. We were able to correctly simulate and prototype this on a breadboard. By using standard CMOS fabrication, price points of these conversion chips can become low, speeding up the transition from pure binary to mixed-radix circuits.

ACKNOWLEDGMENT

We thank the members of the ternary research group for their feedback on this paper.

REFERENCES

- Arjmand, M.M., Soryani, M., Navi, K. and Tehrani, M.A. (2012). A novel ternary-to-binary converter in quantum-dot cellular automata. *2012 IEEE Computer Society Annual Symposium on VLSI*. pp. 147–152.
- Atanasoff, J.V. (1940). Computing Machine for the Solution of large Systems of Linear Algebraic Equations. *Origins of Digital Computers: Selected Papers, Springer-Verlag, Berlin and Heidelberg, 1982*, pp. 315–336.
- Bos, S., and Risto, H.N. (2021). High speed bi-directional binary-ternary interface with CNTFETS. [Computer software] github.com/aiunderstand/MixedRadixConversion
- Gulak, P.G. (1998). A review of multiple-valued memory technology. *Proceedings 1998 28th IEEE International Symposium on Multiple-Valued-Logic*, pp. 222–231.
- Hurst, S.L. (1984). Multiple-valued logic - its status and its future. *IEEE Transactions on Computers*, 33(12). pp. 1160–1179.
- Iguchi, Y., Sasao, T. and Matsuura, M. (2006). On designs of radix converters using arithmetic decompositions. *IEEE International Symposium on Multiple-Valued Logic*. pp. 1 – 6.
- Jeong, J.W., Choi, Y.-E., Kim, W.-S., Park, J.-H., Kim, S., Shin, S., Lee, K., Chang, J., Kim, S.-J. and Kim, K. (2019). Tunnelling-based ternary metal-oxide-semiconductor technology. *Nature Electronics*, vol. 2, pp. 307-312.
- Kim, S., Lim, T., and Kang, S. (2018). An optimal gate design for the synthesis of ternary logic circuits. *23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*. pp. 476–481.
- Li, F.-Q., Morisue, M., and Ogata, T. (1995). A proposal of josephson binary-to-ternary converter. *IEEE Transactions on Applied Superconductivity*, 5(2), pp. 2632–2635.
- Markov, I. (2014). Limits on fundamental limits to computation. *Nature*, 512. pp. 147–54.
- Morozov, D.V., Pilipko, M.M. and Korotkov, A.S. (2009). The design of ternary logic units on the basis of the standard MOS technology. *Russian Microelectronics*, 38(3). p. 206.
- Rafiev, A. (2011). Mixed radix design flow for security applications. *Ph.D. dissertation, Newcastle University*.
- Risto, H.N., Bos, S. and Gundersen, H. (2020). Automated synthesis of netlists for ternary-valued n-ary logic functions in CNTFET circuits. *Linköping University Press in Linköping Electronic Conference Proceedings*, 176. pp. 483–485.
- Shahangian, M., Hosseini, S.A., and Pishgar Komleh, S.H. (2019) Design of a multi-digit binary-to-ternary converter based on cntfets. *Circuits, Systems, and Signal Processing*, 38(6). pp. 2544–2563.
- Zahoor, F., Hussin, F.A., Khanday, F.A., Ahmad, M.R., Mohd Nawi, I., Ooi, C.Y. and Rokhani, F. Z. (2021). Carbon nanotube field effect transistor (CNTFET) and resistive random access memory (RRAM) based ternary combinational logic circuits. *Electronics*, 10(1), 79.

F

Beyond CMOS: Ternary and mixed radix CNTFET circuit design, simulation and verification

**Not available online due to publisher
restrictions**

G

Additional material

G.1 Continuous-time and discrete-time signals

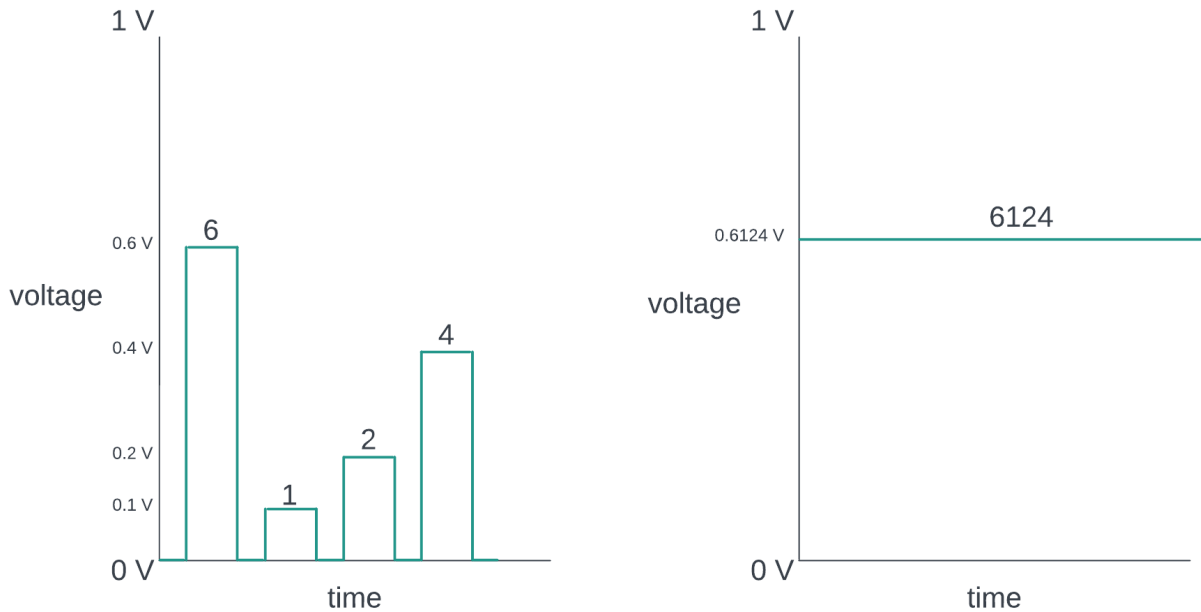


Figure G.1: Example comparison of a decimal (radix-10) digital signal to an analog signal for the number 6124. Four clock pulses are needed to convey the encoded number to a single output (serial transmission). Analog computers encode information in a continuous range of states.

McClellan et al. [326] defines signals to be "*patterns of variations that represent or encode information*" and that virtually all signals originate as continuous-time signals. These signals are *analogous* to another quantity such as voltage, temperature, pressure etc. It is often desired to transform continuous-time signals to discrete-time signals for the reasons below. The difference can be seen in Fig. G.1. Digital signals are processed sequentially and in discrete time and amplitude steps. Discretization of the time-domain is done by sampling at the Nyquist rate which allow perfect reconstruction of the analog signal. Discretization of the amplitude-domain is done by quantization, the mapping of a continuous value to a discrete value. Binary quantization (1-bit quantization) results in 2 discrete values and has the highest possible separation (noise immunity). The combination of sampling and quantization is known as analog-to-digital conversion (ADC). The inverse is digital-to-analog conversion (DAC). Discretization gave huge benefits for digital computers over analog computers: scalable precision at the cost of additional positions (such as transistors, memory cells), reliability through noise immunity and determinacy using a clock [20], [38], [304]. Analog computers at the time were large and their precision was determined by how precise the subdivision of the voltage signal could be made. With every subdivision the reliability also suffered as any noise in the signal affected the interpretation of the number. They had one major advantage though: computations were fast as they are done in parallel and in continuous-time.

G.2 Rebuttal of Buchholz' 9 arguments

Buchholz's 9 arguments on why radix-2 trumps radix-10 are shown in bold [12].

1. **Information Content.** The plot of the equation $E = \frac{\log_2(N)}{\#positions}$ assumes that radix-10 is implemented with bi-stable devices and encodings like binary coded decimals (BCD). The number of positions are clearly much less with higher radices when implemented with multi-stable devices or memory cells.
2. **Arithmetic Speed.** The same BCD implementation assumption is mentioned. Higher radices have fewer carry signals than binary as less positions need switching thus resulting in shorter delays.
3. **Numerical Data.** The argument that general purpose computers should not include multiple ALU's for different radices depends on the technology and application. Artificial intelligence loads have become general purpose computing and require less data movement and thus conversion. Higher radix arithmetic voids the need for conversion.
4. **Non-numeric Data.** Non-numerical information is radix independent. Higher radices allow more efficient encoding.
5. **Addresses.** A BCD implementation is assumed for memory addresses making it far less compact. It is clearly more compact with higher radices when implemented with multi-stable devices or memory cells
6. **Transformation.** A BCD implementation is assumed for performing data-to-address transformation (ALU operations) thus requiring handling of illegal states per BCD (the numbers 10 to 15). The balanced ternary adders and multipliers in Chapter 4 show that ALU operations can actually become less complex as they can handle negative numbers.
7. **Partitioning of Memory.** Binary has a clear advantage in terms of resolution when representing data. If data is inherently binary, then bi-stable devices encode this best. However if data is ternary then it would be inefficient to encode with bi-stable devices (25% loss per trit). Chapter 4 shows that a ternary ISA can be extremely compact as many useful signals are inherently ternary.
8. **Program Interpretation.** Direct ASCII encoding would be possible with a higher radix (such as radix-128) voiding the need for conversion. A alphanumeric subset (such as radix-36) is possible with a smaller radix.
9. **Other Number Bases.** Buchholz considers radix-10 as the only alternative to radix-2 as all other radices require conversion to radix-10 for human readable numerical data. He ignores the radix economy, the limits of radix-2 and reduced need for human readable data in domains like Internet-Of-Things (IoT).

G.3 A radix compatible form of the CMOS power equation

The formula for power consumption in CMOS at the transistor level is well known [8], [63], [64], [336] and shown in Equations G.1-G.5. These equations are extremely simplified as for example not all short-channel effects (SCE) are modelled. To control power each term is minimized. Most emphasis is put on dynamic power consumption as this is typically the dominant term. Dynamic power consumption estimation is difficult as switching activity (α) is data or use case dependent [337, p. 14].

$$P_{device} = P_{dynamic} + P_{static} \quad (G.1)$$

$$P_{dynamic} = P_{switching} + P_{short-circuit} \quad (G.2)$$

$$P_{switching} = \alpha * C * V_{dd} * V_{swing} * F_{clk} \quad (G.3)$$

$$P_{short-circuit} = \tau * \alpha * V_{dd} * I_{short} * F_{clk} \quad (G.4)$$

$$P_{static} = V * I_{leakage} = V * (I_{subthreshold} + I_{gateoxide}) \quad (G.5)$$

Where α is switching activity, C is capacitive load, F_{clk} is the clock frequency and τ is a short period of short circuit $\frac{t_{rise} + t_{fall}}{2}$ when doing a CMOS state transition. This short circuit happens due to transitions between the pull-up networks and pull-down networks and this a feature of CMOS. V_{swing} is often the same as V_{dd} , a full rail to rail swing, hence the common V^2 term seen in literature. For higher radix signals this is not the case, as the output can also be charged to for example $\frac{V_{dd}}{2}$, a half swing. A more generalized form of the CMOS power equation is given in [73]. A derivation of this generalized form can be found in [338, p. 24] and [335]. Capacitive load is a function of gate and interconnect capacitances and fanout [2, p. 40] and is roughly proportional to chip area [337, p. 23].

Equation G.3 and G.4 can be improved to better reflect the important role of switching activity α . A single switch cycle or pulse has two stages, charging (transition from $a- > b$, where $a > b$) and discharging (transition from $b- > a$, where $b > a$) of the load capacitances. Each stage dissipates half the energy. Secondly, switching activity α is a probability of state transitions ($0 \leq \alpha \leq 1$) since a transistor does not need to switch every clock cycle. It depends on the logic function [337, p. 12]. For example, a binary NOR gate with inputs $P[A = 1] = 0.5$ and $P[B = 1] = 0.5$ has 16 possible states [337, p. 14]. The

G.3 A radix compatible form of the CMOS power equation

probability of an actual state transition from 0 to 1, $P_{0 \rightarrow 1}$, given the input probabilities is $P_{0 \rightarrow 1} = P[Out = 0] * P[Out = 1] = 3/16$. Note that the reverse, $P_{1 \rightarrow 0}$ is also 3/16 and that the other states ($0 \rightarrow 0$ and $1 \rightarrow 1$) are not state transitions. This also explains why a binary clock tree is expensive (such as 30% of CPU power budget [8]) since it transitions twice every clock cycle. The amount of possible transitions depend on the radix r and can be calculates with $r * r - 1$.

$$P_{switching_{a \rightarrow b}} = \left(\sum_{i=0, j=1, i < j}^{\frac{r*(r-1)}{2}} P_{a_i \rightarrow b_j} \right) * 0.5 * C * V_{dd} * V_{swing} * F_{clk} \quad (G.6)$$

Ternary with $r=3$ results in 6 state transitions. The three positive transitions are shown in Eq. G.6. The summation term should be read as $P[0 \rightarrow 1] + P[0 \rightarrow 2] + P[1 \rightarrow 2]$. It also interesting to note that Tau is dependent on the transition (see Eq. G.7). In the case of binary transistors for a ternary transition, the transistor gate might already be at $\frac{V_{dd}}{2}$, thus requiring less time to reach V_{dd} or ground. The same effect can be seen with SRAM and DRAM that reduce latency with a $\frac{V_{dd}}{2}$ pre-charge circuit.

$$P_{short-circuit_{0 \rightarrow 1}} = \tau_{0 \rightarrow 1} * P_{0 \rightarrow 1} * V_{dd} * I_{short} * F_{clk} \quad (G.7)$$

At the chip level, equation G.1 depends on the amount of transistors n . Depending on the implementation, logic gates with a higher radix can be made with less transistors. The complete power equation shown in Eq. G.8 thus depends on the amount of transistors and their usage.

$$P_{chip} = \sum_{i=1}^n P_{device_i} \quad (G.8)$$

G.4 Using the radix economy argument

The analogy to see a truth table as uncompressed data with possible redundancies and dependence on practical device implementation will be important to show that the information limit of 58.5% (see Appendix G.7) and the radix economy does not apply for logic in practice. In CMOS two devices are used to encode binary (NMOS for 0, PMOS for 1) which is a radix-1 implementation. In practice optimal electrical circuits are not guaranteed even when the form is optimal [132, p. 522].

In Chapter 4 logic synthesis is discussed. Logic synthesis has two phases; synthesis and technology mapping. In the synthesis step a truth table is transformed to another representation that allows better compression using for example algebraic rules. In the technology mapping step the compressed representation is mapped to a circuit using available devices. Examples are bi-stable devices, bi-stable devices with different thresholds, tri-stable devices or a combination. Surprisingly, two similar phases are found in data compression theory [118, p. 6]; modelling and coding. In the modelling phase redundancy is extracted from the data and together with the data described compactly. In the coding phase this description is encoded in a radix, normally radix-2 as computers are binary. Extracting redundancy falls in the art of pattern finding which is "*an experimental science at best*" [118, p. 6]. Optimal methods, like the Quine-McCluskey method explode in runtime with large circuits such that non-optimal approximate methods are preferred. A compression result is evaluated in different ways such as how fast it was made, the compression ratio, the size, etc. For logic, the final size (area), the power consumption and delay are often the prime requirement, although synthesise speed becomes more relevant as the amount of transistors increases.

The more capabilities a synthesis algorithm has, such as knowledge about the technology mapping possibilities with various devices, the better logic functions can be modelled and exploited. Some patterns are better modelled in radix-3 than radix-2. A ternary signal like {forward, stop, backwards} can be modelled with 1 tri-stable transistor while in binary 2 bi-stable transistors are needed. In terms of device count 100% more transistors are needed which is more than the baseline of 58.5%. The overhead for binary is 25% as one state is not used (see Eq. G.10 in **Appendix G.5**). A similar ternary signal {forward, backwards, backwards} can be modelled with 1 bi-stable transistor with a shifted threshold. It is still a ternary signal at the gate as the middle value is $\frac{V_{DD}}{2}$ but the transistor response is binary. Implementing this with only binary signals would cost more since now the powerful binary input - binary output relation (Boolean logic/bi-stable transistor) is broken. The middle value can only be made by using another pin and encode this state with inefficiency (not using 1 state). This makes the radix-2 implementation cost more than the 58.5% baseline.

The radix economy does not model uncompressed data such as a logic gate well. The $\mathbf{r}^* \mathbf{w}$ cost function models information storage with r-stable state devices and not state

transitions mappings with r -stable state devices. This is not surprising considering the two phases in compression theory. Coding is a different step working under different assumptions than modelling. It might therefore be inconclusive to use the radix economy to argue which radix is optimal for logic, use the information limit (such as 58.5%) to judge design efficiency or use the information limit to predict how it scales. There are too many practical variables. As a rough approximation it might still be useful though. For example, a binary (3,2) ripple carry adder encodes its inputs and outputs without redundancy. A balanced ternary (4,2) parallel adder [202] does the same. For both scaling to higher number resolution means chaining them, which means that there is a direct correlation between pin encoding and transistor count of these two implementations. Pin encoding is number encoding, not number transformation (logic function) encoding. In this case a direct comparison between the binary and ternary solution is possible and the static information limit 1.585 can be used as a decision threshold. It might be that there exist a better binary design or a better ternary design with fewer transistors, so the comparison does not conclude anything on how close each one is to the optimal circuit given certain devices and design constraints.

G.5 Overhead calculation

The change of base (change of radix) equation $\log_b(s) = \frac{\log_a(s)}{\log_a(b)}$ shows mathematically how to express any number in some radix, such as the number of states s in radix-a, in another radix, radix-b. The relation between binary and ternary was described in Eq. 1.8 in Chapter 1.4.3. The number n and m indicate how many positions are needed in binary, respectively ternary to represent s . These positions are typically implemented as discrete devices such as memory cells or transistors whereby each device can read/write the number of symbols in the radix. The number of states s often represents a huge number like the bus architecture (such as 64-bit). As digital computers use discrete devices, rounding to integers is needed. The rounding operation is a floor operation such that the comparison is done against the maximum capacity of the rounded number. This is done such that the overhead is always positive. The rounding difference is the overhead and is calculated with one radix being precise and the other one being approximated. The overhead equations for binary and ternary are Eq. G.9 and Eq. G.10:

$$Overhead_{ternary}(m) = \frac{3^m - \lfloor 2^{\log_2 3^m} \rfloor}{3^m} * 100\%, m \in N. \quad (G.9)$$

$$Overhead_{binary}(n) = \frac{2^n - \lfloor 3^{\log_3 2^n} \rfloor}{2^n} * 100\%, n \in N. \quad (G.10)$$

Overhead are states that are never used but can be made with the given radix and positions. With specific pairs the overhead can be increasingly smaller, such as {5,8} (5.1%), {12,19} (1.3%), {53,84} (0.2%) and {306,485} (0.1%). For more pairs see [318], [319]. A nice visualisation of the overhead can be found in [282, p. 9]. The resulting pairs are "weird" device numbers and not very practical so one radix is typically chosen as the reference. If radix-2 is chosen as the reference, radix-3 will always be inefficient, especially for common architectures like 16, 32 or 64-bit (see [282, p. 9]. The smallest pair with reasonable small overhead (in this case disadvantage for binary) is the mentioned {5,8}. However since 5 is not a multiple of 3, logic design is more complex when scaling to more inputs. Ternary architectures should therefore ideally use radix-3 to be efficient.

Approximate overhead calculations are also found in literature [160] which use the ratio of discrete numbers in different bases to show its approximation to a baseline like $\log_2(3)$. For example, the {5,8} pair results in an approximate overhead *overhead – approximation* = $|\frac{8}{5} - \log_2(3)| \approx 0.015$ or 1.5%. This is significantly less than the 5.1 % found in the exact calculation and should thus be used with care.

G.6 Comparing baselines to 58.5% or to 63.1%

Various paper use different reference points with radix-3 or radix-2 as the baseline. For radix-3 the baseline is $\frac{\log(3)}{\log(2)} = \log_2(3) \approx 1.585$. This can be interpreted as "if ternary needs 100 tri-stable transistors to encode number s , binary would need 158.5 bi-stable transistors to encode the same information according to information entropy". Note that transistors can be substituted for pins or interconnects. For memory this interpretation can easily be found in practice: one tri-stable memory cell like a memristor or capacitor can differentiate (read/write) 3 stable states compared to two states in a bi-stable memory cell. Encoding 9 states (independent of balanced or unbalanced encoding) requires 2 tri-stable cells and $2 * 1.585 = 3.17$ bi-stable cells.

Instead of the baseline being radix-3 with $\log_2(3) \approx 1.585$, the baseline can also be radix-2 with $\frac{\log(2)}{\log(3)} = \log_3(2) \approx 0.631$. A nice table with this baseline can be found in [126]. This can be interpreted as "if binary needs 100 bi-stable transistors to encode number s , ternary would need 63.1 tri-stable transistors to encode the same information according to information entropy". Encoding 8 states needs 3 bi-stable devices and $3 * 0.631 = 1.893$ tri-stable devices. Since ternary architectures should use radix-3 for their architecture, 1.585 should be the baseline.

G.7 58.5% is an information limit, not a system limit

The information theoretical density difference of ternary compared to binary is $\log_2(3) \approx 1.585$. As discussed in Chapter 1.5.3 and **Appendix G.4** memory is fairly straightforward to compare but logic is difficult. The bigger picture is that memory and logic devices are integrated into a functional system. The system limit is a practical and often economical limit and is obviously more important than the information limit. If ternary memory devices can encode information at or above the information limit (due implementation with discrete components, see **Appendix G.4**) but cost twice as much then special circumstances are needed to justify it such as the potential to be cheaper. The same holds for other metrics like performance, area and power. Individually a tri-stable transistor might be superior, but it is at the system level what counts [138].

This makes comparison between radices even more difficult because the design and fabrication challenges are different for trivial circuits compared to dense, high performance circuits. Interconnect density becomes a problem when cramming transistors in a specific area with some target performance requirements. The realisation that transistor count is not the only factor of importance was already noted in 1955 [12, p. 6]. Many of the benefits of using a richer computer alphabet as discussed in Chapter 2 and paper D happen at the system level.

G.8 Ternary computers architectures from 2004-2022

- Alexander Shabarshin (2004) [306]. 3-nity alpha: 9-trit ternary computer architecture. The website also links to a java emulator and to the largest ternary computing forum (to my knowledge). The architecture has not been implemented but Shabashin has made ternary logic IC with CMOS in 2015 [307].
- Connelly et al. (2008) [308]. 3-Trit ternary computer architecture. The PDF report covers many ternary computing concepts and features a PCB implementation with off-the-shelf components.
- Viktor Lofgren (2008) [309]. Tunguska: 6-trit ternary computer architecture. The CPU is based on the (binary) MOS Technology 6502 processor. The source code includes an emulator, compiler and assembler.
- Thomas Leathers (2016) [310]. The Simple Balanced Ternary Computer Virtual Machine (SBTCVM) project written in python. A complete balanced ternary computer virtual machine with operating system, graphics capabilities, 18-trit CPU, ternary assembly and several ternary applications/games. The Github repository contains detailed documentation.
- Douglas W. Jones (2017) [311]. Trillium: 9-trit ternary computer architecture. The architecture is specification only. The website is a treasure trove for many ternary researcher.
- Dmitry V. Sokolov (2017) [313]. Triador: 3-trit balanced ternary computer architecture. This project features a custom PCB, emulator and off the shelf components to build this computer. Several parts were documented on Youtube.
- Wust et al. (2018) [320]. CMOS/Memristor MIPS ternary computer architecture. This paper simulated (with SystemC/Spectre) a MIPS based binary and ternary CPU and compared three filter algorithms. They report that ternary outperforms binary in PDP when the architecture is larger than 32-bit.
- Louis Duret-Robert (2019) [314]. SAP-1: 9-trit balanced ternary computer architecture. This "Simple-As-Possible" architecture by Albert Paul Malvino is made in both binary and ternary. The ternary computer is made with ternary verilog, a custom made HDL extension and features a emulator. The blog[315] also contains a 2-trit balanced ternary ALU implementation with off-the-shelf components. The ALU is compared to a binary equivalent in cost, energy consumption, performance and complexity.
- Cesare Di Mauro (2019) [316]. 3ISA: 20-trit ternary computer architecture. The ALU is part of the 24-trit computer project 5500FP by Claudio La Rosa [317]. This project is made with a custom PCB and off-the-shelf components.

Appendix G Additional material

- Reichenbach et al. (2021) [188]. RISC-V3: 64-bit CPU architecture with ternary datapath. Simulation shows that ternary arithmetic (ALU) can improve performance of a binary CPU with 130nm and 150nm open and commercial PDKs. A slow down is seen in logic and branch-intensive applications. They note that with larger architectures ternary has an increasing benefit [188, p. 43698]: "processors made with a ternary data path allow a much wider word width without having any impact on the clock frequency".
- Kam et al. (2022) [187]. ART-9: 9-trit RISC-based balanced ternary computer architecture. This paper described a pipeline converting regular RISC-V assembly to ternary assembly. The ternary assembly can be executed on a 150 MHz Intel Stratix-V FPGA with binary encoded ternary or emulated with multi-threshold CNTFET devices. The work uses the widely used Dhrystone benchmark and several others benchmarks to assess power, memory, ISA complexity and performance. Reported is that their 24 instruction ternary ISA outperforms modern ARMv6M and RV32i instruction sets.
- Gadgil et al. (2023) [287]. A 3-trit unbalanced ternary CNTFET CPU architecture. The 14 instruction ternary ISA is not based on RISC. No CPU benchmark was performed to validate performance.

G.9 Experimental 2-trit memristor results using uMemristorToolbox

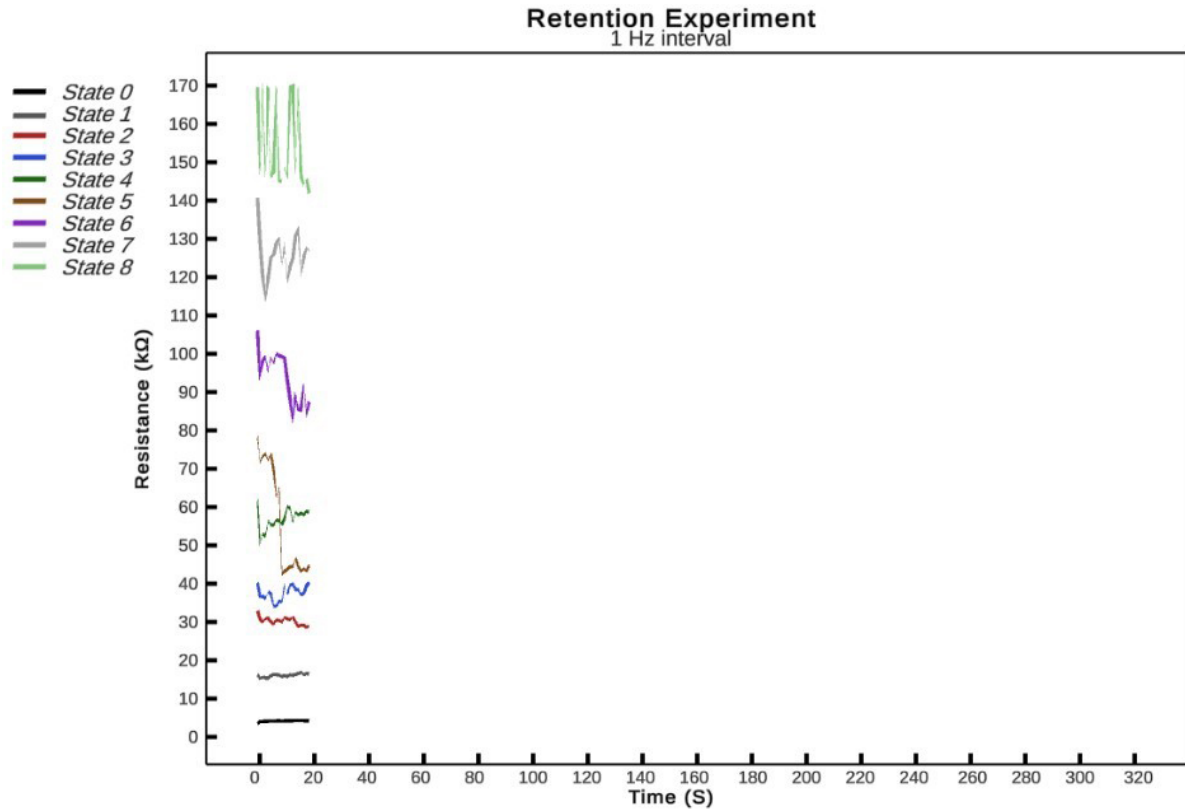


Figure G.2: Measurement of nine memristance levels using the Known PCB and uMemristortoolbox. Shown is an erroneous switching event after 5 seconds. A similar event is reported in **Paper A**. Image source: MSc thesis by Virk [241].

Appendix G Additional material

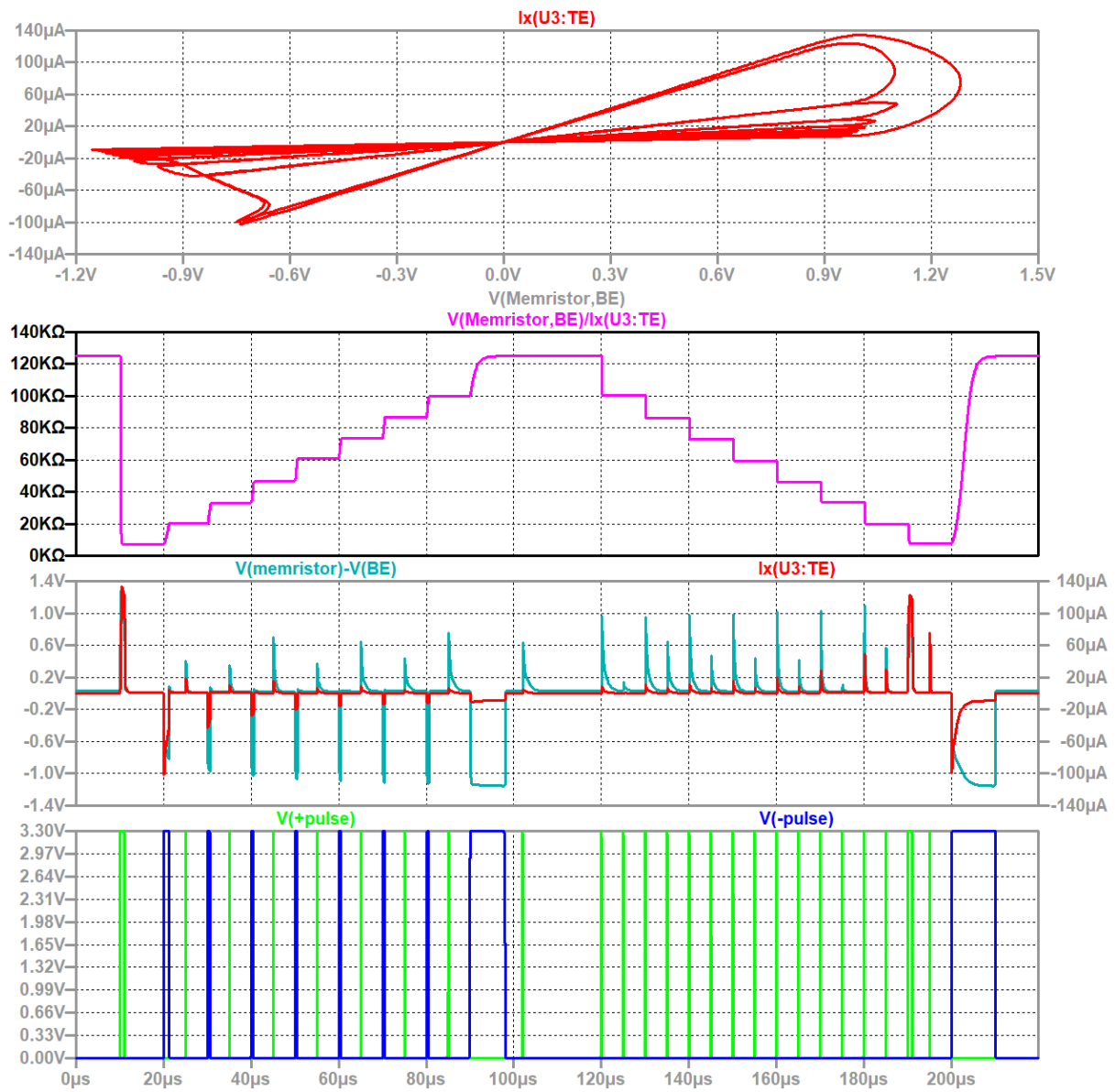


Figure G.3: Simulation of nine memristance levels using the multi-state RRAM development platform described in Chapter 3.3, LTspice and Knowm's Mean Metastable Switch Memristor Model [256]. The delta programming scheme and other details can be found in [241]. Image source: MSc thesis by Virk [241].

G.10 Multi-state RRAM development platform prototype

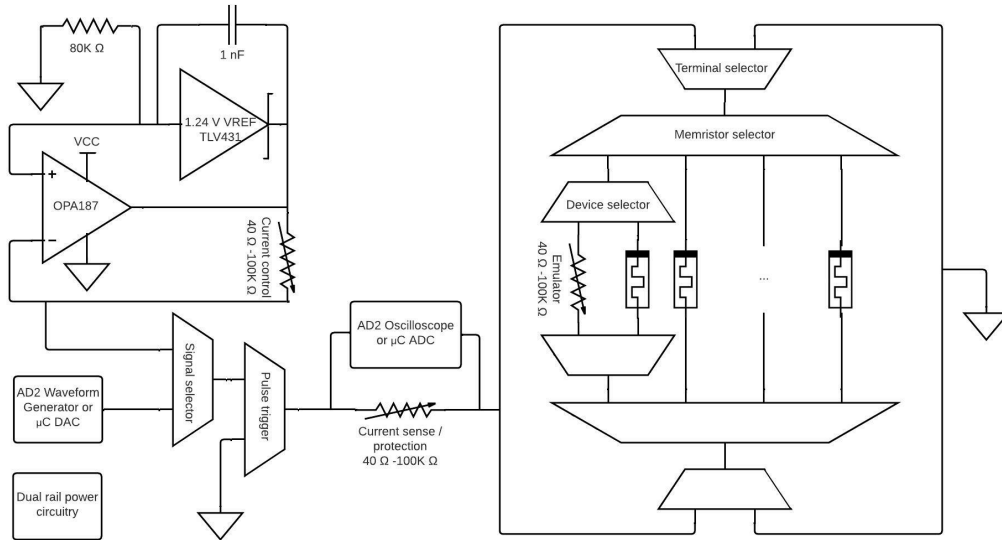


Figure G.4: A novel multi-state RRAM development platform with programmable voltage and current pulses and programmable resistor for memristor emulation.

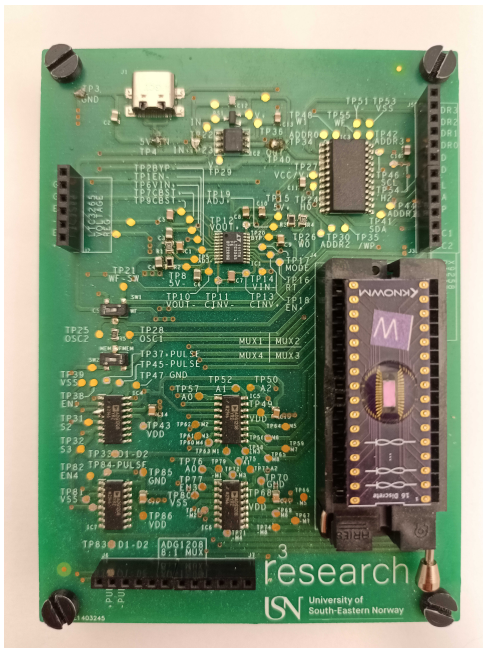


Figure G.5: First PCB implementation of the multi-state RRAM development platform. More details, see [241]



Figure G.6: Overview of used memristors. Shown are four different SDC models (W is Tungsten, C is carbon, Sn is Tin, Cr is Chromium with different characteristics in two different packages. See Knowm datasheet [238].

G.11 Getting started with MRCS

Installation

MRCS runs on three platforms; the web, Windows and the Unity editor. The online version of MRCS does not require installation and can be accessed via ternaryresearch.com [262] using a modern browser. The Windows version can be downloaded from the Github repository [261]. The executable does not require installation. The Unity editor version is designed for software developers. It requires the installation of Unity version 2023.1 or later with WebGL plugin and Visual Studio. After cloning the Github repository [261], the project can be opened via the Unity Hub and selecting *Open > Add project from disk*. The added project now points internally to the file main.unity which is the entry point containing the UI. After opening the project the project should look like Fig. 4.2 from Chapter 4 which includes a hierarchy and inspector view of each component.

User interface and user experience (UIUX)

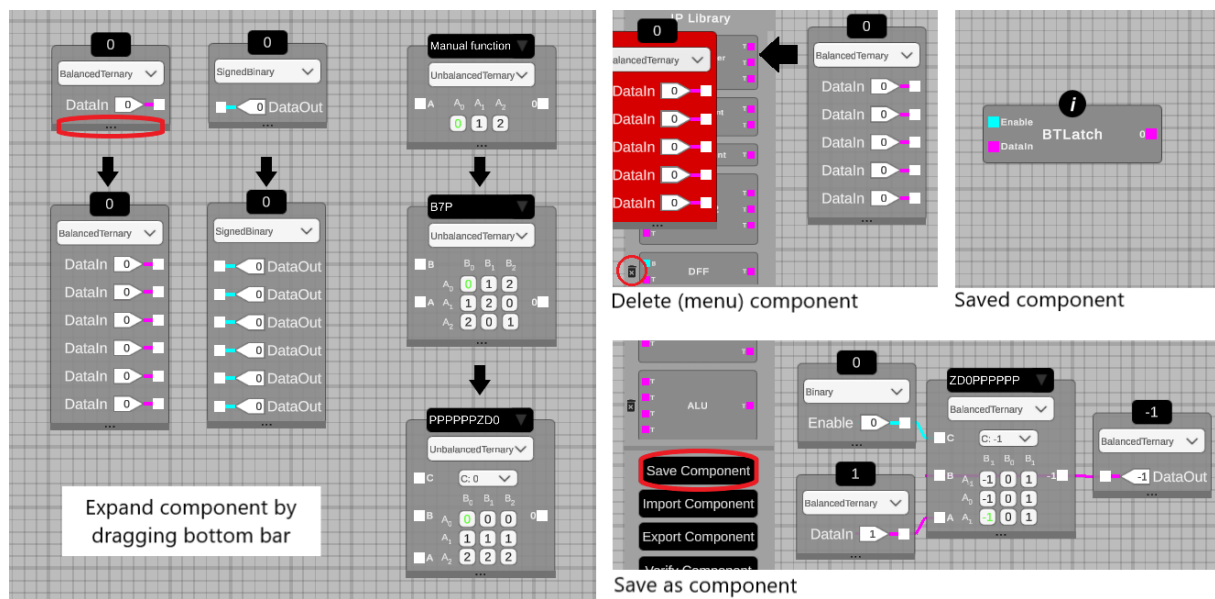


Figure G.7: **Left.** A standard component can be dragged from the drag&drop panel. Change radix with the dropdown box. Depending on the type of component, the arity can be changed by dragging the bottom bar. **Bottom-Right.** Components can be saved as a new component after attaching input and output to it and clicking "Save component". **Top-Right.** A saved component has an *i* button which reveal the statistics (such as the amount of transistors) and the schematic. **Middle.** Delete saved component by clicking the trash icon next to it. Canvas components can be deleted by dragging them left off-screen.

In **Paper F** the menu options and general interface of MRCS is shown. Figure 4 of **Paper F** contains a step-by-step guide to create a simple binary latch circuit in MRCS.

An animated video is shown in [340]. In Fig. G.7 the most important user interface interactions are shown which were not visualized in the paper. The interface of MRCS is still in flux and is likely to change in future versions. For example, a large IP library requires better organisation than the currently implemented scrollbar. The workspace has no zoom functionality and space for about 20 pins which limits creating large designs.

Persistent storage and file management

A saved component in MRCS has a set of files associated to it with a fixed file/folder structure. This structure is persistent for all three versions but has different locations. All data is stored locally, nothing is stored on remote servers. For the WebGL version, all saved components and settings are stored in cache [339]. It is important that the user exports the IP library regularly as a backup since the files are stored in browser cache. For Windows and the Unity editor version, the filepath is `C:\Users\[username]\AppData\LocalLow\USNKongsberg\MRCS`. The name of a saved component will be added to the settings file named *library.csv* such that it can appear in the IP library panel. The generated files associated to the saved component can be found in the folder `filepath\Generated\[componentname]`. This folder contains two folders; HSPICE and verilog. The HSPICE folder contains a set of .sp files and the Stanford CNTFET .lib file. It also contains a main.sp which is the top level interface. This file contains the CNTFET settings MRCS uses for ternary signals and includes a standard test pattern for simulation with Synopsys PrimeSim HSPICE version 2020+. The verilog folder contains a set of .v files and the file `[filename]_singlefile.v` which is the flattened version of the set of .v files in a single file.

G.12 MRCS Limitations

The tool is still experimental and has the following limitations:

Simulator

The event-based gate-level simulator uses discrete-time simulation steps (unit delays). Each gate that is triggered by an input change (event) can only evaluate all of its inputs once per simulation step. The new output is propagated to the next connected gate or output component after 1 unit delay. Wires have zero delays associated with them and are always driven to a known logic state. High-z or tri-state logic is not supported yet. Glitches that happen faster than a unit delay are not caught which might affect proper simulation of some designs such as edge detectors. Components can have unlimited fan-out and only one input which is both unrealistic and impractical for some circuits.

Sequential circuits with Verilog

Circuits with feedback loops such as latches and flip-flops have strict timing requirements (set-up and hold times) to be functionally correct in practice. Mixed signal simulation and post-layout verification of FPGA and ASIC tooling might show different behavior than the gate-level simulator of MRCS. Clock signals are not annotated in MRCS and the generated verilog code does not contain for example *always @(posedge clk)* blocks. The workaround used in tapeout-4 [266] was to design a reusable d-flip-flop (0tZD0PPPPPP) as presented in Fig. G.12 in **Appendix G.15** using two identical d-latches. The generated verilog of that latch was replaced with the code block shown in Fig. G.8. A future version of MRCS might introduce a setting where these blocks are automatically generated when feedback loops are used in digital designs.

```

module f_ZD0PPPPPP_bet (
    input wire[1:0] portC,
    input wire[1:0] portB,
    input wire[1:0] portA,
    output reg[1:0] out
);

    always @(posedge portC[1])
    out <=
        (portA == 2'b01) ? 2'b01 :
        (portA == 2'b10) ? 2'b10 :
        2'b11 ;
endmodule

```

Figure G.8: Verilog workaround for BCT with ternary d-latch. The generated *assign* block was replaced with the shown *always* block. The interface was unchanged.

Synthesis engine

The synthesis algorithm is not optimal. More efficient manually designed circuits are certainly possible as discussed in paper B and [282]. An example is the balanced ternary full adder discussed in section 4.4. The synthesized implementation reported in [155] requires 118T but can be reduced to 110T by wiring the output of the SUM gate to the CARRY gate. Extracting reusable components such as input inverters or whole logic functions and the usage of clever wiring requires a powerful ternary algebra. For Boolean functions no universal synthesis method has been discovered that produces optimal result in little runtime and for all types of logic [322]. Trivial situations where an inverter at the output cost less than inverting the inputs are not yet considered by the algorithm nor reusing shareable inverters of multiple sub-circuits. This optimization needs to be done manually. As can be seen in Fig. 4.6, optimizations are done locally in one of the four maps (networks), not across. This is a huge optimization opportunity.

The original C++ version of the algorithm allows higher arities than 3, but these are not yet supported in MRCS. Arity-4 circuits as basic building blocks make a lot of sense for ternary circuits. For example T-gates are arity-4. Arity-4 balanced ternary counters (4:2 compressors) are also optimal as 4 1-trit inputs have a total range from -4 to +4, which is exactly a 2-trit output.

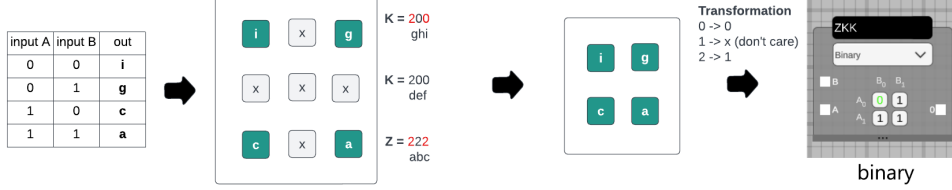
The synthesis engine adds metadata as comments to the .sp files when saving logic gates as a component. MRCS expects this format when loading components and restore the location and settings of the design. This makes editing the subcircuit .sp files manually error prone. The top-level module .sp file can be modified without worry.

Heptavintimal implementation

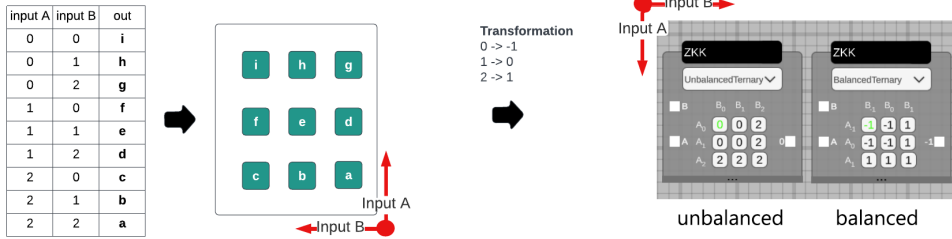
The largest quirk of MRCS is a coordinate system misalignment between the synthesis engine (bottom-right) and the user interface (top-left) as shown in Fig. G.9. In hindsight a top-left notation for both would enable reading the heptavintimal index row by row from a truth table starting from the top. Alignment is a breaking change which unfortunately requires new heptavintimal indexes for all truth tables discussed this thesis.

Appendix G Additional material

arity 2 binary



arity 2 ternary



arity 3 binary and ternary

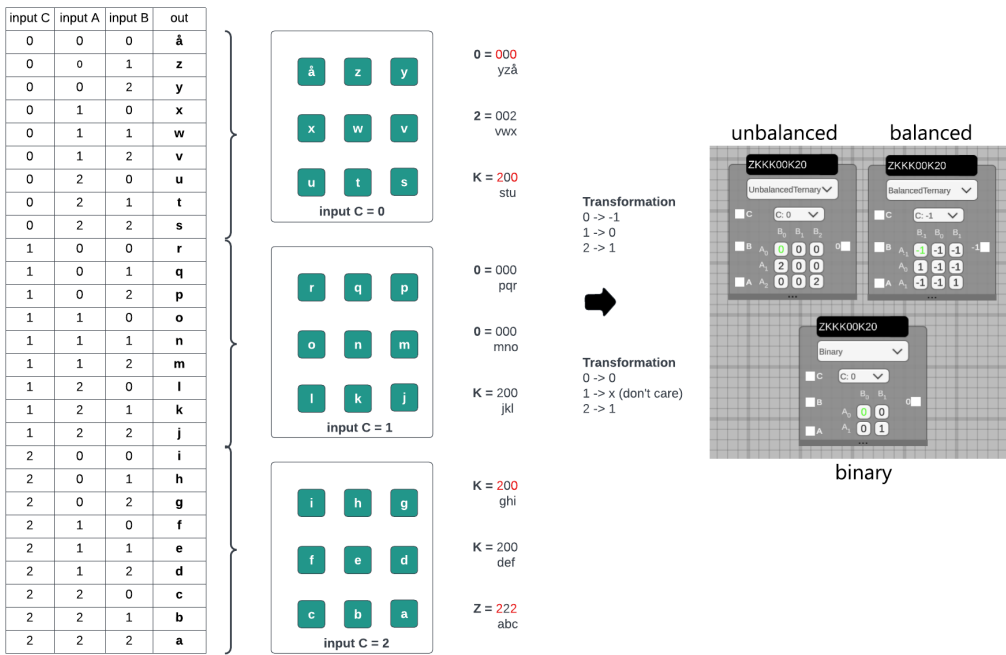


Figure G.9: Heptavintimal implementation in MRCS. Digits in red are binary truth table cells.

G.13 Ternary algebra

Early literature in the field of ternary algebra and switching theory researched mathematical optimizations without caring too much for circuit implementation [139], [166], [168], [170], [267]. Logic representation, algebra and circuit implementation are intimately related. Picking the right combination is needed to make ternary competitive to binary [141, p. 32]: *"In considering MVL algebra, we are most interested in those where the algebraic operations represent functions which have straightforward circuit implementations and which have sufficient representative power that effective circuit implementations can be constructed for general p-valued functions"*. In other words, switching devices require ternary logic representations [325] that can model them efficiently and ternary algebra that can use these models to construct optimal circuits according to an VLSI objective.

Literature on ternary computers from 1964 show that ternary algebra's that are currently heavily used such as chain-based Post algebra's [141, p. 38] are not optimal. For example, one of the most essential logic gates, the adder, is not optimal with this type of algebra without efficient implementation of the cycling gate [171, p. 38]: *"The major disadvantage of the post algebra is that it requires a large number of 'cycling' and 'or' units when functional circuits, say for addition or subtraction of numbers, are attempted."*

Chain-based Post algebra's for ternary logic is a functionally complete ternary input to binary output algebra [99]. This means that the algebra can map all possible ternary (M=3) input combinations of arity n to a binary valued output:

$$f: M^n \rightarrow \{0,1\} \quad (\text{G.11})$$

Eq. G.11 is fundamentally sub-optimal for ternary due to the ternary-to-binary encoding (see theorem and proof in [141, p. 31]). The three values of M form a single totally ordered chain $0 < 1 < 2$, hence the name chain-based Post algebra. In binary (M=2) a functionally complete algebra can be achieved with three operations: MIN, MAX and the literal operation NOT. For ternary input with binary output, the binary MIN and MAX operations can also be used. However an extension is needed for the NOT operation in the form of several literal operations (see proof in [99]). The short notation of Post algebra for ternary is given in Eq. G.12:

$$\langle M; +, \cdot, L; \{0,1\} \rangle \quad (\text{G.12})$$

Where M is the input set $\{0,1,2\}$, + the binary MAX operation, \cdot the binary MIN operation, L the literal operations and $\{0,1\}$ the output set.

Appendix G Additional material

Table G.1: Bi-stable subset of ternary unary functions. Green functions miss efficient implementations.

		LOW	NTI	MTI	PTI	\overline{PTI}	\overline{MTI}	\overline{NTI}	HIGH
		0t0	0t2	0t6	0t8	0tK	0tN	0tV	0tZ
IN	0	OFF	ON	OFF	ON	OFF	ON	OFF	ON
	1	OFF	OFF	ON	ON	OFF	OFF	ON	ON
	2	OFF	OFF	OFF	OFF	ON	ON	ON	ON

For chain-based Post algebra, a literal operation is defined as a unary (1 ternary input to 1 binary output) function of the type $0, 1, 2 \rightarrow 0, 1$ [99], [168]. Physically a binary signal operates at the extremes, so the mapping is better represented as $\{0, 1, 2\} \rightarrow \{0, 2\}$ [168]. In the case of binary logic circuits, the logical state $\{0, 2\}$ can be implemented with a bi-stable single threshold transistor with the operating regions $\{TransistorOFF, TransistorON\}$. In CMOS logic $\{0, 2\}$ is implemented with $\{NMOS, PMOS\}$. For ternary logic with bi-stable multi-threshold transistors more options exist. Multi-threshold transistors can be ON or OFF for the middle value. From the 27 unary functions in Table 2.2 from Chapter 2, only eight possibilities can be extracted that feature the set $\{0, 2\}$ or $\{OFF, ON\}$. This subset of literal operations are ideally suited for bi-stable transistors and is shown in Table G.1:

Two literals in Table G.1, LOW and HIGH are trivial and require only proper wiring to GND or VDD. From the 6 remaining only 4 (NTI , \overline{NTI} , PTI and \overline{PTI}) are commonly found in literature [225], [278], [280]. With multi-threshold CNTFET the V_{th} can be shifted to create OFF-ON-ON or OFF-OFF-ON behavior (and their inverses). The remaining two toggling gates are only found in literature as compositions of other gates and transistors [323], [155]. No names were found for the missing two literals in literature and are abtly named MTI ("middle toggling inverter") inline with the PTI and NTI naming. With complementary logic, the MTI should be made with 2 devices to handle ON and OFF just like the other literals. It should be noted that the hypothetical MTI still produces a binary signal. To create a ternary signal either two literals in voltage division mode are needed or a dual power source solution. In the latter case Table G.1 is duplicated with ON meaning ON_{VDD} or $ON_{\frac{VDD}{2}}$ and thus significantly increases the literal count in Post-algebras.

Much is uncertain how to fabricate the missing MTI 's without composition and is not the focus of this thesis. Literature on ternary device technologies are provided in Chapter 2.1. For example, one possible direction could be to create a more sophisticated device gate that only responds to the middle value such as resonating tunneling devices [324, p.358]: "Externally, resonant-tunneling compressed-function transistor circuits are binary, while internally certain circuit nodes are multivalued, to achieve the increased function per device.". Another possible direction could be based on NEMS (nano-electro-mechanical) such as rotating discs with two states (toggling) or three states (cycling).

G.14 Towards a ternary standard cell library

The amount of logic functions one can make is based on the radix r and arity n . The arity of a function is the number of inputs, argument or operands (synonymous terms). Assuming input and output in the same radix the amount of logic functions can be formulated as

$$\sum F_{r,n} = r^{r^n} \quad (\text{G.13})$$

In binary there are $2^{2^1} = 4$ monadic (1-ary) logic functions of the type $y = F(a)$ and $2^{2^2} = 16$ diadic (2-ary) logic functions of the type $y = F(a,b)$. Each function is named (such as AND, OR, BUF) but not all of them are equally useful. By substituting logic functions in each other higher arity functions are possible. For example triadic (3 input, 1 output) function can be made with 2 diadic ones $y = F(c, F(a,b))$. The proof for this can be found in [167]. Perhaps surprisingly, only a small set of functions is needed to construct all functions. This is called a functionally complete set or a set of universal gates. This similar to the concept of *basis* in linear algebra which spans a vector space with just a few linear independent vectors. This also implies that more than one basis or functionally complete set can be found which is proven in [167]. The usefulness of this set depends on the complexity (costs) of the implementation. For binary the absolute minimum set is just one gate: the NAND or NOR gate. By chaining ("compounding") and wiring the inputs together great flexibility with a single cell is gained at the cost of (much) more gates. In modern chip design standard cell libraries are often constructed for a certain technology node. These libraries often include the functionally complete set INVERT,AND,OR as individual gates and as a single 4-ary AndOrInvert (AIO) gate as well as commonly used 2-ary XOR gates. A standard cell library contains transistor layout variations of this set such as high density, high speed or high drive strength. No standard cell library for ternary exist yet. This section proposes a list of useful monadic (1 input), diadic (2 inputs) and triadic (3 inputs) logic functions that could evolve in a ternary standard cell library when ternary device technology becomes mature.

In ternary there are $3^{3^1} = 27$ monadic logic functions and a stellar $3^{3^2} = 19683$ diadic ones. Some of these been named and identified in literature, but most of them have not. Since binary is a subset of ternary, all binary functions can be represented as ternary functions. Unlike binary, ternary also has sets of universal gates which can be made with 3 unary functions STI, PTI, NTI. This set is comprised of three inverters: the standard ternary inverter (STI), positive ternary inverter (PTI) and negative ternary inverter (NTI). Like binary, many other functionally complete sets can be made but no consensus is found in literature which set is the most economical in terms of implementation as inherently ternary devices are not available. Other functionally complete sets with proof can be found in [168], [169]. An often cited universal set is the *ternary gate* or T-gate [167]. This

Appendix G Additional material

single gate is implemented as a multiplexer of 3 ternary functions with 1 ternary select signal. The term *ternary gate* should be reserved exclusively to refer to generic ternary logic gates. The term T-gate should be used when referring to them.

Tables G.2, G.3 and G.4 are a compilation of common single-gate logic functions to design mixed-radix circuits. Since the full list from arity 1 to arity 3 contains $27 + 19683 + 7625597484987 = 7625597504697$ logic functions, much is still to be discovered. The large majority of designs found in this thesis can be made with this set of building blocks.

Table G.2: Overview of useful arity-1 building blocks

Hepta Index	Name/Alias	Radix	Comment
2	INVERT	2	
K	BUFFER	2	
0	CONST_LOW	$\bar{3}$	
2	NTI	$\bar{3}$	DETECT_LOW
5	STI	$\bar{3}$	
6	MTI	$\bar{3}$	DETECT_MIDDLE
7	INCREMENT	$\bar{3}$	NEXT, SUCCESSOR
8	PTI	$\bar{3}$	
B	DECREMENT	$\bar{3}$	PREV, PREDECESSOR
C	CLAMP_DOWN	$\bar{3}$	
D	CONST_MIDDLE	$\bar{3}$	
K	\overline{PTI}	$\bar{3}$	DETECT_HIGH
N	\overline{MTI}	$\bar{3}$	$\overline{DETECT_MIDDLE}$
P	BUFFER	$\bar{3}$	
R	CLAMP_UP	$\bar{3}$	
V	\overline{NTI}	$\bar{3}$	
Z	CONST_HIGH	$\bar{3}$	

G.14 Towards a ternary standard cell library

Table G.3: Overview of useful arity-2 building blocks

Hepta Index	Name/Alias	Radix	Comment
20K	SUM	2	XOR
K02	NXOR	2	
K00	MIN	2	AND
RDC	MAX	2	OR
22Z	NMIN	2	NAND
002	NMAX	2	NOR

B7P	SUM	3	
C90	CONS	3	
EHZ	NCONS	3	
R99	ANY	3	
4HH	NANY	3	

7PB	SUM	$\bar{3}$	TRISHIFT (DEC,BUF,INC)
RDC	CONS	$\bar{3}$	CONSENSUS
4DE	NCONS	$\bar{3}$	
PC0	MIN	$\bar{3}$	Ternary AND
ZRP	MAX	$\bar{3}$	Ternary OR
045	NMIN	$\bar{3}$	
5EZ	NMAX	$\bar{3}$	
5DP	XOR	$\bar{3}$	
PD5	MULTIPLY	$\bar{3}$	DIVIDE (div/0 is 0)
PRZ	IMPLICATION	$\bar{3}$	Kleene Logic version
XP9	ANY	$\bar{3}$	
15H	NANY	$\bar{3}$	
H51	COMPARE	$\bar{3}$	MORE,LESS,EQUAL (MLE)
RD4	ENABLE	$\bar{3}$	ENABLE w/ binary
VP0	DESELECT	$\bar{3}$	A or B, with one being \overline{ENABLE}

Appendix G Additional material

Table G.4: Overview of useful arity-3 building blocks

Hepta Index	Name/Alias	Radix	Comment
KKKK00Z00	2:1 MUX	2	FE D-LATCH (feedback to B)
Z00K00KKK	2:1 MUX	2	RE D-LATCH (feedback to B)
K0200020K	SUM	2	XOR
ZKKK00K00	CARRY	2	
ZZZZKKZKK	MAX	2	OR
K00000000	MIN	2	AND

PPPPPPZD0	2:1 MUX	$\bar{3}$	FE D-LATCH (feedback to B)
ZD0PPPPPP	2:1 MUX	$\bar{3}$	RE D-LATCH (feedback to B)
ZD0DDDDPPP	2:1 TRIMUX	$\bar{3}$	Tristate with zero, not HZ
B7P7PBPB7	SUM	$\bar{3}$	
XRDRDCDC9	CARRY	$\bar{3}$	
ZZZZRRZRP	MAX	$\bar{3}$	
PC0CC0000	MIN	$\bar{3}$	

G.15 Combinatorial and sequential building blocks

In this section ternary and mixed-radix combinatorial and sequential building blocks are discussed that were developed in this thesis work. These HSPICE and FPGA verified building blocks are essential for building a ternary computer.

Ternary data-latch

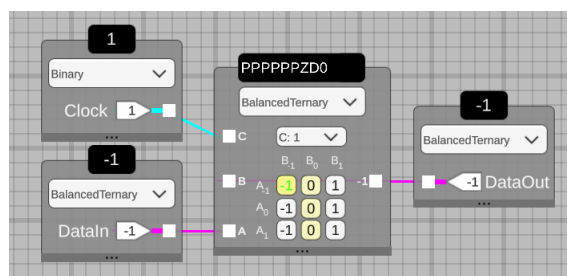


Figure G.10: 28T gated balanced ternary d-latch based on 2:1 MUX

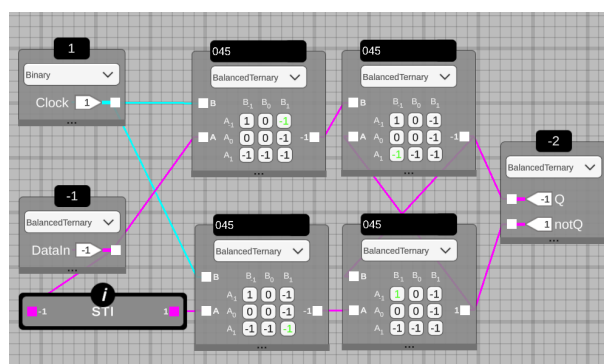


Figure G.11: 46T gated balanced ternary d-latch based on NMN

The latch is one of the most important memory elements. In Fig. G.10 a 28T level-controlled balanced ternary data-latch (d-latch) implementation is shown that was presented in paper F. The d-latch is effectively a 2:1 MUX with feedback from the output to input B. Compared to a naive binary CMOS implementation of a 2:1 MUX with 2 AND, 1 OR and 1 INV (=20T) the MRCS synthesized design is within the 58.5% margin. There exist many other types of d-latch designs such as cross-coupled inverters with transmission gates or gated Set-Reset latches using 4 NAND gates. These are often found in high density binary SRAM cells. The same topologies can be used for ternary SRAM [321], [332]. A NAND (NMN) ternary d-latch shown in Fig. G.11 costs 46T when made with MRCS and is not efficient. Both the NMN and MUX d-latch design cost many more transistors/area than a 8T d-latch design based on two 2T cross-coupled STI's and two 2T transmission gates [149]. Transistor count is not always the most important metric. The 20T ternary SRAM made with CNTFET reported in [321] has 85% better performance and PDP compared to ternary 6 transistor, 2 capacitor (6T2C) DRAM also made with CNTFET.

Ternary data-flip-flop

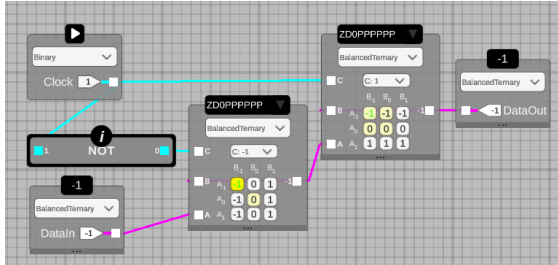


Figure G.12: 54T rising-edge master-slave configuration balanced ternary d-flip-flop

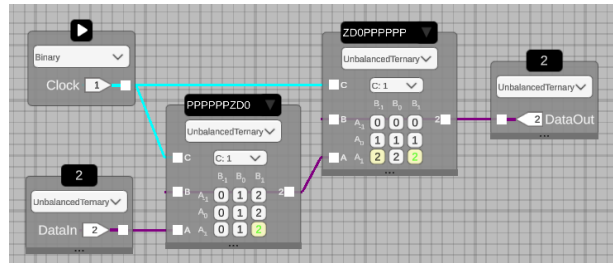


Figure G.13: 52T rising-edge master-slave configuration unbalanced ternary d-flip-flop

A plethora of binary flip-flop designs exist[331]. While the latch is a level controlled memory element, the flip-flop is designed to be edge-controlled. Contrary to binary's bi-stable flip-flops ternary's flip-flops are tri-stable. Sometimes tri-stable flip-flops are called flip-flap-flops [332]. This term should be avoided as it is confusing and the name becomes rather grotesque with higher radices. Ternary flip-flops can be either binary clocked or ternary clocked. This is important as in modern (synchronous) computers the clock tree network (CTN) is the always-on backbone that can consumes 30% of the CPU power budget [8]. The rising-edge master-slave configuration of a MUX based ternary flip-flop is discussed in paper F and shown in Fig. G.12. This configuration consists of two ternary d-latches with a binary inverter. The inverter can be integrated (reducing 2T) by flipping the heptavintimal index of the second latch from 0tPPPPPPZD0 to 0tZD0PPPPPP. By reversing the latches the flip-flop becomes either a rising-edge or falling-edge flip-flop. Note that in Fig. G.12 a balanced ternary version of the D-flip-flop is shown while in Fig. G.13 an unbalanced ternary version of the d-flip-flop is shown. Both have identical implementation which is not always the case (such as the carry function).

Ternary data-flip-flop with double/quad data rate

The d-flip-flop design in Fig. G.12 is a single data rate (SDR) or single edge d-flip-flop. It only transitions with a rising-edge. If tighter timing is permissible, then double data rate (DDR) d-flip-flops or both rising and falling edge triggered d-flip-flops make sense. A novel 76T DDR ternary d-flip-flop design is shown in Fig. G.14 which uses both edges of the binary clock. The design is based on the latches 0pPPPZD0PPP and 0pZD0PPPZD0. With a ternary clock the transitions can be increased to 4 edges. Quad data rate or quad edge triggered d-flip-flops require even tighter timing. The power benefits of QDD memory versus SDD is substantial as the voltage swings are smaller in both the CDN and d-flip-flops. Kim et al. [330] show that QDD ternary flip-flops are 31% more efficient

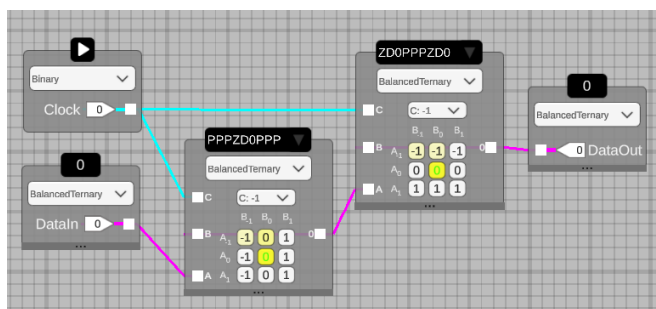


Figure G.14: 76T DDR master-slave configuration balanced ternary d-flip-flop

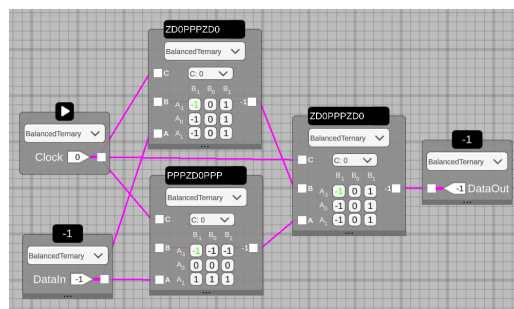


Figure G.15: 110T QDR master-slave configuration balanced ternary d-flip-flop

than SDD ternary flip-flop (only slightly higher delay) while the clock tree consumes 75% less power. A novel MUX-based 110T QDR ternary flip-flop design is shown in Fig. G.15. This design is based on the latches 0tPPPZD0PPP and 0tZD0PPPZD0 and one ternary 2:1 MUX. This MUX has the same heptavintimal index but has no feedback making it a combinatorial block. The top latch is responsible (active) for the edges -1 to 0 and 1 to 0 while the bottom latch is responsible for the edges 0 to 1 and 0 to -1. The wiring is important as the MUX responds to the active latch only during an edge. During level it "listens" to the inactive latch thus ignoring any changes.

Ternary register

The ternary register can be constructed with a mixture of binary and ternary gates. The *Write-enable* flag is a binary signal and can be made with a binary 6T AND (0tK00) or 4T NAND gate (0t22Z) when a binary clock is used. The memory element can be made with MUX-based ternary d-flip-flops shown in Fig.G.12. The *Read* flag is a binary signal while the output is ternary, thus requiring the usage of a ternary logic gate. Enabling the read flag allows the output to reflect the truth table output, else a constant zero is output. This is useful in combination with a DESELECT component (0tVP0) allowing multiple registers to be connected without using a transmission gate and bus architecture. The register design is thus a pure logic gate based design. Transmission gates however are far more efficient, costing just 2T instead of 16T for the *Read* flag. Transmission gates seem a better fit for this functionality in combination with higher radix circuits as transmission gates signals are analog in nature. It should be noted that Kim et al. [280] report that transmission gates "have worse power, speed, and noise margin than static gates for ternary". The ternary register is an example of a mixed-radix design, combining various binary and ternary signals which reduces the transistor count compared to a pure ternary implementation.

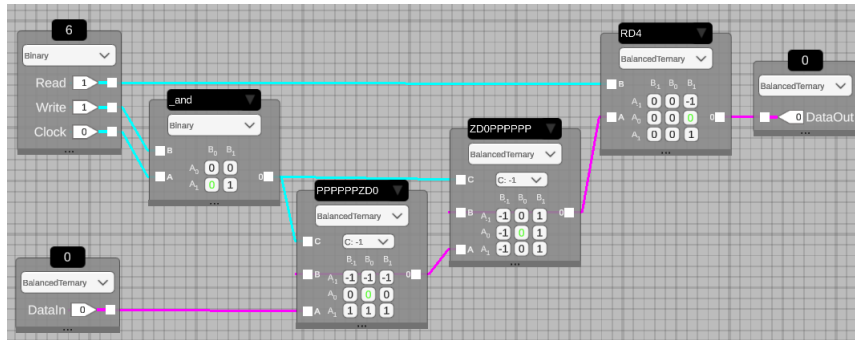


Figure G.16: 80T balanced ternary register

Ternary ROM/RAM

Registers give a blueprint to construct larger memory elements such as blocks of RAM or ROM. Typically application code and immutable data (constants) is stored in ROM while processed data is put in RAM. ROM is often a non-volatile memory array such as FLASH while processed data is stored in SRAM or DRAM and is lost after a power cycle. RAM/ROM have the same interface as registers but feature a DEMUX/MUX to select individual (blocks of) memory elements. The building blocks of both ROM and RAM are clusters of balanced ternary d-flip-flops and is shown in Fig. G.17 .

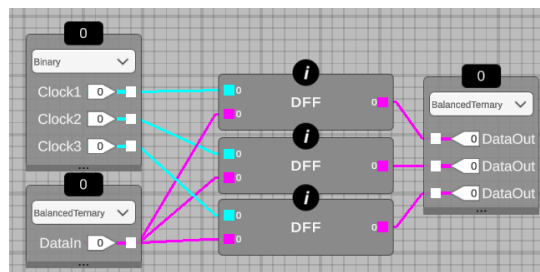


Figure G.17: RAM-3 implementation with three d-flip-flops

A 2x1 block of RAM is shown in Fig. G.18. The 2x1 RAM has 2-trit addresses meaning thus can refer to 9 unique addresses for both reading and writing. Each address refers to a single ternary d-flip-flops. By adding another 2x1 block of RAM in parallel a 2x2 RAM block can be made, thus expanding the content while keeping the address width the same. Parallel read actions are made possible by adding another MUX with two 2-trit register-source addresses (RsAddr and RsAddr2). Parallel write actions are slightly more complex as two write actions might want to update the same register with different data. In MRCS the RAM/ROM content is *uninitialized* at first and needs to be reset

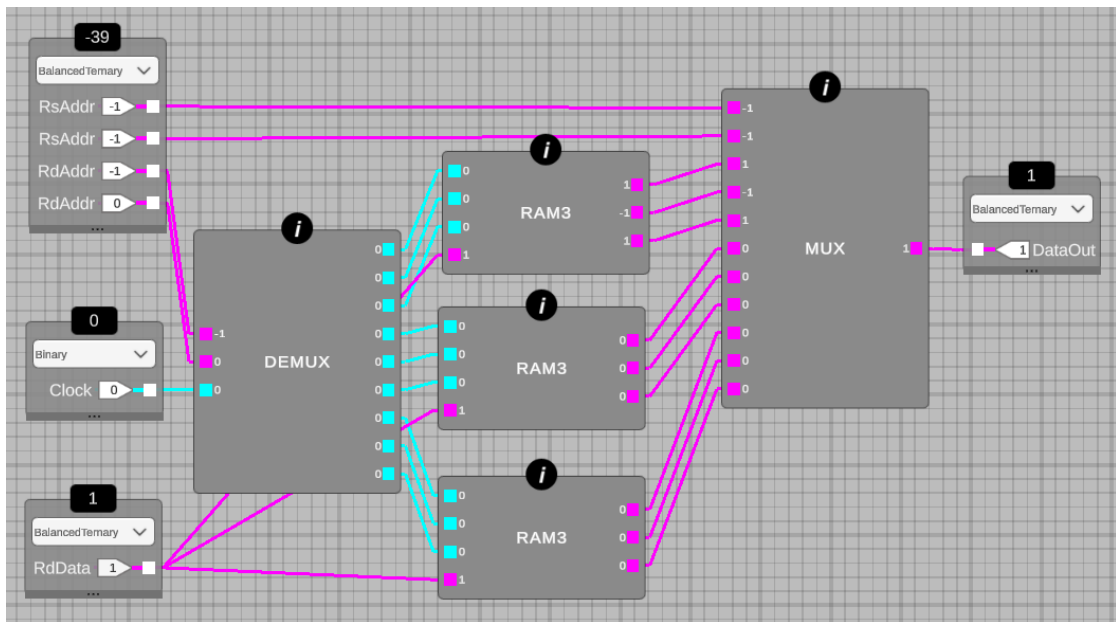


Figure G.18: Ternary ROM/RAM. Implementation of (DE)MUX are shown in Subcomponents

(for instance to 0_3 , 1_3 or 0_2). This is similar to actual behavior of physical memory elements which exhibit unknown states at initialization. Initializing or programming the RAM/ROM can be accomplished manually or automated by reusing the *verify component* functionality.

Ternary full adder

The addition instruction is the cornerstone of most CPU architectures [2], [328]. It is one of the basic arithmetic operations next to subtraction, multiplication and division. These four operations are not uniformly used as they are often reduced to addition and shift operations. Subtraction is addition with one input inverted (2's complement), multiplication is repeated addition and division is repeated subtraction (complete algorithms for all three operations are slightly more complex, see [2]). Even the program counter uses an adder with a constant (such as a hardwired 1) to compute the next instruction address from the present instruction address. Statistics vary, but in [2] the most common instruction of RISC-V CPU's is the ADD instruction. Optimizing the adder result in massive, system-wide performance boost. For this reason the binary adder is historically well researched and is still being improved [327], [328].

The recent large scale survey on ternary full adders (TFA) by Nemati et al. [143] show that a multitude of adder designs exist including many based on multi-threshold CNTFET. Surprisingly, 90% of the reported ternary adders in literature used unbalanced ternary

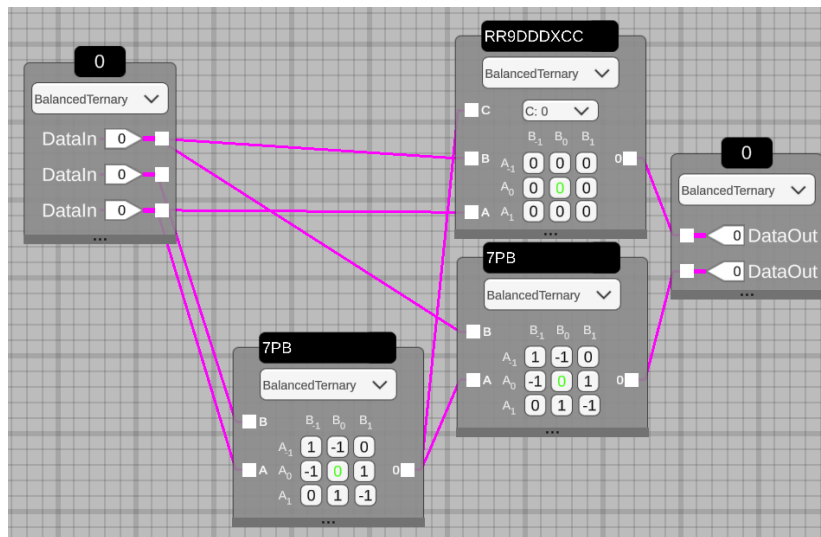


Figure G.19: 110T BTA design with SUM-based CARRY

encoding. Consider for example the ternary half adders (THA) consisting of a SUM and CONS (carry) gate. The 2-ary unbalanced SUM circuit (0tB7P) is 31T and CONS (0tC90) is 10T plus 8T for shared NTI/PTI inverters, totalling 49T. For balanced ternary SUM (0t7PB) is 32T, CONS (0tRDC) is 10T plus 8T for shared inverters, totalling 50T. With nearly identical transistor count, lower switching activity due to balanced ternary arithmetic is preferred. This advantage is discussed in more detail in Chapter 2.

The survey papers by Nemati et al. [143] concludes: "A TFA with faster operation, lower power consumption, and fewer transistors is needed to be considered a potential rival for the binary counterparts". A similar conclusion is found in a survey by Etiemble [225]. Although full adder designs certainly exist with competitive transistor count [202], they have not been demonstrated to be competitive in direct comparison according to PPAC metrics. A survey on binary full adders usings CNTFET [327] show that 14T is possible with transmission gate logic (TG) compared to the well known 28T design using static logic. As mentioned in Chapter 2, for fair comparison to ternary similar logic styles, functionality, resolution, input pattern, etc should be used. Only then will PPAC comparison makes sense. This is unfortunately rarely done in survey papers. For example, the balanced ternary SUM gate made with MRCS is 32T which is 4x larger than the 8T SUM (CMOS XOR) gate in binary. However, the difference in transistor count becomes small when compensating for identical features. The binary 2-bit signed addition circuit requires 2x 2-bit input to cover the range of 2x1 trit input, additional circuitry, wiring, and a add/sub functionality pin. A straightforward implementation would require 3 SUM gates and a AND (carry) for a total of 30T. Even when compensating for theoretical identical resolution as the 2 bit signed binary adder can compute 3 more states (-3,-4 and +2), the difference in transistor count is competitive.

Kim et al. [155] shows a 118T balanced ternary full adder which is replicated in [282]. Just like in a logical level optimized 28T binary full adder design, a TFA with carry that depend on SUM output can be constructed (see Fig. G.19). This design cost 110T and is thus 8T smaller. The SUM components are made with 0t7PB and carry with 0tRR99DDDXCC. Unclear is if this design has been reported earlier.

Ternary asynchronous counter

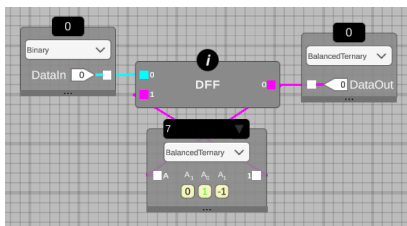


Figure G.20: Balanced ternary ripple counter

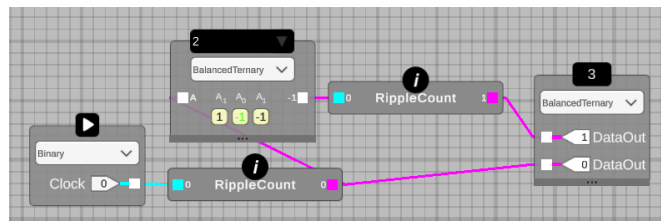


Figure G.21: 2-trit balanced ternary ripple counter

With a memory and adder block another common building block can be constructed: the counter. The counter is used for many types of functionality in digital electronics such as counting clock pulses or as part of a finite state machine(FSM). When counting program instructions it fulfills a role as program counter (PC). Two classes of counters exist, asynchronous or ripple counters and synchronous or parallel counters [329]. Asynchronous balanced ternary counters can be constructed with a d-flip-flop such as Fig. G.12 and INCREMENT (0t7). Contrary to binary counters using JK-flip-flops ternary counters don't toggle between two states such that the output is reusable as a binary clock signal. By adding a NTI (0t2) to detect the overflow, multiple ternary ripple counters can be stacked (see Fig. G.20 and Fig. G.21).

Ternary synchronous program counter

In [260] binary and balanced ternary synchronous counters are shown which were made with MRCS. The various designs are verified with HSPICE simulations. These counters are classified as up/down counters and can be loaded to a specific count value. This makes them usable as a program counter (PC). The PC normally increments linearly but some instructions jump to other instructions at a different address (count value), creating the data-driven flows needed for non-trivial computations. The design of the PC consists of 3 components: ADDER (0t7PB) to count up/down or not count, MUX to choose between the adder or load input and a d-flip-flop (see Fig. G.12) to store the input. In Fig. G.22 a single balanced ternary counter is shown. A CONSENSUS (0tRDC) gate is used between the counters to detect the positive or negative overflow, depending on the direction. This

Appendix G Additional material

design is another example of a mixed-radix design where some signals are binary and some are balanced ternary resulting in a smaller transistor count than binary encoded ternary or pure ternary.

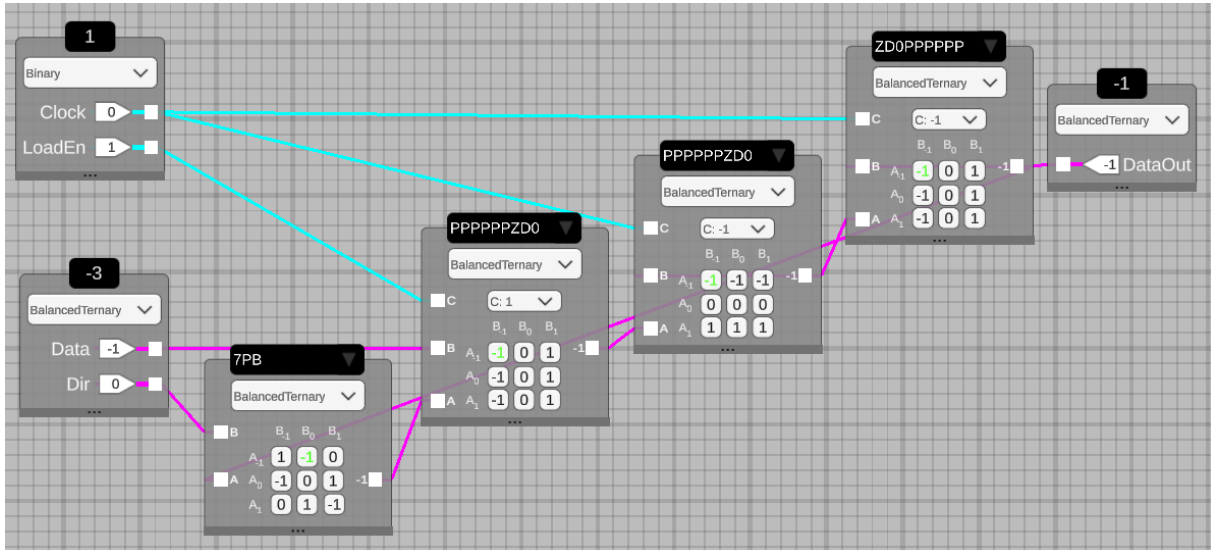


Figure G.22: 1-trit synchronous balanced ternary tri-directional loadable program counter

Both the 6-bit binary counter in [260] and 4-trit balanced ternary counter in Fig. G.23 have been submitted for tape-out [266], [334]. These counters are designed to have identical features. With CNTFETs the 6-bit binary counter needs 542 transistors while the 4-trit ternary needs 8 less, 534 transistors. Less transistors are needed for ternary while the resolution of 4-trits being 81 is higher than the resolution of 6-bits (= 64). Note that no radix economy compensation is applied as the resolution is more or less comparable. A slight compensation to have identical resolution would benefit ternary even more but is purely theoretical since it would require non-discrete devices. The small transistor/die area advantage for ternary increases with higher trit comparisons since the designs are compoundable.

Although in this comparison ternary has a lower transistor count, the average power consumption and delay and thus the PDP is much worse compared to binary. The binary design used $18 \mu\text{W}$ for a basic testbench (see [260]) while the ternary design used $137 \mu\text{W}$. This big difference is the result of the synthesis method discussed earlier. The $\frac{VDD}{2}$ state is made by voltage division and consumes a disproportional amount of current. In [155] Kim et al. show that this state consumes 266.36 nW while logical -1 and logical 1 consume 0.15 nW and 0.31 nW respectively. In the same work they propose to use the body effect to reduce the static power consumption of the middle voltage. They improved the power consumption from 266.36 to $3.57 \mu\text{W}$. New simulations are needed to ascertain if the PDP is below the binary design with this improvement.

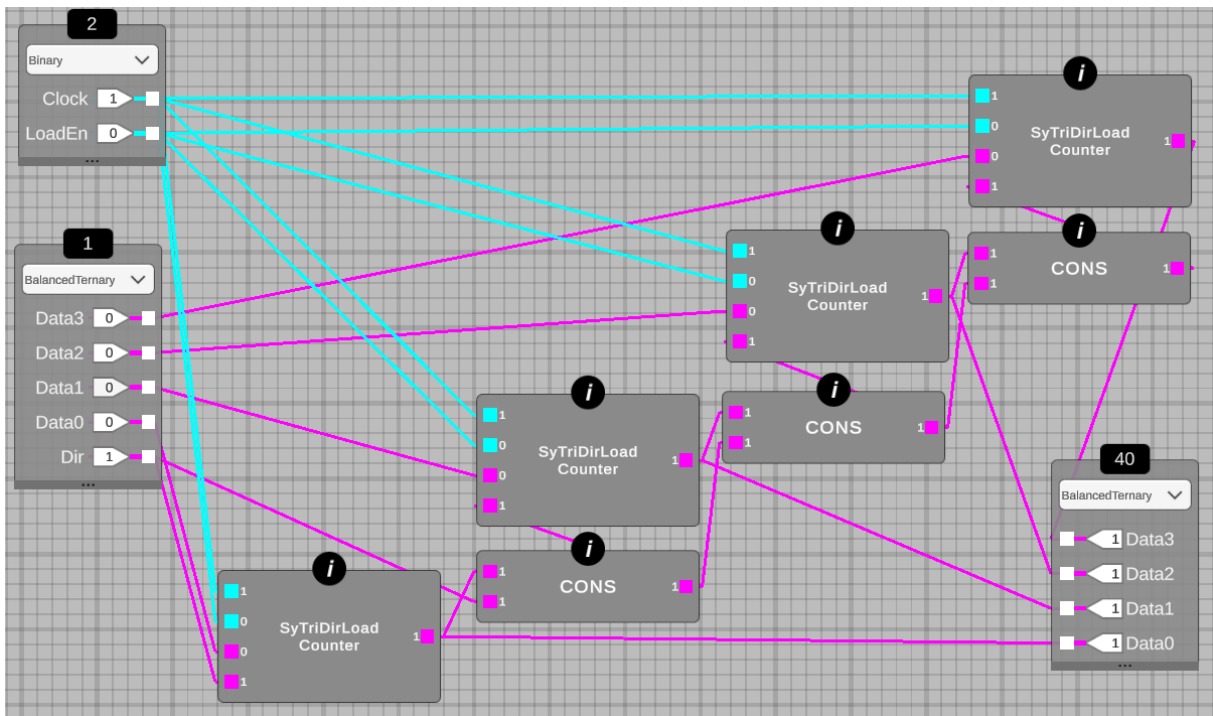


Figure G.23: 4-trit synchronous balanced ternary tri-directional loadable program counter

G.16 Subcomponents

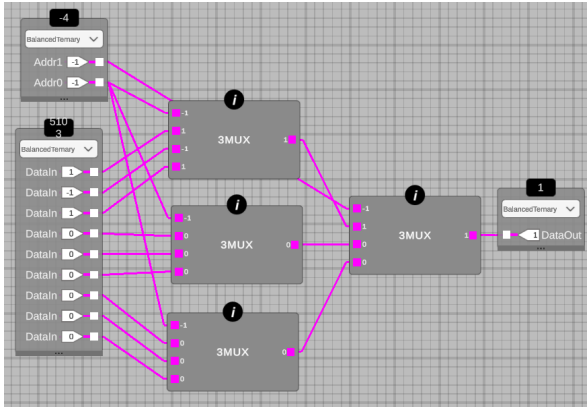


Figure G.24: MUX level 2 implementation, part of Fig. G.18

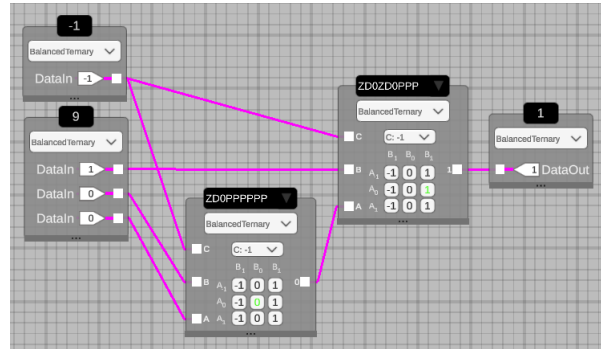


Figure G.25: MUX level 1 implementation, part of Fig. G.18

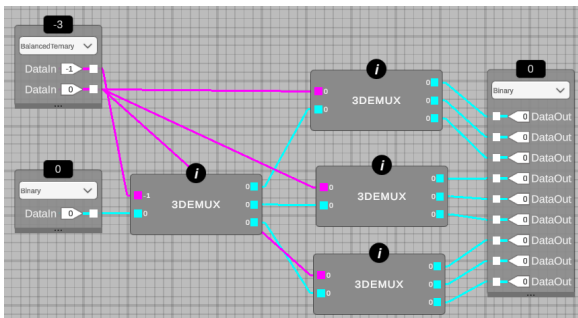


Figure G.26: DEMUX level 2 implementation, part of Fig. G.18

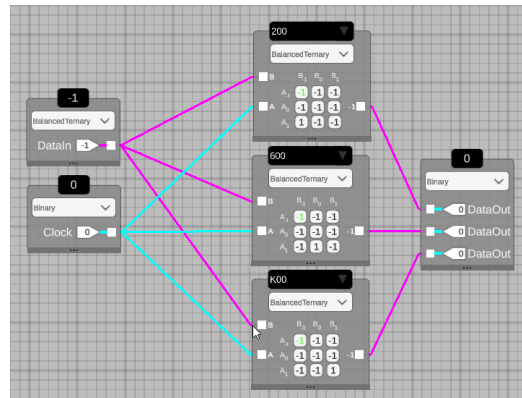


Figure G.27: DEMUX level 1 implementation, part of Fig. G.18

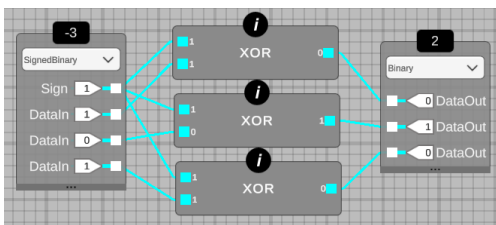


Figure G.28: XOR-3 implementation, part of Fig. 4.9. The XOR gate is made with binary 0r20K.

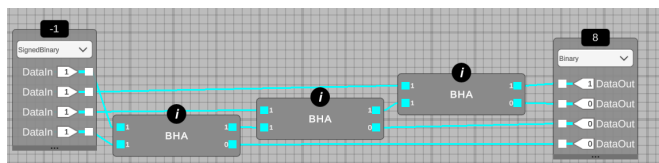


Figure G.29: BHA-3 implementation part of Fig. 4.9. The BHA gate is made with binary 0r20K for the sum and binary 0rK00 for the carry.

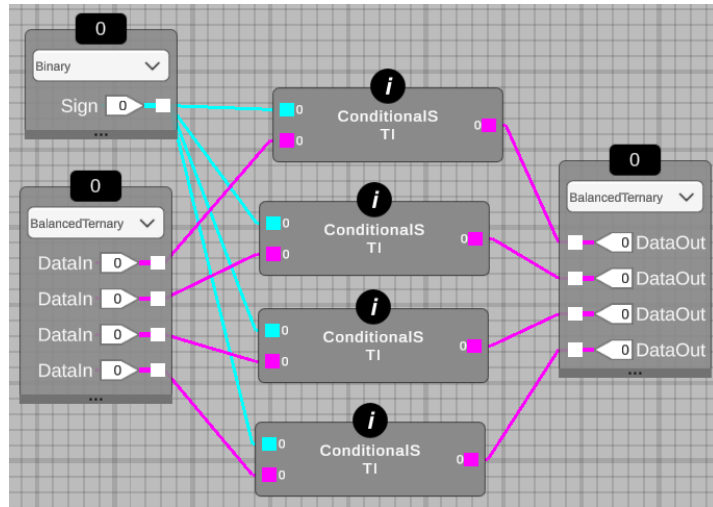


Figure G.30: Conditional-STI-3 implementation part of Fig. 4.9. The Conditional-STI gate is made with *0t5DP*.

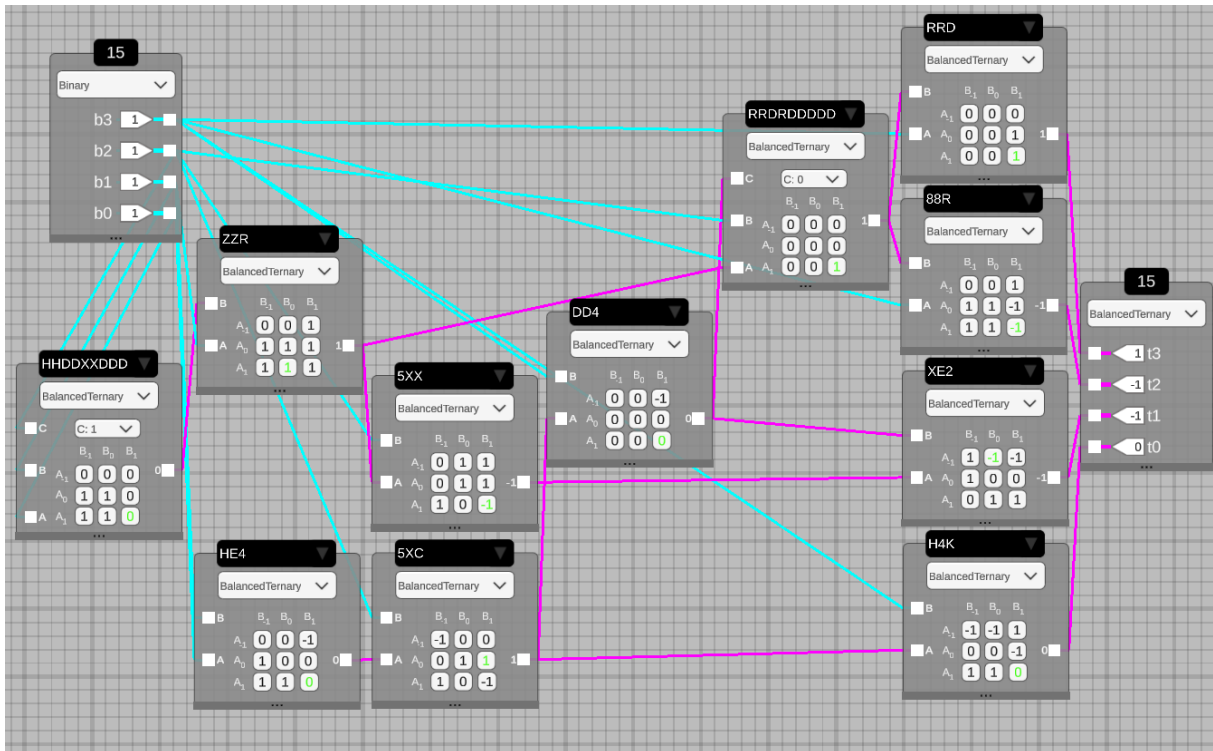


Figure G.31: The 170T 4-bit unsigned binary to 4-trit balanced ternary radix converter in paper E and part of Fig. 4.9

Appendix G Additional material

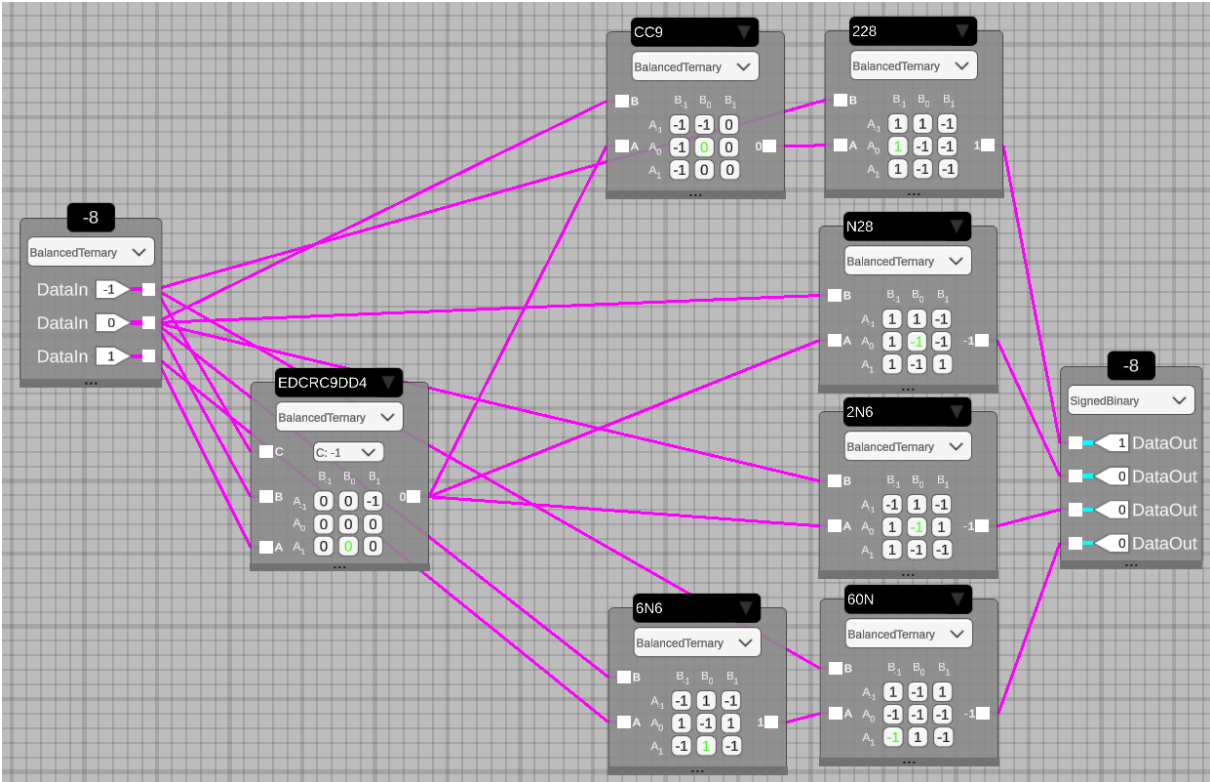


Figure G.32: A novel 139T 3-trit balanced ternary to 4-bit 2's complement signed binary radix converter

G.17 Online radix conversion tool

Mixed Radix Converter

From **Base 3 Balanced**

To **Base 10**

Status: *Successful conversion*

- 0 +

Figure G.33: User interface of the online radix converter tool.

During the development of MRCS and the radix conversion circuits many conversions were needed. For ternary to binary and the inverse a very fast software converter can be found [312]. A more general approach between all possible radices is found in [295]. No all-in-one and browser-based radix converter could be found that was able to convert between radix-2, radix-3 and radix-10 in both signed and unsigned encoding. This led to the development of the open source radix converter tool [333]. It was made with Unity WebGL and is hosted on TernaryResearch.com/mixed-radix-converter/. It is planned to add heptavintimal, octal, hexadecimal and base-64 conversion in a future version.

H

TNNN 2023: Ternary VLSI with CMOS

Ternary VLSI with CMOS using MRCS

Steven Bos* 

Dept. of Science and Industry Systems
University of South-Eastern Norway
Kongsberg, Norway
steven.bos@usn.no

Henning Gundersen 

Dept. of Science and Industry Systems
University of South-Eastern Norway
Kongsberg, Norway
henning.gundersen@usn.no

Abstract—Moore’s exponential scaling law became unsustainable after Dennard scaling stopped in 2006. This has not stopped the industry to continue transistor scaling, leading to exponentially increasing inefficiency and complexity: the power wall, the memory wall and the EDA wall. Radix-3 or ternary is a higher radix than binary and allows information to be more compressed. A higher radix improves utilization of the inherently analog interconnect infrastructure. The design and verification of ternary silicon and especially ternary Very Large Scale Integration (VLSI) has been notoriously hard due to the lack of Electronic Design Automation (EDA) tools and flows. In this one page paper we discuss our latest version of Mixed Radix Circuit Synthesizer (MRCS) that can automatically translate ternary truth tables to binary encoded ternary (BET) high level register transfer level (RTL) code in verilog. The work enables rapid digital design and verification of both ternary ASIC using industry standard CMOS as well as ternary FPGA using off-the-shelf hardware such as Basys 3 with Vivado software. The Openlane flow with various open PDK’s is used to convert RTL to GDS-II and has been used to tape-out designs at Skywater 130nm foundry using the affordable TinyTapeout service.

Index Terms—ternary computing, mixed-radix chip design, ternary EDA

I. INTRODUCTION

The Shannon limit in eq. 1 shows mathematically what the highest theoretical information rate (capacity C) of error-free communication through a noisy channel such as an interconnect is.

$$C = B \log_2 \left(1 + \frac{\text{Signal}}{\text{Noise}} \right) \quad (1)$$

The theorem also shows that more signal levels ie. a higher radix is feasible by adjusting bandwidth (B), power and/or noise. This might seem costly but the relentless focus on computation rather than communication results in 1300x more energy consumption and 100x more clock cycles to communicate data when it is not in cache compared to a typical 32-byte ALU operation [1]. The complexity of binary arithmetic is often taken for granted but a higher radix can make design and verification much more intuitive. For example balanced ternary notation using -1, 0, 1 symbols does not require 2’s complement for negative numbers. Unfortunately, higher radix EDA tooling and flows for chip design are missing. A focus on efficiency is needed for future scaling.

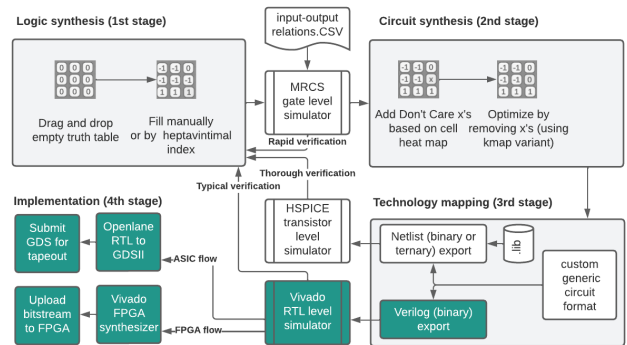


Fig. 1. Mixed-radix chip design and verification flows in MRCS. New flows highlighted in green.

II. TERNARY VLSI WITH MRCS

We first introduced MRCS at ISCAS in 2022 [2] and gave a talk about it at the 1st TNNN. In the previous version we only enabled SPICE simulation of ternary signals using multi-threshold CNTFET as efficient transistors for ternary signals are not yet available. This new version enables automatic conversion of ternary truth tables to binary encoded truth tables and generates the resulting hierarchical netlist in verilog (see Fig. 1). Ternary logic can thus be physically emulated using binary CMOS technology while native transistors are being developed. Emulation is not as efficient as the underlying CMOS is binary valued. We tested the new MRCS workflow with complex multi-trit designs such as ALU’s in HSPICE. A 2-trit multiplier and adder/subtract ALU design has been submitted for tape-out using Openlane, the sky130 OpenPDK and Tinytapeout service. The open source design has also been successfully deployed on a Basys 3 (Artix-7) FPGA [3].

REFERENCES

- [1] P. Ruch, T. Brunswiler, W. Escher, S. Paredes, and B. Michel, “Toward five-dimensional scaling: How density improves efficiency in future computers,” *IBM Journal of Research and Development*, vol. 55, no. 5, pp. 15:1–15:13, 2011.
- [2] S. Bos, H. N. Risto, and H. Gundersen, “Beyond cmos: Ternary and mixed radix cntfet circuit design, simulation and verification,” in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 80–85.
- [3] S. Bos. (2023, May) Tinytapeout: Balanced ternary calculator. [Online]. Available: <https://github.com/aiunderstand/tt03-balanced-ternary-calculator>

Ternary VLSI with CMOS

The next chip design paradigm is 3?

Steven Bos and Henning Gundersen

steven.bos@usn.no

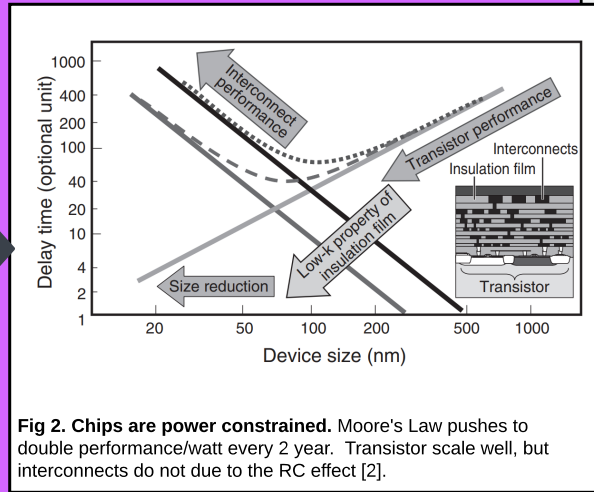
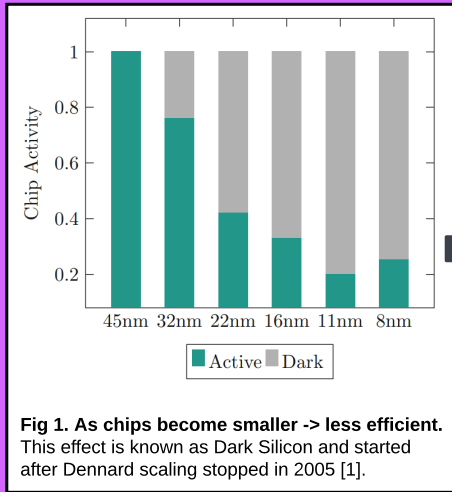
henning.gundersen@usn.no

Takeaway: With our open source tool Mixed Radix Circuit Synthesizer (MRCS) you can design, verify and make ternary logic chips with industry standard CMOS

USN³ research



2nd TNNN Conference Vestfold 2023



"The energy required for off-chip communication was 260 times greater than that for arithmetic operations at 130 nm, whereas this ratio increased to 1300 for the 45 nm node" [3]

Interconnect bottleneck as a consequence of device-centric scaling

Shannon limit: increase efficiency of interconnect

$$C = B \log_{2,3,4,..} \left(1 + \frac{Signal}{Noise} \right)$$

Strategies to improve efficiency

Strategy	Frequency	Signal/Noise Ratio (noise margin)	Effect	Action
1	F ↓	SN ↑	Radix ↑	Reduce frequency to improve SnR
2	F =	N ↓	Radix ↑	Reduce noise to improve SnR
3	F =	S ↑	Radix ↑	Increase Vdd to improve SnR
4	F =	S ↓	Radix ↑	Reduce Vdd and improve device noise tolerances
5	F =	S/N =	Radix ↑	Improve device noise tolerances

Only binary digital Electronic Design Automation (EDA) tools exist

New! Mixed Radix Circuit Synthesizer. An open source EDA tool to design and verify binary, ternary and mixed radix circuits.

Literature (DOI links)
 [1] 10.1007/978-3-319-31596-6_1
 [2] 10.1016/j.mee.2014.10.019
 [3] 10.1147/JRD.2011.2165677
 [4] 10.1109/ISCAS48785.2022.9937259
 [5] 10.5281/zenodo.3557410

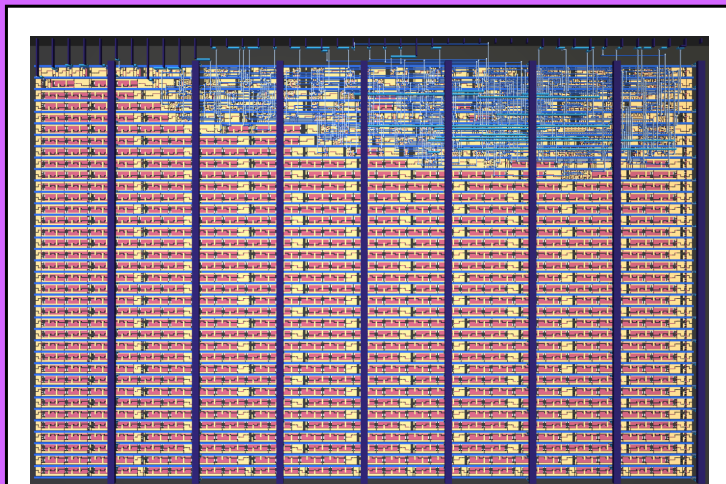


Fig 4. A ternary logic chip generated with MRCS and submitted for tapeout in May 2023. This ASIC contains a 4-trit tri-directional loadable program counter[5].

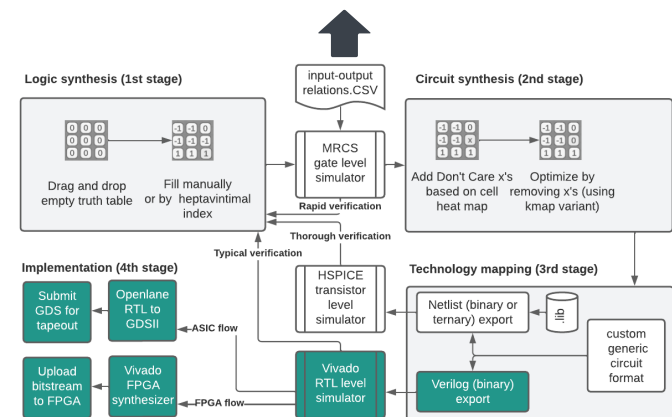
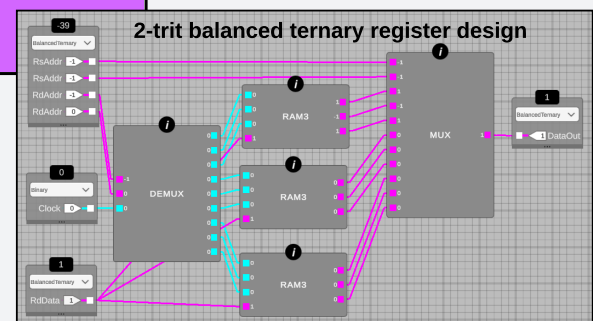


Fig 3. The three workflows of MRCS. Design a ternary state machine with sequential and combinatorial logic using binary and ternary truth tables. Verify correct gate-level behavior using the internal simulator. Automatically optimize the circuit for transistor count and generate a CNTFET netlist and a CMOS verilog file. **Flow 1)** Use netlist file for mixed-signal verification with HSPICE. **2)** Use verilog file to generate a GDS file with OpenLane. Tapeout with TinyTapeout for ~\$100! **3)** Use verilog file to upload bitstream to FPGA. **Read more in [4]**



Did you know? State-of-the-art SSD's, videocards network cards and other peripheral cards increasingly switch to a higher radix for more bandwidth?

Benefits despite binary devices? Higher order logic needs fewer carry switching thus saving power and increasing performance

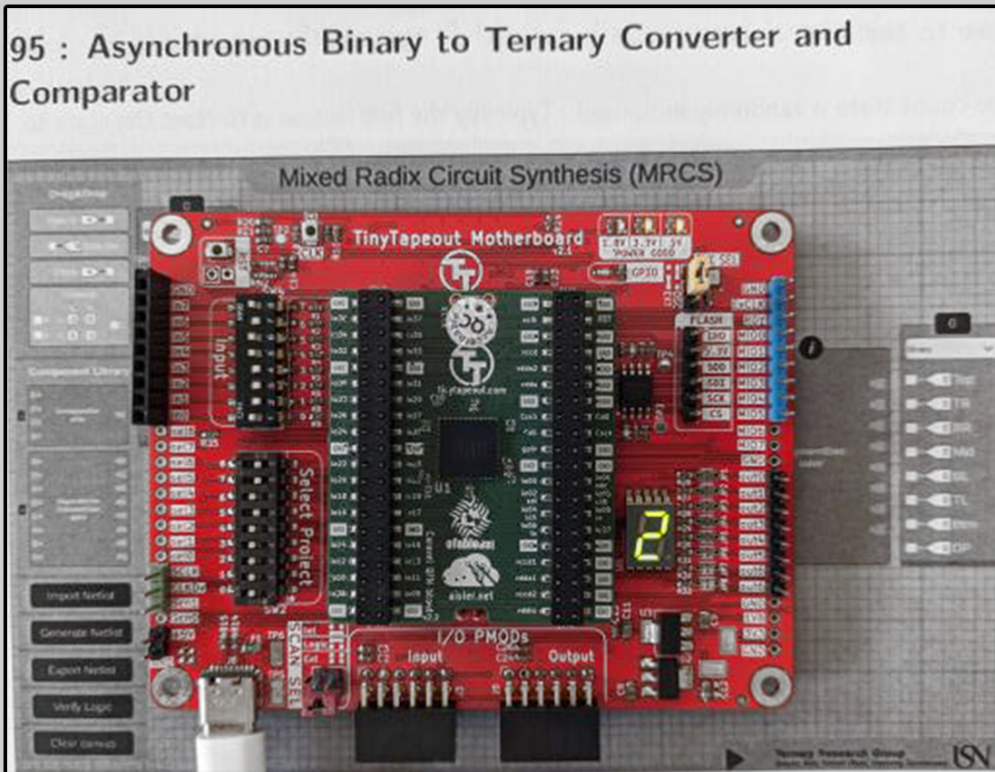
Why 3? 3 is the closest integer to the optimum e (Radix economy theorem)

I

Tape-outs

Hardware verification

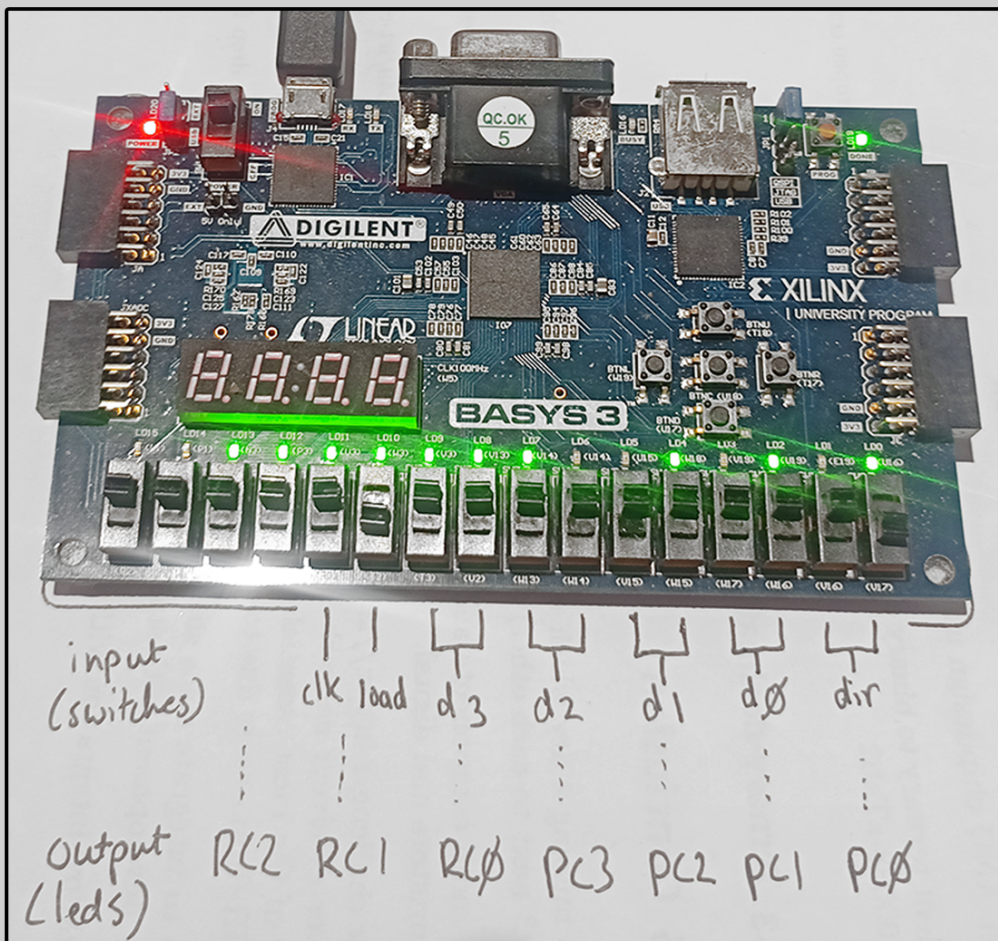
(Top) TinyTapeout 2 ASIC (Bottom) Digilent Basy3-3 FPGA



binary to ternary converter & comparator

github.com/aiunderstand/tt02-async-binary-ternary-convert-compare

Correct unary coded ternary to decimal conversion on 7 segment display (Image: Matt Venn)

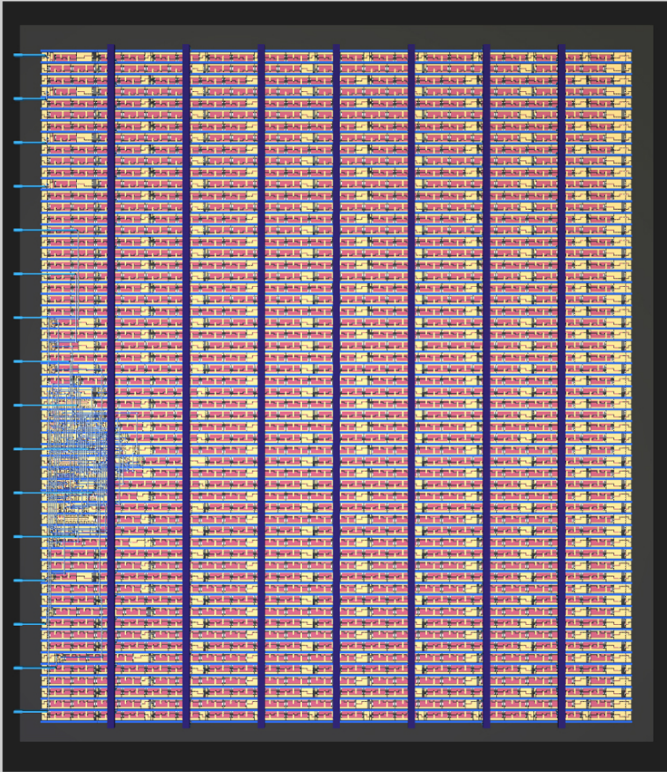


4-trit tri-directional loadable program counter & radix converter

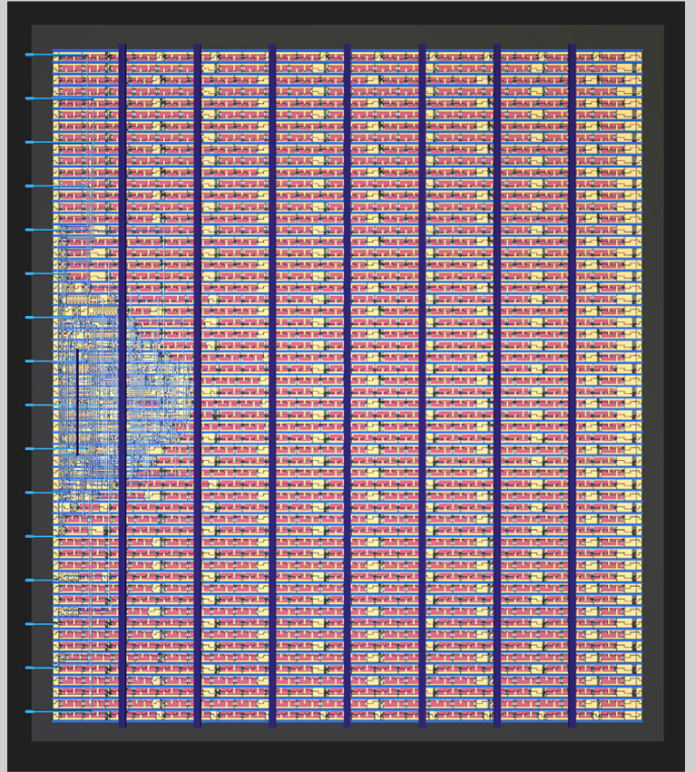
https://github.com/aiunderstand/tt03p5-4-trit-balanced-ternary-counter-bt_signb_bt-radix-converter

Demonstrating balanced ternary count down with current value being 14 on LED's (+---)

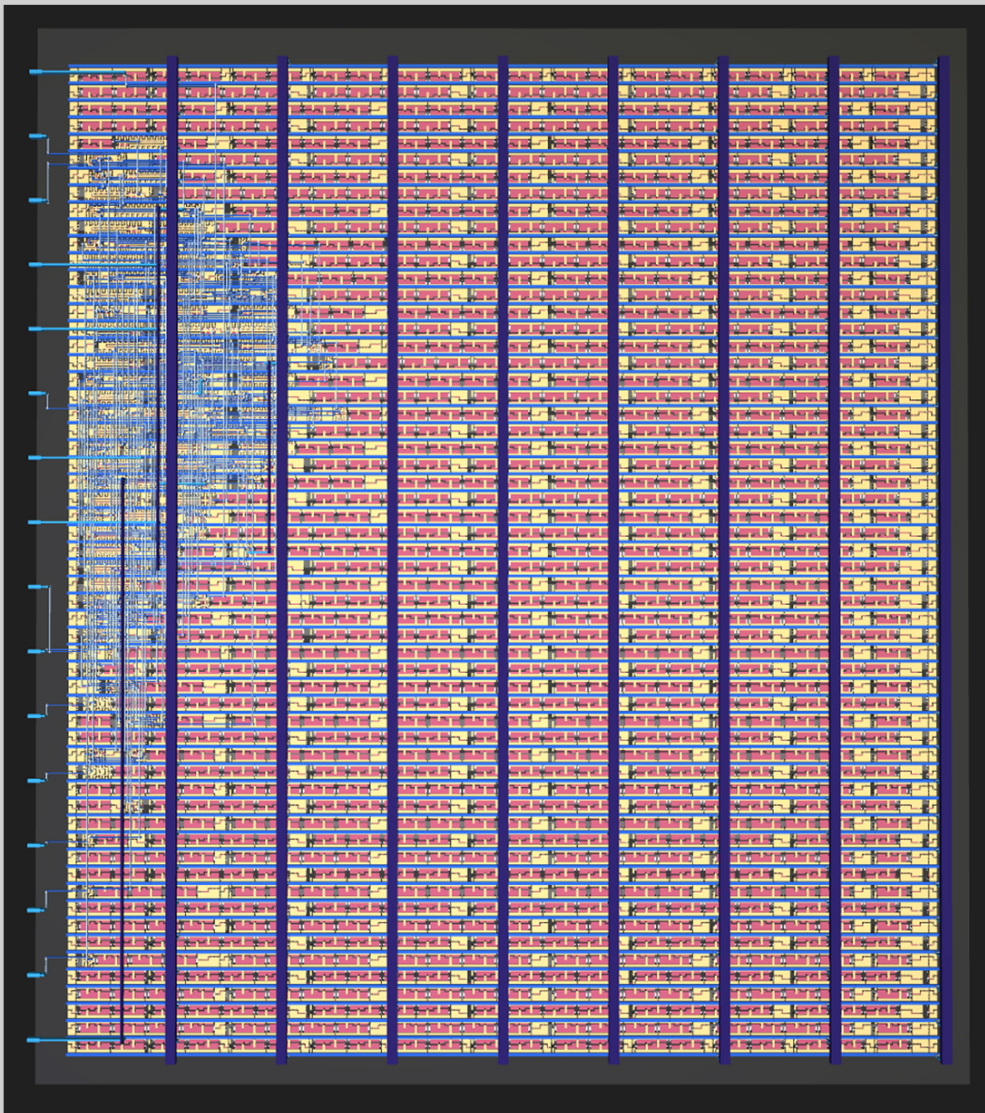
Skywater 130nm Tapeouts



4-bit tri-directional loadable program counter
github.com/aiunderstand/tt02-4bit-tristate-loadable-counter

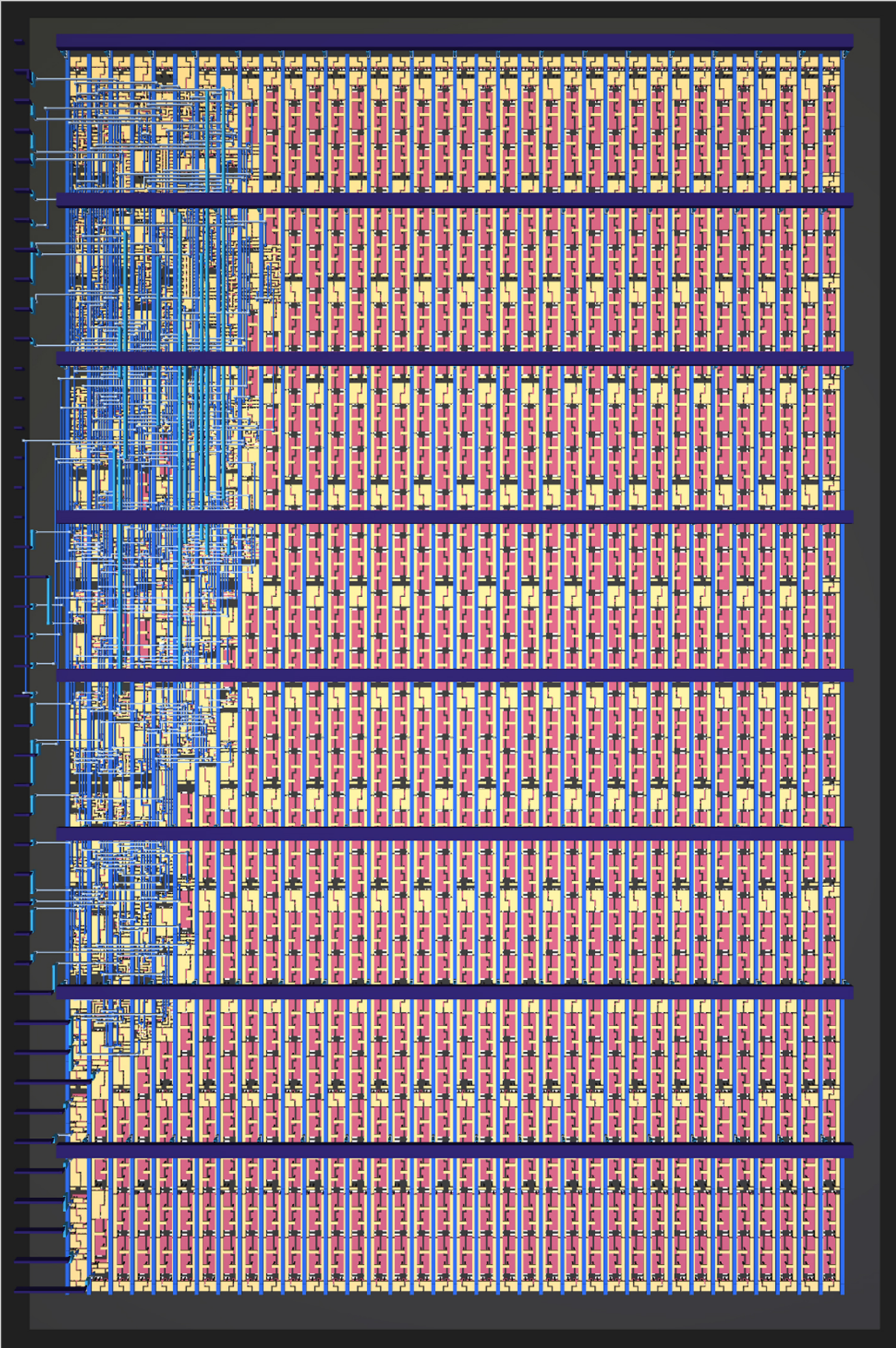


binary to ternary converter & comparator
github.com/aiunderstand/tt02-async-binary-ternary-convert-compare



binary encoded balanced ternary calculator
github.com/aiunderstand/tt03-balanced-ternary-calculator

Skywater 130nm Tapeouts



4-trit tri-directional loadable program counter & radix converter

github.com/aiunderstand/tt03p5-4-trit-balanced-ternary-counter-bt_signb_bt-radix-converter

**Beyond 0 and 1: A mixed radix
design and verification workflow
for modern ternary computers**
Steven Bos

**Doctoral dissertations at the
University of South-Eastern
Norway no. 189**

ISBN 978-82-7206-854-6 (print)
ISBN 978-82-7206-855-3 (online)

usn.no