UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

DEPARTAMENT D'ENGINYERIA ELECTRÒNICA

Variability-aware Architectures based on Hardware Redundancy for Nanoscale Reliable Computation

by

Nivard Aymerich

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy
in the Universitat Politècnica de Catalunya
within the Doctoral Program in Electronic Engineering

Thesis advisor: Antonio Rubio

December 2013

A la Mª Amparo…
alegria i suport inestimable.

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Abstract

Electronic Engineering Department

Doctor of Philosophy

by Nivard Aymerich

During the last decades, human beings have experienced a significant enhancement in the quality of life thanks in large part to the fast evolution of Integrated Circuits (IC). This unprecedented technological race, along with its significant economic impact, has been grounded on the production of complex processing systems from highly reliable compounding devices. However, the fundamental assumption of nearly ideal devices, which has been true within the past CMOS technology generations, today seems to be coming to an end. In fact, as MOSFET technology scales into nanoscale regime it approaches to fundamental physical limits and starts experiencing higher levels of variability, performance degradation, and higher rates of manufacturing defects. On the other hand, ICs with increasing number of transistors require a decrease in the failure rate per device in order to maintain the overall chip reliability. As a result, it is becoming increasingly important today the development of circuit architectures capable of providing reliable computation while tolerating high levels of variability and defect rates.

The main objective of this thesis is to analyze and propose new fault-tolerant architectures based on redundancy for future technologies. Our research is founded on the principles of redundancy established by John von Neumann in the 1950s and extends them to three new dimensions:

1. Heterogeneity: Most of the works on fault-tolerant architectures based on redundancy assume homogeneous variability in the replicas like von Neumann's original work. Instead, we explore the possibilities of redundancy when heterogeneity between replicas is taken into account. In this sense, we propose compensating mechanisms that select the weighting of the redundant information to maximize the overall reliability.

2. Asynchrony: Each of the replicas of a redundant system may have associated different processing delays due to variability and degradation; especially in future nanotechnologies. If we design our system to work locally in asynchronous mode then we may consider different voting policies to deal with the redundant information. Depending on how many replicas we collect before taking a decision we can obtain different trade-off between processing delay and reliability. We propose a mechanism for providing these facilities and analyze and simulate its operation.

3. Hierarchy: Finally, we explore the possibilities of redundancy applied at different hierarchy layers of complex processing systems. We propose to distribute redundancy across the various hierarchy layers and analyze the benefits that can be obtained.

Drawing on the scenario of future ICs technologies, we push the concept of redundancy to its fullest expression through the study of realistic nano-device architectures. Most of the redundant architectures considered so far do not face properly the era of Terascale Computing and the nanotechnology trends. Since von Neumann applied for the first time redundancy at electronic circuits, never until now effects as common in nanoelectronics as degradation and interconnection failures have been treated directly from the standpoint of redundancy. In this thesis we address in a comprehensive manner the reliability of digital processing systems in the coming technology generations.

Acknowledgements

I would like to acknowledge and thank many people who, in some or another way, contributed to this thesis, especially

To my advisor, Antonio Rubio for his guidance throughout this research contributing their valuable knowledge and experience.

To Sorin Cotofana, for his qualified opinion and advice on some specialized topics.

To my colleagues in the Department of Electronic Engineering, Carmina, Esteve, Sergio, Jordi, Peyman, Gerhard, David, Francesc, and Antonio for their collaboration and support.

Finally, I would like to thank my sweet wife Mª Amparo and my lovely daughter Meritxell, who supported me all this time urging me on and eagerly waiting for me with great patience as I spent long hours working away from them.

Contents

Abstract **iv**

Acknowledgements **vi**

List of Figures **xi**

List of Tables **xvii**

Abbreviations **xix**

1 Introduction **1**
1.1 The Challenges of Nanoelectronics 2
1.2 Reliable Computation and Fault-tolerance 4
1.3 Outline of the Thesis 4

2 Nanoscale Technology Precedents **7**
2.1 Nanoelectronics and Nanodevices 8
2.1.1 Nanowire Field-Effect Transistors (NWFETs) 8
2.1.2 Tunnel Field-Effect Transistors (TFETs) 9
2.2 Variability and Fault Modeling 10
2.3 Aging and Degradation Modeling 12
2.4 Reliability Characterization 15

3 Reliable Redundancy Architectures **17**
3.1 Redundancy by von Neumann 18
3.1.1 Error framework 18
3.1.2 Difficulty of errors 18
3.1.3 Problem of reliable computing 19
3.2 Fault-tolerant Architectures based on Redundancy 20
3.2.1 Modular Redundancy 20
3.2.2 Multiplexing 22
3.2.3 Reconfiguration 22
3.3 Redundancy Managers 23
3.3.1 Majority Gate (MAJ) 24
3.3.2 Averaging Cell (AVG) 25
3.3.2.1 Systematic Effects 27

3.4 Fundamental Error Bounds . . . 28
3.4.1 Basic Definitions . . . 28
3.4.2 State of the Art . . . 29
3.4.3 Asymmetric Error Designs . . . 30
3.4.3.1 Asymmetric Error NAND Gates . . . 31
3.4.3.2 Asymmetric Error Reliable Computation with 2-input NAND gates . . . 32
3.4.3.3 Asymmetric Error Reliable Computation with k-input NAND gates . . . 35
3.4.3.4 Concluding Remarks . . . 38

4 Thesis Motivation and Objectives 41

5 Heterogeneous-aware Reliable Design 45
5.1 The Unbalanced Averaging Cell (U-AVG) . . . 46
5.1.1 Heterogeneous Variability Scenario (Simple Case) . . . 47
5.1.2 The Existence of Optimal Weighted Averages in General AVGs . . . 49
5.1.3 Optimal Unbalanced Weights . . . 54
5.1.4 AVG versus U-AVG . . . 55
5.2 The Adaptive Averaging Cell (AD-AVG) . . . 56
5.2.1 Adaptive Learning versus Static AVG . . . 58
5.2.2 Noisy Variability Monitor in the AD-AVG . . . 60
5.2.3 AD-AVG Implementation . . . 61
5.2.3.1 Variability Monitor . . . 62
5.2.3.2 Weight Drivers and Switching Resistive Crossbar . . . 63
5.3 DSR-aware AD-AVG . . . 64
5.3.1 DSR Fundaments . . . 67
5.3.1.1 DSR in 2-input AD-AVG . . . 67
5.3.1.2 AD-AVG yield sensitivity analysis . . . 69
5.3.2 DSR in AD-AVG . . . 71
5.3.3 DSR-aware AD-AVG design . . . 73
5.3.3.1 DSR Occurrence Control . . . 73
5.3.3.2 Controllable Noise Injectors Implementation . . . 77
5.4 Averaging Cell Linear Threshold Gate (AC-LTG) . . . 78
5.4.1 AC-LTG Mathematical Model . . . 78
5.4.2 Implementable 2-input AC-LTG Boolean Functions . . . 79
5.4.3 N-redundant 2-input NAND AC-LTG . . . 80
5.4.3.1 Simulation results . . . 85
5.4.3.2 Measurement of Tolerance against Manufacturing and Environment Variability . . . 85
5.4.4 Reliability Comparison of NAND AC-LTG versus NAND multiplexing . . . 87
5.4.4.1 NAND multiplexing topology . . . 87
5.4.4.2 Parameter Equivalences . . . 88
5.4.4.3 Comparison Results . . . 90
5.5 Conclusion . . . 91

6 Time-aware Reliable Design 93

6.1 The partially-Asynchronous R-fold Modular Redundancy (pA-RMR) . . 94
6.1.1 pA-RMR Model 95
6.1.2 pA-RMR Reliability 96
6.1.3 pA-RMR Performance 99
6.1.3.1 Tokens' Arrival Time 100
6.1.3.2 Policies' Response Time 101
6.2 Reliability vs Performance Trade-off 102
6.2.1 Fault-free Tokens in pA-RMR Architectures 103
6.2.2 Faulty Tokens in pA-RMR Architectures 104
6.3 Conclusions 106

7 Multiple-layer Reliable Design **107**
7.1 Cross-Layer Reliable Architecture 108
7.2 Fault Model 109
7.3 Analysis Method 109
7.4 Redundancy Distribution Simulation Results 111
7.4.1 Redundancy at Different Layers 111
7.4.2 Redundancy at Multiple Layers Simultaneously 113
7.5 Conclusions 116

8 Final Conclusions **117**
8.1 Thesis Contributions 119
8.2 Future Work 120

A Publications related to this thesis **121**
A.1 Journal papers 121
A.2 Book chapters 122
A.3 Conference papers 122

Bibliography **123**

List of Figures

1.1 Transistor count of Intel's processors from 4004 to Xeon Phi against the year of introduction. 2
1.2 Cumulative interdependent challenges as a function of time and technology generation. 3

2.1 Taxonomy for emerging research information processing devices (technology entries are representative but not comprehensive). 8
2.2 Structure of a Nanowire Field-Effect Transistor (NWFET). 9
2.3 Structure of p-type TFET with applied source (V_S), gate (V_G) and drain (V_D) voltages. 9
2.4 Error probability P_e against the ratio between the standard deviation σ_i and V_{cc}. The highlighted point corresponds to the reliability reference value $P_e^{\max} \equiv 10^{-4}$ and the corresponding maximum admissible standard deviation $\sigma_{i\ \max}/V_{cc} = 0.1344$. 13
2.5 Five samples of variability σ_i^2 profile against degradation in time units generated using our degradation model. 15

3.1 Memory system. 19
3.2 Triple Modular Redundancy (TMR) schematic. 21
3.3 (a) R-fold Modular Redundancy (RMR); and (b) Cascaded R-fold Modular Redundancy (CRMR) schematic. 21
3.4 NAND multiplexing schematic. 22
3.5 Basic structure for the reconfiguration technique theory. 23
3.6 Majority Gate (MAJ) schematic. 24
3.7 Output error probability p_{out} against input error probability p_{in} of 3-input MAJ for different values of gate failure rate $\epsilon = \{0, 0.1, 1/6, 0.3\}$. 25
3.8 Averaging Cell (AVG) schematic. 25
3.9 Asymmetric error NAND gate. 31
3.10 Circuit schematic built out of 2-input NAND gates. 32
3.11 Fundamental error bound for asymmetric error 2-input NAND gates. 34
3.12 Maximum tolerable variability $\sigma_{\max}$ against threshold level l for reliable computation using 2-input NAND gates. 35
3.13 Fundamental error bound for asymmetric error k-input NAND gates. 36
3.14 Maximum tolerable variability $\sigma_{\max}$ against threshold level l for reliable computation using k-input NAND gates. 37
3.15 Maximum tolerable variability $\sigma_{\max}$ of symmetric and asymmetric NAND gate designs against the number of inputs k. The exact error threshold for k-input gates (k odd) is also shown with black markers. 38

4.1 Schematic representation of the three main contributions of this thesis to extend the principles of redundancy established by von Neumann in [2]. The three considered dimensions, namely heterogeneity, asynchrony, and hierarchy, correspond to Chapters 5, 6, and 7 respectively. 43

5.1 Unbalanced Averaging Cell (U-AVG) schematic. 46
5.2 Output standard deviation of AVG and U-AVG with 2, 3, 5 and 10 inputs subject to heterogeneous variability scenario. The scenario is modeled with a standard deviation $\sigma = 0.1\ V$ in all the inputs except from the qth that has σ_q ranging between $0.1\ V$ and $0.4\ V$. 48
5.3 Optimal weight c_q^{opt} against the ratio of input variances σ_q^2/σ^2 in a simple case of heterogeneous variability scenario. All the replicas have the same variance σ^2 except form the input qth that has variance σ_q^2. 49
5.4 Variation of the error probability P_e against the weight c_q. Standard deviation in the input q ranges from $\sigma_q = 0.0\ V$ to $\sigma_q = 0.4\ V$ and the standard deviation of the weighted average due the rest of inputs is $\sigma_{y'}|_{\{c_q=0\}} = 0.2\ V$. 51
5.5 Yield analysis of AVG and U-AVG against the redundancy factor for different technology scenarios. The simulated scenarios identified with letters (from A to F) have associated heterogeneous input variances following the Gamma distribution. The mean values for the standard deviation range from $E\{\sigma_i\} = \sigma_{\max}$ in scenario A to $E\{\sigma_i\} = 6\sigma_{\max}$ in scenario F. $\sigma_{\max}$ corresponds to the reference value $0.1344\ V$. 57
5.6 Adaptive Averaging Cell (AD-AVG) schematic. 57
5.7 Yield of different size AD-AVG, U-AVG and AVG against the degradation in time units. 58
5.8 Yield analysis of different size AD-AVG and AVG against degradation. Redundancy levels are chosen to meet the reliability requirements: 90% yield after 7, 25, 45 and 75 degradation in time units. 59
5.9 Yield analysis of different size AD-AVG against degradation. Simulations include the effect of different noise levels in the Variability Monitor (from $\sigma_s = 0\ V$ to $\sigma_s = 0.08\ V$). 61
5.10 Adaptive Averaging Cell (AD-AVG) implementation in switching resistive crossbar technology. 62
5.11 Crossbar layout view of AD-AVG to configure the averaging weights. . . . 63
5.12 Monte Carlo simulation of a 5-input AD-AVG with the proposed topology. The subplots correspond to the five input replica variances (σ_i^2), the five adaptive averaging weights (c_i), and the variance of the weighted average ($\sigma_{y'}^2$) against the degradation. 65
5.13 Yield of 20-input AD-AVGs against degradation with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$. 66
5.14 Monte Carlo simulation result of a 2-input AD-AVG yield against σ_2 with null variability in the input 1 ($\sigma_1 = 0\ V$). We consider different levels of noise in the Variability Monitor (σ_s) and a maximum admissible output variability of $\sigma_{\max} = 0.1344\ V$. 68

5.15 Sensitivity analysis of AD-AVG's yield with multiple inputs to the variability of input ith for different levels of noise (σ_s) in the Variability Monitor. The figure above shows the AD-AVG yield and the lower shows the yield loss which is defined as the yield with a null noise in the Variability Monitor ($\sigma_s = 0$) less the yield. 71
5.16 Yield analysis of different size AD-AVGs against degradation for different levels of noise (σ_s) in the Variability Monitor. 72
5.17 Adaptive Averaging cell (AD-AVG) architecture with independent noise generators added to the inputs. 74
5.18 Yield against degradation of 20-input AD-AVGs with different levels of noise added to the inputs. Thick blue line correspond to the AD-AVG cell yield with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$. Thin red lines correspond to the impact of adding noise to the inputs of this cell with different magnitudes: $\sigma_x = 0.05V$, $0.10V$, and $0.20V$. 75
5.19 Yield against degradation of 20-input AD-AVGs with different levels of noise added to the inputs. Thick blue line correspond to the AD-AVG cell yield with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$. Thin colored lines correspond to the impact of adding noise to the inputs of this cell with different magnitudes from $\sigma_x = 0V$, $\sigma_x = 0.9V$. The thick black line corresponds to the curve followed by the yield when the proper input noise magnitude at each degradation in time is applied in order to maximize the reliability. 76
5.20 Magnitude of input noise σ_x against degradation in time that maximizes the reliability of a 20-input AD-AVG with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$ based on the DSR effect. 77
5.21 Schematic of M-input AC-LTG with redundancy level N. 79
5.22 Schematic of N-redundant 2-input NAND AC-LTG with optimized precision-limited configuration of weights and threshold robust to manufacturing inaccuracies ($W = \{-18, -18\}$, $T = -25V$). 81
5.23 P_e color map of N-redundant 2-input NAND AC-LTG with null variability ($\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$) 83
5.24 P_e color map of 10-redundant 2-input NAND AC-LTG with variability parameters ($\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0.1V$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$) and in small white cross the configuration with lower probability of error P_e. Yellow contour outlines the area with error probability $P_e < 0.1$ 84
5.25 P_e color map of 10-redundant 2-input NAND AC-LTG with variability parameters ($\sigma_{\eta_1} = \sigma_t = 0.1V$, $\sigma_{\eta_2} = 0.5V$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$) and in small white cross the configuration with lower output probability of error P_e. Yellow contour outlines the area with error probability $P_e < 0.1$ 84
5.26 Output probability of error P_e versus redundancy level N at different levels of deviation in the assignment of weights and threshold $d \geq max(\Delta w_1, \Delta w_2, \Delta t)$. Case with variability parameters $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0.1V$ 85

5.27 Output probability of error P_e versus redundancy level N at different levels of homogeneous variability $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t$ and null deviation in weights and threshold assignment $d = 0$ 86
5.28 P_e color map of 10-redundant AC-LTG unate functions with variability parameters ($\sigma_{\eta_1} = \sigma_t = 0.1V$, $\sigma_{\eta_2} = 0.5V$). In small red marker the optimized precision-limited configuration and in small white cross the configuration with lower output probability of error P_e. 87
5.29 Schematic of a NAND multiplexing architecture 88
5.30 Schematic of a NAND AC-LTG with N redundant inputs and outputs . . 89
5.31 Reliability comparison between NAND multiplexing with redundancy $N = 10$ and number of restoring stages $n = 7$ and NAND AC-LTG with redundancy $N = 10$ and different levels of inaccuracy in weights and threshold assignment d 90

6.1 partially-Asynchronous R-fold Modular Redundancy (pA-RMR) architecture. 96
6.2 Reliability provided by the 4 different voting policies of a pA-RMR with 7 replicas against the input error probability ϵ. 98
6.3 Maximum admissible error probability per replica (ϵ_{max}) against voting policies. Different lines correspond to different targets of output error probability Pe_{max}. 99
6.4 Arrival time CDF of tokens taking into account the arrival order of a pA-RMR with 7 replicas. 101
6.5 Response Time CDF of the different policies of a pA-RMR with 7 replicas. Solid lines correspond to input error probability $\epsilon = 0.01$ and slashed lines to $\epsilon = 0.1$. 102
6.6 Reliability versus performance trade-off for different size pA-RMR from $R = 1$ to 17 with an input error probability of $\epsilon = 0.2$. 104
6.7 Reliability versus performance trade-off for different size pA-RMR from $R = 1$ to 17 with an input error probability of $\epsilon = 0.2$. 105

7.1 Schematic view of a cross-layer system and its multiple hierarchy layers. We characterize each layer with a parameter D that indicates the number of devices embedded in each element. 108
7.2 Schematic view of terminals and interconnects of a computing system with D_s devices. Interconnects of layer D are depicted with slashed lines. 111
7.3 Global error probability P_e^* of a cross-layer system with $D_s = 10^{10}$ devices against redundancy layer D with n-MR technique. Slashed lines correspond to the global error probability components: black curves with triangular markers correspond to the error probability due to devices (ideal interconnects) and green curves with square markers correspond to interconnects (ideal devices). The levels of redundancy are $n = 3$, 5, 7 and 9. The rest of parameters are $k = 2$, $r = 0.6$, $P_{eo} = 10^{-2}$ and $\delta = 10^{-15}$. . . . 112

7.4 Global error probability P_e^* of a cross-layer system with $D_s = 10^{10}$ devices against redundancy layers D_1 and D_2. The optimum redundancy configurations are highlighted with white dot makers and it is labeled the associated minimum error probability (P^*_{emin}). The yellow contours enclose the regions with global error probabilities lower than $1.1 \cdot P^*_{emin}$. The levels of redundancy are $n = 3$ for both dimensions and the rest of parameters are $k = 2$, $r = 0.6$, $P_{eo} = 10^{-2}$ and $\delta = 10^{-15}$. 114
7.5 Impact of interconnect failure probability δ in the optimum redundancy distribution D_1 and D_2 in a cross-layer system with $D_s = 10^{10}$ devices. Both layers have the same redundancy $n = 3$ and the rest of parameters are $k = 2$, $r = 0.6$ and $P_{eo} = 10^{-2}$. Color lines show the contours of minimum error probability $P_e^* = 1.1 \cdot P^*_{emin}$ for different values of δ. Black dots correspond to the corresponding optimum configurations. . . . 115

List of Tables

3.1 Summary of the main advances in the field of fundamental error bounds for reliable computation. 30

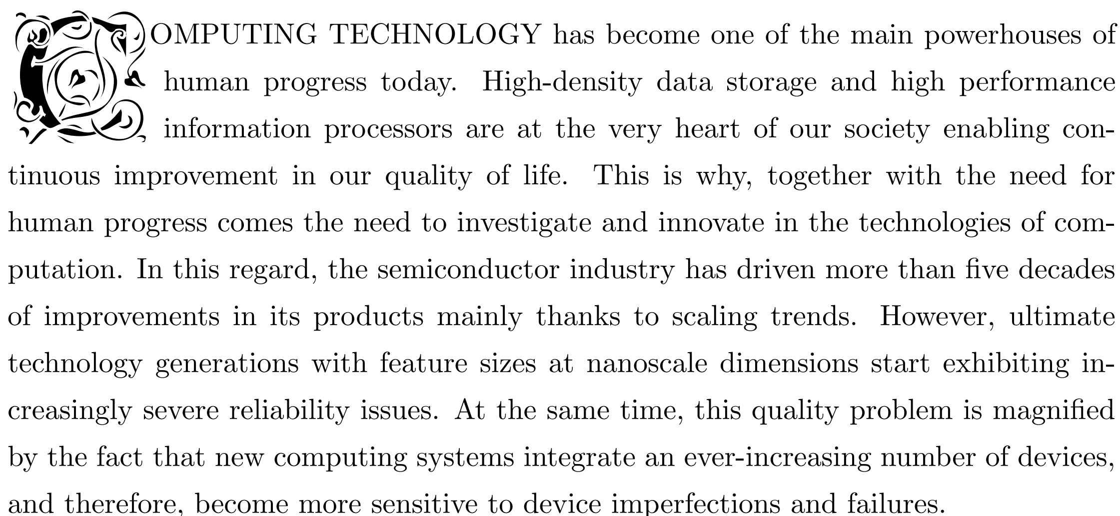
5.1 Critical variability levels (σ_2^*) of DSR effect in a particular case of 2-input AD-AVG with null variability in the first input. 70
5.2 2-input LTG functions and the associated configuration of weights and threshold. 80
5.3 Tolerance Parameter d_{max} of 2-input LTG functions with different levels of variability. 87
5.4 Number of devices versus level of redundancy for NAND multiplexing and AC-LTG 89

Abbreviations

AC-LTG	**A**veraging **C**ell-**L**inear **T**hreshold **G**ate
AD-AVG	**AD**aptive-**AV**era**G**ing cell
AVG	**AV**era**G**ing cell
CMOS	**C**omplementary **M**etal **O**xide **S**emiconductor
CRMR	**C**ascaded **R**-fold **M**odular **R**edundancy
DSR	**D**egradation **S**tochastic **R**esonance
IC	**I**ntegrated **C**ircuit
ITRS	**I**nternational **T**echnology **R**oadmap for **S**emiconductors
MAJ	**MAJ**ority gate
MOSFET	**M**etal–**O**xide–**S**emiconductor **F**ield–**E**ffect **T**ransistor
NWFET	**N**ano**W**ire **F**ield-**E**ffect **T**ransistor
U-AVG	**U**-**AV**era**G**ing cell
pA-RMR	**p**artially **A**synchronous-**R**-fold **M**odular **R**edundancy
RMR	**R**-fold **M**odular **R**edundancy
SE	**S**oft **E**rror
SEU	**S**ingle **E**vent **U**pset
SIA	**S**emiconductor **I**ndustry **A**ssociation
TFET	**T**unnel **F**ield-**E**ffect **T**ransistor
TMR	**T**riple **M**odular **R**edundancy

Chapter 1

Introduction



COMPUTING TECHNOLOGY has become one of the main powerhouses of human progress today. High-density data storage and high performance information processors are at the very heart of our society enabling continuous improvement in our quality of life. This is why, together with the need for human progress comes the need to investigate and innovate in the technologies of computation. In this regard, the semiconductor industry has driven more than five decades of improvements in its products mainly thanks to scaling trends. However, ultimate technology generations with feature sizes at nanoscale dimensions start exhibiting increasingly severe reliability issues. At the same time, this quality problem is magnified by the fact that new computing systems integrate an ever-increasing number of devices, and therefore, become more sensitive to device imperfections and failures.

Taking as a precedent the current state of computer technology and the expected issues of future nanoelectronics, this thesis aims at providing in-depth insight on how to compute reliably in the nanoscale era. Based on the principles of redundancy and reliable computing established by John von Neumann in the 1950s, we explore several fault-tolerant architectures based on hardware redundancy with particular emphasis on nanotechnology issues. In this sense, we extend conventional redundancy and reliability framework to three new dimensions, namely heterogeneity, asynchrony and hardware hierarchy. This chapter is organized as follows. In Section 1.1, the current context of nanoelectronics is presented as well as the main challenges it faces. In Section 1.2, the need for reliable computation is highlighted together with the need for developing fault-tolerant techniques. Finally, the outline of this thesis is presented in Section 1.3.

1.1 The Challenges of Nanoelectronics

The origins of modern electronic computers date back to the invention of the transistor in 1947 [3, 4] and the IC in 1959 [5, 6]. Since then, the semiconductor industry has been able to improve continuously the productivity and performance of its products based on the CMOS scaling trend. Indeed, thanks to clever engineering solutions and improved device architectures, designers and manufacturers have made possible to exponentially decrease the minimum feature sizes of transistors. As a consequence, chip integration levels have been roughly doubled every two years practically following the Moore's Law [7] (see Figure 1.1). Scaling has also enabled the production of faster and less energy consuming devices. Another important benefit of the scaling trend has been the reduction in the manufacturing cost. As a result, high-quality and inexpensive electronic applications with increasingly large memories and higher speeds of computation have been released into the market every year. Not to mention the huge economic impact it has had in our society: according to SIA [8], around 25 billion dollars are being invested every year in semiconductor technologies, of which about one half corresponds to the segment market of microprocessors.

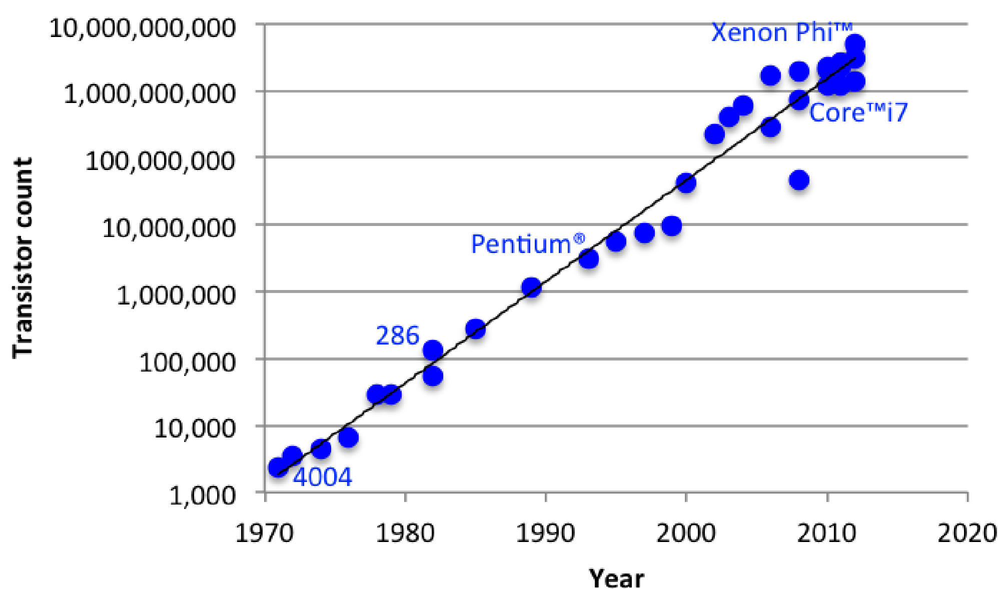


FIGURE 1.1: Transistor count of Intel's processors from 4004 to Xeon Phi against the year of introduction. (Source: Intel)

From the Intel processor 4004 delivered in 1971 until the Intel Xenon Phi delivered in 2012, minimum transistor dimensions have been reduced by three orders of magnitude (from 10 μm to 22 nm), the performance has increased by three orders of magnitude (from 1 MHz to 3.16 GHz), and the transistor count has increased by six orders of magnitude (from 2,300 to 5,000,000,000). However, despite this great success and the relatively

small effort required by the semiconductor industry in the past, the years of "happy scaling" seem to be coming to an end, see Figure 1.2. Every year, technology scaling becomes more and more challenging as it deepens into the nanoscale regime with minimum feature sizes below 100 nm. Current fabrication methods based on photolithography are being pushed toward fundamental physical limitations. Sub-wavelength pattering, deterioration of electrical characteristics due to short-channel, quantum mechanical and transport effects, and intrinsic parameter fluctuations resulting from the discreteness of charge and the granular nature of matter are some of the main difficulties facing the scaling trends today. While traditionally the main focus of semiconductor industry designs was the trade-off between area, delay, and power, reliability appears today as an unavoidable issue [9]. Process variability (random and systematic), SE (soft errors), and device (transistor performance) degradation induce increasing levels of unreliability in the IC components. Therefore, the production of ideal transistors or ICs with very low defect rates becomes increasingly improbable. In the same way, the percentage of operating chips at the end of the manufacturing process, usually referred as yield, decreases drastically with the latest CMOS technology generations requiring adaptive and fault-tolerant designs [10].

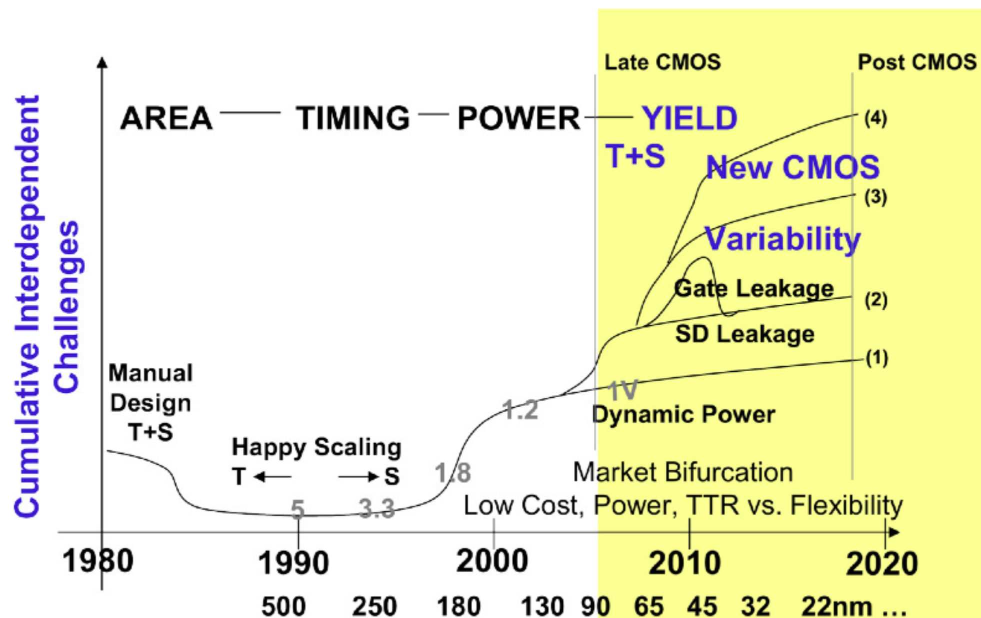


FIGURE 1.2: Cumulative interdependent challenges as a function of time and technology generation. (Source: G. Declerck, Keynote talk, VLSI Technol. Symp. 2005 [11])

On the other hand, research communities are currently investigating several new alternative information processing devices to substitute or extend the functionality of the current CMOS platform [12]. However, high level of variations, performance degradation due to stress of materials, and high rates of manufacturing defects are expected for both ultimate CMOS and beyond CMOS technology generations [9, 13, 14].

1.2 Reliable Computation and Fault-tolerance

As stated before, information processing devices have become almost indispensable to sustain and even enhance the welfare of humankind. From devices that we use everyday such as cell phones or personal computers, to much more complex computing systems including medical equipment or aircraft control systems, they all have in common the use of ICs in order to process information. Obviously a very important characteristic of these systems is the level of reliability they provide. Depending on the particular application a single error could lead to catastrophic consequences, for example the control system of a nuclear plant. In other cases, such as video image encoders, it is perfectly admitted a certain amount of errors. For this reason, the analysis of reliability and the proposal of fault-tolerant techniques is of particular interest today due to the high levels of uncertainty expected for near future computing technologies [15].

Nevertheless, lowering defect rates to a level where perfect or nearly perfect devices can be produced at reasonable yields, as traditionally with conventional CMOS technology, has become impossible today. Reliable computing is no longer possible if we keep assuming ideal compounding devices. Under these circumstances, we need a change in the ICs design paradigm.

Back in the 1950's, John von Neumann analyzed this problem and proposed the production of *reliable organisms/systems from unreliable components* [2]. Indeed, instead of relying on perfect components von Neumann proposed to design defect-tolerant electronic circuits that could operate correctly even though compounding devices included a significant number of manufacturing defects. In this thesis we extend this fundamental principle and explore new dimensions of reliable computation.

1.3 Outline of the Thesis

This thesis provides an overview of fault-tolerant systems based on hardware redundancy focusing on three main issues: 1) heterogeneity, 2) asynchrony and, 3) hardware hierarchy. The organization of this work is as follows. Chapter 2 presents a brief overview of future nanoelectronics with a description of some key emerging nanodevices. A discussion on fault modeling, variability characterization and degradation is also provided together with a discussion on the reliability metrics. In Chapter 3, a review on the concept of redundancy is presented and the framework proposed by John von Neumann is described. It is also provided an overview of the main fault-tolerant architectures proposed in the literature and a particular study on the possible redundancy managers.

Finally, a brief review on the fundamental error bounds for reliable computing is presented together with a consideration on asymmetric errors. In Chapter 4, the motivation for this thesis is presented with some details on the main contributions of this work. In Chapter 5, a first extension on the redundancy concept is made taking into account the heterogeneous nature of real electronic components. An adaptive averaging cell is introduced and analyzed in detail. It is also presented a possible implementation of reliable gates using the averaging cell on linear threshold gates. In Chapter 6, time-aware reliable design is considered by proposing and analyzing a partially-asynchronous structure. In Chapter 7, the hardware hierarchy is taken into account in order to reduce the redundancy effort by properly distributing redundancy throughout system layers. Finally, some concluding remarks are presented in Chapter 8.

Chapter 2

Nanoscale Technology Precedents

THE REALM of NANOELECTRONICS beyond classical CMOS technology has begun and with it a vast new world of challenges and opportunities has been opened up for the research community. While ultimate CMOS technology generations approach inexorably to its physical limits, current nanoelectronic communities including universities, research institutes, and industrial research laboratories are making huge efforts to develop new nanometer-sized information processing devices. It is sought to step from current technology state towards much more highly integrated technologies with billions of devices in the same chip in order to sustain the historical IC scaling pace and the reduction of cost-per-function in the future. This step can be done either 1) by extending the functionality of standard CMOS using heterogeneous technologies with new emerging devices (also known as "More than Moore") or 2) by stimulating the invention of completely new information processing paradigms with new emerging devices and architectures (also known as "Beyond CMOS") [16]. However, in both cases it is required a serious effort in the characterization of these emerging nanodevices as well as in the development of new reliable architectures to account for their expected higher rates of defects and variability.

This chapter revisits several emerging nanotechnologies representatives of the future computing devices and introduce possible models to characterize and analyze them. The organization is as follows. In Section 2.1, a brief review of the most relevant emerging research devices is presented highlighting their main properties. In Section 2.2, a model of faults and variability for the analysis and simulation of future nanodevices is presented, and in Section 2.3, a model to account for aging and degradation is introduced. Finally, a metric for characterizing the reliability of computing systems is presented in Section 2.4.

2.1 Nanoelectronics and Nanodevices

There is a wide range of nanodevices under investigation today. Figure 2.1 depicts a representative taxonomy of emerging research devices. It enumerates the various options in five different categories from the lowest physical layer representing the computational state variable to the highest one representing the architecture. The elements shown in the red-lined yellow boxes constitute the current CMOS platform technology. The other entries summarize individual approaches that may provide new highly scalable information processing paradigms.

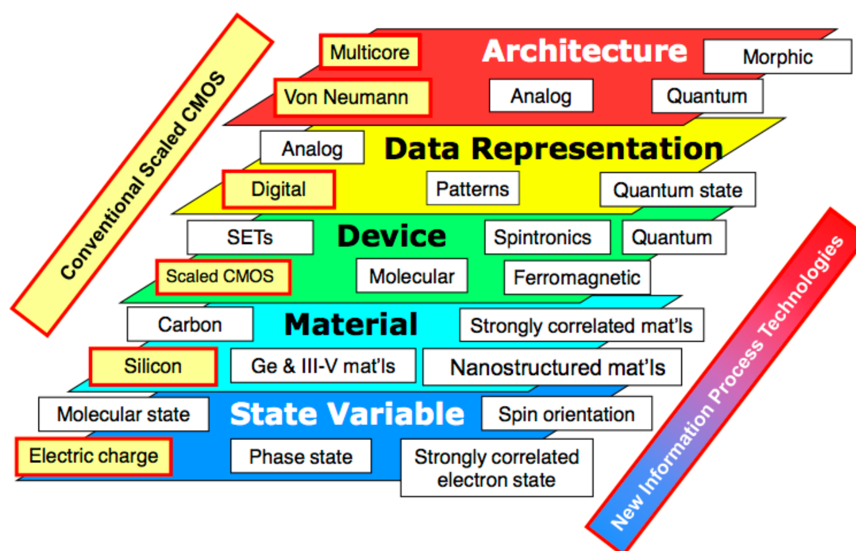


FIGURE 2.1: Taxonomy for emerging research information processing devices (technology entries are representative but not comprehensive). (Source: ITRS [16])

In the following subsections we present a review of two emerging nanoscale research devices, namely Nanowire Field-Effect Transistors (NWFETs), and Tunnel Field-Effect Transistors (TFETs). We have selected these technologies as the most representative and promising candidates to implement future nanoscale processing systems.

2.1.1 Nanowire Field-Effect Transistors (NWFETs)

NWFETs are structures in which the conventional planar MOSFET channel is replaced with a semiconducting nanowire. In these nanodevices, current flows through the nanowire or is pinched off under the control of the voltage on the gate electrode, which surrounds the nanowire, see Figure 2.2. For this reason they are also known as "gate-all-around" transistors. However, because of their small size, single nanowires can't carry enough current to make an efficient transistor. Researchers are currently working on gate-all-around transistor architectures based on small forest of nanowires that are controlled by the same gate and so act as a single transistor [17, 18].

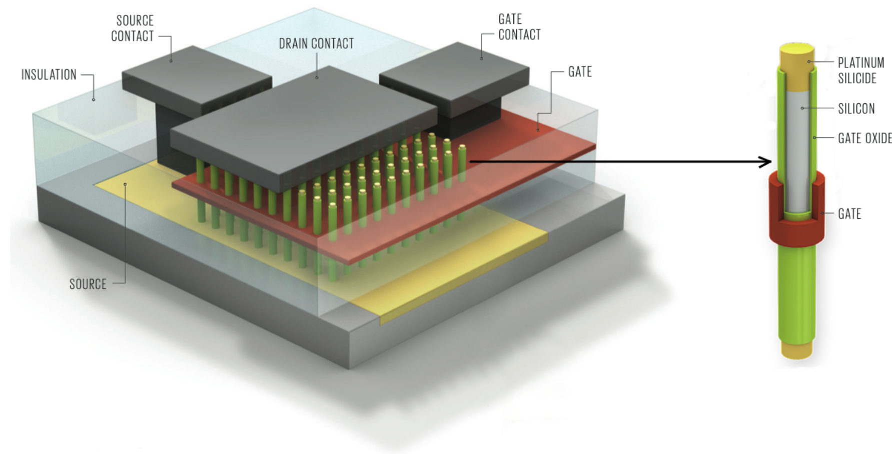


FIGURE 2.2: Structure of a Nanowire Field-Effect Transistor (NWFET). (Source: A. Hellemans, IEEE Spectrum [19])

Nanowires may be composed of a wide range of materials and it has been demonstrated that their diameters may be as small as 5 nm [20]. At low diameters, these nanowires exhibit quantum confinement behavior, i.e, 1-D conduction, that may permit the reduction of short channel effects and other limitations to the scaling of planar MOSFETs. Large logic circuits can be implemented using nanowires because Boolean gates up to a complexity of an XOR gate exhibit signal restoration and they can drive other logic gates [21]. High gain nanowire based inverters with robust noise margins have been also demonstrated.

2.1.2 Tunnel Field-Effect Transistors (TFETs)

TFETs are gated reverse-biased p-i-n junctions, see Figure 2.3, whose switching behavior is expected to be much steeper than conventional MOSFETs, that have 60 mV/dec subthreshold swing at room temperature [22]. Power dissipation is one of the main limitations of future nanoelectronic circuits. Decreasing the supply voltage reduces the energy needed for switching, but current FETs require at least 60 mV of gate voltage to increase the current by one order of magnitude at room temperature. TFETs avoid this limit by using quantum-mechanical band-to-band tunneling, rather than thermal injection, to inject charge carriers into the device channel.

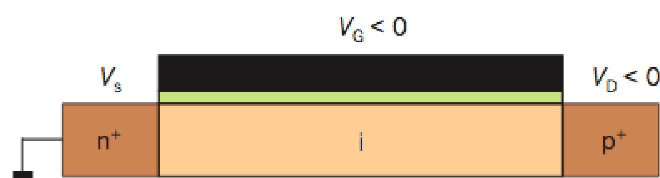


FIGURE 2.3: Structure of p-type TFET with applied source (V_S), gate (V_G) and drain (V_D) voltages. (Source: Ionescu et al., Nature [23])

TFETs are under research due to their potential for low standby leakage current and as promising candidates for future logic circuits operating with a supply voltage less than 0.5 V. Recent reports suggest that TFETs could be also considered for implementing high performance switches by using appropriate heterogeneous structures [24]. For example, TFETs based on ultrathin semiconducting films or nanowires could achieve a 100-fold power reduction over CMOS transistors, so integrating TFETs with CMOS technology could improve low-power integrated circuits [23].

2.2 Variability and Fault Modeling

It is difficult to predict exactly which types of nanodevice will end up implementing future computing systems. However, it is widely accepted by the research community that they will have associated much higher degrees of uncertainty than conventional MOSFETs. For this reason, it is very important to define a comprehensive variability and fault model in order to analyze the reliability of different circuit architectures, compare results from different approaches, and propose new fault-tolerant mechanisms. In this section we introduce the fault model we will use along this thesis.

Since the publication of the first study on circuits reliability in 1950's by John von Neumann [2], many different fault models have been proposed at different levels of abstraction. Some of them are general and can therefore be applied to different architectures and technologies. For example, the stuck-off/stuck-open and stuck-on/stuck-short fault models. Others are specific for a particular technology [25]. In these cases, fault models tend to be much more complex and difficult to use. In this thesis we will use a general and technology-independent fault model based on variability. Our goal is to thoroughly analyze the reliability of computing systems being able to detect, to explore and discuss all its consequences without getting lost in too specific details of technology.

Before presenting our variability-based fault model let us clarify the meaning of three different terms in the context of computing technology which are closely related to system reliability analysis, namely defect, fault, and error. These definitions are coherent with the taxonomy of dependable and secure computing from Avizienis et al. [26].

- **Defect** is a physical problem with a final manufactured system which differs from the intended design as a result of an imperfect fabrication process.
- **Fault** is an incorrect state of a system due to manufacturing defects, component failures, environmental conditions, or even improper design. A fault is active when it causes an error, otherwise it is dormant.

- **Error** is an incorrect output of a system. The cause of an error is always a fault. Errors can be classified into three main groups according to their stability and occurrence:
  - *Permanent errors* are caused by irreversible physical changes/defects in the produced system. Normally, permanent errors are originated in the manufacturing processes, but they can occur also during the usage of the circuit, especially when it gets old and starts to wear out. As its name suggest, once a permanent error occurs, it will not vanish and therefore these type of errors can be easily identified because the test to detect them can be repeated with the same results.
  - *Intermittent errors* are occasional error bursts that usually repeat themselves every now and then but are not continuous as permanent errors. They are caused by unstable or marginal hardware activated by environment changes such as temperature or voltage variations. This type of errors can also be observed when a circuit operates incorrectly only for some input instances because some path of the circuit may be slower than supposed to but not totally inoperable. Intermittent errors are very hard to detect because they may occur only under certain environment constraints or for some specific input combination.
  - *Transient errors* are temporal single errors caused by some temporary environmental conditions, such as external radiation or cross-talk. Transient errors do not make any permanent marks on the chip and therefore they are also called soft errors or single-event upset (SEU). The occurrence of transient errors is commonly random and therefore difficult to detect.

Once stated the difference between defects, faults and errors, we proceed with the presentation of our fault model. We take as a reference the John von Neumann probabilistic computing framework and extend it to account for heterogeneity. In this sense, we use a general model applicable to different technologies and we include a relevant characteristic of future nanoelectronic circuits, i.e., the heterogeneity.

Von Neumann error framework assumes that faults of different gates are uncorrelated. It also assumes that each gate flips the output with a precise probability ϵ, while the interconnections (input and output lines) are assumed ideal/error-free [2]. Our fault model also assumes statistic independence between faults, but we dismiss the assumption of homogeneous faults (same signal flip probability for all the gates). Instead, we use a variablity-based model in order to allow each device to have associated a different fault probability. Basically, we model each device output signal y_i as composed by two

components:

$$y_i = y + \eta_i, \tag{2.1}$$

where y is the error-free version of the signal and η_i is an independent drift that modifies the signal. Drifts η_i account for all transient faults affecting the manufactured devices. As a result of this model, signals y_i are observed in the system as continuous voltage levels, where 0 and V_{cc} stand for ideal logical values '0' and '1', respectively. Without loss of generality, we use $V_{cc} = 1\ V$ to simplify future discussion. Drift magnitudes η_i are modeled as Gaussian random variables with null mean and standard deviation σ_i, $\eta_i \sim N(0, \sigma_i)$.

As stated before, we consider drifts as mutually independent, thus our fault model explores random effects between replicas. Systematic alterations, which affect all the replicas in the same direction, are specifically evaluated in subsection 3.3.2.1 and they are shown to cause independent effects on the reliability of the considered fault-tolerant structures. Therefore, since random and systematic effects can be studied and counteracted independently our results are applicable in any case. Moreover, systematic effects are usually static and they can be easily mitigated by calibration in the initial stages of system operation.

Using this fault model we can calculate now the probability of error of each signal y_i. Assuming that digital signals are interpreted as logic '0's or '1's by thresholding decisions, errors occur if and only if the drift magnitude $\eta_i = y - y_i$ reaches $V_{cc}/2$ or $-V_{cc}/2$, depending on the logic value y. Using the complementary Gauss error function we can analytically formulate the error probability associated to each signal y_i as follows:

$$P_e = \int_{V_{cc}/2}^{\infty} f_\epsilon(\epsilon)\ \mathrm{d}\epsilon = \frac{1}{2} \times \operatorname{erfc}\left(\frac{V_{cc}}{\sqrt{8}\sigma_i}\right). \tag{2.2}$$

Figure 2.4 depicts the relationship between error probability P_e and the ratio σ_i/V_{cc}. It presents a monotonically increasing behavior. Thus, given a maximum admissible error probability $P_e^{\max}$, there is a maximum admissible standard deviation $\sigma_{i\ \max}$ for any given V_{cc}. If we take for example $P_e^{\max} \equiv 10^{-4}$ as the reference value for the maximum admissible error probability, then the maximum admissible standard deviation $\sigma_{i\ \max}$ that fulfills this error specification is 0.1344 V (being $V_{cc} = 1\ V$).

2.3 Aging and Degradation Modeling

Emerging nanodevices will pose new challenges for the design and manufacturing of reliable computing systems. Variability, soft errors, and device (transistor performance)

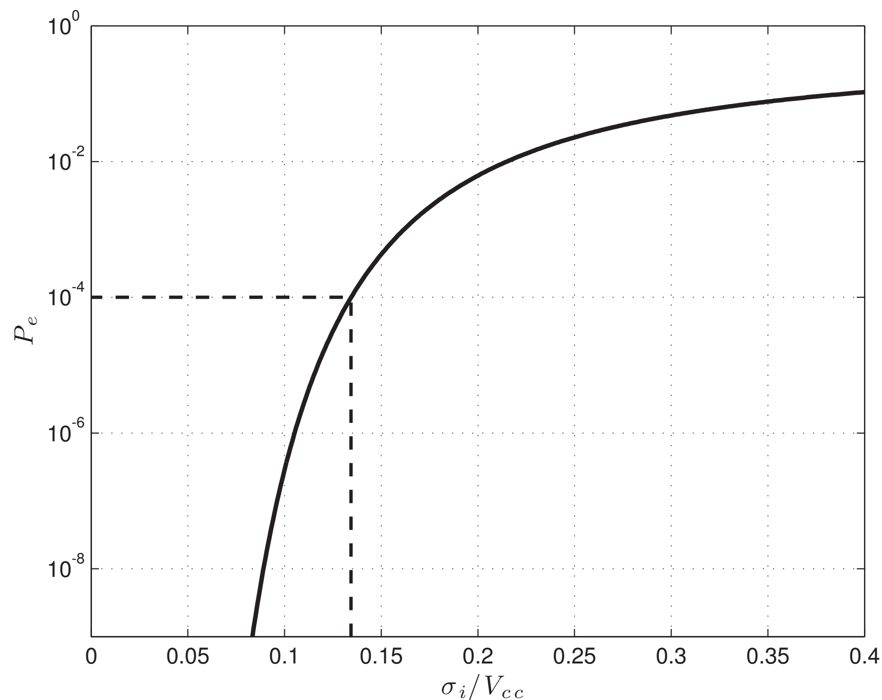


FIGURE 2.4: Error probability P_e against the ratio between the standard deviation σ_i and V_{cc}. The highlighted point corresponds to the reliability reference value $P_e^{\max} \equiv 10^{-4}$ and the corresponding maximum admissible standard deviation $\sigma_{i\ \max}/V_{cc} = 0.1344$.

degradation are considered the main difficulties to be faced in the future [9]. In connection with this, our previously introduced fault model can account for variability and soft errors but always with static rates of errors and faults. In this section we present the degradation model that we use in combination with our fault model in order to include time-varying effects caused by aging, external aggression, or degradation.

The process of aging and the dynamics of the variability associated to each nanodevice depend on the particular technology and the environmental conditions. It is therefore very difficult to establish a relationship between degradation and time. However, based on the main properties of degradation and using a Probability Distribution Function (PDF), we can define a general degradation model that serves to simulate and analyze its effects. In particular, we use the Gamma distribution function to generate positive random values of variability σ_i^2 (variance), see Equation (5.15), and update their values according to increasing levels of degradation. Indeed, thanks to the infinite divisibility property of this statistic distribution we can easily simulate the influence of increasing amounts of degradation by simply adding random Gamma-distributed increments to the initial input variances.

$$\sigma_i^2 \sim \Gamma(x; k, \phi) = \begin{cases} \frac{1}{\phi^k \Gamma(k)} x^{k-1} e^{-x/\phi} & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

The simulation process comprises the following steps:

1- First we generate the initial variability levels σ_i^2 using the Gamma distribution function, see Equation (5.15), with scale parameter $\phi = 2$. These levels of variability correspond to the initial imperfections already present in fresh devices (non utilized) mainly associated to the manufacturing process. In the simulations, this initial stage of the circuits corresponds to the 0 in the axis of degradation in time (horizontal axis) and, unless otherwise specified, we assume a mean value of $E\{\sigma_i^2\} = (2\sigma_{i\ \max})^2$ $(= 0.07V^2)$.

2- Then, in order to simulate successive moments of the circuit's life we need to calculate the input variabilities after the effect of increasing amounts of degradation. To do so we estimate recursively the input variances at consecutive stages of degradation. The variances in the stage $n+1$ are computed by adding a positive random increment α_i to the variance in the previous degradation stage n:

$$\sigma_i^2[n+1] = \sigma_i^2[n] + \alpha_i[n] \tag{2.4}$$

The increments used to update the input variances are also generated with the Gamma distribution function and reflect the effect of degradation occurred during the time between consecutive stages of degradation. In the simulations, we define the degradation in time unit so that it corresponds to a mean increase in the replicas' variance σ_i^2 of magnitude $E\{\alpha_i\} = (\sigma_{i\ \max})^2$ $(= 0.02V^2)$. Using this normalized time-scale we avoid using particular degradation-time relationships and therefore our results are technology independent and can be applied to each particular case.

Figure 2.5 depicts five variability profile samples generated using our degradation model. In the graph as well as in future analysis we use a degradation normalized temporal unit called "degradation in time". With this unit, instead of relating particular amounts of degradation to time, we make our model more general and technology independent. We basically focus our degradation metric on measurable circuit magnitudes that vary with degradation and associate degradation in time units to these increments. This technique was utilized in other studies, e.g., Brown et al. [27]. In these cases time is not directly related to degradation but measurable percentages of parameters shift due to degradation.

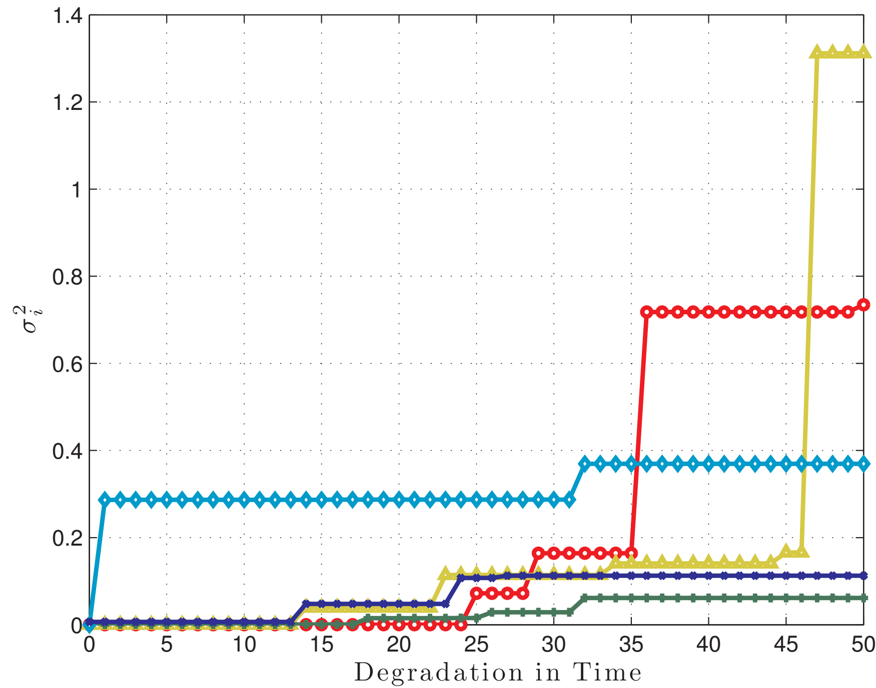


FIGURE 2.5: Five samples of variability σ_i^2 profile against degradation in time units generated using our degradation model.

2.4 Reliability Characterization

We need a reliability metric in order to predict, optimize and design future computing systems composed of unreliable nanodevices. If we look at the literature, we can find a number of parameters defined for this purpose. Some of the most important are summarized below:

- **Reliability** $R(t)$, according to IEEE, is the ability of a system or component to perform its required functions under stated conditions and for a specified period of time.
- **Mean time to failure (MTTF)**, also called *expected life*, is the expected value of the time to failure of a system. It can be calculated by integrating the reliability parameter $R(t)$ from time zero to infinity.
- **Yield** is the percentage of acceptable parts (systems or components) among all parts that are fabricated.

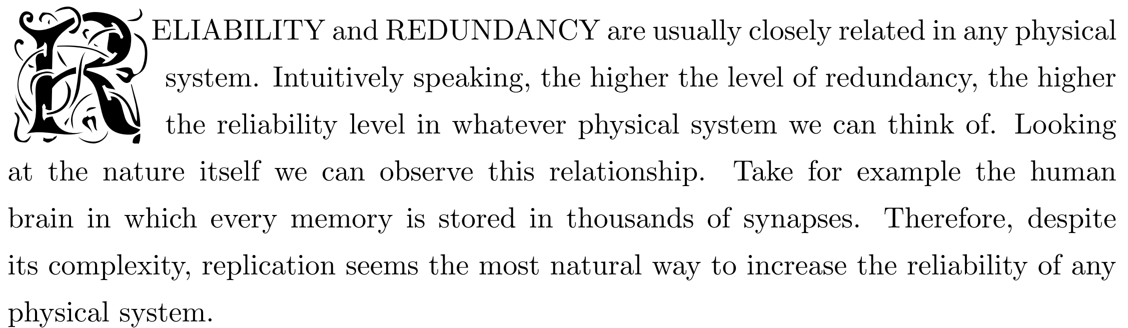
Among all the options available, in this thesis we chose the yield parameter to characterize the reliability of processing systems for several reasons:

- First, it can be computed from the error probability (Yield $= 1 - P_e$), which is directly related to the variability and fault model used along this thesis, see (2.2).
- Second, the yield parameter is an instantaneous measure of reliability (at a particular time) and therefore it allows us to easily simulate time-varying effects like degradation. Using an instantaneous reliability measurement also makes it possible to clearly observe and analyze the system behavior under these circumstances.
- Additionally, parametric yield measurements are widely used by IC designers because they allow to easily compare different techniques and clearly evaluate the impact of each modification or improvement in the methodologies and proposed designs.

Therefore, in the remainder of this thesis we will use yield as a measure of reliability.

Chapter 3

Reliable Redundancy Architectures



RELIABILITY and REDUNDANCY are usually closely related in any physical system. Intuitively speaking, the higher the level of redundancy, the higher the reliability level in whatever physical system we can think of. Looking at the nature itself we can observe this relationship. Take for example the human brain in which every memory is stored in thousands of synapses. Therefore, despite its complexity, replication seems the most natural way to increase the reliability of any physical system.

Obviously, a system with a faultless design and implemented with perfect components does not require any kind of redundancy to improve its reliability, because it is in itself perfect. Still, it is often not possible to produce ideal or almost perfect devices and in that case some mechanism is needed to improve reliability unless we accept the associated risk of failure. In this chapter we revisit the idea of redundancy to produce reliable electronic circuits from unreliable devices as first introduced by von Neumann. The organization of this chapter is as follows. In Section 3.1, we list the basic principles of redundancy exposed by von Neumann in his celebrated work on probabilistic logics. In Section 3.2, a review of some of the most important fault-tolerant architectures based on redundancy that can be found in the literature is presented. In Section 3.3, we focus on the redundancy managers explaining their characteristics and presenting different approaches. Finally, a review on the fundamental error bounds for reliable computing is presented in Section 3.4. A particular contribution is added to this chapter on asymmetric error designs for enhanced fault-tolerant computing.

3.1 Redundancy by von Neumann

Before proposing new ideas in the field of redundancy it is very interesting to go back to the origin of the idea itself. For this purpose we shall focus on the excellent work of von Neumann, who in 1956 wrote a paper entitled “Probabilistic logics and the synthesis of reliable organisms from unreliable components” [2]. In that paper he established the fundamental principles of redundancy in its modern engineering meaning [28]. Afterwards, this idea of constructing reliable computing circuits by the redundant use of unreliable components spread in the scientific community and was developed and applied in other contexts [29, 30].

In the center of von Neumann contribution we find the essence of the reliability problem, which we summarize below.

3.1.1 Error framework

In the first place, von Neumann defines a simple but comprehensive error framework:

- Every basic component has associated a (precise) error/failure probability ϵ in any operation.
- This malfunctioning is assumed to occur statistically independently of the general state of the system and the occurrence of other errors.

Although he observed that another error model that does not assume independent failures would be more realistic, von Neumann prefers to adopt the simpler assumption for ease of analysis.

3.1.2 Difficulty of errors

After defining a model for errors, von Neumann shows through a very clear example the difficulty introduced by errors in digital computing systems

Consider the memory system of Figure 3.1, which is clearly described in the von Neumann paper [2]. Once stimulated, this circuit should continue to emit pulses forever. However, suppose that the basic component has associated an error probability ϵ. If the system receives a stimulation at time t and no later ones we can compute the probability of correct output in subsequent cycles:

Let the probability that the system is still excited after s cycles be denoted ρ_s, this corresponds to the probability of correct operation. Then the recursion formula for small ϵ can be written as:

$$\rho_s - \frac{1}{2} = (1-2\epsilon)^s \left(\rho_0 - \frac{1}{2}\right) \approx e^{-2\epsilon s} \left(\rho_0 - \frac{1}{2}\right).$$

The quantity $\rho_s - \frac{1}{2}$ can be taken as a rough measure of the amount of discrimination in the system after the s-th cycle. According to the above formula, $\rho_s \to \frac{1}{2}$ as $s \to \infty$, a fact which is expressed by saying that, after a long time, the memory content disappears, since it tends to equal likelihood of being right or wrong. i.e. to irrelevancy.

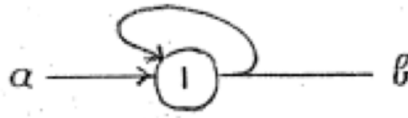


FIGURE 3.1: Memory system.

3.1.3 Problem of reliable computing

After defining the context for reliable computing, von Neumann introduces for the first time the idea of using control mechanisms to prevent the accumulation of errors. Among the two possible options to increase the reliability of computing systems, namely fault-intolerance (or fault-avoidance) and fault-tolerance [31, 32], he proposes to organize the system itself in order to mask or remove the effects of faults, i.e., fault-tolerant design. In fact, he is the first one to propose and analyze in detail fault-tolerant techniques for reliable computing.

He defines the problem of reliable computing that underlies all studies on fault-tolerance, reliable architectures and also this thesis. The problem is divided in two main questions:

- Given $\delta > 0$, can a corresponding system be constructed from faulty components, which will perform the desired function and will commit an error (in the final result. i.e. output) with probability $\leq \delta$? How small can δ be prescribed?
- Are there other ways to interpret the problem which will allow us to improve the accuracy of the result?

During the rest of his work, von Neumann proposes several fault-tolerant techniques based on redundancy to improve the reliability of computing systems. The basic idea is to use more resources than normally necessary so as to upgrade system reliability.

He analyses the reliability of triplicated systems with single line and also proposes the use of multiplexing strategy. For each structure we also provides interesting results about the associated fault tolerance capabilities. Some of these techniques will be discussed in Section 3.2 together with other techniques well-known in the literature.

Apart from studying particular architectures, he also finds fundamental limits in the maximum admissible error rates of the compounding devices. These limits along with other developed to date will be presented in Section 3.4.

3.2 Fault-tolerant Architectures based on Redundancy

Nowadays, we can find in the literature a large collection of fault-tolerant techniques based on redundancy [1, 2, 31, 33]. All of them use the same fundamental principle introduced by von Neumann but applied in many different ways. A general classification for all redundancy based techniques distinguishes two main groups:

- **Static redundancy techniques**, also called masking or massive redundancy techniques, in which fault tolerance is implemented into the system structure and is therefore inherent to the operation of the system. They are able to mask all types of faults (permanent and transient) and can be classified into three categories: hardware, time, and information redundancy techniques.
- **Dynamic redundancy techniques**, also called selective, stand-by or sparing redundancy techniques, are based on fault detection, location, containment, and recovery. These techniques are not useful for transient faults because they need significant amount of time to detect a fault and activate the corresponding circuitry to perform the corrective action. The main benefit of dynamic redundancy is higher reliability for permanent and multiple faults and lower overhead than static techniques.

In the following subsections we revisit some of the most relevant fault-tolerant techniques based on redundancy.

3.2.1 Modular Redundancy

Triple Modular Redundancy (TMR) is probably the most intuitive and well-known redundancy based technique. It was introduced by von Neumann [2] and later developed in many papers [34–36]. The TMR technique, graphically depicted in Figure 3.2, basically consists of triplicating a functional module that we want to improve and combining the

outputs with a majority criterion. The voting circuitry is usually referred as majority gate or voter.

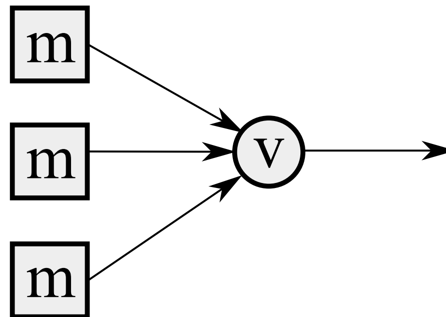


FIGURE 3.2: Triple Modular Redundancy (TMR) schematic.

TMR can be easily generalized to structures with more than three replicated modules by using majority gates with higher number of inputs. The only restriction in this case is that the number of replicas has to be odd ($R = 3, 5, 7, \dots$) in order to avoid ties in the voting operation. This scheme is known as R-fold Modular Redundancy (RMR), see Figure 3.3-a. There are also other approaches based on the TMR such as the Cascaded R-fold Modular Redundancy (CRMR) of Figure 3.3-b in which each module working in parallel is another RMR structure [37–39].

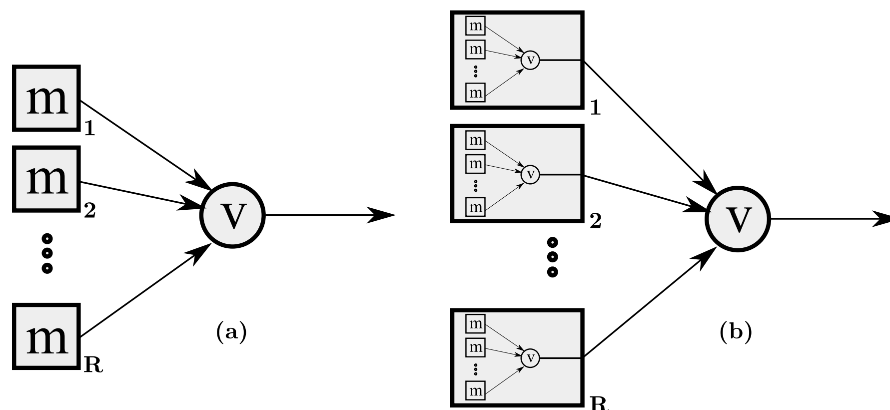


FIGURE 3.3: (a) R-fold Modular Redundancy (RMR); and (b) Cascaded R-fold Modular Redundancy (CRMR) schematic.

The weak point of this technique is probably the voter, since a fault in this component could cause the whole circuit to fail. This problem was already identified by von Neumann, who also deduced, without proving it, that any circuit built out of 3-input components (TMR voters) cannot be made reliable if the error probability of the components ϵ is higher than $1/6$. In Section 3.4, we will see that indeed $1/6$ is the threshold value for reliable computation with 3-input components.

3.2.2 Multiplexing

The multiplexing technique was also introduced by von Neumann in his celebrated paper [2] in 1956. This technique was conceived as method to remove the weak point of the RMR and construct reliable structures whose malfunction could not be caused by the failure of a single component (the RMR voter) or a small set of components. Von Neumann presented two versions of the multiplexing technique based on NAND gates of MAJs (majority gates), although he focused his analysis on the former. Figure 3.4 depicts a schematic view of the NAND multiplexing technique. The grey boxes (U) correspond to permutation units that

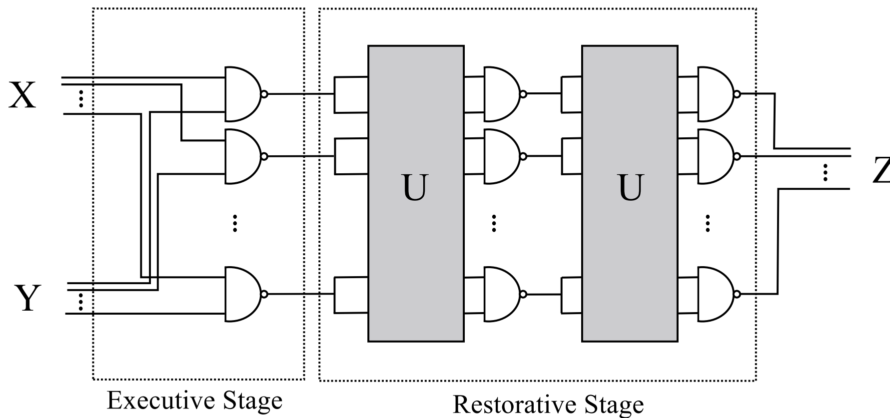


FIGURE 3.4: NAND multiplexing schematic.

The multiplexing technique incorporates what von Neumann called the "multiple line trick". Instead of having a single output line and therefore a single voter that could cause the whole system to fail, the trick consists of carrying the output on a bundle of N lines. A positive number $\Delta(< 1)$ is chosen and the stimulation of $\geq (1 - \Delta)N$ lines of the bundle is interpreted as a positive message, i.e. logic '1', and the stimulation of $\leq \Delta N$ lines as a negative message, logic '0'. Any other number of stimulated lines is interpreted as malfunction.

In the literature we can find a lot of papers on the multiplexing technique that analyze its behavior, suggest improved designs, and explore its application in future nanoscale technolgies [40–47].

3.2.3 Reconfiguration

While modular redundancy and multiplexing are static redundancy techniques, reconfiguration is a dynamic redundancy technique. This involves quite a different way of

applying redundancy. In dynamic redundancy techniques, redundant parts of the design are only activated when a fault appears and a correcting action is needed. The most representative example of this technique is undoubtedly the “Teramac” computer: a reconfigurable defect-tolerant architecture [48]. “Teramac” is a massively parallel experimental computer with a defect-tolerant architecture that incorporates a high communication bandwidth that enables it to easily route around defects. Indeed, the fundamental idea behind reconfiguration techniques is to avoid faults by modifying the system organization so that the defective parts are no longer used. In the “Teramac” computer, basic unreliable components are assembled in groups to form a configurable logic block (CLB, shown as a sub-unit in Figure 3.5). At the same time, a number of CLBs are grouped together to form an atomic fault-tolerant block (AFTB, larger units in Figure 3.5). The AFTB can be configured to perform some basic set of operations and they are also grouped in clusters which perform some desired functions. Finally, the chip is filled with identical and independent copies of a cluster and thereby, after the appearance of a fault, it is possible to reconfigure the system and continue operating reliably.

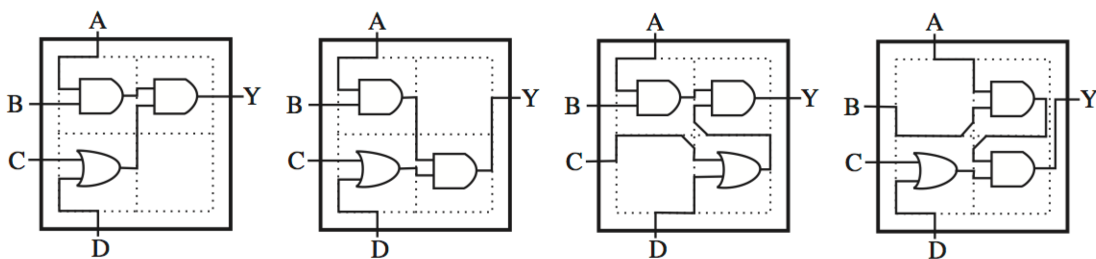


FIGURE 3.5: Basic structure for the reconfiguration technique theory. (Source: Nikolić, et al. [1])

Another well-known approach to reconfigurable redundancy techniques is based on embryonics [49]. As a general comment, reconfiguration techniques are mainly suitable for dealing with manufacturing defects rather than transient errors. This is because the time required to detect, locate and correct a fault is usually much larger than the time limit to suppress the impact of such fault.

3.3 Redundancy Managers

Except the dynamic approach, redundancy techniques typically require the use of redundancy managers. That is, devices that combine the information provided by the

replicas (independent and identical copies of a functional block) according to a majority criterion. The whole system reliability highly depends on the performance of these managers, and therefore, there is a great interest in their design [50].

In the following subsections we review two well-known voters, namely the majority gate and the averaging cell, and present their main features.

3.3.1 Majority Gate (MAJ)

Von Neumann was the first to introduce the "majority organ" (MAJ) and use it to control the errors in redundant structures [2]. The MAJ, graphically depicted in Figure 3.6, outputs the logic value carried by the majority of its inputs. The number of inputs per MAJ (R in Figure 3.6) may vary, but should always be an odd number in order to avoid ties ($R = 3, 5, 7, \ldots$). Thus, an R-input MAJ is able to mask any set of simultaneous faults provided it is in minority, i.e., the number of masked faults is always $\leq (R-1)/2$.

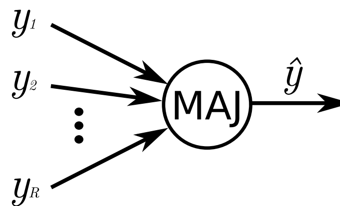


FIGURE 3.6: Majority Gate (MAJ) schematic.

In the following we take as example a 3-input MAJ and show some interesting properties. As clearly analyzed by von Neumann, its operation within the context of redundant structures can be described by the following equation:

$$p_{\text{out}} = \epsilon + (1 - 2\epsilon)\left(3p_{\text{in}}^2 - 2p_{\text{in}}^3\right), \tag{3.1}$$

being p_{in} the error probabilities in the input lines (independent of each other), p_{out} the error probability at the output, and ϵ the failure rate of the MAJ. Figure 3.7 depicts the reliability characteristic described in (3.1) for different values of ϵ. From this result von Neumann deduced in 1956, without proving it, that any network of ϵ-faulty 3-input elements can not be made reliable if $\epsilon > 1/6$. In fact, observing the reliability characteristic of 3-input MAJ we can verify that $\epsilon > 1/6$ implies that p_{out} is always higher than p_{out}. 45 years later, in 1991, Hajek and Weller proved that, actually, 1/6 is the threshold value for reliable formulas of 3-input elements [51]. In Section 3.4, we review all the error bounds for reliable computation that have been found so far.

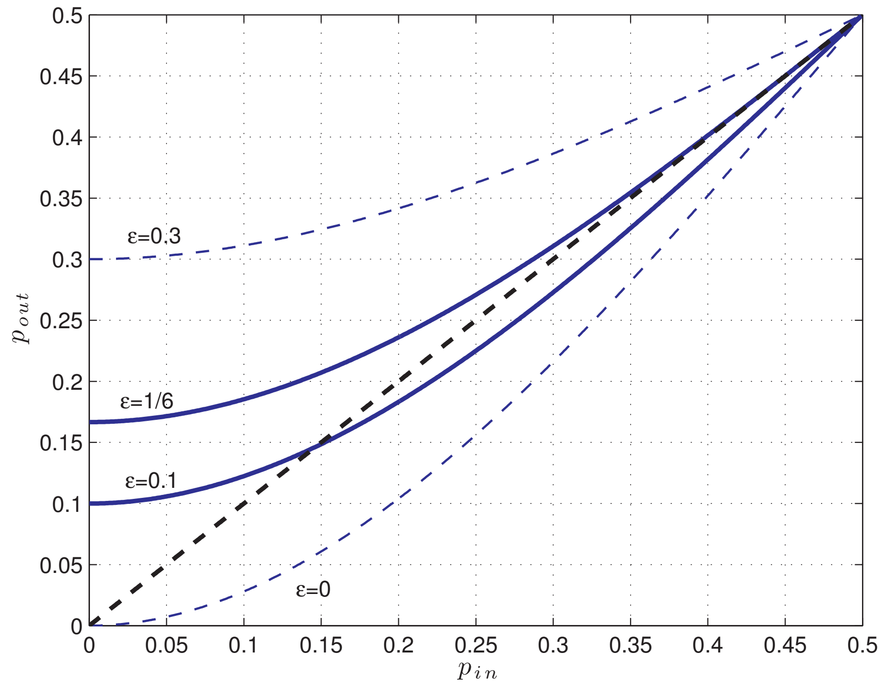


FIGURE 3.7: Output error probability p_{out} against input error probability p_{in} of 3-input MAJ for different values of gate failure rate $\epsilon = \{0, 0.1, 1/6, 0.3\}$.

3.3.2 Averaging Cell (AVG)

The Averaging Cell (AVG), graphically depicted in Figure 3.8, is an analog approach to the majority voting. While the MAJ operates in the digital domain, the AVG performs a weighted average of the replicated inputs in the analog domain, thus is potentially more robust. The AVG stems from the perceptron, the McCulloch-Pitts neuron model [52, 53] and it is widely known for its application in the four-layer reliable hardware architecture (4LRA) [54]. It is associated with fault-tolerant techniques based on redundancy and can calculate the most probable value of a binary variable from a set of error-prone physical replicas.

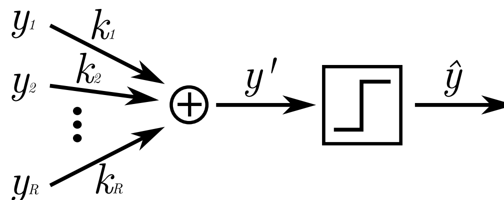


FIGURE 3.8: Averaging Cell (AVG) schematic.

The AVG output $\hat{y}$ is an estimation of the ideal input variable y according to (3.2) and (3.3).

$$y' = W(y_1, \ldots, y_R) = \frac{1}{\sum_{i=1}^{R} k_i} \sum_{i=1}^{R} k_i y_i \tag{3.2}$$

$$\hat{y} = T(y') = \begin{cases} V_{cc} & \text{if } y' \geq V_{cc}/2 \\ 0 & \text{if } y' < V_{cc}/2 \end{cases} \tag{3.3}$$

Where y_i are R error-prone replicas of the ideal input variable y. Using the fault-model introduced in Section 2.2, we can describe the replicas as

$$y_i = y + \eta_i \qquad i = 1, \ldots, R, \tag{3.4}$$

where each replica y_i has associated an independent drift η_i that modifies its ideal value y with a Gaussian random distribution $\eta_i = N(0, \sigma_i)$. As a consequence, input signals y_i are observed in the system as continuous voltage levels. All the averaging weights are positive $k_i \geq 0$, $i = 1, \ldots, R$ and we use normalized weights $c_i = k_i / \sum_{j=1}^{R} k_j$ instead of k_i in order to simplify the mathematical formulation.

When y' is processed by the threshold operation $T(y')$ an error will be produced if and only if the deviation in the weighted average $\Delta = y' - y$ reaches $V_{cc}/2$ or $-V_{cc}/2$, depending on the logic value y. Since this deviation parameter Δ can be expressed as a linear combination of normally distributed variables η_i, by the properties of the normal distribution the probability density function (PDF) $f_\Delta(\Delta)$ can be described as a normal distribution with parameters

$$\mu_\Delta = E\left\{\sum_{i=1}^{R} c_i y_i\right\} - y = 0, \text{ and} \tag{3.5}$$

$$\sigma_\Delta^2 = E\left\{\left(y' - y\right)^2\right\} = \sigma_{y'}^2. \tag{3.6}$$

The variance of the weighted average $\sigma_{y'}^2$ can be expressed in terms of the input variances σ_i^2 and the averaging weights c_i, $i = 1, \ldots, R$:

$$\sigma_{y'}^2 = \sum_{i=1}^{R} c_i^2 \sigma_i^2. \tag{3.7}$$

Despite the general definition of AVG entails the possibility of using unbalanced weights, most studies using this structure are restricted to the case of balanced weights ($c_i = 1/R$ for all i) for different reasons. From now on, we refer to this balanced AVG approach as

the conventional AVG or B-AVG. On one hand, most of the works use the assumption of homogeneous input drifts equivalent to von Neumann's hypothesis of homogeneity (see Subsection 3.1.1), thereby all the inputs are considered to have the same variability level ($\sigma_i = \sigma$ for all i), for example Martorell et al. in [55]. Under these conditions, a balanced weight set produces a weighted average y' with the minimum possible standard deviation $\sigma_{y'} = \sigma_y/\sqrt{R}$ and maximum reliability $(1 - P_e)$. The output error probability P_e decreases with decreasing standard deviation $\sigma_{y'}$ as observed in Section 2.2 (see Figure 2.4). On the other hand, due to the symmetry of the structure the balanced weights approach is also the best possible choice when there is no knowledge about the different levels of variability present in each of the input replicas.

Finally, as a general comment, we would point out that the conventional AVG is an efficient redundancy manager that operates in the analog domain. It is difficult to compare with MAJ because it requires a fault-model that enables analog signals. It does not require odd redundancy factors and it always provides a reliability enhancement since it preserves the mean value y (assumed to be correct) and diminishes the variance of the signal to be read ($\sigma_{y'}^2$).

3.3.2.1 Systematic Effects

This thesis does not consider systematic effects when characterizing the reliability level of averaging structures. However, future scenarios targeted here will probably include this kind of effects. Fluctuations in temperature and voltage as well as systematic variations from manufacturing processes usually imply coherent deviations in many devices of the system at the same time. In this subsection we prove that even when taking into account these systematic effects the results provided using our random drift-based fault model are correct.

In (3.5) and (3.6) the statistical model of the AVG deviation parameter Δ ($= y' - y$) was developed assuming no bias in the input variables, only random effects. The consideration of systematic effects can be taken into account at this point by adding the same non-zero mean value δ to all the input drift variables $\eta_i \sim N(\delta, \sigma_i)$. Applying this change, the statistical distribution of the deviation parameter becomes

$$\mu_\Delta = E\left\{\sum_{i=1}^{R} c_i y_i\right\} - y = \delta, \tag{3.8}$$

$$\sigma_\Delta^2 = E\left\{\left(y' - y\right)^2\right\} - \delta^2 = \sigma_{y'}^2 \tag{3.9}$$

while the variance of the weighted average $\sigma_{y'}^2$ expressed in (3.7) remains unaltered due to the fact that the bias effect δ is common to all the replicas. Under these conditions,

the corresponding error probability P_e is

$$P_e = \int_{V_{cc}/2}^{\infty} f_{y'}(y')\mathrm{d}\epsilon = \frac{1}{2} \times \mathrm{erfc}\left(\frac{V_{cc}/2 - \delta}{\sqrt{2\sigma_{y'}^2}}\right) \tag{3.10}$$

from which we can draw the conclusion that systematic effects and random effects worsen the AVG reliability in two independent ways. The effect of systematic deviation translates into a reduction in the margin of tolerable variation $V_{cc}/2 - \delta$. On the other hand, random fluctuation increases the variance of the weighted average $\sigma_{y'}^2$, see (3.7). By applying averaging techniques with different weighting schemes we can mitigate the impact of random variations through the reduction of the deviation parameter variance $\sigma_{y'}^2$. This is the main target of the AVG techniques. As for the bias effect δ, which is independent of the random variations, it is not a critical issue. Systematic effects are usually static and can be mitigated by calibration in the initial stages of system operation.

3.4 Fundamental Error Bounds

This section provides a review of the fundamental error bounds for reliable computation. The study of fault-tolerant architectures is of great interest today due to the less robust and more error-prone components expected in next technology generations. One of the most challenging problems of this research area consists in finding the fundamental error bounds beyond which reliable computation is not possible. A good understanding of these limits allows the scientific community to take the best approach for the design of reliable computing redundancy techniques.

In the following subsections we present the basic concepts of error bound theory, a detailed list of all the advances in the field of fundamental error bounds until today, and finally, our own contribution in the case of asymmetric error designs.

3.4.1 Basic Definitions

Before proceeding with the fundamental error bounds for reliable computation, we define here the basic concepts of the error bound theory for better understanding. Notice that the error framework used in this field of research is always the one proposed by von Neumann 3.1.1.

ϵ-noisy gate: Logic gate that computes a Boolean function with an error probability ϵ.

Noisy circuit: Circuit composed of ϵ-noisy gates. The errors of each gate occur independently of the other gates.

Circuit and formula: Both circuits (also referred as networks) and formulas have several Boolean inputs and one Boolean output without any feedback interconnection. However, while in circuits gates fan-out may be higher than one, in formulas gates fan-out is at most one.

Error probability: The maximum over all input combinations of the probability that a noisy circuit (or formula) computes a Boolean function f incorrectly (outputs its complement).

Reliable Computation Being $\delta < 1/2$, for every Boolean function there exists a noisy circuit (or formula) that computes its value with an error probability not higher than δ. In general this is only possible for specific circuit circumstances: ϵ-noisy gates, fan-in and fan-out.

On one hand, it is evident that a circuit composed of perfect/error-free gates ($\epsilon = 0$) will compute reliably, provided it has a correct design. On the other hand, it is also evident that in general a circuit composed of very unreliable gates ($\epsilon \to 1/2$) will not be able to provide an output error rate below 1/2. Thus, between these two extremes, there must be a threshold value for the gates error rate ϵ^* such that reliable computation is possible if $\epsilon < \epsilon^*$ and not possible if $\epsilon > \epsilon^*$. This characteristic error rate ϵ^* is conceptually regarded as the threshold error rate and it varies depending on the circuit configuration, i.e., the type of gates used or, in general, their fan-in, and if it is a circuit or a formula (fan-out limited to 1 or not). Among the advances to find the threshold error rate ϵ^* in each case, often partial results are obtained which can be separated into two classes:

Positive result: Reliable computation is possible if $\epsilon < \epsilon^{\text{lb}}$. We refer to this value as a lower bound for ϵ^* since $\epsilon^{\text{lb}} \leq \epsilon^*$.

Negative result: Reliable computation is not possible if $\epsilon > \epsilon^{\text{ub}}$. We refer to this value as an upper bound for ϵ^* since $\epsilon^* \leq \epsilon^{\text{ub}}$.

3.4.2 State of the Art

Next, we present a summary of the most relevant advances in the field of fundamental error bounds. Table 3.1 lists in chronologic order the main contributions and their corresponding proven error bounds. In the table it is also specified the circuit configuration to which the bound applies taking into account the type of gates and the fan-out (1 in the case of formulas and non-restricted for circuits). Some contributions are able to

prove lower error bounds ϵ^{lb}, other show upper error bounds ϵ^{ub}, and finally, few of them are able to determine exactly the error threshold ϵ^* under specific circuit configurations.

TABLE 3.1: Summary of the main advances in the field of fundamental error bounds for reliable computation.

Year	Author	Circuit configuration		Error bounds	
		Type of gates	fan-out	ϵ^{lb}	ϵ^{ub}
1956	von Neumann [2]	3-input MAJ	circuits	0.0073	
1988	Pippenger [56]	k-input gates	formulas		$\frac{1}{2} - \frac{1}{2k}$
1989	Feder [57]	k-input gates	circuits		$\frac{1}{2} - \frac{1}{2k}$
1991	Hajek and Weller [58]	3-input gates	formulas	$\epsilon^* = 1/6$	
1998	Evans and Pippenger [59]	2-input NAND	formulas	$\epsilon^* = \frac{3-\sqrt{7}}{4}$	
1999	Evans and Schulman [60]	k-input gates	circuits		$\frac{1}{2} - \frac{1}{2\sqrt{k}}$
2003	Evans and Schulman [61]	k-input gates (k odd)	formulas	$\epsilon^* = \frac{1}{2} - \frac{2^{k-2}}{k\binom{k-1}{k/2-1/2}}$	
2005	Gao, Qi and Fortes [62]	k-input NAND	circuits	ϵ^* =numeric solution	
2008	Unger [63]	2-input gates	formulas	$\epsilon^* = \frac{3-\sqrt{7}}{4}$	

Nevertheless, looking carefully at the list of advances we can notice that there are still some gaps to fill such as:

- Finding the error threshold for k-input gates ($k > 2$ even) for formulas and circuits.
- Generalizing the Weller's error threshold (k odd) for circuits.
- Generalizing the basic von Neumann assumptions for more realistic cases.
- Considering other types of gate or circuit designs, such as asymmetric error and analog logic.

In the following subsection we extend the fundamental error bound for k-input NAND gates for asymmetric error designs.

3.4.3 Asymmetric Error Designs

In the literature we can find the exact error threshold for circuits built out of noisy NAND gates under the von Neumann's probabilistic computing framework [62]. In the

following we extend this result for asymmetric error designs and demonstrate that it is possible to compute reliably with 2-input noisy NAND gates beyond the well known error bound: $\epsilon^* = (3 - \sqrt{7})/4$. We further determine the fundamental error bound for asymmetric error multiple-input NAND gates. To this end we study the nonlinear interactions among noisy NAND gates, and thereby, we are able to derive the exact error threshold using elementary bifurcation theory.

3.4.3.1 Asymmetric Error NAND Gates

Until now, the vast majority of work concerned with fundamental error bounds and fault-tolerant computation use the von Neumann error model framework [2]. According to this model every logic gate will fail to function correctly in any operation with the precise probability ϵ. Statistical independence between gate errors is also assumed. Here, we propose to extend the von Neumann error model considering asymmetric errors for logic values '0' and '1'.

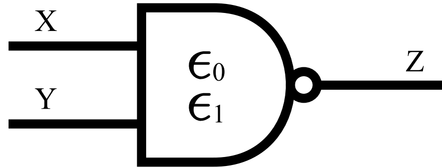


FIGURE 3.9: Asymmetric error NAND gate.

We denote by ϵ_1 the probability of an output logic '1' being misread as logic '0' and ϵ_0 the probability of an output logic '0' being misread as logic '1'. Assuming statistical independence between inputs we can calculate the probability of logic '1' at the output of a NAND gate (Z) in terms of the input probabilities of logic '1' (X, Y) and the error probabilities ϵ_0 and ϵ_1 as follows:

$$Z = (1 - \epsilon_1)(1 - XY) + \epsilon_0 XY. \quad (3.11)$$

However, error probability parameters ϵ_1 and ϵ_0 mix key information on two different aspects of NAND gates: namely, the amount of uncertainty and the level of error asymmetry. In the following we describe a parametric fault model that allows us to separate both dimensions in the analysis of asymmetric error NAND gates. We assume that output logic values '0' and '1' are detected by a threshold device reading a voltage signal z affected by variability. Signal z can be decomposed into two parts:

$$z = z_0 + \delta,$$

where z_0 corresponds to the ideal value and δ includes the variability/uncertainty of the NAND gate. We model δ as a normal random variable with null mean and standard deviation σ ($\delta \sim N(0, \sigma)$). Without loss of generality we assume that voltage levels $0V$ and $1V$ are associated to logic values '0' and '1' respectively. The threshold level l of the decision device can be adjusted to achieve different levels of asymmetry.

With this model we can easily separate the uncertainty of NAND gates, reflected by the standard deviation parameter σ, and the asymmetry of errors, which is determined by the threshold level l of the decision device. Under these conditions, the probabilities of error ϵ_0 and ϵ_1 can be expressed in terms of σ and l as follows:

$$\epsilon_0 = \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{l}{\sqrt{2}\sigma}\right)\right) \tag{3.12}$$

$$\epsilon_1 = \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{1-l}{\sqrt{2}\sigma}\right)\right). \tag{3.13}$$

Note that $l = 1/2$ corresponds to the symmetric case for which $\epsilon_0 = \epsilon_1$.

3.4.3.2 Asymmetric Error Reliable Computation with 2-input NAND gates

Taking as a reference the basic von Neumann NAND multiplexing scheme, we determine the fundamental error bound for circuits built out of asymmetric error 2-input NAND gates. It is assumed that NAND multiplexing scheme has no feedback loops and the output of each gate is connected to an input of only one other gate like the circuit schematic shown in Figure 3.10. Under these conditions the inputs of each gate are statistically independent.

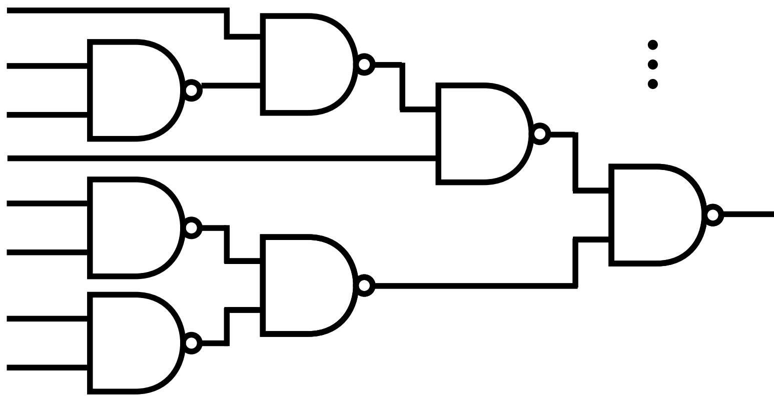


FIGURE 3.10: Circuit schematic built out of 2-input NAND gates.

Proceeding with the bifurcation analysis we take the worst case scenario in which $X = Y$. In this case, (3.11) reduces to a nonlinear map:

$$Z = f(X) \equiv 1 - \epsilon_1 - (1 - \epsilon_1 - \epsilon_0)X^2. \tag{3.14}$$

We first determine the equilibrium points of a single computation step by solving the equation $f(X) = X$, whose roots are:

$$X_{1,2} = \frac{-1 \pm \sqrt{5 + 4\epsilon_1^2 - 8\epsilon_1 - 4\epsilon_0(1 - \epsilon_1)}}{2(1 - \epsilon_0 - \epsilon_1)}. \tag{3.15}$$

Analyzing the argument of the square root we observe that it is always positive $5 + 4\epsilon_0^2 - 8\epsilon_0 - 4\epsilon_1(1 - \epsilon_0) \geq 0$ in the region $0 < \epsilon_0, \epsilon_1 < 1$. However, since any probability must be contained in the interval $[0, 1]$ we reject root X_2 leaving X_1 as the unic equilibrium point for single computation step.

We analyze now the equilibrium points of two consecutive computation steps by solving the equation $f(f(X)) = X$. The resulting roots include $X_{1,2}$, as in the previous case, and two new solutions:

$$X_{3,4} = \frac{1 \pm \sqrt{1 + 4\epsilon_1^2 - 8\epsilon_1 - 4\epsilon_0(1 - \epsilon_1)}}{2(1 - \epsilon_0 - \epsilon_1)}. \tag{3.16}$$

These $X_{3,4}$ roots correspond to the branches of asymmetric error 2-input NAND bifurcation map. Therefore, analyzing the argument of the square root and imposing it to be greater or equal to zero, we find the fundamental error bound for reliable computation with asymmetric error 2-input NAND gates:

$$\epsilon_0 \leq 1 - \epsilon_1 - \frac{3/4}{1 - \epsilon_1}. \tag{3.17}$$

If we impose $\epsilon_0 = \epsilon_1$ ($\equiv \epsilon$), the previous condition obviously reduces to the well-known error bound $\epsilon \leq \epsilon^* = (3 - \sqrt{7})/4$. However, as we will see next, if we admit asymmetric error designs then 2-input NAND gates can tolerate higher levels of variability/uncertainty.

Figure 3.11 depicts the fundamental error bound for reliable computation with asymmetric error 2-input NAND gates. We observe a clear asymmetry in the error bound regarding the error rates ϵ_0 and ϵ_1. Indeed, according to this result reliable computation with NAND gates tolerates higher rates of ϵ_0 than ϵ_1. For example, reliable computation is possible with $\epsilon_0 = 0.2$ (as long as $\epsilon_1 < 0.028$), but not with $\epsilon_1 = 0.2$ (in fact, never if $\epsilon_1 > 0.134$). This asymmetric fault tolerance is associated to the fact that the logic NAND function is an unbalanced gate and it has a different number of '1's and '0's in its output. In fact, given random independent inputs, it is more probable to obtain a logic

'1' at the output than a logic '0'. Therefore, misreading an output logic '1' has a greater impact on the overall reliability of the logic NAND gate than misreading a logic '0'. Figure 3.11 also shows the contour lines of parameters σ and l for the symmetric design

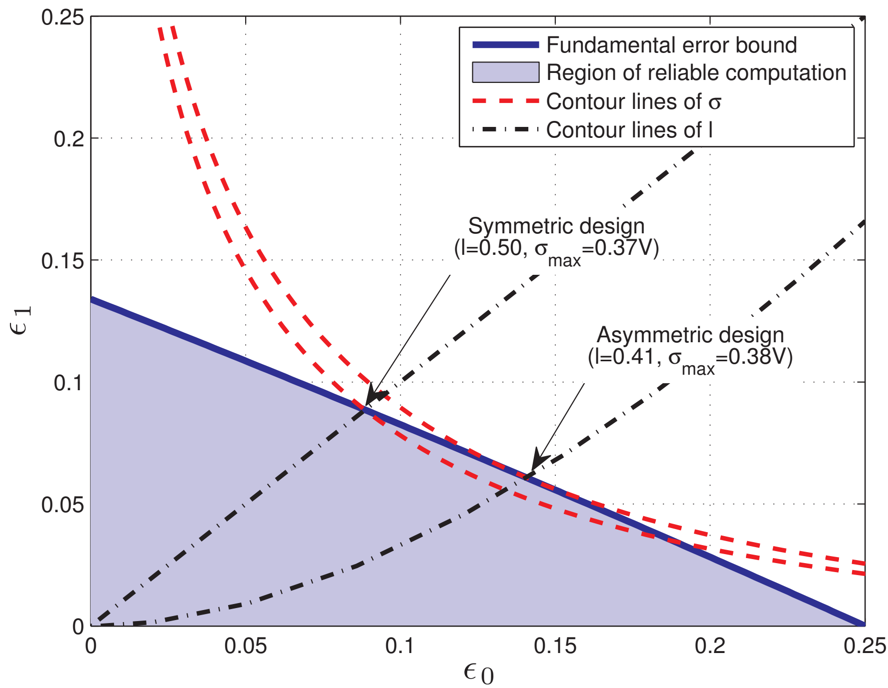


FIGURE 3.11: Fundamental error bound for asymmetric error 2-input NAND gates.

and the optimum asymmetric design. Symmetric design has $l = 0.5$ and $\epsilon_1 = \epsilon_0 = \epsilon^*$. Consequently, the associated maximum variability level that can be tolerated with the symmetric error design is:

$$\sigma^* = \frac{1}{\sqrt{8}\,\mathrm{erf}^{-1}\left(1 - 2\epsilon^*\right)} \approx 0.37V.$$

Extending the problem to asymmetric error designs we are able to increase the maximum admissible variability level for reliable computation. Figure 3.12 shows the maximum admissible variability level $\sigma_{\max}$ against the threshold level l of 2-input NAND gates. The resulting curve corresponds to all the configurations (l, σ) over the fundamental error bound. Because of the asymmetry of the problem, the maximum fault-tolerance for the NAND gate is achieved by asymmetric threshold devices, i.e. $l \neq 0.5$. In the case of 2-input NAND gates we should set the threshold at $l = 0.41$. With this configuration we could tolerate a maximum variability level of $\sigma_{\max} = 0.38V$, which is larger than the one obtained with the symmetric design ($\sigma^* = 0.37V$). Now, using (3.12) and (3.13) we calculate the equivalent symmetric error rate $\epsilon_{\max}$ associated to optimum asymmetric design. Comparing both results we can see how much the fundamental error bound for

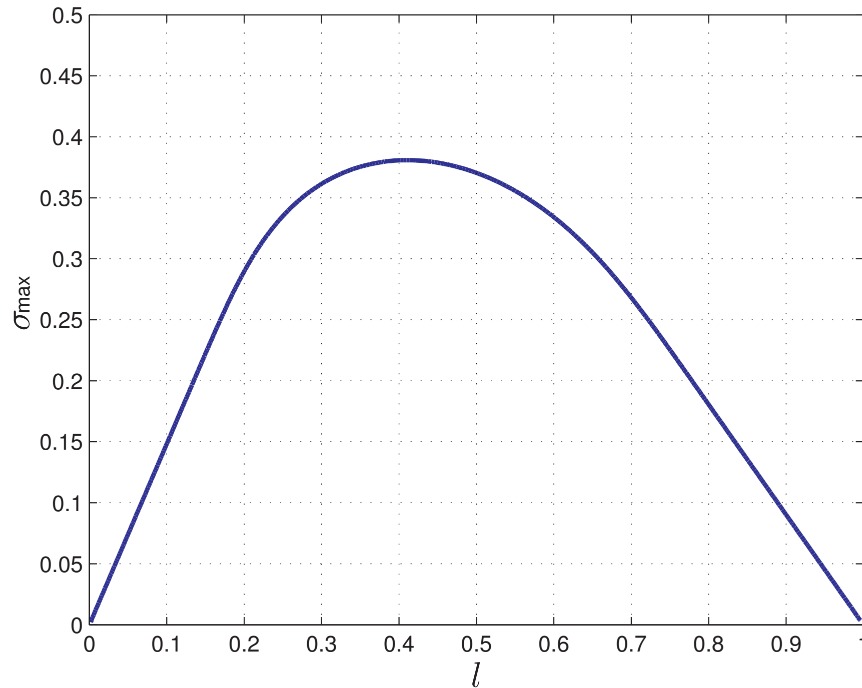


FIGURE 3.12: Maximum tolerable variability $\sigma_{\max}$ against threshold level l for reliable computation using 2-input NAND gates.

2-input NAND gates has been extended :

Asymmetric design	Symmetric design
$\sigma_{\max} \approx 0.38V$	$\sigma^* \approx 0.37V$
$\epsilon_{\max} \approx 0.0946$	$\epsilon^* \approx 0.0886$

From $\epsilon = 0.0886$ to $\epsilon = 0.0946$ the equivalent error bound has been extended a 6.8%. In this particular case of noisy 2-input NAND gates the gain in fault tolerance is not very big. However, this is a significant result since it proves that logic gates asymmetry (unbalanced gates) can be exploited to improve fault tolerance. Additionally, larger gains are expected for more unbalanced logic gates, such as NAND gates with higher number of inputs.

3.4.3.3 Asymmetric Error Reliable Computation with k-input NAND gates

We proceed now with the study of multiple-input NAND gates. First, we reproduce the nonlinear map of (3.14) for k-input NAND gates:

$$Z = f_k(X) \equiv 1 - \epsilon_1 - (1 - \epsilon_1 - \epsilon_0)X^k. \tag{3.18}$$

In this case it is hard to obtain an explicit form for the branches of the bifurcation map. Instead, to determine the condition of fundamental error bound we combine the following conditions as suggested in [62]:

- The pairs (ϵ_0, ϵ_1) lying on the fundamental error bound have to be an equilibrium point X_0 of a single computation step $(f_k(X_0) = X_0)$.
- At the same time, the derivative of $f_k(X)$ evaluated at the equilibrium point X_0 has to be -1, $f'_k(X_0) = -1$ (in order to find an unstable equilibrium point).

Hence, the condition of fundamental error bound for pairs (ϵ_0, ϵ_1) is:

$$1 - \epsilon_0 - \epsilon_1 = \frac{1}{k^k}\left(\frac{1+k}{1-\epsilon_0}\right)^{k-1}. \tag{3.19}$$

Figure 3.13 depicts the fundamental error bounds of (3.19) for $k = 2$ to $k = 10$.

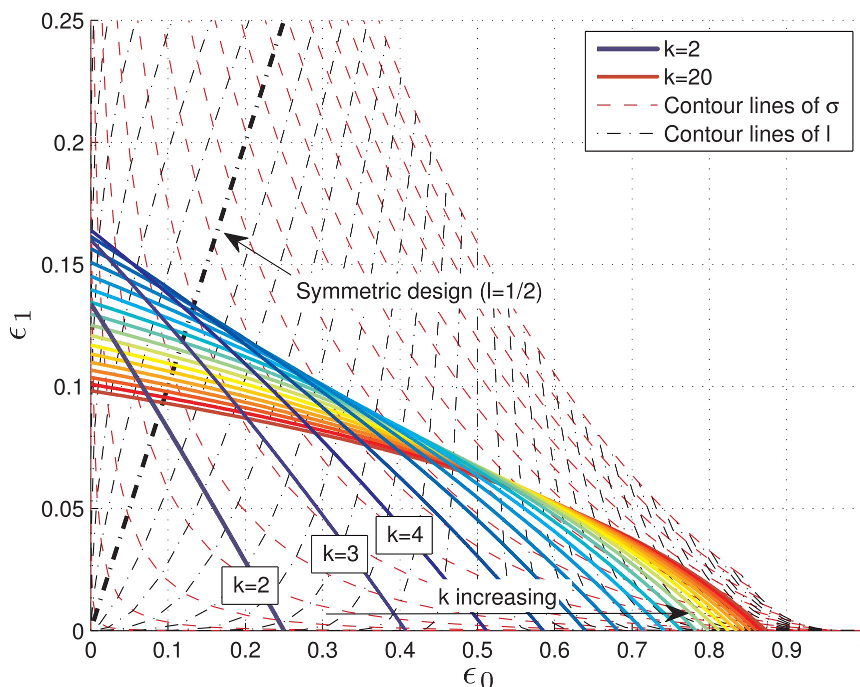


FIGURE 3.13: Fundamental error bound for asymmetric error k-input NAND gates.

In the figure we can observe how the asymmetry of error bound increases with k. The greater the number of inputs k to the NAND gate, the better the errors ϵ_0 are tolerated and the worse the errors ϵ_1. As a consequence of this increasing asymmetry, the gain in the maximum level of tolerable uncertainty $(\sigma_{\max})$ increases with k. In fact, observe in Figure 3.14 the maximum admissible variability level $\sigma_{\max}$ against the threshold level

l of k-input NAND gates. If we compare $\sigma_{\max}$ of symmetric designs (with $l = 0.5$)

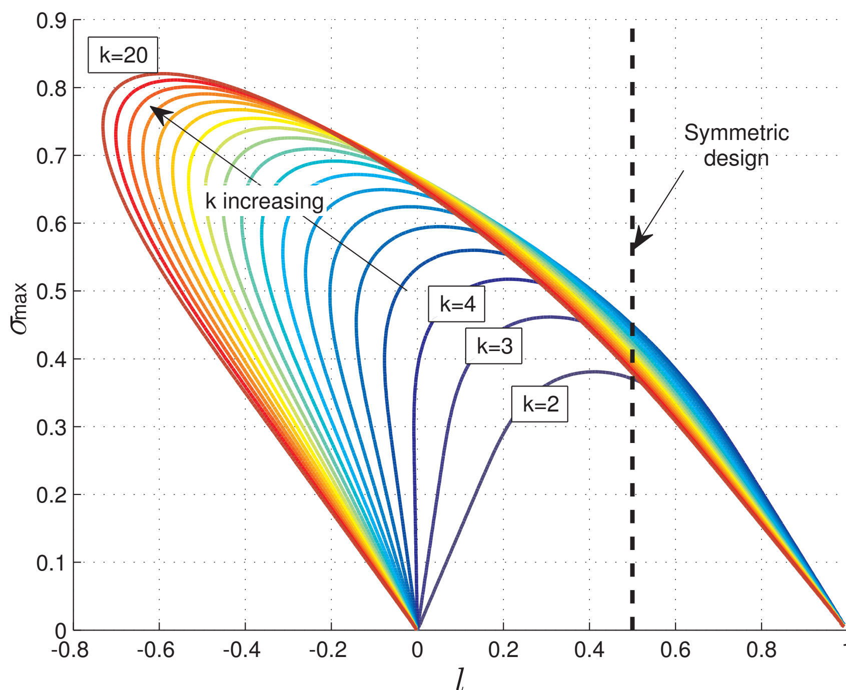


FIGURE 3.14: Maximum tolerable variability $\sigma_{\max}$ against threshold level l for reliable computation using k-input NAND gates.

and optimum asymmetric designs we can see how in the case of symmetric designs it increases for $k < 5$ and then decreases again while in the case of asymmetric designs it keeps increasing with k. Figure 3.15 compares the maximum equivalent symmetric error bound $\epsilon_{\max}$ of k-input NAND gates between symmetric and optimum asymmetric designs. We use (3.12) and (3.13) to perform the (σ, l) to $(\epsilon \equiv \epsilon_1 = \epsilon_0)$ conversion. As we have seen before, symmetric designs have the maximum admissible variability for $k = 5$. This is consistent with previous work [62]. In the case of asymmetric error designs, the fundamental error bound keeps increasing with k and significantly outperform the symmetric designs. For example, optimum asymmetric designs of noisy 3-input NAND gates have 17.5% gain in fault tolerance and 5-input NAND gates have 39.8% gain with respect to symmetric designs. However, if we compare this error bounds with the exact error threshold for k-input gates (k odd) found by Evans and Schulman in [61], we observe that they are still below the theoretical maximum, as expected: 16.4% below in the case of $k = 3$, 20.3% in the case of $k = 5$ and gradually increasing with k. That is, we have significantly extended the error bounds of multiple input NAND gates by using asymmetric error designs, but for number of inputs $k \geq 3$ the resulting error bounds are still lower than the fundamental error threshold of k-input gates.

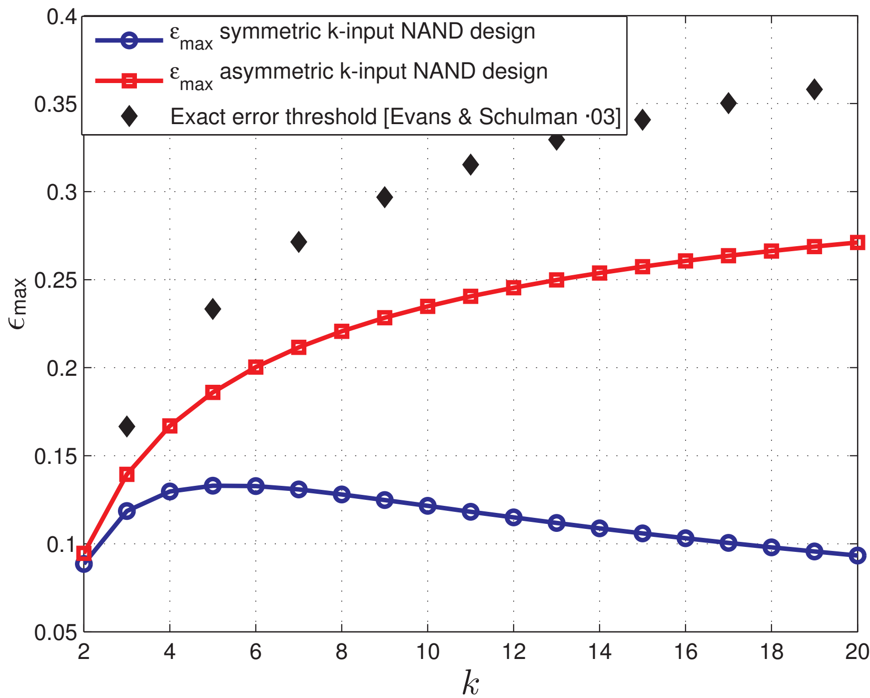


FIGURE 3.15: Maximum tolerable variability $\sigma_{\max}$ of symmetric and asymmetric NAND gate designs against the number of inputs k. The exact error threshold for k-input gates (k odd) is also shown with black markers.

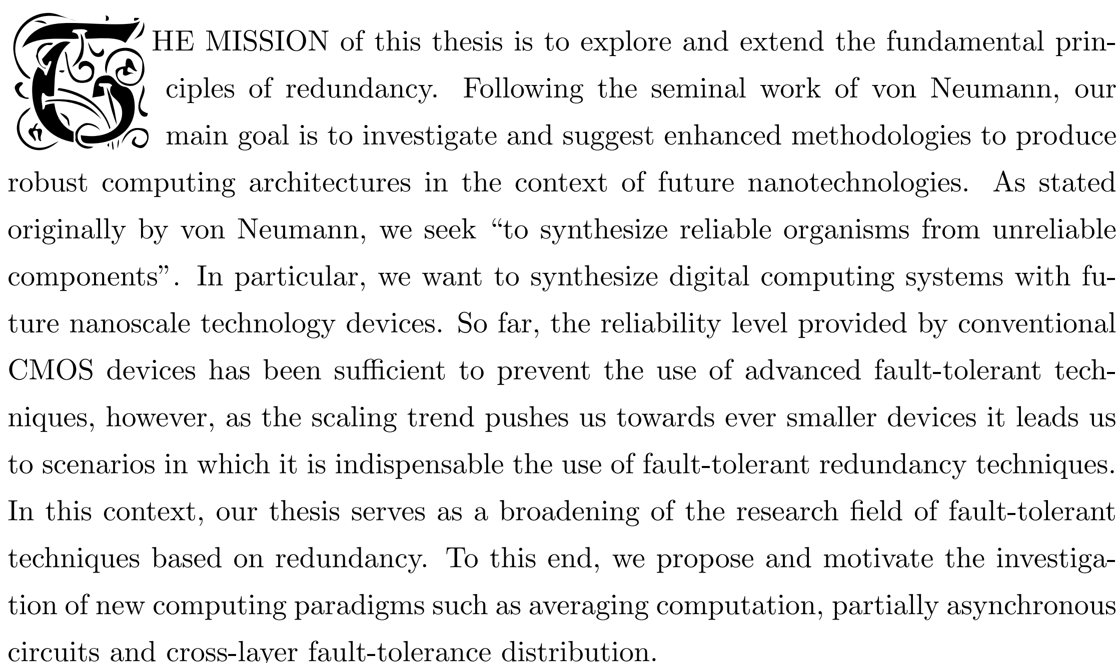
3.4.3.4 Concluding Remarks

Our contribution on asymmetric error designs extends the fundamental error bounds for reliable computation. We define a parametric fault model for logic gates that allows us to separate the variability/uncertainty dimension from the level of asymmetry between the error rates of logic '0's and logic '1's. Using the bifurcation theory we generalize the well known error bound for noisy 2-input NAND gates and find the optimum asymmetric design by which the maximum fault tolerance is obtained; in this case, a 6.8% increase in the equivalent error bound with respect to classic symmetric implementations. Based on the analysis we associate the asymmetric tolerance to errors in logic '0's and logic '1's to the unbalanced condition of NAND logic function. We also explore the gain in fault tolerance of optimum asymmetric designs of NAND gates with higher number of inputs and the obtained results confirm our expectations. Optimum asymmetric designs of noisy 3-input NAND gates have 17.5% gain in fault tolerance and 5-input NAND gates have 39.8% gain. The higher the asymmetry or unbalance level of a logic gate is, the higher the gain in fault tolerance can be achieved by an asymmetric error design. We also compare the extended error bounds of multiple input NAND gates with the theoretical error threshold for k-input gates of Evans and Schulman. While our results

are consistent, the comparison shows that our proposed asymmetric design approaches the theoretical fundamental limit for reliable computation.

Chapter 4

Thesis Motivation and Objectives



THE MISSION of this thesis is to explore and extend the fundamental principles of redundancy. Following the seminal work of von Neumann, our main goal is to investigate and suggest enhanced methodologies to produce robust computing architectures in the context of future nanotechnologies. As stated originally by von Neumann, we seek "to synthesize reliable organisms from unreliable components". In particular, we want to synthesize digital computing systems with future nanoscale technology devices. So far, the reliability level provided by conventional CMOS devices has been sufficient to prevent the use of advanced fault-tolerant techniques, however, as the scaling trend pushes us towards ever smaller devices it leads us to scenarios in which it is indispensable the use of fault-tolerant redundancy techniques. In this context, our thesis serves as a broadening of the research field of fault-tolerant techniques based on redundancy. To this end, we propose and motivate the investigation of new computing paradigms such as averaging computation, partially asynchronous circuits and cross-layer fault-tolerance distribution.

In order to describe in more detail the specific aims of this thesis, we list in the following a set of fundamental questions that summarize all the points treated in this work. Some of them can be answered by analyzing the state of the art of computing technology, other questions are developed in different parts of this thesis.

- **Which will be the technologies used to implement future nanoscale computing systems?**

  We review the main nanotechnologies that have been proposed so far in order to envision the scenario of future computation. We identify the main features that the different alternatives have in common, such as reliability level, integration capability and type of device.

- **Which will be the main sources of unreliability associated to future nanoscale computing devices?**

  Being the system reliability our main concern, we analyze the sources of unreliability present in future computing devices. We need to know what kind of faults will appear in the basic components in order to design robust circuits.

- **How can we characterize and model the performance of future nanoscale computing devices?**

  Being aware of all the types of device malfunctions, we develop realistic fault and degradation models to simulate and characterize future nanoscale devices.

- **Which are the fundamental reliability limits of computing devices beyond which reliable computation is not possible?**

  We revisit the fundamental limits of reliable computation present in the literature and try to find new ways to address the problem of reliable computing with unreliable devices.

- **Which will be the most appropriate hardware architectures based on redundancy to implement robust computing systems in the future?**

  We review the main fault-tolerant architectures proposed for the design of robust computing systems. We identify the main advantages of each fault-tolerant approach in order to gather the necessary knowledge to be able to suggest improved designs adapted to specific technology conditions.

- **How can we extend the principles of redundancy for the future nanotechnologies?**

  Combining the forecasts of future nanoscale technology with the currently available enhanced fault-tolerant techniques, we want to extend the conventional redundancy approach to adapt and optimize it for reliable computation in future computing scenarios.

  In the following chapters we propose three extensions of the von Neumann redundancy approach. We can describe these contributions as three new dimensions to consider during fault-tolerant design based on redundancy. Figure 4.1 reproduces a schematic view of these dimensions: (1D) Heterogeneous-aware Reliable Design, (2D) Time-aware Reliable Design, and (3D) Multiple-layer Reliable Design.

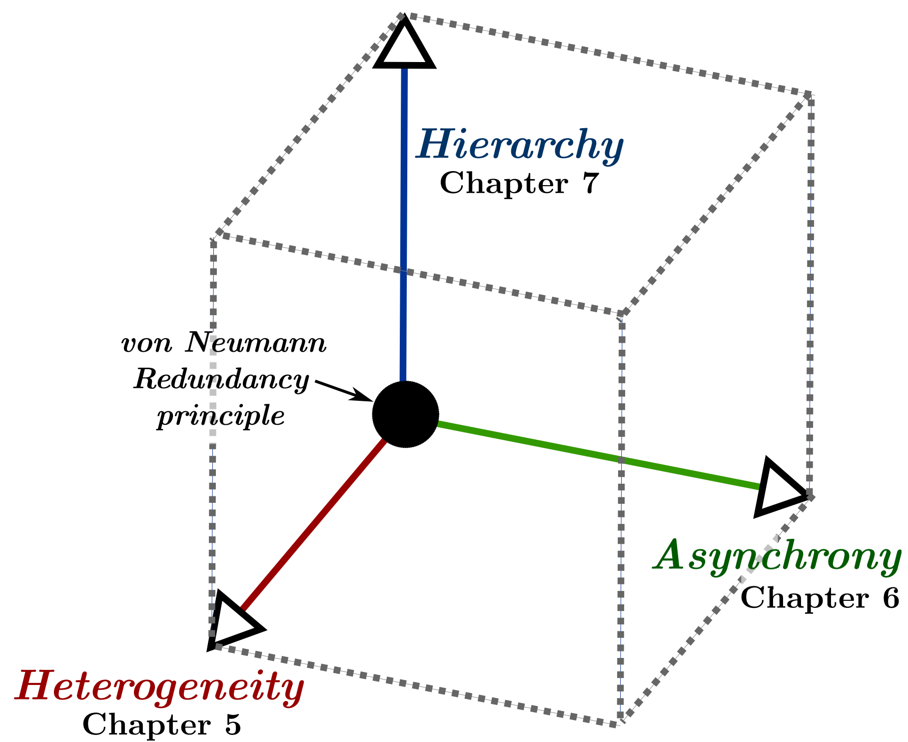


Figure 4.1: Schematic representation of the three main contributions of this thesis to extend the principles of redundancy established by von Neumann in [2]. The three considered dimensions, namely heterogeneity, asynchrony, and hierarchy, correspond to Chapters 5, 6, and 7 respectively.

Chapter 5

Heterogeneous-aware Reliable Design

HETEROGENEITY between ideally identical devices will most probably appear in future technology generations as a result of the expected high levels of variability and defects. As ICs enter the nanoscale regime, fundamental physical limits and manufacturing constraints arise that make it impractical to work with design schemes based on quasi-perfect devices.

In this Chapter, we propose redundancy-based fault-tolerant techniques that take into account the heterogeneity of future nanoscale technologies. Instead of assuming a precise error probability ϵ equal for all the components following the von Neumann's hypothesis of homogeneity, we take into account the heterogeneity of real nanodevices and use adaptive architectures to cope with non-homogeneous variability environments. More specifically, we focus on the averaging cell (AVG) principle, which was previously introduced in Chapter 3 (see page 25), and improve its performance through adjustable weights systems. Our study gives rise to various fault-tolerant techniques which correspond to different sections of this chapter.

The rest of this Chapter is organized as follows. In Section 5.1, we consider static heterogeneity between replicas and derive a methodology to determine the specific averaging weights that maximize the reliability of the AVG structure. The implementation of these optimal weights in the AVG structure gives place to the unbalanced AVG approach (U-AVG). The U-AVG extends the conventional AVG approach, which uses balanced weights (B-AVG). In Section 5.2, we take into consideration that circuits are exposed to degradation, which can induce significant changes on the levels of variability during the circuit lifetime, and introduce the adaptive AVG structure (AD-AVG).

The AD-AVG approach embeds a learning mechanism based on a variability monitor that allows for the on-line input weight adaptation such that the actual weight configuration properly reflects the aging status. To evaluate the potential implications of this proposal we compare the conventional AVG architecture (B-AVG), the unbalanced AVG (U-AVG), and the adaptive AVG (AD-AVG) approaches in terms of reliability and redundancy overhead by means of Monte Carlo simulations. Subsequently, we include temporal variation of input drifts in the simulations to reproduce the effects of degradation and aging. In Section 5.2.3, specific reconfigurable hardware based on resistive switching crossbar structures is proposed for the implementation of the AD-AVG, the weights reconfiguration and the input variability measurement.

5.1 The Unbalanced Averaging Cell (U-AVG)

Starting from the conventional AVG architecture that uses balanced weights (B-AVG), which was previously introduced in Section 3.3.2 (see page 25), we concentrate here on the special case of AVGs designed with unbalanced weights (U-AVG) to counteract the heterogeneity present in the input replicas due to variability. Figure 5.1 shows a schematic of the Unbalanced Averaging Cell (U-AVG) architecture. Our proposal consists of adjusting the AVG weighting scheme according to the following principle: assign greater weight to the less degraded and more reliable inputs, and lower weight to the ones that are more prone to be unreliable. Intuitively speaking such an approach should improve the overall reliability.

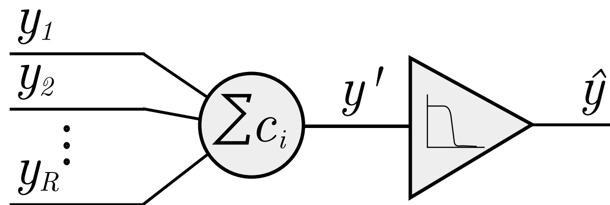


FIGURE 5.1: Unbalanced Averaging Cell (U-AVG) schematic.

When the input drifts lose homogeneity due to variability effects, the output standard deviation increases dramatically for the conventional AVG approach (B-AVG), as shown in the following subsection, and so does the output error probability P_e. In this section, we study in detail the benefits that can be further obtained from the AVG structure by considering extra information related to the specific input variability levels. The main objective of the U-AVG proposal is to improve the fault-tolerant capabilities of the AVG structure by adjusting the values of the averaging weights in accordance with the

particular levels of input variability. This approach allows us to deal with scenarios of heterogeneous variability.

In order to motivate the use of unbalanced averaging, we first analyze a simple case of heterogeneity where all inputs are exposed to homogeneous variability except one that has a higher standard deviation than the others. This example demonstrates that a significant reduction in the variability of the weighted average y' (in terms of $\sigma_{y'}$) can be achieved if we set up the averaging weights according to the U-AVG proposal, when compared with the conventional AVG with balanced weights (B-AVG). After this experiment we analytically demonstrate the existence of optimal averaging weights that minimize the output error probability P_e (or equivalently $\sigma_{y'}$) in any possible heterogeneous variability scenario. Then, we find a general formula that allows us to calculate the optimal set of weights in any circumstance. We conclude this section with the theoretical savings in the redundancy level (R) that our U-AVG proposal can provide against the B-AVG.

5.1.1 Heterogeneous Variability Scenario (Simple Case)

In this experiment we reproduce the most simple case of heterogeneous variability scenario. We simulate the AVG with homogeneous level of variability for all the input replicas except to the qth input; we assign standard deviation σ_q to the qth input while all the rest have the same standard deviation σ. In this experiment, we first consider the conventional AVG (B-AVG), which uses balanced weights $c_i = 1/R$, $i = 1, \ldots R$. In this case, the resulting output variance can be expressed as (5.1). This formula is deduced from (3.7) and the σ_i model presented above.

$$\sigma_{y'}^2 = \frac{R-1}{R^2}\sigma^2 + \frac{1}{R^2}\sigma_q^2 \tag{5.1}$$

Figure 5.2 reproduces, in continuous lines, the analytic expression of the output standard deviation $\sigma_{y'}$ against different levels of variability in the qth input. One can observe that for the modeled heterogeneous scenario (with $R = 3$, $\sigma = 0.1\ V$, and σ_q from $0.1\ V$ to $0.4\ V$), the output standard deviation for the conventional AVG (B-AVG) dramatically increases from $\sigma_{y'} = 0.06\ V$ when $\sigma_q = \sigma = 0.1\ V$ to more than its double $\sigma_{y'} = 0.14\ V$ when $\sigma_q = 0.4\ V$.

Next we investigate how much better the overall performance can be when we set up the weight values c_i properly such that they reflect the fact that the qth input has a different variability. Using the general output variance expression in (3.7), an optimization problem can be derived, see (5.2). There are $R-1$ inputs with standard deviation

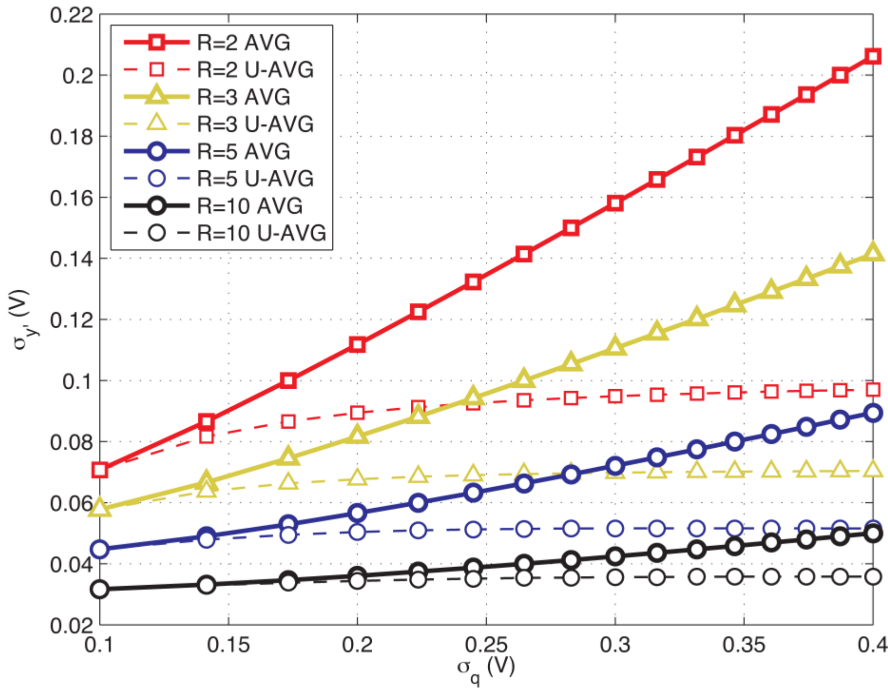


FIGURE 5.2: Output standard deviation of AVG and U-AVG with 2, 3, 5 and 10 inputs subject to heterogeneous variability scenario. The scenario is modeled with a standard deviation $\sigma = 0.1\ V$ in all the inputs except from the qth that has σ_q ranging between $0.1\ V$ and $0.4\ V$.

σ and normalized weight c, while the remaining input has a higher standard deviation $\sigma_q > \sigma$ and a different normalized weight c_q. Notice that (5.2) must hold the normalized weights condition: $(R-1)c + c_q = 1$.

$$\min(\sigma^2_{y'})|_{c_q} \Rightarrow \frac{\mathrm{d}}{\mathrm{d}c_q}\left((R-1)c^2\sigma^2 + c_q^2\sigma_q^2\right) = 0 \tag{5.2}$$

Making the appropriate calculations to minimize the output variance $\sigma^2_{y'}$ by adjusting the value of weights c and c_q, the following expression for the optimum weight c_q^{opt} can be deduced

$$c_q^{\text{opt}} = \frac{1}{(R-1)\frac{\sigma_q^2}{\sigma^2} + 1}. \tag{5.3}$$

This formula, depicted in Figure 5.3, clearly demonstrates that the optimal distribution of weights only depends on the ratio between the input variances, or equivalently on the relative reliability levels. In the particular case of $R = 2$ we can easily verify that the set of optimal weights is $(1/2, 1/2)$ when the variances or reliabilities are equal. If the variances ratio is higher than 1 then the input qth is less reliable and the optimal weight c_q^{opt} decreases. And vice versa, if the variances ratio is lower than 1 the optimal weight c_q^{opt} increases with respect to the equilibrated case. Figure 5.3 also shows the

impact of redundancy level R on the relationship between c_q^{opt} and the input variances. We observe that it corresponds to a compression to the origin with scale factor $R-1$.

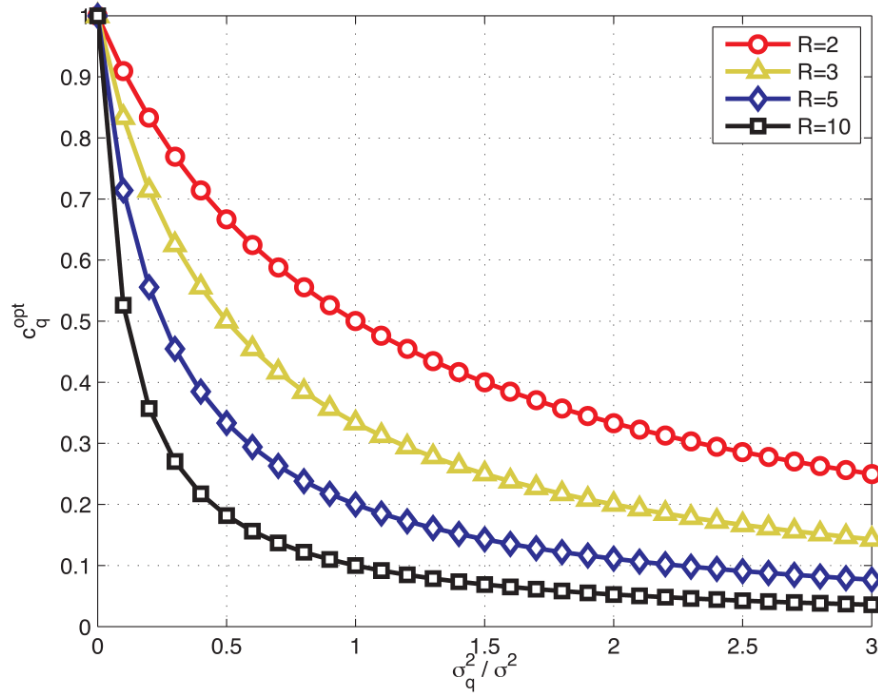


FIGURE 5.3: Optimal weight c_q^{opt} against the ratio of input variances σ_q^2/σ^2 in a simple case of heterogeneous variability scenario. All the replicas have the same variance σ^2 except form the input qth that has variance σ_q^2.

The improvements achieved by the new weight configuration are depicted in Figure 5.2 with dashed lines. One can observe in the figure that the U-AVG approach minimizes the output standard deviation compared to the AVG. Retrieving the previous example ($R = 3$ and $\sigma = 0.1\ V$), but now considering the U-AVG approach, we can observe that an increase in the qth standard deviation σ_q from $0.1\ V$ to $0.4\ V$ only increases the output standard deviation from $0.06\ V$ to $0.07\ V$. Thus, in this particular example, a 50% net reduction in the output standard deviation $\sigma_{y'}$ is achieved by the U-AVG with respect to the AVG. Moreover, one can deduce from the figure that the adjustment of weights is critical for low R (redundancy) values, which suggests that U-AVG may potentially require a lower redundancy than AVG for the same targeted reliability.

5.1.2 The Existence of Optimal Weighted Averages in General AVGs

Continuing with the idea of making use of non-balanced weights, we demonstrate in the following that it always exists a set of weights c_i^{opt}, $i = 1, \ldots, R$, that optimally minimizes the output error probability P_e, or equivalently the standard deviation of the

weighted average $\sigma_{y'}$. First we present a Lemma that displays this concept for individual input replicas.

Lemma 1: Given a weighted average

$$y' = \frac{1}{\sum_{i=1}^{R} k_i} \sum_{i=1}^{R} k_i y_i$$

of R input random variables y_i, $i = 1, \ldots R$, following statistically independent normal distributions and fixed positive weights $k_i \geq 0$ for all $i = 1, \ldots R$ except $i = q$, there always exists a unique value k_q^{opt} for the unfixed qth weight, $q \in (1, \ldots R)$, that optimally minimizes the standard deviation of the weighted average $\sigma_{y'}$.

Proof: Recalling the expression of the weighted average variance in (3.7), and substituting the definition of normalized weights ($c_i = k_i / \sum_{j=1}^{R} k_j$), we can express the dependence of $\sigma_{y'}^2$ on the weight k_q

$$\sigma_{y'}^2 = \frac{k_q^2 \sigma_q^2 + \sum_{i=1, i \neq q}^{R} k_i^2 \sigma_i^2}{\left(k_q + \sum_{i=1, i \neq q}^{R} k_i\right)^2}. \tag{5.4}$$

Differentiating with respect to k_q and matching to zero, we get the following optimal value for the qth weight:

$$\frac{\mathrm{d}\sigma_{y'}^2}{\mathrm{d}k_q} = 0 \Rightarrow k_q^{\mathrm{opt}} = \frac{1}{\sum_{i=1, i \neq q}^{R} k_i} \sum_{i=1, i \neq q}^{R} k_i^2 \frac{\sigma_i^2}{\sigma_q^2}. \tag{5.5}$$

Notice that k_q^{opt} is also positive. Differentiating two times with respect to k_q and substituting $k_q = k_q^{\mathrm{opt}}$, we obtain a positive value. This confirms that indeed the value obtained k_q^{opt} corresponds to the unique weight value that optimally minimizes the $\sigma_{y'}^2$ value, and the standard deviation $\sigma_{y'}$ as well:

$$\left.\frac{\mathrm{d}^2\left(\sigma_{y'}^2\right)}{\mathrm{d}k_q^2}\right|_{k_q = k_q^{\mathrm{opt}}} = \frac{2\sigma_q^2 \sum_{i=1, i \neq q}^{R} k_i}{\left(k_q^{\mathrm{opt}} + \sum_{i=1, i \neq q}^{R} k_i\right)^3} \geq 0.$$

□

Lemma 1 demonstrates that if we look at the value of a particular weight k_q, we can find a value that optimally minimizes the weighted average variance, and therefore minimizes the output error probability. In the Figure 5.4, we plot a sensitivity of the output error probability P_e to the modification of a specific weight in its normalized form c_q. For the analysis, we assume different levels of variability in the qth input modeled with the parameter σ_q ranging between 0.0 V and 0.4 V. The rest of inputs are considered altogether with a fixed contribution to the weighted average variability when the value of weight c_q is null: $\sigma_{y'}|_{\{c_q=0\}} = 0.2\,V$. One can observe in the figure, the different locations of the P_e minimum and the relation between optimal weights and different levels of variability. It clearly shows that it is possible minimize the output error probability P_e if we properly tune the value of weight c_q.

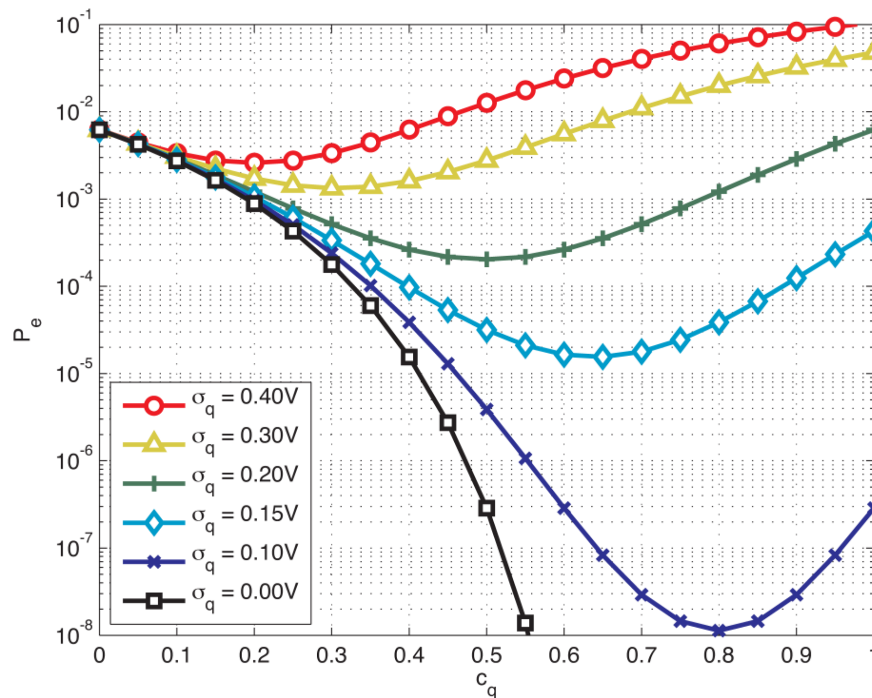


FIGURE 5.4: Variation of the error probability P_e against the weight c_q. Standard deviation in the input q ranges from $\sigma_q = 0.0\ V$ to $\sigma_q = 0.4\ V$ and the standard deviation of the weighted average due the rest of inputs is $\sigma_{y'}|_{\{c_q=0\}} = 0.2\,V$.

The following conclusions can be drawn from Figure 5.4:

- There is always one and only one c_q value in the range from 0 to 1 that minimizes the error probability P_e in each possible variability environment.
- The optimum c_q value is never exactly equal to 0. Even for large levels of deviation in the qth input with respect to the others, it is useful to have a contribution from the input q.

- The optimum c_q value is never exactly equal to 1, except from the singular case with $\sigma_q = 0\ V$, which in practice never occurs. Even for small levels of deviation in the qth input with respect to the others, it is useful to have a contribution from the rest of inputs.

- To minimize the output error probability P_e, higher weights (close to 1) should be assigned to less deviating inputs and vice versa.

- When the deviating effect in the qth input σ_q is equal to the deviating effect of the rest of inputs $\sigma_{y'}|_{\{c_q=0\}}$, the optimum c_q value is 0.5. Note in Figure 5.4 that the curve with the minimum at $c_q = 0.5$ holds $\sigma_q = \sigma_{y'}|_{\{c_q=0\}}$.

So far we have proved the existence of an optimal value for a particular weight independently from the rest of weights. However, the optimization of the entire set of weights requires an extra effort because the isolated optimization of averaging weights does not give rise to a global optimum. For example, if we optimize the values of the weights k_i from $i = 1$ to $i = R$ following the Lemma 1, we do not achieve a globally optimum. The optimal weight values found in the proof of Lemma 1 depend on the values of the other weights [see (5.5)]. Therefore, when we modify a weight the optimal value for the rest of weights changes. As we are looking for the globally minimum output error probability we need a further step.

In the following we state and prove the Theorem that leads to the global optimization of weights. Before this we want to note the existence of multiple sets of optimal weights if we use non-normalized weights. Let us assume that the optimal set of weights that minimizes the variance $\sigma^2_{y'}$ of a particular AVG is $(k^*_1, k^*_2, \ldots, k^*_R)$. Then $(\alpha k^*_1, \alpha k^*_2, \ldots, \alpha k^*_R)$ is also an optimal set of weights. Indeed, as the weighted average variance only depends on the normalized weights [see (3.7)], any scaling operation in the set of weights does not affect its value. The following Theorem uses normalized weights c_i in order to avoid multiple solutions.

Theorem 1: Given a weighted average $y' = \sum_{i=1}^{R} c_i y_i$ of R variables normally distributed $y_i \sim N(y, \sigma_i)$, $\forall i \in (1 \ldots R)$, and averaging weights that satisfy the normalization condition $\sum_{i=1}^{R} c_i = 1$, there always exists a set of weights c_i^{opt}, $i = 1, \ldots, R$, that optimally minimizes the standard deviation of the weighted average $\sigma_{y'}$.

Proof: We use recursion to prove this Theorem. Let us express the variance of the weighted average as follows:

$$\sigma_{y'}^2 = \sum_{i=1}^{R} c_i^2 \sigma_i^2 = c_R^2 \sigma_R^2 + \sum_{i=1}^{R-1} c_i^2 \sigma_i^2$$
$$= c_R^2 \sigma_R^2 + (1-c_R)^2 \underbrace{\sum_{i=1}^{R-1} \left(\frac{c_i}{1-c_R}\right)^2 \sigma_i^2}_{S_{R-1}^2} \tag{5.6}$$

If we look at the S_{R-1}^2 definition, we observe that it corresponds to a weighted average with the first $R-1$ input replicas. The averaging weights defined as $c_i^* \equiv c_i/(1-c_R)$ add the unity, and therefore accomplish the normalization condition. Hence, S_{R-1}^2 can be regarded as a new minimization problem with $R-1$ normalized weights c_i^* independent from c_R.

Thereby, the $\sigma_{y'}^2$ minimization problem can be split into two independent subproblems (type 1 and type 2):

1. First, to minimize S_{R-1}^2 as a function of $R-1$ normalized weights c_i^*, $i = 1, \ldots, R-1$.
2. Second, to minimize $\sigma_{y'}^2$ as a function of a single weight c_R ($\sigma_{y'}^2 = c_R^2 \sigma_R^2 + (1-c_R)^2 S_{R-1}^2$). In this case S_{R-1}^2 is regarded as a constant factor obtained from the first minimization subproblem.

The second subproblem presents a straightforward solution. The minimization of $\sigma_{y'}^2$ as a function of c_R, being S_{R-1}^2 independent from c_R, leads to the optimal value $c_R^{opt} = S_{R-1}^2/(S_{R-1}^2 + \sigma_R^2)$. Substituting c_R^{opt} in the second derivative it yields a positive value and confirms that c_R^{opt} corresponds to the unique minimum of the function.

In turn, the first subproblem can be split again into two independent subproblems as before (type 1 and type 2). If we keep splitting the problem this way, $R-2$ times in succession, we get a subproblem that cannot be split again. It corresponds to a function S_2^2 that only depends on two normalized weights and two input variances. Thus, it can be solved like the second subproblem type.

So far we have decomposed the problem into $R-2$ subproblems of type 2 plus one of type 1 that can be solved optimally. As a result, we have demonstrated that the whole problem can be solved optimally, and there exists an optimal set of weights that minimizes the weighted average variance.

□

5.1.3 Optimal Unbalanced Weights

In the previous subsection we demonstrated that it exists a optimal set of weights that minimizes the output error probability for the U-AVG structure. Therefore, there must be a way to express the optimal values of weights independently of the other weights and that is only function of the input variability levels. In order to find this formula we perform the following analytic computation. We minimize the variance of the weighted average $\sigma^2_{y'}$, or equivalently its standard deviation that is directly related to P_e as (2.2) indicates. To perform this minimization considering all the weights c_i simultaneously, we have to use the Lagrange multipliers introducing the additional restriction of normalized weights. The target function is the variance $\sigma^2_{y'}$ and the variables to optimize are the magnitudes of the averaging weights c_i. The normalized weights condition ($\sum_{j=1}^{R} c_j = 1$) must hold. Therefore the target function is

$$F(c_1, c_2, ..., c_R, \lambda) = \sigma^2_{y'} - \lambda \left(\sum_{j=1}^{R} c_j - 1 \right). \tag{5.7}$$

Differentiating with respect to the normalized weights c_i, $i = 1, \ldots R$, recall (3.7), and the Lagrange multiplier λ, we obtain the following equations

$$\frac{\mathrm{d}(F)}{\mathrm{d}c_i} = 2c_i^{\mathrm{opt}}\sigma_i^2 - \lambda \qquad i = 1, \ldots, R, \tag{5.8}$$

$$\frac{\mathrm{d}(F)}{\mathrm{d}\lambda} = 1 - \sum_{j=1}^{R} c_j^{\mathrm{opt}}. \tag{5.9}$$

Matching to zero (5.8), we obtain that the optimal weights are inversely proportional to the input variances

$$c_i^{\mathrm{opt}} = \frac{\lambda}{2\sigma_i^2} \tag{5.10}$$

Equation (5.9) equal to zero expresses the condition of normalized weights; combining both conditions, we deduce the value of λ

$$\lambda = \frac{2}{\sum_{j=1}^{R} 1/\sigma_j^2}. \tag{5.11}$$

Now we can calculate the explicit formula for the optimal weights

$$c_i^{\mathrm{opt}} = \frac{1}{\sum_{j=1}^{R} \sigma_i^2/\sigma_j^2} \qquad i = 1, \ldots, R. \tag{5.12}$$

This is the configuration of weights that optimally minimizes the error probability P_e. Observe that c_i^{opt} values are only dependent on the input variances σ_i^2. Depending on the input drifts distribution, each weight should be tuned according to (5.12) in order to achieve the lowest possible output error probability P_e. Now we can verify all the conclusions extracted from Figure 5.4. We can see how the optimal weights are small when the input variance is large and vice versa.

We can also calculate the magnitude of the minimum possible variance for the weighted average that we obtain when we apply the optimal set of weights. Using (3.7) and the expression of optimal weights $c_i^{\text{opt}} = \lambda/(2\sigma_i^2)$, we get a closed expression for the minimum weighted average variance

$$\sigma_{y'\ \text{min}}^2 = \frac{\lambda}{2} = \frac{1}{\sum_{j=1}^{R} 1/\sigma_j^2}. \tag{5.13}$$

Therefore, it is possible to express each optimal weight in terms of its input variance and the minimum possible variance of the weighted average

$$c_i^{\text{opt}} = \frac{\sigma_{y'\ \text{min}}^2}{\sigma_i^2} \qquad i = 1, \ldots, R. \tag{5.14}$$

The optimal configuration has all the weights c_i^{opt} directly proportional to the constant $\sigma_{y'\ \text{min}}^2$ and inversely proportional to the respective input variances σ_i^2. We note that the particular case in which one or more inputs have null variability $\sigma_i = 0$ has to be treated separately. If this situation happens, then the output error probability minimization is straightforward: it would suffice us to assign the maximum weight to the input with null variability $c_q = 1$ and we would obtain the lowest possible output error probability, $P_e = 0$, according to (3.7) and (2.2). However this can never reflect a real case as, in practice, there is always at least a small noise contribution that affects all the inputs.

5.1.4 AVG versus U-AVG

To assess the implications of our proposal, we carry on a reliability analysis for AVG with balanced and unbalanced weights. Given a non-uniform input drift scenario, we calculate the yield of AVG and U-AVG taking as a reliability requirement the reference value presented in Section 2.2 ($P_e < 10^{-4}$). The used definition of yield can be found in Section 2.4. In order to simulate realistic environments with heterogeneous variability, we use the aging and degradation model introduced in Section 2.3 but in an static form (not changing with time). We basically generate the per replica drift variances following

the Gamma distribution function

$$\sigma_i^2 \sim \Gamma(x; k, \phi) = \begin{cases} \frac{1}{\phi^k \Gamma(k)} x^{k-1} e^{-x/\phi} & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.15}$$

with scale parameter $\phi = 2$ and different shape values k reproducing increasing values of the input replica variances.

In the following experiment, we simulate the AVG behavior in different technology scenarios. Each scenario is modeled with heterogeneous input drift variances σ_i^2 following the Gamma distribution function with different mean values. We basically perform 10 000 Monte-Carlo simulations and estimate the yield for both architectures (AVG and U-AVG). Figure 5.5 presents the simulation results against the redundancy factor. One can observe that U-AVG can deliver the same yield than AVG with a much lower redundancy level R. For example, if we required a 90% yield, given a variability scenario with $E\{\sigma_i\} = 3\sigma_{\max}$, we would need eight replicas with the AVG while only two with the U-AVG. This corresponds to a $4\times$ redundancy saving.

Thus far we have demonstrated that if we know the distribution of deviations among the replicas at the design time, we can provide better reliability levels at lower cost by configuring the AVG weights accordingly. However, this improvement is optimal only for static variability conditions of the system. If the levels of variation gradually change in time due to degradation, the unbalanced design may become suboptimal. In the next section, we make a step further and propose a dynamically adapting structure that modifies the weight values at run-time in order to keep track with the possible variations of the input deviations.

5.2 The Adaptive Averaging Cell (AD-AVG)

In the previous section, we analyzed the reliability of AVGs in static environments of variability. We introduced the concept of heterogeneous variability scenario and we demonstrated the advantages of configuring the averaging weights according to the different input levels of variability. In this section we also consider time-variation of the input levels of variability in order to take into account the effects of degradation and aging and dynamically adjust the AVG configuration such that we can tolerate the maximum possible amount of variation. For this reason, we have to equip the AVG with the capability of dynamically reconfiguring the averaging weights according to the time-varying input variabilities measured at run-time. We call this dynamic architecture the Adaptive Averaging Cell (AD-AVG). Our AD-AVG proposal, graphically represented in

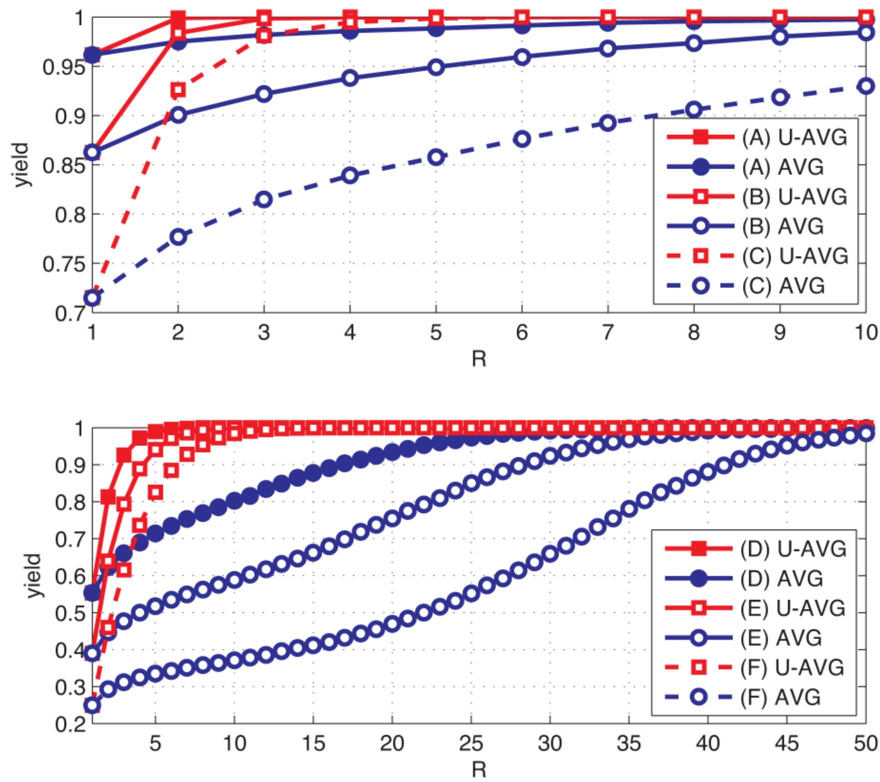


FIGURE 5.5: Yield analysis of AVG and U-AVG against the redundancy factor for different technology scenarios. The simulated scenarios identified with letters (from A to F) have associated heterogeneous input variances following the Gamma distribution. The mean values for the standard deviation range from $E\{\sigma_i\} = \sigma_{\max}$ in scenario A to $E\{\sigma_i\} = 6\sigma_{\max}$ in scenario F. $\sigma_{\max}$ corresponds to the reference value 0.1344 V.

Figure 5.6, is based on a Variability Monitor implemented as a disagreement detector like the one suggested in [35]. In Subsection 5.2.3, we develop in detail the proposed

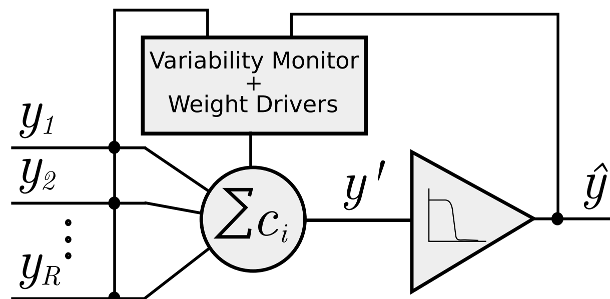


FIGURE 5.6: Adaptive Averaging Cell (AD-AVG) schematic.

technology implementation. In the following we analyze the main characteristics of the AD-AVG structure: redundancy overhead, reliability, and tolerance against degradation.

5.2.1 Adaptive Learning versus Static AVG

Using the degradation model introduced in Section 2.3, we can simulate realistic situations of circuits that degrade in time. In the following, we analyze the difference between using adaptive weights (AD-AVG) and static weights (AVG and U-AVG). To do so, we perform 10 000 Monte Carlo simulations of the AVG structures with increasing amounts of degradation. Figure 5.7 depicts the simulation results of yield for different size AVG circuits against the degradation.

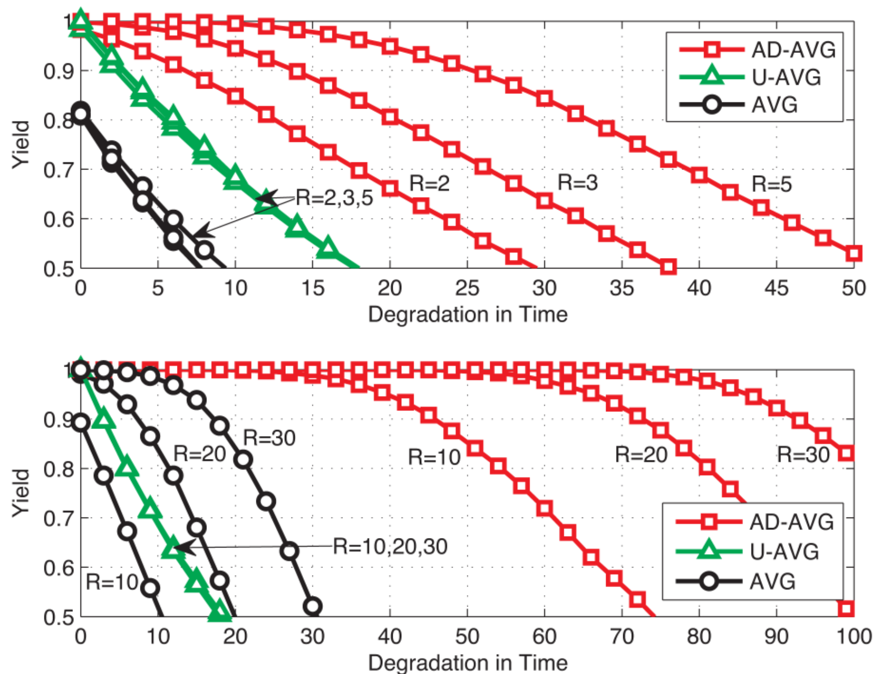


FIGURE 5.7: Yield of different size AD-AVG, U-AVG and AVG against the degradation in time units.

We observe that at degradation in time 0, there is no difference between AD-AVG and U-AVG; both structures are configured with the optimal set of weights and the AVG yield is maximized. However, as the degradation increases, the U-AVG loses yield quickly regardless of the redundancy level. The U-AVG weights configuration is static and only optimal for the initial input variability levels; when the circuit degrades, the configuration of weights becomes suboptimal and after 18 degradation in time units the U-AVG yield drops below the 0.5. On the other hand, the AD-AVG technique is capable of tolerating much higher amounts of degradation and it improves notoriously with the redundancy level R. We can conclude that the AD-AVG always provides the maximum yield given any particular redundancy factor R.

Comparing the characteristic curves of U-AVG and AVG in Figure 5.7, we also observe that the unbalanced approach is better than the AVG only for low redundancy levels (< 15). Since the U-AVG does not improve with redundancy but AVG does, in the conditions of this experiment, the AVG outperforms U-AVG for redundancy levels higher than 15. This is an interesting property of the static weight approaches. If we configure the weights according to the information of variability levels of fresh devices, the U-AVG will provide always better results at degradation in time 0. However, as we consider higher levels of degradation the AVG happens to outperform the U-AVG. For this reason, we dismiss the U-AVG technique and focus now on the comparison between AVG and AD-AVG.

Figure 5.8 depicts the results of 10 000 Monte Carlo simulations comparing the AD-AVG versus AVG, but instead of using the same redundancy level, we impose the same reliability target under specific degradation conditions. The figure clearly demonstrates that the AD-AVG consumes less redundancy than AVG: from 9.5× redundancy saving for a 90% yield after seven degradation in time units to 4.1× redundancy saving for a 90% yield after 75 degradation in time units.

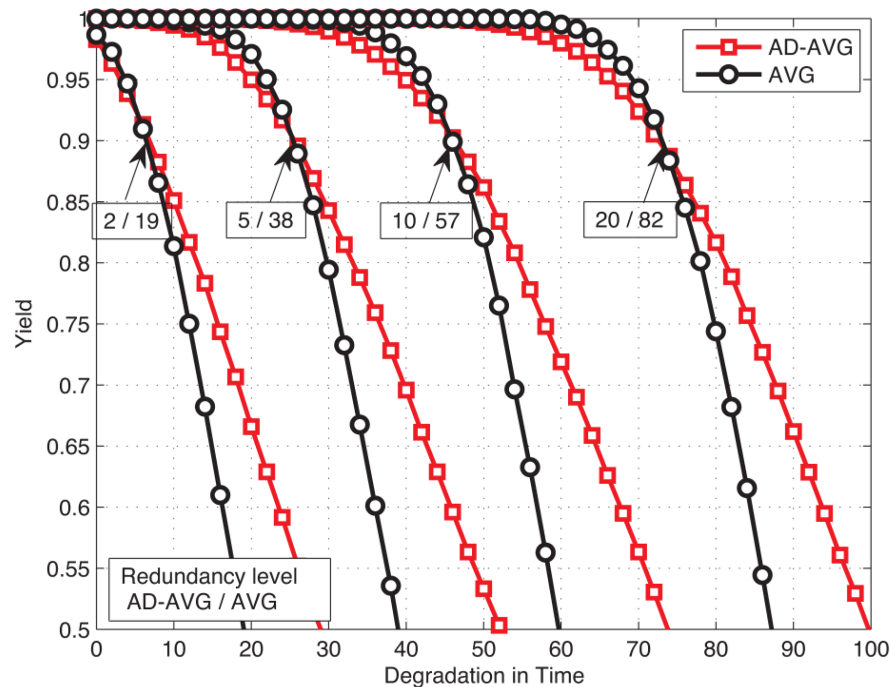


Figure 5.8: Yield analysis of different size AD-AVG and AVG against degradation. Redundancy levels are chosen to meet the reliability requirements: 90% yield after 7, 25, 45 and 75 degradation in time units.

5.2.2 Noisy Variability Monitor in the AD-AVG

Thus far we assumed that the information which is driving the learning process is provided by an ideal Variability Monitor. In practice this cannot be the case as there are many sources of noise in a real environment such as temperature, external radiation, interference within the same operating circuit, and discretization noise. Therefore, the input variances σ_i^2 estimated by the Variability Monitor are subject to noise:

$$\widehat{\sigma_i^2} = \sigma_i^2 + \xi_i. \tag{5.16}$$

We model the noise level ξ_i as a Gaussian random variable with null mean and standard deviation σ_s, $\xi_i \sim N(0, \sigma_s)$. The Variability Monitor is based on the computation of the disagreements between the AD-AVG output $\hat{y}$ and the signal provided by each replica y_i. This mechanism was introduced by Mathur and Avizienis in [35]. The averaging weights are reconfigured by the Weight Drivers according to the input variance estimators and the optimal averaging weights' formula found in Subsection 5.1.3 [see (5.12)]:

$$c_i = \frac{1/\widehat{\sigma_i^2}}{\sum_{j=1}^{R} 1/\widehat{\sigma_j^2}} \tag{5.17}$$

In the following experiment, we analyze the influence of this noise in the effectiveness of the learning process. We perform 10 000 Monte Carlo simulations of the AD-AVG against degradation including the effect of different levels of noise in the Variability Monitor. Figure 5.9 reproduces the obtained results for the yield of different size AD-AVG.

As expected, the noise added to the measures provided by the Variability Monitor worsens the characteristic reliability of the AD-AVG. We observe that the negative impact of noise increases with the redundancy level. For example, if we want a 90% yield in two-input AD-AVG with a noise affecting the monitor with standard deviation $\sigma_s = 0.06\ V$, then the lifetime is reduced in two degradation in time units with respect to the noise-free case, whereas the same noise in the case of five-input AD-AVG reduces the lifetime in seven degradation in time units. From this experiment, we can conclude that the use of sensors to learn the variability changes over time is useful to improve the reliability of AVG structures as long as the noise in the Variability Monitor is small or comparable to the variability levels that it has to measure. The characteristic reliability of AD-AVG degrades with unreliable Variability Monitors, and therefore the particular tradeoff between reliability and overhead should be analyzed in each case.

If we extend this experiment to AD-AVGs with larger number of replicas, we discover an interesting but counter-intuitive resonance phenomenon that deserves further study.

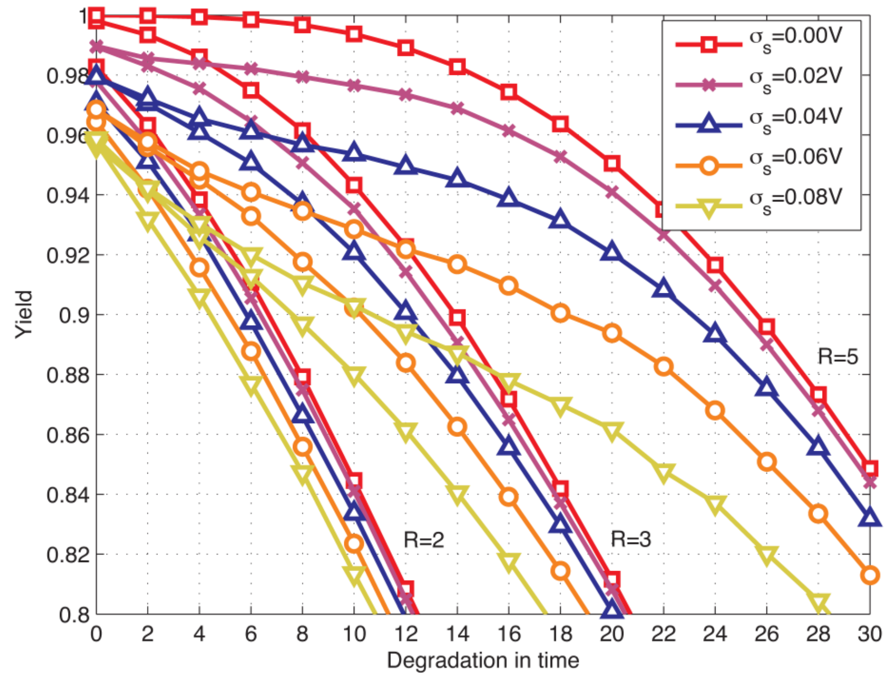


FIGURE 5.9: Yield analysis of different size AD-AVG against degradation. Simulations include the effect of different noise levels in the Variability Monitor (from $\sigma_s = 0\ V$ to $\sigma_s = 0.08\ V$).

Under particular conditions of noise in the Variability Monitor and redundancy level the AD-AVG reliability starts diminishing against degradation, as expected, but at a certain moment in time it increases due to a resonance phenomenon and reaches a maximum. After this peak of maximum reliability it decreases again with degradation. We refer to this effect as Degradation Stochastic Resonance (DSR) and devote the next Section 5.3 to analyze in detail its fundaments, the impact it has on the reliability of AD-AVGs, and the possibility to control it.

5.2.3 AD-AVG Implementation

In the following, we suggest a possible implementation for the AD-AVG technique. Although we did not propose a specific implementation for the U-AVG in the previous section, this could be obtained similarly to the AD-AVG but simplifying the Variability Monitor to a Detector that would operate only in an initial state of setting up. The rest of the structure have exactly the same functionality, and thus uses the same implementation. The AD-AVG structure graphically depicted in Figure 5.10 is based on the use of a Variability Monitor, a Weight Drivers block and a crossbar of switching resistive devices, such as memristors [64, 65]. The dynamic adjustment of weights requires, on one hand, a technique for gathering the required information and, on the other hand, a technology

with high reconfiguration capabilities to implement the adjustable weights. The use of resistive switching crossbars provides this reconfiguration feature and represents a good candidate for future technology with a high integration capability.

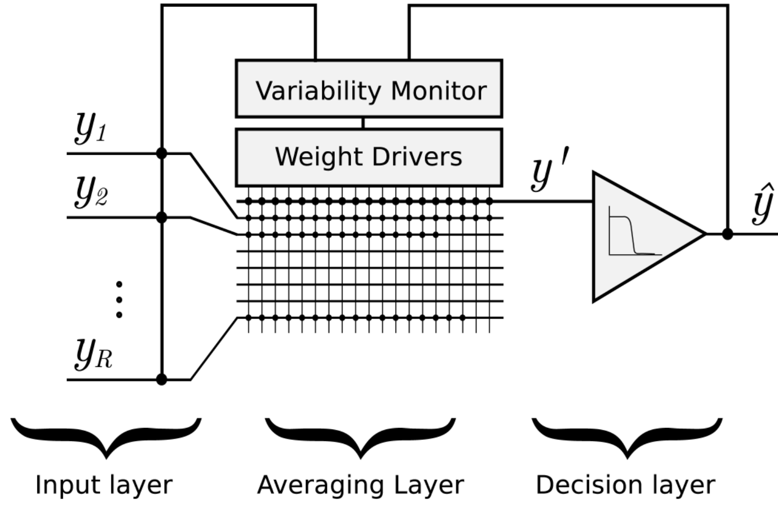


FIGURE 5.10: Adaptive Averaging Cell (AD-AVG) implementation in switching resistive crossbar technology.

The whole architecture can be decomposed in three layers. The first one corresponds to the input layer and receives the input signals from the replicas. The second layer performs the averaging operation and is composed by the resistive switching crossbar, the Variability Monitor, and the Weight Drivers. Finally, the third layer is the decision layer, it restores the binary output value by means of a threshold function.

5.2.3.1 Variability Monitor

We first propose a topology for the Variability Monitor implementation. It is based on a disagreement detector between the AD-AVG output $\hat{y}$ and the signal provided by each replica y_i, $i = 1, \ldots R$. This mechanism was already used by Francis P. Mathur and Algirdas Avizienis in [35]. Checking at runtime the differences between the AD-AVG output, which is statistically more reliable than the inputs, and each of the input replicas, we can establish a criterion for evaluating the relative variability of each replica. For example, we can estimate the standard deviation associated to each replica $\hat{\sigma_i}$ by counting the number of times n_i that the difference $\hat{y} - y_i$ exceeds a certain level L in a given number of clock cycles N. With this number we can access a look up table that stores the relationship between n_i and $\hat{\sigma_i}$:

$$\hat{\sigma_i} = \frac{L}{\sqrt{2}\,\text{inverfc}\,(2n_i/N)} \qquad i = 1, \ldots, N \tag{5.18}$$

This mechanism allows us to keep the adapting process running during the circuit normal operation, and therefore it does not require any test implementation to obtain the information of input variability levels.

5.2.3.2 Weight Drivers and Switching Resistive Crossbar

In order to implement the adjustable averaging weights required by the AD-AVG architecture, we propose the use of switching resistive crossbars. Figure 5.11 reproduces the basic crossbar layout for the AD-AVGs. In this structure, each input replica is connected to a horizontal metal line that can be connected or disconnected to N vertical lines by means of resistive switching devices which are located on the crossing points of the structure. In Figure 5.11, black dots correspond to connected devices. The state of each device can be easily controlled by the Weight Drivers by selectively applying specific configuring voltages to the vertical and horizontal lines. The basic idea to implement an adjustable averaging weight for each input line is to connect or disconnect more or less devices in the corresponding input line, and thus obtain a configurable connection resistance. Using this feature we can set up a network of interconnects that averages the input replicas with specific reconfigurable averaging weights. We refer by n_i to the number of connected devices in the input line i.

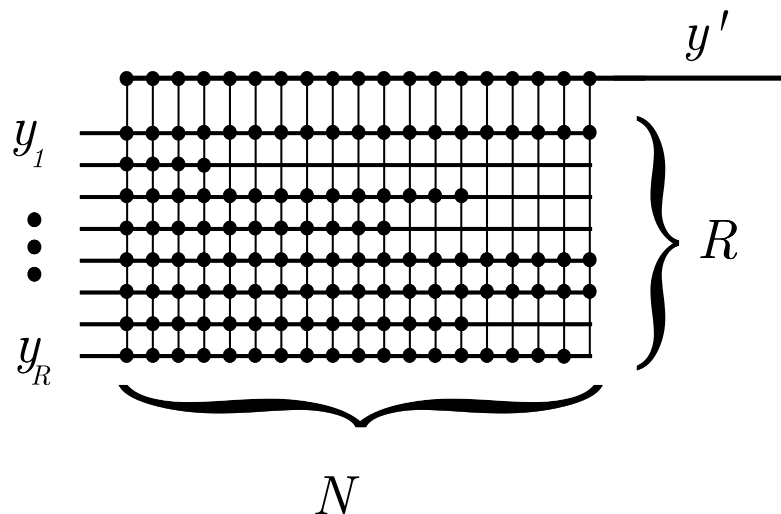


FIGURE 5.11: Crossbar layout view of AD-AVG to configure the averaging weights.

Hence, given a redundancy factor R from the initial stages of the AD-AVG design; this parameter is associated to the number of horizontal metal lines. The number of vertical metal lines N determines the maximum number of interconnects per input, and therefore the minimum resistance value and the maximum accuracy to configure the weight values. The total area in the crossbar structure is proportional to the number of vertical metal lines and the redundancy factor ($Area \propto R \times N$). During the AD-AVG operation

Weight Drivers receive the information regarding the relative replica variabilities from the Variability Monitor. Based on these data a different number of cross-points n_i in the switching resistive crossbar are connected or disconnected. In order to increase the weight value of a particular input, the number of connected cross-points is increased, and vice versa, to decrease the value cross-points are disconnect in the corresponding horizontal line. Apart from the configurable region of $R \times N$ devices, figure 5.11 also shows an additional metal line that corresponds to the output signal y'. This line is connected to all the vertical lines and collects the weighted average of the inputs. The resulting signal is fed to a threshold function that amplifies the signal and restores the binary value.

The resistive switching devices exhibit a different resistance value depending on the configuration state (R_{on} and R_{off}). For example, in the case of memristors the characteristic resistances are: $R_{\text{on}} \simeq 1$ MΩ and $R_{\text{off}} \simeq 1$ GΩ. Analyzing the equivalent circuit we can calculate the analytic expression of signal y' in terms of the inputs y_i and the crossbar configuration of connections n_i, $y = 1, \ldots, R$, see (5.19).

$$y' = \frac{1}{RNR_{\text{on}} + \sum_{i=1}^{R} n_i \left(R_{\text{off}} - R_{\text{on}}\right)} \sum_{i=1}^{R} \left[NR_{\text{on}} + n_i \left(R_{\text{off}} - R_{\text{on}}\right)\right] \times y_i \tag{5.19}$$

In order to prove the feasibility of the proposed implementation we perform a Monte Carlo simulation of a 5-input AD-AVG with the described topology. The simulation uses the degradation model introduced in Subsection 2.3. Figure 5.12 depicts the temporal evolution of the input variances (σ_i^2), the adaptive averaging weights (c_i), and the variance of the weighted average ($\sigma_{y'}^2$). We can observe how the input variances keep increasing over time and how the averaging weights are modified in order to minimize the weighted average variance.

5.3 DSR-aware AD-AVG

In this section, we analyze in detail the Degradation Stochastic Resonance (DSR) phenomenon, which was first mentioned in Subsection 5.2.2, and increase the AD-AVG design to take notice of it. This counter-intuitive effect takes place in the AD-AVG structure as a result of the combined effect of hardware degradation and the noise present in the Variability Monitor, i.e., the AD-AVG part in charge with the evaluation of the variability levels in the input replicas. The DSR effect is related to the well-known Suprathreshold Stochastic Resonance (SSR), which was first analyzed by Stocks in 2000 [66]. Some interesting applications of SSR phenomenon are sigma-delta modulators [67] and analog-to-digital converters [68]. In the case of AD-AVG, DSR

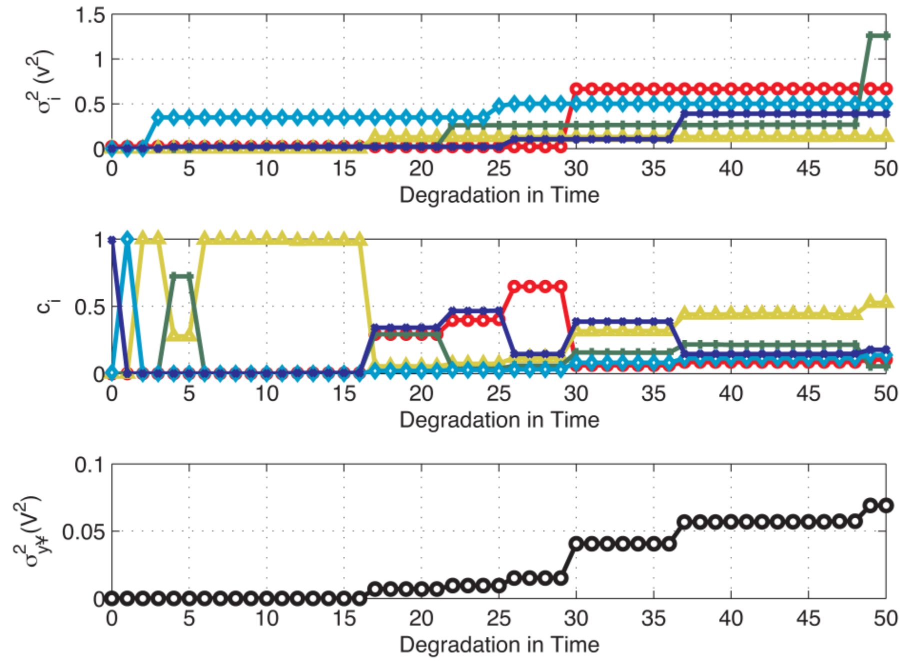


Figure 5.12: Monte Carlo simulation of a 5-input AD-AVG with the proposed topology. The subplots correspond to the five input replica variances (σ_i^2), the five adaptive averaging weights (c_i), and the variance of the weighted average ($\sigma_{y'}^2$) against the degradation.

occurrence provides an unexpected reliability enhancement after a certain degradation level is reached and under specific noise conditions. Our simulations indicate that a DSR induced reliability peak is reached at a particular amount of degradation in time, which depends on the AD-AVG reliability level and input variability level. Thus, the DSR phenomenon implies that while the system degradation increases the AD-AVG reliability evolves as follows: after an initial decrease, it improves until a maximum value is reached at the DSR peak, after which it finally steadily decreases to zero.

For example, if we take a 20-input AD-AVG with a noise level in the Variability Monitor of 0.06V the reliability evolves as follows (see Figure 5.13): (i) The yield starts from a high value of 0.97 in fresh devices and gradually decreases to 0.89 when the degradation level is reaching 35 units; (ii) As the degradation is increasing, due to the DSR effect, the yield starts increasing up to 0.94, at the DSR peak which corresponds to an accumulated degradation in time of 60 units; (iii) After the DSR peak the circuit's accumulated degradation causes a drop in the yield characteristic. This particular AD-AVG example was chosen to clearly show the DSR effect. In general, DSR causes different yield evolutions depending on the circumstances of redundancy, noise and degradation level. We also want to point out that even in current technology a 20x replication appears to be unacceptable, due to device scaling, which results among other positive aspects in

a low reliability, such a solution might be unavoidable. Moreover if we consider some emerging technologies such as solid-state nanopores high replications levels of up to 20 are not prohibitive any longer [69, 70].

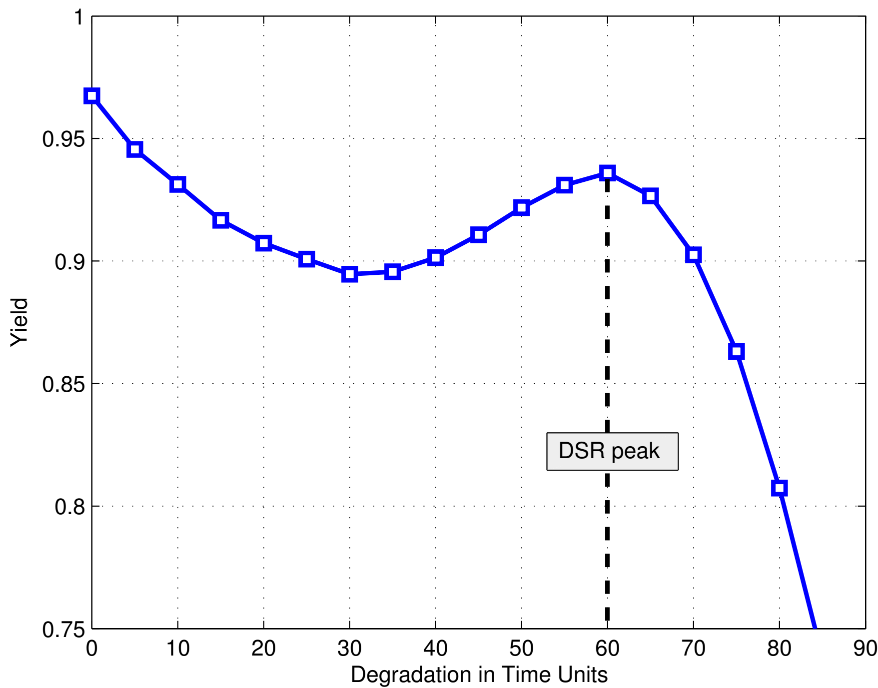


FIGURE 5.13: Yield of 20-input AD-AVGs against degradation with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$.

As DSR results in yield improvement under certain aging conditions a legitimate question to ask is: *Can one take advantage of DSR over the entire lifetime of the system?* In the following we address this question and propose a method to artificially create the conditions such that DSR induced yield peaks are obtained for a large part of the system life time, i.e, from fresh devices (in particular, from the moment when the yield falls under a minimum acceptable value) up to the end-of-life (the degradation is that high that no yield improvement is possible any longer).

To this end we first demonstrate that by artificially changing the variability level of the AD-AVG inputs we can control the DSR peak occurrence and position, thus for any degradation level we can find an input variability level that induces a DSR yield peak. Subsequently, we propose to augment the AD-AVG structure with per input controllable noise injectors and to extend the Variability Monitor such that it can compute, apart of the required input weight values, also the noise level that virtually increases the circuit amount of degradation. In this way the AD-AVG is capable of creating the required conditions for the DSR peak occurrence, regardless of the actual system degradation level.

Our simulations indicate that the augmented AD-AVG scheme is capable, besides of eliminating the yield decrease range that normal AD-AVGs exhibit prior to the DSR peak, of providing higher yield levels. This extra benefit can be explained by the fact that even we are placing our structure under the DSR peak conditions the system presents a lower level of accumulated degradation than the one associated to the DSR peak, which results in an yield improvement.

If we apply the proposed DSR control by noise injection to the same example as before, i.e., a 20-input AD-AVG with a noise level in the Variability Monitor of 0.06V, the reliability evolves as follows (see Figure 5.19): (i) The yield starts from a high value of 0.97 in fresh devices and gradually decreases to 0.94 during the first 6 units of degradation in time. In this lifetime part no noise injection is needed; (ii) From 6 degradation units further noise injection is applied and a nearly flat yield curve is obtained until the DSR peak degradation level (60 units) is reached. In this part of the yield characteristic we get a minimum guaranteed yield level of 0.94 and a maximum of 0.97; (iii) After the DSR peak the circuit's accumulated degradation causes a drop in the yield characteristic until it eventually reaches zero.

5.3.1 DSR Fundaments

In this subsection, we analyze the DSR phenomenon aiming at better understanding its basics fundaments. To do so we first focus on a particular case of DSR with a 2-input AD-AVG structure (R=2). This example reveals us the circumstances of noise under which the DSR effect occurs. After this example we also perform a sensitivity analysis of the AD-AVG yield to the degradation of one particular input. This result permits us to generalize the DSR effect to AD-AVG structures with an arbitrary number of inputs. Finally, we show simulation results for the AD-AVG structure with different redundancy factors R and noise levels in the Variability Monitor (σ_s) in order to conclude a clear overview of the DSR impact in AD-AVG structures.

5.3.1.1 DSR in 2-input AD-AVG

Focusing on a simple AD-AVG case with 2-inputs we are able to perform an exact analytical study of the DSR effect. We analyze the case of a 2-input AD-AVG when one of the inputs has very small, even null variability level ($\sigma_1 \approx 0\ V$) and the variability of the other one (σ_2) increases over time as a consequence of degradation. In order to perform the calculations we model the statistics of the input variability levels with parameters $\sigma_1(\approx 0\ V)$, σ_2, the noise in the Variability Monitor σ_s, and the reliability specification of maximum admissible output variability $\sigma_{\max}$ (above this level we assume cell failure).

First, we perform a Monte Carlo simulation; Figure 5.14 depicts the resulting yield against the variability in the input 2 (σ_2). We observe that the yield is always 1 when

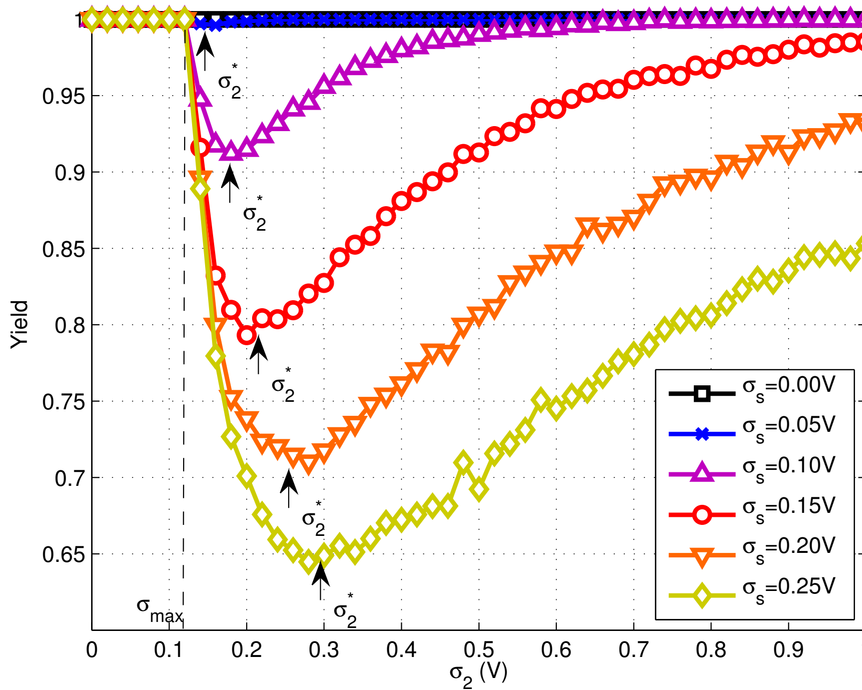


FIGURE 5.14: Monte Carlo simulation result of a 2-input AD-AVG yield against σ_2 with null variability in the input 1 ($\sigma_1 = 0\ V$). We consider different levels of noise in the Variability Monitor (σ_s) and a maximum admissible output variability of $\sigma_{\max} = 0.1344\ V$.

$\sigma_2 < \sigma_{\max}$. This is because, in these conditions, any possible combination of weights gives place to an output with a variability level lower than the maximum tolerated. However, as soon as σ_2 becomes greater than $\sigma_{\max}$ the yield begins to decrease at different rates depending on the level of noise in the Variability Monitor. Obviously, the greater the noise the greater the yield loss. This detrimental effect of degradation, or increase in the variability, happens only during a certain range of input variability. After a critical variability level in the input 2 (σ_2^*) the effect of further degradation changes and becomes beneficial; this is the starting point for the DSR effect in this particular case (Figure 5.14). We can understand this phenomenon by realizing that as the input variability σ_2 grows the AD-AVG is capable of calculating more precisely the optimal value of the averaging weights. The ratio between the variability σ_2 and the noise in the Variability Monitor σ_s increases and, therefore the measure of input variability necessary to calculate the averaging weights becomes more reliable. In fact, the AD-AVG gives more weight to the input 1, which has null variability, and the yield grows up to 1 again.

For this particular case it is possible to prove the effect analytically. Given the conditions described for this example we can calculate the probability density function of the

averaging weights $f_{c_1}(c_1)$, $f_{c_2}(c_2)$ and the yield of the AD-AVG structure Y. Taking the derivative of the yield with respect to the variability level σ_2 we obtain a closed analytic expression that reveals the impact of degradation in the reliability of the structure, see (5.20).

$$\begin{aligned}\frac{dY}{d\sigma_2} =& \frac{1}{\sqrt{2\pi}\sigma_s}\sqrt{\frac{\sigma_{\max}}{\sigma_2}}e^{-\frac{\sigma_{\max}\sigma_2}{2\sigma_s^2}}\operatorname{erf}\left(\frac{\sqrt{\sigma_2(\sigma_2-\sigma_{\max})}}{\sqrt{2}\sigma_s}\right)\\ &-\frac{\sigma_{\max}}{\pi\sigma_2\sqrt{\sigma_2(\sigma_2-\sigma_{\max})}}e^{-\frac{\sigma_2^2}{2\sigma_s^2}}\end{aligned} \tag{5.20}$$

Matching to zero (5.20) we obtain the condition in terms of accumulated degradation (σ_2) that the AD-AVG system must satisfy in order to start experiencing the DSR effect, see (5.21). In this example we define this characteristic point of change from detrimental to beneficial degradation as the critical variability level σ_2^*.

$$\frac{1}{\sqrt{\pi}a} \times e^{-a^2} = \operatorname{erf}(a) \tag{5.21}$$

Where

$$a^2 = \frac{\sigma_2^*(\sigma_2^* - \sigma_{\max})}{2\sigma_s^2}. \tag{5.22}$$

Solving this equation we get the expression of σ_2^*:

$$\sigma_2^* = \frac{\sigma_{\max}}{2} + \sqrt{\left(\frac{\sigma_{\max}}{2}\right)^2 + 0.769\ \sigma_s^2}. \tag{5.23}$$

Substituting this formula with the simulation parameters of Figure 5.14 we can verify the analytical model, see Table 5.1, when compared with simulation results. Using this relation it is possible to find the critical variability level σ_2^* of the AD-AVG for any given noise level and reliability requirement.

5.3.1.2 AD-AVG yield sensitivity analysis

In order to extend the previous result to AD-AVG systems with arbitrary number of inputs, we perform a sensitivity analysis of the AD-AVG's yield to the variability level of one particular input (the ith one). With this experiment we demonstrate the occurrence of the DSR effect in a general AD-AVG case. To do this we assume an R-input AD-AVG and analyze the sensitivity of the weighted average variance against the contribution of

Table 5.1: Critical variability levels (σ_2^*) of DSR effect in a particular case of 2-input AD-AVG with null variability in the first input.

σ_s (V)	σ_2^* (V)
0.00	0.1344
0.05	0.1474
0.10	0.1777
0.15	0.2149
0.20	0.2550
0.25	0.2965

the ith input. We separate the general output variance expression as follows:

$$\sigma_{y'}^2 = c_i^2\sigma_i^2 + \sum_{j=1, j\neq i}^{R} c_j^2\sigma_j^2. \tag{5.24}$$

We generate random sets of $R-1$ variance values σ_j^2, $j = 1, \ldots, R$, $j \neq i$, following the Gamma distribution function as in previous studies [71]. As for the ith variability (σ_i), we swept its value from 0 to 0.4 V in the Monte Carlo simulations. In these conditions, we estimate the AD-AVG's yield taking into account different levels of noise in the Variability Monitor (σ_s). Figure 5.15 depicts the simulation results against the influence of σ_i.

We note in this experiment that ideally (when the noise in the Variability Monitor is null) the AD-AVG yield is 1 as long as $\sigma_i < \sigma_{\max}$, and after this value the yield decreases gradually to 0.93. This happens because the AD-AVG structure with null noise perfectly optimizes the reliability. Therefore, while the ith variability is lower than the reliability requirement $\sigma_{\max}$ the AD-AVG yield is 1, and after this value the yield decreases gradually to the maximum yield achievable with the remaining $R-1$ inputs. On the other hand, when the Variability Monitor is affected by noise the optimal weight values are calculated imperfectly and the yield is not maximized. We observe in the figure that the impact of this imperfect weight configuration is not homogenous throughout all the values of ith input variability. The yield loss presents a resonance effect with the ith input variability for different levels of noise in the Variability Monitor (σ_s). This result together with the previous example evidences the relevance of the DSR effect in the AD-AVG structure.

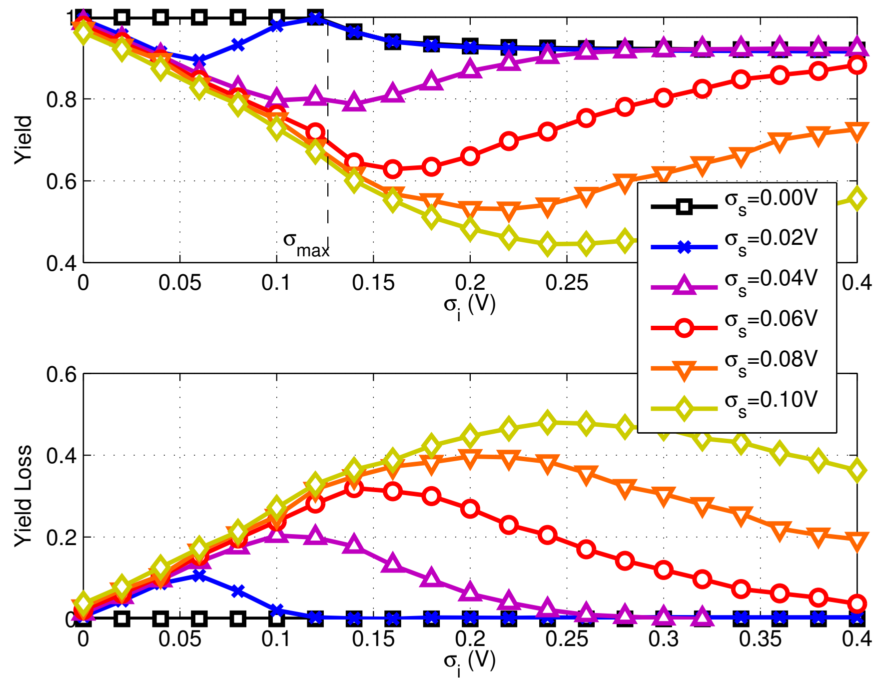


FIGURE 5.15: Sensitivity analysis of AD-AVG's yield with multiple inputs to the variability of input ith for different levels of noise (σ_s) in the Variability Monitor. The figure above shows the AD-AVG yield and the lower shows the yield loss which is defined as the yield with a null noise in the Variability Monitor ($\sigma_s = 0$) less the yield.

5.3.2 DSR in AD-AVG

Once we have perceived the intuition of the DSR effect and demonstrated its applicability to general AD-AVG cases we focus in this subsection on the analysis of typical AD-AVG. Following with the previous simulations we analyze now AD-AVG structures against the degradation in time and observe the evolution of reliability. This time we need to use the degradation model described in Section 5.2. We simulate different size AD-AVGs with different levels of noise in the Variability Monitor (σ_s) and estimate the corresponding yield. Figure 5.16 depicts the simulation results for redundancy factors $R = 3$, 10, and 20.

In the figure we clearly observe how the yield characteristic of AD-AVG changes over time due to the DSR effect. We also note that the influence of this resonance phenomena is more significative at higher levels of redundancy. In the examples of Figure 5.16 the DSR effect is not perceived for the 3-input AD-AVG whereas it does for the 10 and 20 inputs AD-AVG. Another conclusion that we can extract from the figure is that DSR implies a diminution of the negative influence of noise in the Variability Monitor when the accumulated level of degradation is significantly higher than the noise level. In fact, both Figure 5.15 and Figure 5.16 show a negative impact of noise in the Variability Monitor

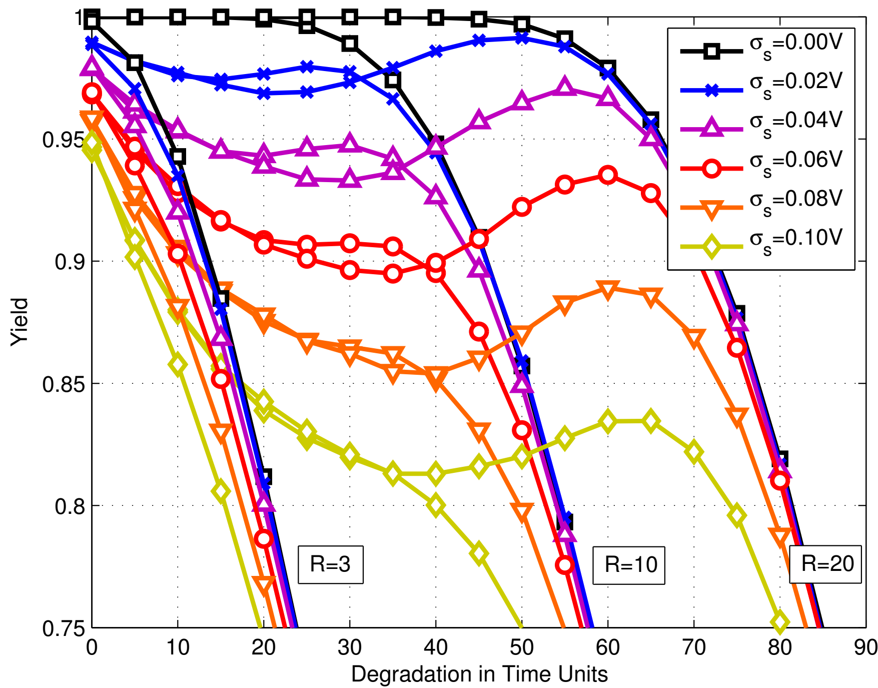


FIGURE 5.16: Yield analysis of different size AD-AVGs against degradation for different levels of noise (σ_s) in the Variability Monitor.

that increases for low levels of degradation or input variability and then decreases after a critical amount of degradation in time. From this point we identify the characteristic degradation in time units associated to the resonance peak as the DSR virtual age of the circuit. For example, the DSR age of 20-input AD-AVG with a noise in the Variability Monitor of $\sigma_s = 0.06V$ is 60 units of degradation in time.

Thanks to this effect it is possible to obtain higher factors of AD-AVG yield after specific amounts of degradation. Regarding the experiment in Figure 5.16 on the DSR effect we can extract the following counter-intuitive conclusions:

- The DSR effect is more relevant in AD-AVGs with larger number of inputs.
- Given an AD-AVG in a particular situation of degradation in time and noise level it is not necessarily the best option to use all the available replicas. There are situations in which less input replicas provide higher yield with the same degradation in time and noise in the Variability Monitor due to the DSR phenomenon.
- We would increase the system yield if we could artificially increase the amount of degradation in time up to the resonance peak. This operation is feasible as long as the level of degradation in time is below the DSR peak degradation level. This will be contemplated in next section.

5.3.3 DSR-aware AD-AVG design

In previous subsections we demonstrated that after a certain degradation level the DSR phenomenon occurs, which results in an yield improvement. Based on this we conclude that degradation can also have beneficial effects and one can embrace it and try to take advantage of it whenever possible. However the DSR enhancement becomes effective only after a certain system degradation is accumulated because one cannot take advantage of it in fresh devices. In order to extend the DSR benefit over the entire system lifetime we propose in this subsection a DSR tailored AD-AVG structure. The main idea behind our proposal is to artificially create degradation symptoms in the AD-AVG such that it exhibits, regardless of the real degradation level, the DSR peak behavior/yield. To this end we first analyze the relation between AD-AVG degradation, input variability, and the DSR occurrence and demonstrate that by increasing the input variability we can induce DSR peak conditions at any degradation level. In other words by increasing the input variability a virtual degradation correspondent to the DSR peak can be induced in the AD-AVG system. Based on this we further demonstrate that by run-time controlling the input variability we can place the system yield above the DSR peak level for a large range of degradation levels, i.e., most of the system lifetime span. Finally, we briefly discuss the practical implications of this method on the AD-AVG cell organization and sketch a potential implementation of DSR tailored AD-AVG cells.

5.3.3.1 DSR Occurrence Control

The main idea behind extending the DSR effect occurrence is to add virtual degradation to the system in order to create DSR peak degradation conditions regardless of its actual degradation level. Thus, to virtually make the system behave as it had the amount of degradation associated to the DSR peak. Degradation affects the hardware and causes a variability increase in the input signals. Therefore, one way to achieve this virtual degradation is to increase the input variability levels to make the system behave like it would have been in a higher degradation status. However, we have to take into account that only reversible approaches can avoid reducing the circuit lifespan. In fact, if we increase the circuit degradation using an irreversible procedure we are maximizing the reliability at the present moment but we are compromising the future. As degradation always keeps increasing irreversibly the amount of virtual degradation we need to get to the resonance peak decreases over time. In this line of reasoning we propose the use of controllable noise injectors that can easily modify the magnitude of noise added to the inputs. A detailed explanation of the proposed controllable noise injectors is given in Section 5.3.3.2. In the augmented AD-AVG structure the practical control of DSR effect takes place inside the Variability Monitor and Weight Drivers block, see Figure 5.17.

Based on the estimated input variability levels $\widehat{\sigma_i^2}$ it is possible to approximate the optimum noise level as shown next.

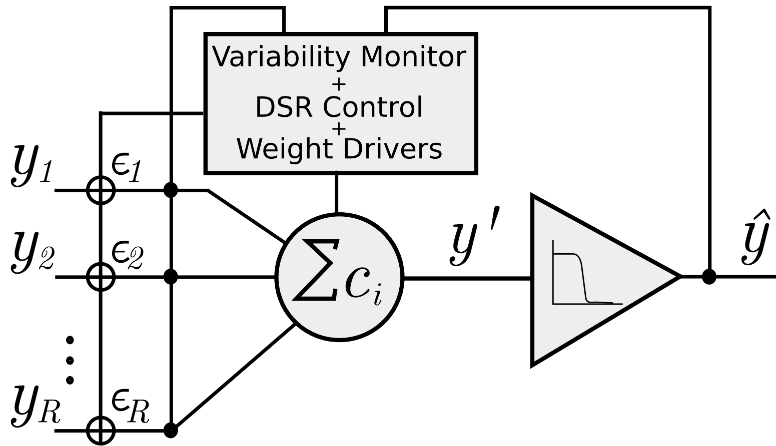


FIGURE 5.17: Adaptive Averaging cell (AD-AVG) architecture with independent noise generators added to the inputs.

In order to test the efficiency of this idea we analyze the impact of adding Gaussian noise to the AD-AVG inputs on the overall system reliability. To perform this analysis we take as a reference the curve in Figure 5.16 corresponding to 20-input AD-AVG with a noise in the Variability Monitor of $\sigma_s = 0.06V$ and compare it with the case when noise with different magnitudes is added to the AD-AVG inputs. We basically repeat the same Monte Carlo simulations which results are depicted in Figure 5.16 but now we are adding a noise signal ϵ_i to each input y_i.

$$y_{i'} = y_i + \epsilon_i. \tag{5.25}$$

We assume that the ϵ_i signals are independent from each other and follow a Gaussian distribution with null mean and standard deviation σ_x ($\epsilon_i \sim N(0, \sigma_x)$). Figure 5.18 depicts in thick blue line the yield of the AD-AVG cell with noise of magnitude $\sigma_s = 0.06V$ in the Variability Monitor and in thin red lines the impact of adding noise to the inputs of magnitude $\sigma_x = 0.05V$, $0.10V$, and $0.20V$. As one can easily observe in Figure 5.18 the DSR peak is shifted towards lower levels of degradation by applying different input noise levels σ_x. For example, if we add an input noise of $\sigma_x = 0.2V$ we shift the DSR peak from 60 to 45 units of degradation in time. This result proves that we can really control the DSR phenomenon in the AD-AVG structure, and we can get the DSR peak enhancement conditions at any level of degradation below the DSR peak. Besides, one can also observe in the figure that this method is not only providing an enhanced yield earlier than the natural DSR peak but it also improves the yield significantly over the DSR peak depending on the particular amount of degradation. For example, in the previous simulation given a degradation level of 45 units we can

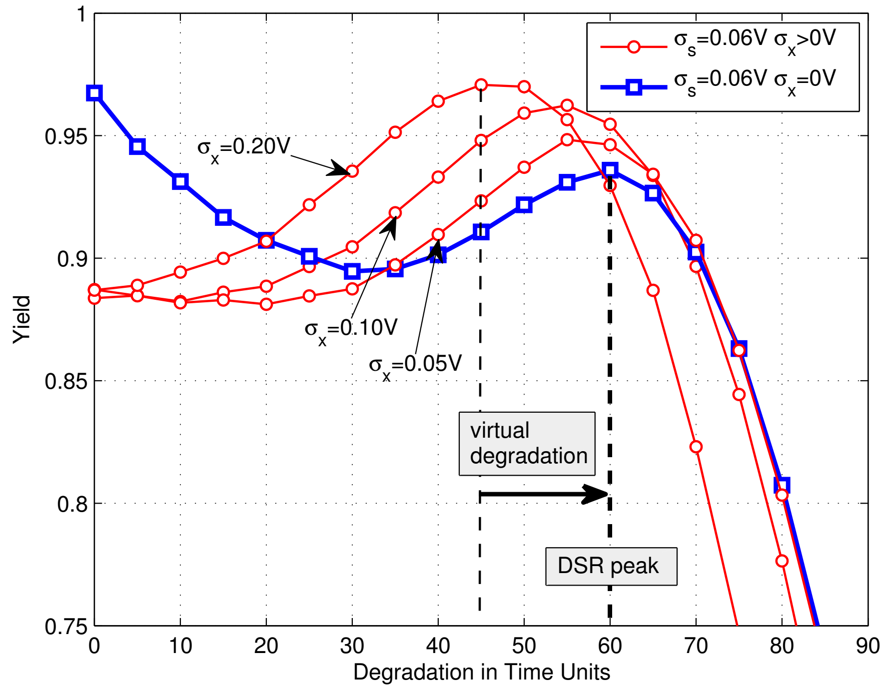


FIGURE 5.18: Yield against degradation of 20-input AD-AVGs with different levels of noise added to the inputs. Thick blue line correspond to the AD-AVG cell yield with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$. Thin red lines correspond to the impact of adding noise to the inputs of this cell with different magnitudes: $\sigma_x = 0.05V$, $0.10V$, and $0.20V$.

increase the yield from 0.91 to 0.97, which is higher than the resonance peak yield, i.e., 0.94.

Given that we demonstrated that we can control the DSR peak position by means of input variation levels we can make a steep forward and define a strategy that allows us to get the maximum yield during all the circuit lifetime, by enabling DSR peak relocation as a consequence of degradation evolution. The basic principle of the DSR control is to check at runtime the instantaneous amount of degradation in terms of the input variability estimators and update the input noise magnitude accordingly. The target is to keep the reliability characteristic at the highest value regardless of the particular degradation level. In order to observe which is the input noise magnitude that we have to inject into the circuit in order to accomplish our goal we present in Figure 5.19 simulation results for a 20-input AD-AVG with a noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$ sweeping over different input noise levels from $\sigma_x = 0V$ to $\sigma_x = 0.9V$. Figure 5.19 depicts in thick blue line the curve associated to the null input noise case, thin colored lines are the curves associated to the sweeping values of σ_x from 0V to 0.9V. We also highlight in the figure in thick black line the curve that the yield follows when we apply the proper input noise magnitude at each degradation in time.

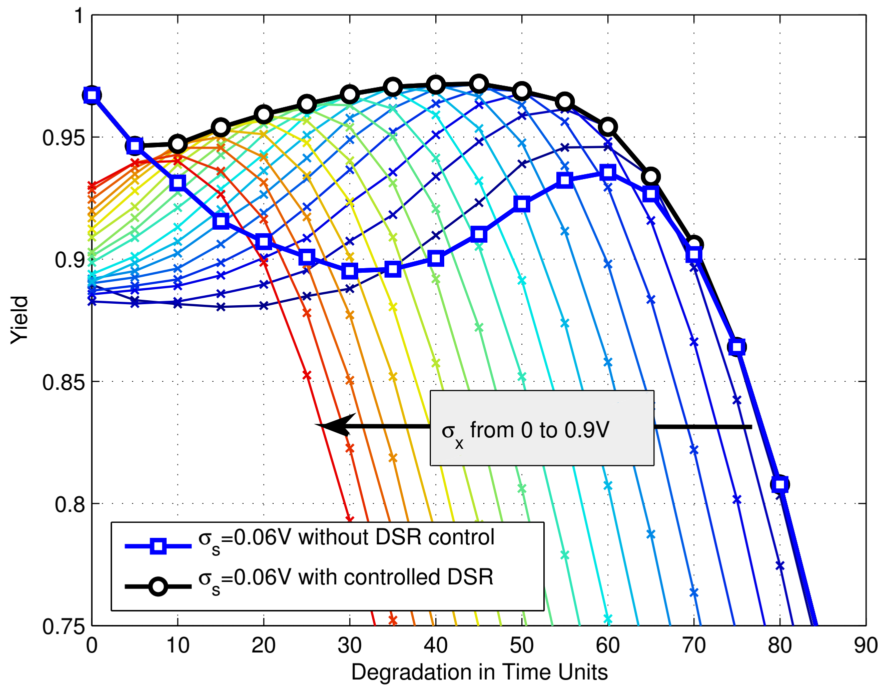


FIGURE 5.19: Yield against degradation of 20-input AD-AVGs with different levels of noise added to the inputs. Thick blue line correspond to the AD-AVG cell yield with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$. Thin colored lines correspond to the impact of adding noise to the inputs of this cell with different magnitudes from $\sigma_x = 0V$, $\sigma_x = 0.9V$. The thick black line corresponds to the curve followed by the yield when the proper input noise magnitude at each degradation in time is applied in order to maximize the reliability.

If we apply the proper noise magnitude we can move along the involute of the thin colored curves obtaining a yield even higher than that provided by the resonance peak. Since finding the exact relation between optimum input noise magnitude σ_x and degradation level is impractical, if not impossible, we propose a numerical approach. In the same example simulated before, a 20-input AD-AVG with a noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$, we find numerically this relationship and the result is presented in Figure 5.20. The curve indicates that a nearly linear decreasing relation between both magnitudes exists. It is, therefore, possible to achieve a good approximation of the noise magnitude σ_x based on a numeric approach. DSR control unit only has to implement a linear decreasing function with the estimated input variability levels. We can describe the evolution of the injected noise magnitude σ_x for the DSR control as: (i) We start with an initial stage in which the circuit is young and healthy and no DSR yield enhancement is needed. During this period of time a null input noise magnitude is the best option $\sigma_x = 0V$. (ii) After six units of degradation in time the DSR control becomes useful with an input noise magnitude of $\sigma_x = 0.8V$. From this moment and during the rest of the circuit lifetime we have to apply an input noise magnitude that

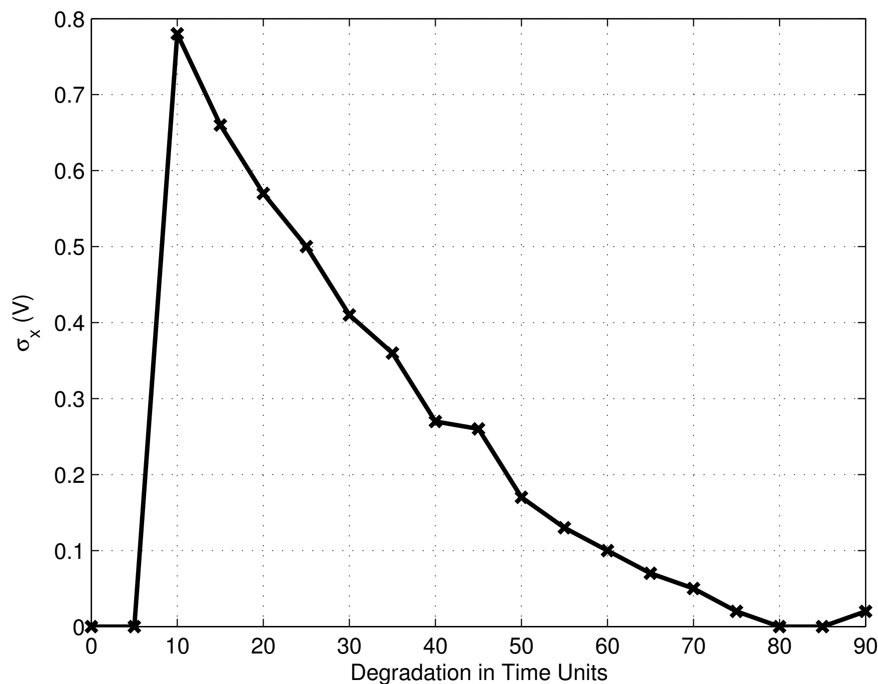


FIGURE 5.20: Magnitude of input noise σ_x against degradation in time that maximizes the reliability of a 20-input AD-AVG with noise in the Variability Monitor of magnitude $\sigma_s = 0.06V$ based on the DSR effect.

decreases approximately linearly against the degradation level. (iii) Finally, when the natural DSR peak degradation level is reached, the optimum input noise magnitude arrives to zero and afterwards the circuit reliability drops.

5.3.3.2 Controllable Noise Injectors Implementation

In order to virtually increase the instantaneous amount of degradation we may think of different strategies. In this thesis, we choose the option of adding independent noise injectors of controllable magnitude ϵ_i to the inputs, see Figure 5.17. Using this technique we can increase the input variability levels at any time with a particular magnitude. We can also gradually reduce the magnitude of added noise to zero as the circuit continues to experience degradation.

To implement the noise injectors we propose to make use of diodes designed to work through avalanche breakdown. We can artificially generate electrical noise for each input by controlling the avalanche breakdown phenomenon occurring in R different diodes in the AD-AVG structure, one for each input replica. This phenomenon has been widely studied [72] and it offers us an easy solution for the DSR control.

By adding a controllable noise injector to each input we modify the input variability levels as

$$\sigma_{i'}^2 = \sigma_i^2 + \sigma_x^2. \tag{5.26}$$

We assume a normal distribution for the noise added ϵ_i to the input signals with a null mean and standard deviation σ_x. With the proposed implementation the noise magnitudes generated by the injectors are independent as required.

5.4 Averaging Cell Linear Threshold Gate (AC-LTG)

In this section, we present a fault-tolerant nanoscale architecture based on the implementation of logic systems with averaging cells linear threshold gates (AC-LTG). We compare the tolerance to manufacturing and environment deviation of our approach and the well known NAND multiplexing technique. We show that the AC-LTG is a valuable alternative in specific nanoscale conditions. The analyzed structure combines the concepts of Averaging Cell (AC) [55] and Linear Threshold Gate (LTG) [73]. Both architectures share the same structure of weighting average and threshold operation, as a result the extension of the LTG architecture with the AC technique takes full advantage of its natural capabilities and does not require any additional effort in the design stages. The AC-LTG combination was first proposed as a way to implement fault-tolerant computing architectures by Ferran Martorell et al. in 2006 [69]. We revisit this strategy and perform an extensive reliability analysis taking into consideration the sources of variability that affect both the behavior of devices and the manufacturing process.

5.4.1 AC-LTG Mathematical Model

Every logical operation implemented by an AC-LTG can be expressed by the following equations [see (5.27) and (5.28)].

$$\hat{y} = sign\left(\sum_{i=1}^{M} w_i x_i - T\right) \tag{5.27}$$

$$x_i = \sum_{j=1}^{N} c_j x_i^j \qquad \forall i \in (1 \dots M), \tag{5.28}$$

M being the number of input variables (x_i) composing the Boolean function, N the number of available replicas of each input and $T(= t \cdot V)$ the threshold decision level. Weights $W = (w_1, w_2, \dots, w_M)$ define the specific synthesized Boolean function whereas

c implement the average of each input bundle ($c = 1/N$). Figure 5.21 shows the generic scheme of a M-input AC-LTG with redundancy level N.

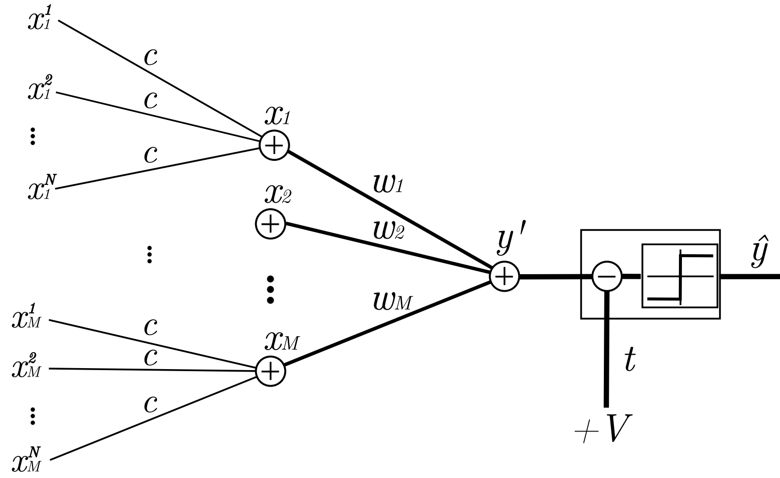


FIGURE 5.21: Schematic of M-input AC-LTG with redundancy level N.

Each input x_i, $\forall i \in (1 \dots M)$, comes from the average of N error-prone physical replicas x_i^j, $\forall j \in (1 \dots N)$, which represent the ideal binary variable $x_i^* \in (0, V)$. Logic values 0 and 1 are physically represented by voltage levels 0 and V respectively. We use the fault model introduced in Section 2.2 to model the variability in the input voltage levels x_i^j due to internal noise, device parameter deviations, and physical defects:

$$x_i^j = x_i^* + \eta_j, \tag{5.29}$$

where $\eta_j \sim N(0, \sigma_{\eta_j})$. Therefore, by the properties of the normal distribution and considering homogeneous variability, each input x_i follows a normal statistical distribution with mean $\mu_{x_i} = x_i^*$ (the ideal input value) and standard deviation $\sigma_{x_i} = \sigma_{\eta_j}/\sqrt{N}$.

After the initial redundancy layer, each input x_i is weighted by a parameter w_i that controls its impact on the final average y'. Many different Boolean functions can be synthesized adjusting the configuration of weights. And finally, a threshold operation is performed with an equivalent decision level T. As a result, the restored binary variable $\hat{y}$ is obtained.

5.4.2 Implementable 2-input AC-LTG Boolean Functions

LTG are capable of implementing unate functions of any number of inputs. All non-linearly separable Boolean functions have to be constructed by combinations of more than one LTG. In this subsection, we present all possible non-redundant 2-input unate functions and the corresponding configuration of weights and threshold needed to implement them in the AC-LTG structure introduced above. The information has been

obtained from the work by Goparaju et al. [74]. Functions are identified with notation f_i, where i represents the decimal equivalent of binary identification of functional output values of mimterms of f. It is also assumed a manufacturing process that restricts the values of weights and threshold to integers in the range [-25,25].

TABLE 5.2: 2-input LTG functions and the associated configuration of weights and threshold.

Function f_i	Configuration $\{w_1, w_2, t\}$
f_1	{-25,-25,-9}
$f_2 \equiv f_4$	{-25,16,6}
f_7	{-18,-18,-25}
f_8	{18,18,25}
$f_{11} \equiv f_{13}$	{-16,25,-6}
f_{14}	{25,25,9}

5.4.3 N-redundant 2-input NAND AC-LTG

In this subsection we analyze the particular case of 2-input NAND AC-LTG, see Figure 5.22. We want to study the effects of variability in all possible sources of variability associated to the AC-LTG architecture. Although we focus on the NAND gate, the results are easily transportable to other versions of 2-input AC-LTG, such as AND, OR and NOR gates. We apply the configuration presented in the Table 5.2 for function f_7, which corresponds to the 2-input NAND function:

$$w_1^* = w_2^* = -18 \qquad t^* = -25 \qquad (T^* = -25V) \tag{5.30}$$

This configuration is optimum for tolerating deviations in the assignment of weights due to non-ideal manufacturing process. In the literature, the capability to tolerate deviations in the weights is measured with a parameter d_{max} that corresponds to the maximum deviation that can affect the weights and threshold before changing the synthesized Boolean function. In the particular case of NAND function the robustness parameter is $d_{max} = 3.5$.

$$\Delta w_1 = |w_1 - w_1^*|, \qquad \Delta w_2 = |w_2 - w_2^*|, \qquad \Delta t = |t - t^*| \tag{5.31}$$

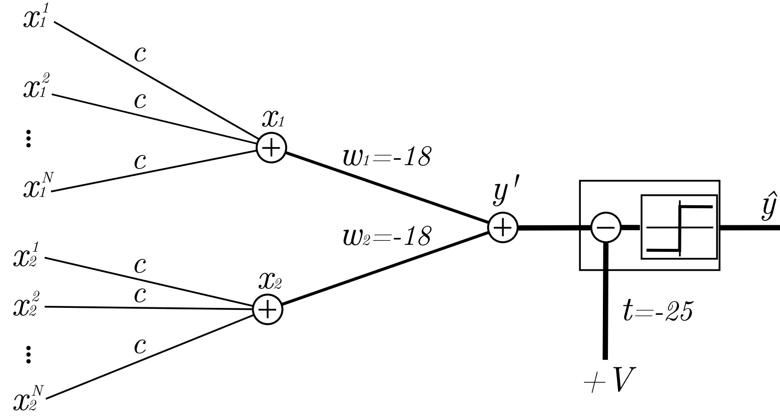


FIGURE 5.22: Schematic of N-redundant 2-input NAND AC-LTG with optimized precision-limited configuration of weights and threshold robust to manufacturing inaccuracies ($W = \{-18, -18\}$, $T = -25V$).

$$max(\Delta w_1, \Delta w_2, \Delta t) \leq d_{max} \tag{5.32}$$

We make here a parametric analysis of a generic 2-input AC-LTG. Using variables for the values of weights and threshold $W = \{w_1, w_2\}$ and $T = t \cdot V$ we can compute the probability of producing an erroneous output given the required Boolean function to synthesize (NAND in this particular case) and the four possible input combinations $\{00, 01, 10, 11\}$. Averaging the four results we obtain a formula to express the gate output error probability P_e:

$$\begin{aligned} P_e &= \frac{1}{8}\left(1 - erf\left(\frac{(w_1 + w_2 - t)V}{\sqrt{2\left(w_1^2 \frac{\sigma_{\eta_1}^2}{N} + w_2^2 \frac{\sigma_{\eta_2}^2}{N} + t^2\sigma_t^2\right)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{(w_1 - t)V}{\sqrt{2\left(w_1^2 \frac{\sigma_{\eta_1}^2}{N} + w_2^2 \frac{\sigma_{\eta_2}^2}{N} + t^2\sigma_t^2\right)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{(w_2 - t)V}{\sqrt{2\left(w_1^2 \frac{\sigma_{\eta_1}^2}{N} + w_2^2 \frac{\sigma_{\eta_2}^2}{N} + t^2\sigma_t^2\right)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{-tV}{\sqrt{2\left(w_1^2 \frac{\sigma_{\eta_1}^2}{N} + w_2^2 \frac{\sigma_{\eta_2}^2}{N} + t^2\sigma_t^2\right)}}\right)\right). \end{aligned} \tag{5.33}$$

We observe that weights $W = \{w_1, w_2\}$, and threshold parameter t do not depend on any scaling factor. They can be normalized and no change is produced in the resulting error probability P_e. Applying this observation we define here the normalized weights $W^n = \{w_1^n, w_2^n\} \equiv \{w_1/t, w_2/t\}$ in order to reduce the number of variables involved in the Equation (5.33). Before introducing this change we present another definition in order to reorganize and normalize the remaining intervening terms, converting them into unitless parameters. It refers to the input standard deviations σ_{x_1}, σ_{x_2}, the drift in the threshold level σ_t, the voltage V and the redundancy N:

$$\begin{aligned} \sigma_1^n &\equiv \sigma_{\eta 1}/\sigma_t\sqrt{N} \\ \sigma_2^n &\equiv \sigma_{\eta 2}/\sigma_t\sqrt{N} \\ V^n &\equiv V/\sigma_t \end{aligned}$$

Applying all the above definitions into equation (5.33) it yields:

$$\begin{aligned} P_e &= \frac{1}{8}\left(1 - erf\left(\frac{(w_1^n + w_2^n - 1)V^n}{\sqrt{2((w_1^n\sigma_1^n)^2 + (w_2^n\sigma_2^n)^2 + 1)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{(w_1^n - 1)V^n}{\sqrt{2((w_1^n\sigma_1^n)^2 + (w_2^n\sigma_2^n)^2 + 1)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{(w_2^n - 1)V^n}{\sqrt{2((w_1^n\sigma_1^n)^2 + (w_2^n\sigma_2^n)^2 + 1)}}\right)\right) \\ &+ \frac{1}{8}\left(1 + erf\left(\frac{-V^n}{\sqrt{2((w_1^n\sigma_1^n)^2 + (w_2^n\sigma_2^n)^2 + 1)}}\right)\right). \end{aligned} \tag{5.34}$$

Equation (5.34) describes P_e and allows us to analyze the effects of variability due to input drift sources ($\sigma_{\eta 1}$, $\sigma_{\eta 2}$), drift in the threshold decision level (σ_t) and deviation in weights and threshold assignment (Δw_1, Δw_2, Δt). It also provides us a way to study how much the impact of this effects can be diminished by increasing the level of redundancy N or the source voltage V.

- **Ideal Case (Null variability):**

  Figure 5.23 depicts the P_e color map in the plane of normalized weights for the ideal case. Cool colors represent low probabilities while hot high probabilities. In this ideal case, with null variability, there is a region in the W^n-plane where

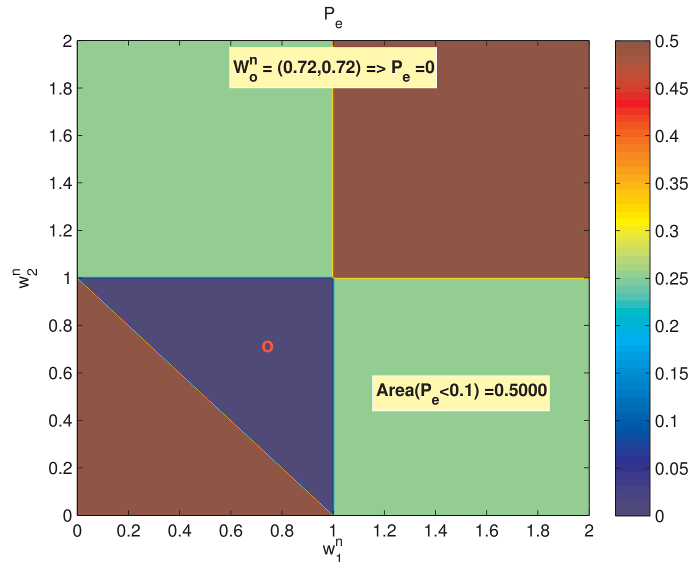


FIGURE 5.23: P_e color map of N-redundant 2-input NAND AC-LTG with null variability ($\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$)

the error probability P_e is null (the blue triangle). With small red marker the position of the previously referred configuration is shown (See Equation (5.30), $W = (-18, -18)$, $t = -25 \Rightarrow W^n = (0.72, 0.72)$). This configuration is robust to small displacements of the configuration (red marker) in the W^n-plane since it has been designed to withstand deviations in weights and threshold assignment.

- **Non-Ideal Homogeneous Case:**

  When variability is introduced in the input variables σ_{η_1}, σ_{η_2} as well as in the threshold value σ_t, the shape of the P_e color map deforms as shown in Figure 5.24. It is observed in this case that the optimal configuration previously presented (small red marker) is not necessarily the configuration with lower probability of error P_e (small white cross) as shown in Figure 5.24. Optimizing weights considering only deviation in the weights assignment produces optimal solutions slightly different from the ones obtained when considering also other variability sources.

- **Non-Ideal and Non-Homogeneous Case:**

  We consider here different levels of variability depending on the input in order to model the effect of degradation, which affects randomly different parts of the system. Figure 5.25 shows an example of non-homogeneous variability.

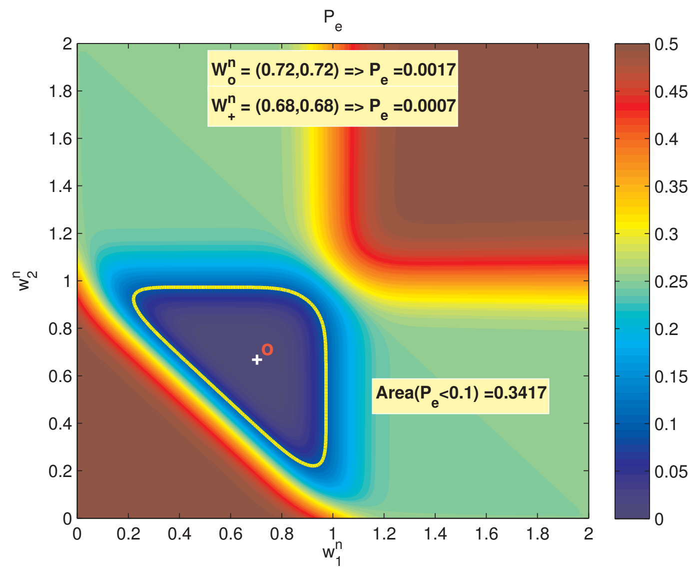


FIGURE 5.24: P_e color map of 10-redundant 2-input NAND AC-LTG with variability parameters ($\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0.1V$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$) and in small white cross the configuration with lower probability of error P_e. Yellow contour outlines the area with error probability $P_e < 0.1$

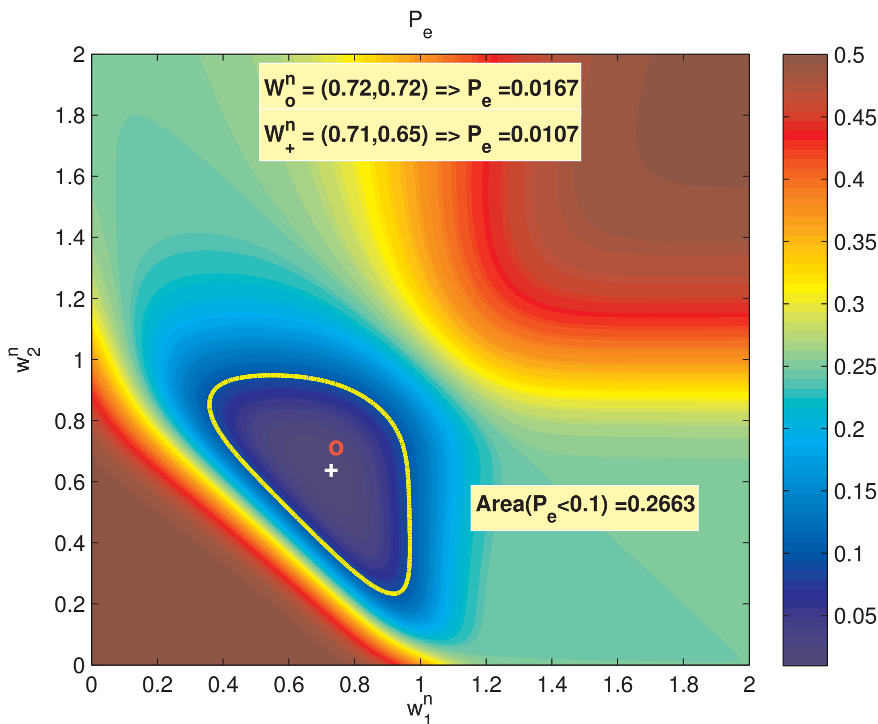


FIGURE 5.25: P_e color map of 10-redundant 2-input NAND AC-LTG with variability parameters ($\sigma_{\eta_1} = \sigma_t = 0.1V$, $\sigma_{\eta_2} = 0.5V$). In small red marker the optimized precision-limited configuration $W = (-18, -18)$, $t = -25$ ($W^n = (0.72, 0.72)$) and in small white cross the configuration with lower output probability of error P_e. Yellow contour outlines the area with error probability $P_e < 0.1$

5.4.3.1 Simulation results

Figure 5.26 depicts the impact of redundancy level N in a typical case with variability parameters $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0.1V$ and different levels of deviation in the assignment of weights and threshold $d \geq max(\Delta w_1, \Delta w_2, \Delta t)$. Results correspond to the worst case obtained within the respective range of deviation. In this simulation, deviation parameter d is configured to sweep levels from 0% to 40% of the maximum admissible deviation (remember $d_{max} = 3.5$).

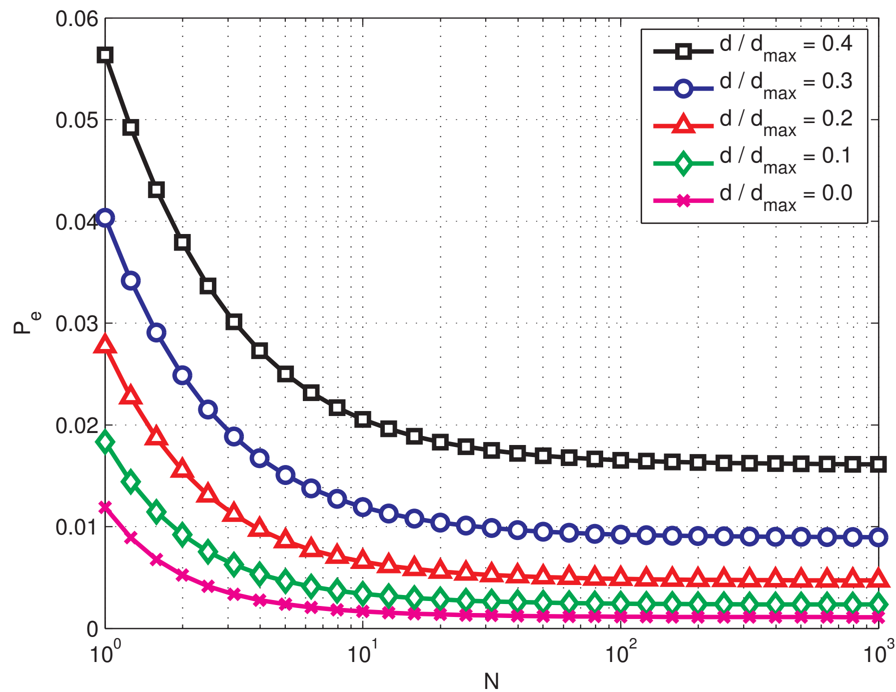


FIGURE 5.26: Output probability of error P_e versus redundancy level N at different levels of deviation in the assignment of weights and threshold $d \geq max(\Delta w_1, \Delta w_2, \Delta t)$. Case with variability parameters $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t = 0.1V$

Figure 5.27 shows the effect produced on the probability of error P_e versus the redundancy N at different levels of homogeneous variability $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t$. A great improvement from $N = 1$ to 10 is observed. Input levels of variability are extracted from the previsions of ITRS 2012 [16].

5.4.3.2 Measurement of Tolerance against Manufacturing and Environment Variability

We have seen that 2-input NAND LTG can tolerate deviations in the weights and threshold up to 3.5 according to the restriction of integer weights within $[-25, 25]$. This value is directly related with the radium of the maximum circumference that can be inscribed

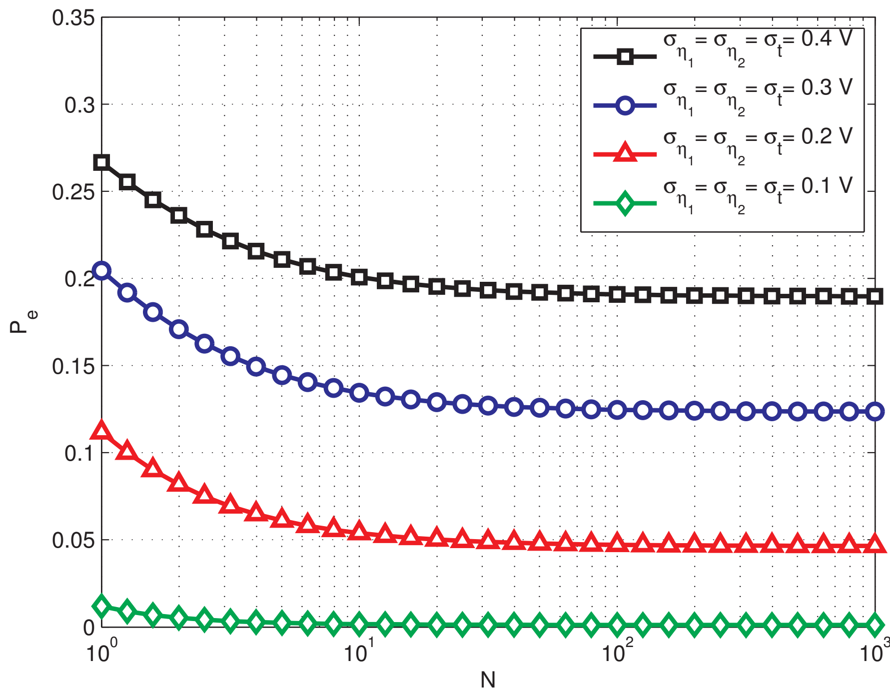


FIGURE 5.27: Output probability of error P_e versus redundancy level N at different levels of homogeneous variability $\sigma_{\eta_1} = \sigma_{\eta_2} = \sigma_t$ and null deviation in weights and threshold assignment $d = 0$

in the blue region ($P_e = 0$) of the P_e-colormap of Figure 5.23. However, this tolerance parameter only takes into account deviations in the weights and threshold. In this subsection we propose a metric to estimate the tolerance against the combined effect of variability in the inputs and source voltage and the deviations in the weights and threshold of the LTG configuration.

Our approach consists on determining a satisfiability boundary for weights and threshold configuration in the P_e-plane. This boundary depends on the levels of variability in the input signals and in the source voltage. It encloses all the LTG configurations that achieve output error probability lower than a given probability P_{max}. We calculate the area of the regions of satisfying configurations at different levels of variability and correct the fault-tolerance parameter d_{max} to see how it decreases in environments with high variability. Table 5.3 shows these values of parameter d_{max} with homogeneous variability in the inputs and source voltage ranging from $\sigma = 0.0V$ to $\sigma = 0.4V$ with requirement $P_{max} = 0.1$. We can observe that f_7 and f_8 functions (NAND and AND) have a weak tolerance against variability while f_1 and f_{14} (NOR and OR) have the best behavior in spite of the variability in the inputs.

Figure 5.28 reproduces the P_e-plane for the functions f_1, f_2 and f_7. The results for the rest of implementable functions are symmetric to these three cases.

Table 5.3: Tolerance Parameter d_{max} of 2-input LTG functions with different levels of variability.

Function	Fault-tolerance d_{max}				
f_i	$\sigma = 0.0V$	$\sigma = 0.1V$	$\sigma = 0.2V$	$\sigma = 0.3V$	$\sigma = 0.4V$
f_1	7.99	7.95	7.78	7.59	7.37
$f_2 \equiv f_4$	4.99	4.80	4.29	3.54	2.29
f_7	3.49	3.08	0.00	0.00	0.00
f_8	3.49	3.08	0.00	0.00	0.00
$f_{11} \equiv f_{13}$	4.99	4.80	4.29	3.54	2.29
f_{14}	7.99	7.95	7.78	7.59	7.37

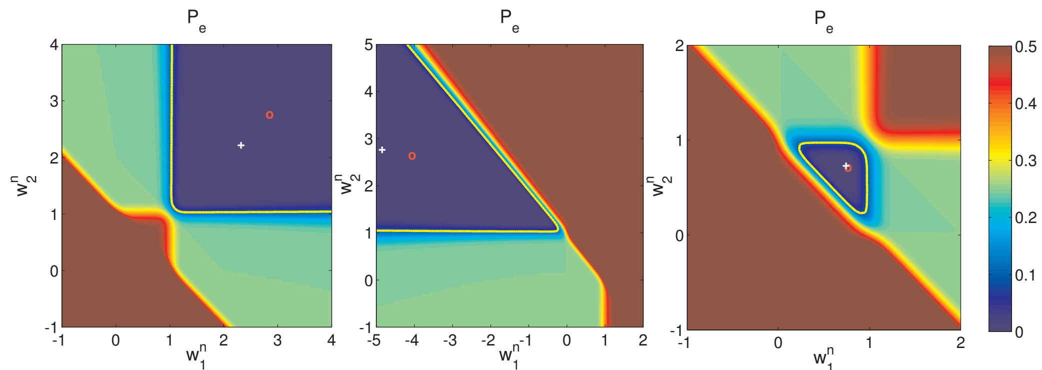


Figure 5.28: P_e color map of 10-redundant AC-LTG unate functions with variability parameters ($\sigma_{\eta_1} = \sigma_t = 0.1V$, $\sigma_{\eta_2} = 0.5V$). In small red marker the optimized precision-limited configuration and in small white cross the configuration with lower output probability of error P_e.

5.4.4 Reliability Comparison of NAND AC-LTG versus NAND multiplexing

In this subsection, we compare the presented AC-LTG architecture with the well known NAND multiplexing technique. Both designs tolerate variability and faulty behavior of its compounding devices by means of redundancy.

5.4.4.1 NAND multiplexing topology

There are many studies on the NAND multiplexing architecture. Therefore, a set of useful formulations of its performance have been provided. They allow us to analyze it and have a clear view of its capabilities. We take advantage of these contributions to guide our discussion [44].

Figure 5.29 presents the general topology of a NAND multiplexing unit. It consists of a first stage performing the NAND operation and a second stage to restore the output value. Restoration is implemented with two NAND operations in series and intercalated randomizing blocks (U). This restoring unit can be replicated as many times as necessary to improve reliability level although this implies an additional increase in overhead.

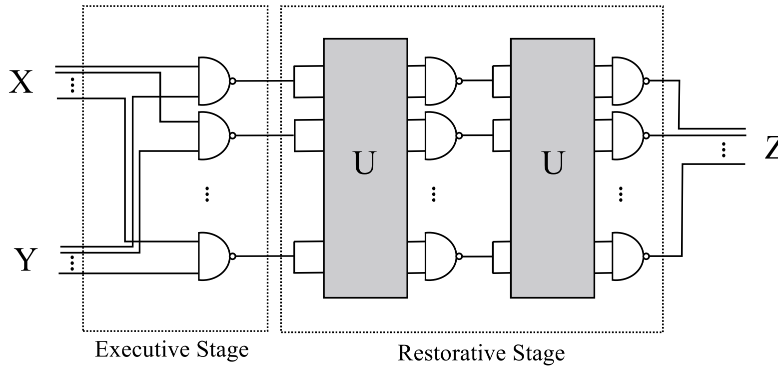


FIGURE 5.29: Schematic of a NAND multiplexing architecture

Parameters usually used to formulate the NAND multiplexing characteristics are:

- N, the number of redundant inputs and outputs,
- δ, the ratio between the faulty input lines and the total number of lines N,
- ϵ, the probability of a device producing a faulty output
- and n, the number of restoring stages added at the output (the NAND multiplexing scheme of Figure 5.29 has $n = 1$).

5.4.4.2 Parameter Equivalences

In order to compare both techniques, NAND AC-LTG and NAND multiplexing, some equivalences between characteristic parameters must be established. Some of them are direct, like the level of redundancy N, but others, like ϵ and δ, require a more detailed analysis.

- **Redundancy N and number of restoring stages n:**

  We add N threshold operations in parallel at the output of AC-LTG architecture so as to have the same topology in both fault-tolerant techniques: N redundant inputs and N redundant outputs. Figure 5.30 presents the general scheme of the NAND AC-LTG considered in this section. It is remarkable that the effective number of devices of both architectures differ linearly from each other with the

number of restoring stages n (See Table 5.4). NAND multiplexing requires more devices than NAND AC-LTG for the same redundancy N.

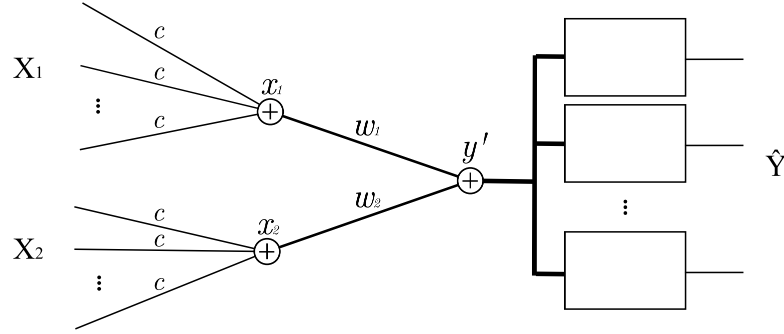


FIGURE 5.30: Schematic of a NAND AC-LTG with N redundant inputs and outputs

TABLE 5.4: Number of devices versus level of redundancy for NAND multiplexing and AC-LTG

Architecture	Redundancy level	Number of devices
NAND multiplexing	(N, n)	$N \cdot (1 + 2n)$
AC-LTG	N	N

- **Ratio of faulty input lines δ:**

  δ parameter from NAND multiplexing is directly related with the input level of variability in AC-LTG, which is expressed by σ_{η_1} and σ_{η_2}. Parameter δ expresses the probability of an input line being faulty. It corresponds to the probability of a given level of input variability σ_η deviating the correct value more than $V/2$. This relation is expressed by equation (5.35).

$$\delta = \frac{1}{2}\left(1 - erf\left(\frac{V/2}{\sqrt{2\sigma_\eta^2}}\right)\right) \tag{5.35}$$

- **Ratio of faulty operations per device ϵ:**

  NAND multiplexing parameter ϵ is related with the remaining AC-LTG variability parameters: drift in the threshold level σ_t and deviation level in weights and threshold assignment d. All of them concern to device faulty behavior. Given a certain deviation d, drift in threshold level σ_t must complete the level of variability expressed by ϵ. Equation (5.36) manifests this relation.

$$\epsilon = \frac{1}{2}\left(1 - erf\left(\frac{(d_{max} - d)V}{\sqrt{2t^2\sigma_t^2}}\right)\right) \tag{5.36}$$

5.4.4.3 Comparison Results

The above relations along with the presented formulation of NAND multiplexing architecture allow us to make a reliability comparison.

Given a logic input $X_1 = 1$, $X_2 = 1$, the worst case, having 10% of errors in the input bundle ($\delta = 0.1$), the probability of having less than 10% of errors in the output bundle is computed for both strategies. Results for redundancy level $N = 10$ and $n = 7$ are depicted in Figure 5.31. We have picked these parameters because they imply a good performance in both strategies and do not require too high redundancy.

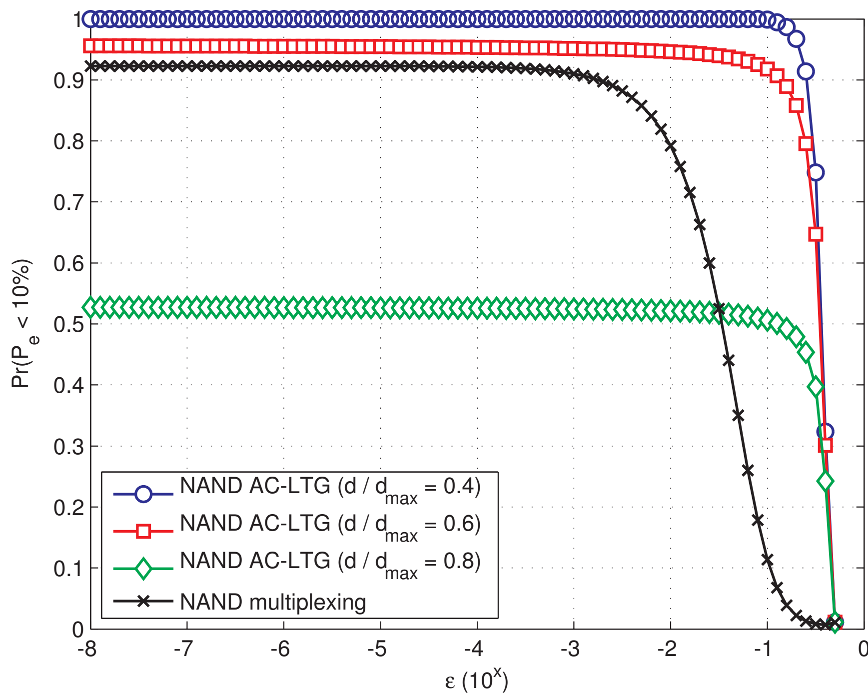


Figure 5.31: Reliability comparison between NAND multiplexing with redundancy $N = 10$ and number of restoring stages $n = 7$ and NAND AC-LTG with redundancy $N = 10$ and different levels of inaccuracy in weights and threshold assignment d

We can see in Figure 5.31 that AC-LTG have better performance against device failure rate than NAND multiplexing provided that deviation levels in weights and threshold assignment d are lower than 60% of the maximum admissible d_{max}. Given a restriction for the probability of having less than 10% of errors in the output bundle, it is easy to see how NAND AC-LTG improves NAND multiplexing performance. For example, imposing $Pr(P_e < 10\%) > 90\%$ implies having $\epsilon < 10^{-2.75}$ for the NAND multiplexing technique while $\epsilon < 10^{-0.60}$ for the NAND AC-LTG with deviation level in weights and threshold assignment of $d = 40\%$ of d_{max}.

5.5 Conclusion

This Chapter introduces several redundant reliable structures based on the averaging cell (AVG) principle. In first place, we introduce the unbalanced averaging cell (U-AVG) targeting the reliability improvement of nanoscale circuits and systems suffering from heterogeneous variability. The U-AVG basically consists of adjusting the configuration of weights in the averaging structure according to the input variability levels that can be measured prior to operation. A method for determining the optimal averaging weight values that maximize the reliability of the AVG output is derived. Monte Carlo simulations of U-AVGs operating in different heterogeneous variability scenarios show that our proposal substantially improves the output reliability at a lower cost than the classic balanced AVG; i.e., it requires 4x less redundancy for the same reliability requirement.

In second place, we extend the U-AVG by proposing a methodology to on-line learn the temporal variations on the input drift levels induced by external aggressions and aging, when the circuit is deployed in the field. This learning augmented scheme is called the adaptive averaging cell (AD-AVG). Further Monte Carlo simulations of different AVG, U-AVG, and AD-AVG instances in non-static heterogeneous environments are performed and provide meaningful information about the tolerance against degradation of the different AVG approaches. It is observed that the U-AVG rapidly loses performance with the circuits degradation and it is outperformed by the conventional AVG for redundancy factors higher than 15. Comparing the characteristic reliability of the AVG versus the AD-AVG against degradation, significant redundancy savings are obtained for the adaptive approach: from 9.5x to 4.1x redundancy reduction when imposing reliability requirements of 90% yield after increasing amounts of accumulated degradation. Apart from the significant advantages in terms of redundancy overhead savings and improved tolerance to degradation, the implications of using a non-ideal Variability Monitor for the learning process are also studied. Based on this analysis it is observed that monitor's noise has a greater negative effect on AD-AVGs with larger redundancy factors. Given a requirement of 90% yield with Variability Monitor with noise level $\sigma_s = 0.06\ V$, the lifespan of 2-input AD-AVG reduces 2 degradation in time units while that of 5-input AD-AVG reduces 7 degradation in time units. We also show a possible implementation of the AD-AVG structure based on resistive switching crossbars. The proposed topology is capable of reconfiguring the averaging weights at run-time and to measure the changes in the input variability levels. A simulation of the final AD-AVG implementation is provided. It is demonstrated that the AD-AVG proposal is potentially implementable with state of the art technology.

In third place, we present the Degradation Stochastic Resonance (DSR) effect in the context of Adaptive Averaging (AD-AVG) architectures. This important but counter-intuitive effect implies an enhancement in the system reliability against hardware degradation for specific noise conditions. For example, the yield of a 20-input AD-AVG, with a noise level of 0.06V in the Variability Monitor, decreases from 1 to 0.89 as the system degradation is increasing, then it grows up to 0.94 at the DSR peak, and finally decreases to zero when the system reaches its end-of-life. We analytically demonstrate this behavior in the particular case of 2-input AD-AVG with null variability in one of the inputs. We perform several Monte Carlo simulations to generalize the DSR effect implications to multiple input AD-AVGs. Exploring the main features of DSR we observe that this effect becomes more relevant in AD-AVGs with large number of inputs and that the DSR peak conditions enhance the system reliability over the one provided by equivalent higher redundancy systems, which despite of including more replicas are outside the DSR peak conditions. Moreover, in order to take full advantage of the DSR effect, we propose to add controllable noise injectors to the AD-AVG inputs to virtually increase the amount of hardware degradation and create the DSR conditions regardless of the degradation level. By this method we shift the characteristic yield to the DSR peak, regardless of the degradation level, and significantly enhance the system yield. Simulation results indicate that by applying the proper noise magnitude we can provide an optimum and nearly flat reliability level at any time before the DSR peak degradation level. Returning to the earlier example of a 20-input AD-AVG with a noise level of 0.06V in the Variability Monitor we obtain a guaranteed yield level of 0.94 during the system lifespan with a maximum yield of 0.97. This clearly demonstrates that by controlling the DSR phenomenon we can guarantee a minimum yield level for the entire life of the system. The particular magnitude of input noise that has to be applied in order to control the DSR effect is numerically calculated and we find that it approximately follows a linearly decreasing relation against the degradation level. Then we also propose a physical implementation of the controllable noise injectors based on the avalanche breakdown phenomenon occurring in diodes.

Finally, we propose the combination of Averaging Cells with Threshold Logic Gates AC-LTG as a way to perform reliable computing in spite of the inherent unreliability of the compounding devices. We demonstrate that this structure exhibits a good performance at moderate levels of redundancy ($N = 10$) against different sources of variability, such as drift in the input signals and deviation in the LTG parameters. The comparison between AC-LTG and NAND multiplexing technique determines that under moderate levels of manufacturing inaccuracies the improvement in tolerance to faulty device behavior is two orders of magnitude higher in AC-LTG with respect to NAND multiplexing.

Chapter 6

Time-aware Reliable Design

THE ASYNCHRONOUS nature of future nanoelectronic computing systems is addressed in this chapter from the perspective of the reliable design. Based on current CMOS technology trends as well as beyond CMOS nanoelectronics, it is expected a significant increase in the variability of the compounding devices. In particular, future nanoelectronic circuits will probably have associated much higher statistical dispersion in the processing times than current CMOS technology.

Most of the fault tolerant architectures based on hardware redundancy improve the system reliability by replicating the basic computing element and combining the results with a majority criterion. In this Chapter, we extend this conventional approach by introducing the time dimension. Our proposal, called partially-Asynchronous R-fold Modular Redundancy (pA-RMR), takes into account the asynchronous nature of future nanoelectronic computing systems by detecting the arrival of each input signal using tokens. The voter behavior is modified in such a way that it sets the output result after a determined number of token arrivals. By doing this, we basically add a second degree of freedom to the RMR structure, which not only has a configurable size (R replicas), but also allows modifying the number of tokens it waits before giving an output. As a consequence of this seemingly simple change, we are able to exploit new possibilities of this redundant structure such as trading system reliability for performance during operation. The remainder of this Chapter is organized as follows. In Section 6.1, we introduce the pA-RMR structure and analyze the reliability and time performance associated to each possible voting policy. In Section 6.2, we show the reliability versus performance trade-off of the pA-RMR structure and discuss some examples. We also show the impact of failures in token transmissions. Finally, in Section 6.3, we briefly summarize our results and make a few concluding remarks.

6.1 The partially-Asynchronous R-fold Modular Redundancy (pA-RMR)

One of the most fundamental and effective fault tolerant mechanisms introduced in von Neumann's work [2] in 1952 was the well-known R-fold Modular Redundancy (RMR), where R replicas ($R = 3, 5, 7\dots$) of a computing subsystem present their outputs to a voter block that generates a reliable output based on a majority criterion (also known as the MAJ gate). The RMR, in its $R = 3$ version (TMR) has been widely used in the design of systems in environments where reliability is considered a key issue (nuclear plant control, space applications, bank organization, etc) [75, 76]. These scenarios consist of low or moderate failure probability subsystems but high reliability requirements at system level. In modern and future processing systems designed from emergent new device generations, reliability is becoming a major challenge due to the poor quality of the basic elements [9, 14]. As a consequence, the research of fault-tolerant techniques based on hardware redundancy is gaining a renewed interest today. In fact, in the literature we can find many recent articles analyzing RMR-based structures. Some of them compare different redundancy factors ($R \geq 3$), others combine multiple RMR cells (MAJ gates) in cascaded or multiplexed architectures [39, 41, 77, 78]. However, despite the wide spectrum of research works based on the RMR technique, almost all of them analyze the reliability issue from a static point of view. In other words, given a set of error-prone data replicas generated by R independent subsystem replicas, the majority of studies seek to determine the reliability characteristics of the RMR structure without any temporal consideration. In real applications though, the way data moves or propagates through electronic circuits has associated significant time-varying processes. The variability in the time it takes the electronic devices to process the information and the propagation of signals through buses of different length or load, among others, induce significant spread in the arrival time of the input signals to the RMR voter. This temporal variability is even more significant in future nanoelectronic systems in which we can expect significant statistical dispersion in the processing time [79].

In order to address this issue, in this Chapter we extend the conventional RMR design considering the time dimension. We propose the utilization of tokens that signal the arrival of input bits to the voter in order to be able to take decisions even before the arrival of all the input bits from the replicas and remove this time-consuming limitation. We name this new RMR concept the pA-RMR, which stands for partially-Asynchronous R-fold Modular Redundancy. Due to the different number of inputs presented at a given time to the pA-RMR voter block, each one with its associated error probability, the problem considered here is an extension of the conventional RMR analysis. The voter may take decisions depending on the reliability level required by the system. It

may present outputs before the arrival of the complete set of replicas, allowing a faster processing of the reliable output data always inside a strategy of decisions taken by the voter. In the analysis of the pA-RMR concept, the conventional case of synchronous data arrival time is a particular case. Extending the RMR analysis from the set of failures probabilities for each replica to the inclusion of arrival time models for each replica's data, the analysis presented in this paper evaluates original strategies applicable to redundant systems. This model could also be used to mimic the behavior of biological neural structures, where the computation is performed with characteristics similar to those introduced in our pA-RMR structure. As supported by the literature, this is a key research topic today [80, 81]. The presented pA-RMR architecture also fits perfectly into de design environment of cross-layer resilient systems [82], since it provides, by means of the voter configurability, an easy way to implement most of the characteristic resilience tasks such as detection, diagnosis, reconfiguration and adaption.

6.1.1 pA-RMR Model

The partially-asynchronous pA-RMR structure, as the RMR, consists of a set of error-prone replicated computing units and a decision gate (voter) that establishes the global output based on a majority criterion applied to the input signals, see Figure 6.1. However, the pA-RMR also includes a token associated to each input replica to indicate the arrival of a data bit. We assume an independent error probability $\epsilon \leq 1/2$ for each replica. When the processing cycle starts all the subcircuit replicas start at once the computation of a new data. This process has associated delay mismatch between replicas since they are subject to variations. As soon as a replica finishes its calculation the corresponding token is activated to signal the voter. In Section 6.2.1 we assume an fault-free token transmission, but in Section 6.2.2 we generalize our analysis to exceptional situations in which tokens may be lost or corrupted during operation. We model the statistics of each token arrival time as an independent exponential distribution with rate parameter λ $[s^{-1}]$. By using the exponential distribution, we target future nanoelectronic architectures with high levels process variability [79]. We call $F_T(t)$ the Cumulative Distribution Function (CDF) of tokens' arrival time:

$$F_T(t) = Pr(T_{\text{token i}} < t) = 1 - e^{-\lambda t} \quad i = 1, \ldots, R, \tag{6.1}$$

$T_{\text{token i}}$ being the arrival time of token i.

The voter is the responsible for setting the most probable value at the system output. Its underlying operation principle consists on providing at any time the "best possible" output from the available inputs, though without activating the associated output token

until it reaches the final decision. In other words, as the input replicas arrive the pA-RMR keeps processing and updating the output signal according to the majority criterion. For instance, if there is only one input available, the pA-RMR outputs this input value and when other inputs arrive it updates the output accordingly. During the process it is possible to reach equilibrium states in which the inputs are tied and the output is not updated. These equilibrium states are immediately broken by the following arrival that tilts the result in its own direction. This operation contrasts with the RMR structure that only starts computing the output when all the inputs have arrived. It is easy to deduce that the pA-RMR can operate faster than RMR in any case. Indeed, while the RMR has to compute the majority voting of the R replicas, the pA-RMR only has to update the previous result with the last information bit. Apart form that, the pA-RMR has an additional feature: it may follow different voting policies as it can set the output before receiving all the input replicas. Different voting policies provide different trade-offs between reliability and performance since they base their final decision on different amounts of information. The pA-RMR output includes a token that is activated as soon as the configured voting policy is completed. By using this arrangement it is possible to construct more complex computing architectures with multiple pA-RMR cells combined in cascade or multiplexed structures.

6.1.2 pA-RMR Reliability

During operation, the pA-RMR voter keeps updating the output according to the received inputs in order to minimize the output error probability. After the arrival of a specific number of tokens, determined by the voting policy, it activates the output token. According to this strategy, each voting policy provides a different compromise

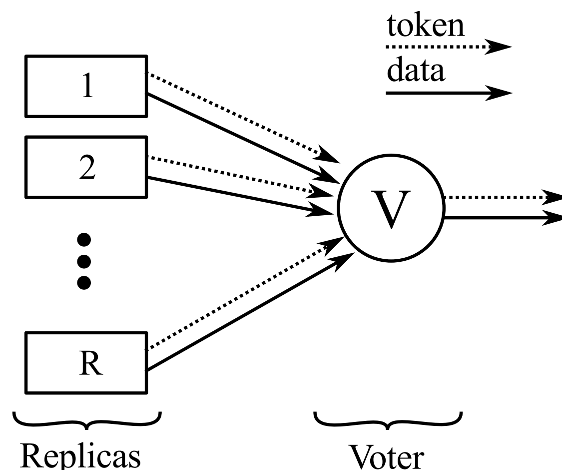


Figure 6.1: partially-Asynchronous R-fold Modular Redundancy (pA-RMR) architecture.

between reliability and performance. In this subsection, we analyze the reliability level guaranteed by each voting policy.

We identify each pA-RMR policy with an index k_{pol}. This parameter corresponds to the number of tokens validating the same input information (logic '1' or '0') that the voter waits before activating the output token. For example, if the chosen policy is $k_{pol} = 2$, the voter waits for two tokens validating a logic '1' or a logic '0'. This policy may imply awaiting the arrival of the first two tokens if they are in consensus, or awaiting the first three tokens if the first two are in disagreement. Notice that in this case the third signal directly determines the pA-RMR output.

The number of different voting policies in a pA-RMR structure is limited. R being the number of replicas, policies with index k_{pol} higher than $(R+1)/2$ are not recommended because high inconsistency situations could lead the system to fail to respond. We choose $k_{pol} = (R+1)/2$ as the maximum voting policy because it is still free of failure response and coincides with the RMR structure.

The reliability provided by each voting policy is directly related to the number of tokens received.

- If we look at the first voting policy ($k_{pol} = 1$), the least reliable one, it waits only for the first token and assigns the received input directly to the output. In this case the output error probability is $Pe = \epsilon$.
- The second voting policy ($k_{pol} = 2$) waits for two coherent inputs. In this case, we have to consider two possibilities: (1) the first two inputs are coherent but incorrect, and (2) the first two inputs are in disagreement and the third one is incorrect. This results in an error probability of $Pe = \epsilon^2 + 2\epsilon^2(1-\epsilon)$, which is lower than the one associated with the first voting policy.
- The third voting policy ($k_{pol} = 3$) waits for three coherent inputs. This may imply waiting only for the first three inputs, if they are in consensus, or waiting for four or even five replicas in the worst case (when the first four are tied). Adding the probability of error associated to each possible scenario it yields a probability of $P_e = \epsilon^3 + 3\epsilon^3(1-\epsilon) + 6\epsilon^3(1-\epsilon)^2$.

By following this pattern we find the general term for the output error probability associated with each voting policy.

$$Pe(k_{pol}) = \sum_{i=0}^{k_{pol}-1} \binom{k_{pol}-1+i}{i} \epsilon^{k_{pol}} (1-\epsilon)^i \tag{6.2}$$

Notice that the reliability levels described by (6.2) are valid for each voting policy independently of the number of replicas R. The only restriction for these results to be applicable in a particular pA-RMR is that the number of available replicas must be larger or equal to $2k_{pol} - 1$.

Figure 6.2 depicts the reliability level $(1 - Pe)$ of a pA-RMR with 7 replicas against the error probability of single replicas (ϵ) when applying different voting policies. In this case, we can choose between 4 different voting policies. It is easy to see that policies requiring higher level of input coherency provide higher reliability. For example, if we require a reliability level of 99% we can only admit an error probability per replica of $\epsilon = 0.01$ when using the first policy. However, if we use the fourth policy ($k_{pol} = 4$) we can admit an error probability per replica of $\epsilon = 0.14$.

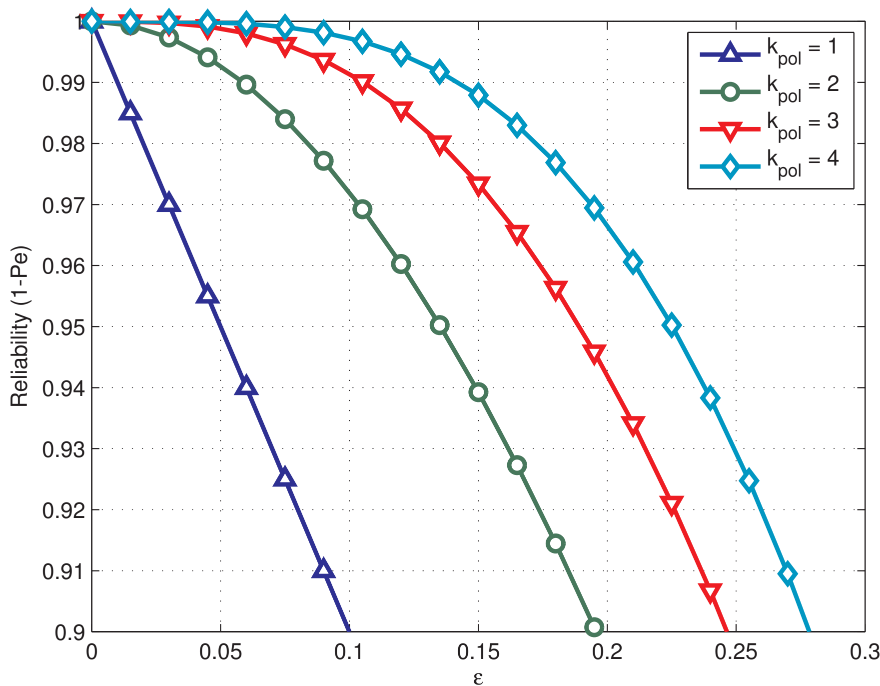


FIGURE 6.2: Reliability provided by the 4 different voting policies of a pA-RMR with 7 replicas against the input error probability ϵ.

One interesting way to analyze the reliability characteristics of a complex system consists in establishing a target reliability level $(1 - Pe)$ and obtain its tolerance level against imperfections in the compounding devices. Applying this method to the pA-RMR architecture we perform an experiment to find its capability of tolerating input errors without degrading the reliability level under a specific bound. In the experiment, we set a minimum admissible reliability or, equivalently, a maximum admissible output error probability (Pe_{max}) and find how much error probability we can tolerate in the replicas (ϵ_{max}). Figure 6.3 depicts the maximum admissible error probability per replica

ϵ_{max} of a pA-RMR against the voting policy and the targeted output error probability Pe_{max}. We can find the minimum voting policy we need to tolerate a certain input error

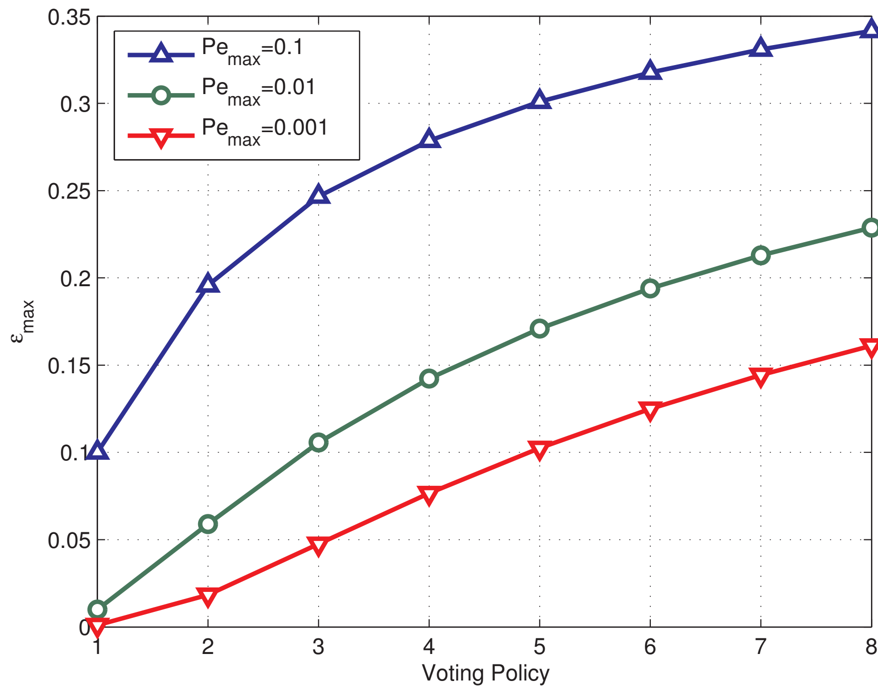


FIGURE 6.3: Maximum admissible error probability per replica (ϵ_{max}) against voting policies. Different lines correspond to different targets of output error probability Pe_{max}.

probability. For example: if our reliability target is an output error probability lower than 0.01 and we use the voting policy $k_{pol} = 2$ then we can admit a maximum error probability per replica of $\epsilon_{max} = 0.06$. If we switch to the voting policy $k_{pol} = 3$, we can admit almost twice as much error probability as the previous policy with the same reliability target.

6.1.3 pA-RMR Performance

Increasing the pA-RMR reliability by switching to higher-level voting policies is associated with a performance cost. Voting policies with higher level imply waiting longer for the required number of tokens. In this subsection, we analyze the timing conditions associated with each voting policy and the number of replicas. In this case, unlike the levels of reliability, there is an influence of the number of replicas on the timing conditions for each voting policy.

6.1.3.1 Tokens' Arrival Time

In order to calculate the timing characteristics of each voting policy, we first need to analyze the tokens' arrival time taking into account the arrival order. In the following, the Cumulative Distribution Function (CDF) associated with each token of the arrival sequence is calculated.

Given a pA-RMR with R replicas, the voter will receive R tokens during the processing cycle. We call $T_{i^{\text{st}}\text{token}}$ the arrival time of the i^{st} token in the sequence of arrivals. The CDF of the first token ($T_{1^{\text{st}}\text{token}}$) can be deduced using the arrival time model described by (6.1) and applying the property of independent random variables. The probability of having received the first token at time t after the cycle starts is:

$$Pr\left(T_{1^{\text{st}}\text{token}} < t\right) = 1 - \prod_{i=1}^{R} Pr\left(T_{\text{token i}} > t\right) \tag{6.3}$$

$$= 1 - \left(1 - F_T(t)\right)^R = 1 - e^{-\lambda R t}. \tag{6.4}$$

In order to calculate the probability of having received the i^{st} token at time t we perform a a summation over all the situations for which the ith token has already arrived:

$$\begin{aligned}
&Pr\left(T_{i^{\text{th}}\text{token}} < t\right) = \\
&Pr\left(i \text{ tokens arrived before } t \text{ and the rest after } t\right) + \\
&Pr\left(i+1 \text{ before } t \text{ and the rest after } t\right) + \\
&\dots \\
&Pr\left(R-1 \text{ before } t \text{ and the remaining one after } t\right) + \\
&Pr\left(\text{all tokens arrived before } t\right)
\end{aligned}$$

Developing these probabilities using the combinatorial numbers we obtain the following expression for the probability of having received the ith token at time t:

$$Pr\left(T_{i^{\text{th}}\text{token}} < t\right) = \sum_{n=i}^{R} \binom{R}{n} \left(1 - e^{-\lambda t}\right)^n e^{-\lambda t (R-n)} \tag{6.5}$$

Figure 6.4 shows the arrival time CDF of each token in a pA-RMR of 7 replicas taking into account the arrival order. In the plot we can see that the first token arrives more

than 95% of the times before $t = 0.5/\lambda$ and the last one lasts more than $t = 3/\lambda$ 30% of the times.

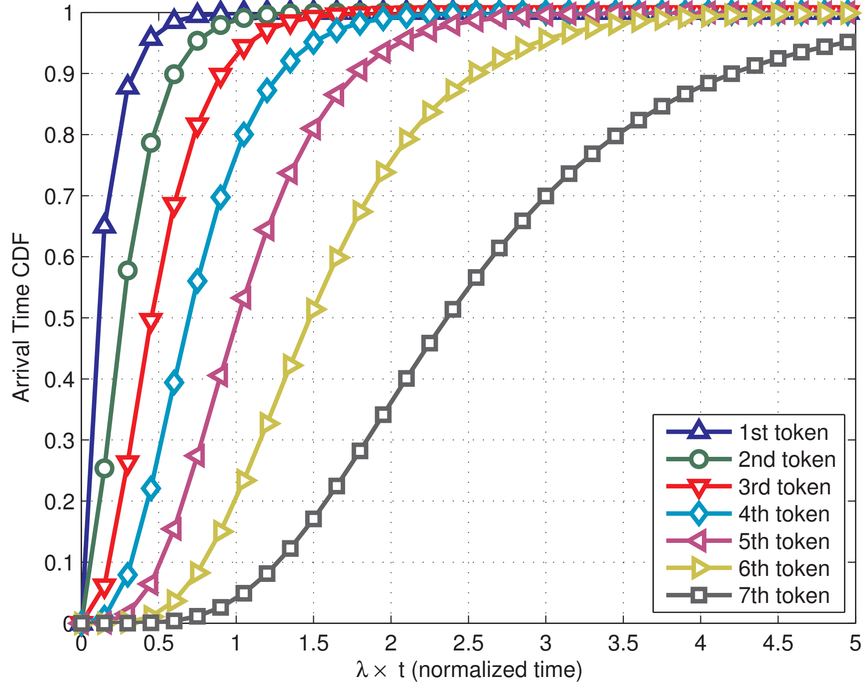


FIGURE 6.4: Arrival time CDF of tokens taking into account the arrival order of a pA-RMR with 7 replicas.

6.1.3.2 Policies' Response Time

The timing characteristics of each voting policy depend on the token's arrival time. This relationship is influenced by the replica error probability ϵ except for the first policy case, whose timing characteristics corresponds directly to the first token. Despite this, in general higher replica error probability means less coherency between inputs, and as a consequence, this implies waiting for more arrivals before completing the level of coherency required by the voting policy. For example, the second policy's timing corresponds to the second token's arrival time only in case the first two tokens are coherent and it corresponds to the third token's time when there is a disagreement between the first two tokens. In the previous example, we can easily see that both agreement and disagreement situations have associated two possibilities: In the case of agreement, the tokens may be correct or incorrect. In the case of disagreement, the last one (the one that fulfills the required coherency and determines the system output) may be correct or incorrect. When computing the relationship between token and policy's timing characteristics we have to account for both possibilities. In the following, we present the relationship between policy's and token's timing characteristics:

$$Pr\left(T_{i^{\text{th}}\text{policy}} < t\right) = \sum_{j=i}^{2i-1} A(i,j,\epsilon) Pr\left(T_{j^{\text{th}}\text{token}} < t\right), \tag{6.6}$$

where the weighting coefficient $A(i,j,\epsilon)$ is:

$$A(i,j,\epsilon) = \binom{j-1}{j-i} \left(\epsilon^i (1-\epsilon)^{j-i} + (1-\epsilon)^i \epsilon^{j-i}\right) \tag{6.7}$$

Note that the timing characteristics of the ith policy only depends on the arrival time of the ith and subsequent tokens.

Figure 6.5 depicts the response time CDF of each possible voting policy of a pA-RMR with 7 replicas and two different input error probabilities. It can be observed that the first policy has a time response independent from the input error probability while the rest experience a delay with the increase of input error probability.

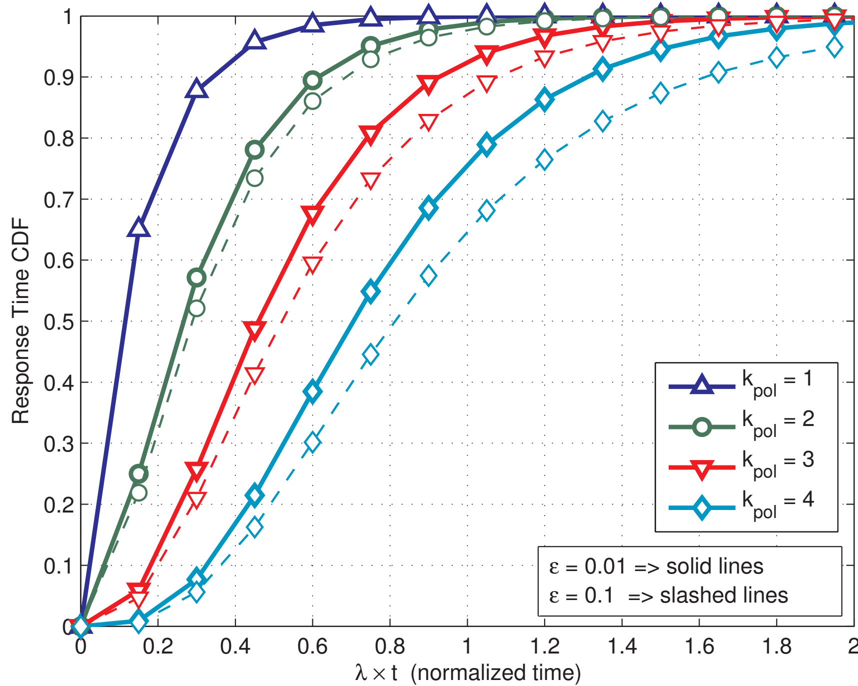


FIGURE 6.5: Response Time CDF of the different policies of a pA-RMR with 7 replicas. Solid lines correspond to input error probability $\epsilon = 0.01$ and slashed lines to $\epsilon = 0.1$.

6.2 Reliability vs Performance Trade-off

Once the reliability and performance aspects of the pA-RMR architecture have been analyzed, we are able to combine in this section both characteristics and analyze the

trade-off achieved by the different possible voting policies. The main advantage regarding the pA-RMR structure consists on the capability of improving performance or reliability by changing the voting policy. The implications of using a different policy to obtain the required performance or reliability target are detailed in the following analysis.

6.2.1 Fault-free Tokens in pA-RMR Architectures

Figure 6.6 shows the characteristic reliability $(1-Pe)$ versus performance of different size pA-RMR assuming fault-free tokens. In this analysis, we apply different voting policies and establish the error probability per input unit as $\epsilon = 0.2$. Each line corresponds to a specific redundancy level ranging from $R = 1$ to 17 and all the possible voting policies are indicated in the figure with markers. As stated before, the maximum voting policy for each redundancy factor R is $k_{pol} = (R+1)/2$. The performance metric is based on the normalized time (T_m), which we define as the time at which the pA-RMR response has arrived in 99% of the cases. We obtain the performance parameter by computing the inverse of the normalized response time $(1/T_m)$. Obviously, this definition is not unique and the resulting measures vary with the chosen percentage. Yet, it is a useful metric for comparing the voting alternatives since analytically the response time follows a probability distribution with infinite tail. Note that the pA-RMR with $R = 1$ has only one point because it has associated only one voting policy. The trade-off characteristic of a pA-RMR with $R = 3$ has two possible voting policies. When we increase the number of replicas we have more options to play with this trade-off, sacrificing reliability for performance. Note also that all the voting policies with the same parameter k_{pol} provide the same reliability level (see all the points are grouped in horizontal levels). The figure also presents connected with a thick slashed line the characteristic reliability points (stars in the figure) of conventional RMRs with number of replicas raging between 1 and 17. We can observe in the figure that the conventional RMR is a particular case of the pA-RMR when it uses the highest possible voting policy.

If we calculate the speedup of each pA-RMR configuration with respect to the classic RMR technique, we obtain the following conclusion: There is a 3x speedup in the structure when we choose the first voting policy instead of the second one in the pA-RMR with three replicas. In general, a pA-RMR is capable of working R times faster than the classic RMR if we choose the policy $k_{pol} = 1$. Given a pA-RMR structure with a redundancy level R the different available voting policies provide us the capability of improving reliability or performance according to our interests. We observe that in general better performance specifications are achieved if we increase the number of

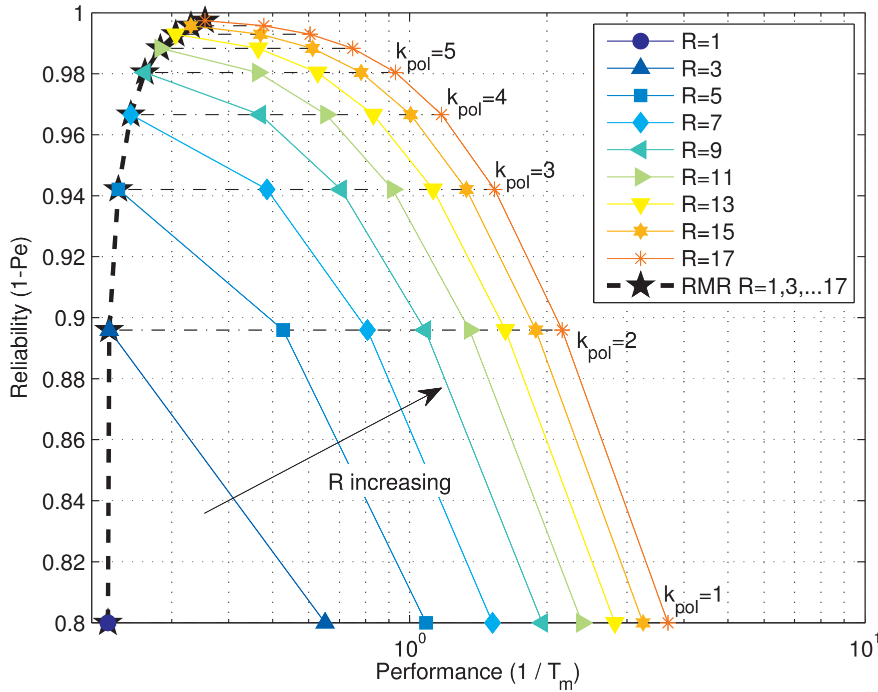


FIGURE 6.6: Reliability versus performance trade-off for different size pA-RMR from $R = 1$ to 17 with an input error probability of $\epsilon = 0.2$.

replicas (R), provided the same voting policy. And the only way to improve the pA-RMR reliability and performance at the same time is to increase the number of replicas and the voting policy.

6.2.2 Faulty Tokens in pA-RMR Architectures

In this subsection, we analyze the impact of non-ideal tokens in the performance and reliability of pA-RMR architectures. In previous analysis, we assumed that no token is lost or corrupted during operation, however, real circuits are subject to this issue. Tokens are utilized to signal the arrival of an input bit, and thus, they are not directly related with the output information. In fact, only tokens that signal incorrect input information may degrade the reliability of the pA-RMR while the rest only degrade the performance. Indeed, a token that fails to signal the arrival of an input bit, no matter the bit is correct or incorrect, it cannot affect the pA-RMR reliability as it is not taken into account, it only degrades a little bit the performance because it implies waiting for one extra token. Similarly, a token activated by error before the arrival of the associated bit, it increases the pA-RMR performance and only in this case, the pA-RMR reliability could also be degraded subject to the following two conditions: (i) the associated bit information was incorrect and (ii) the incorrect token arrived before the pA-RMR structure reached the

configured voting policy. Therefore, the output probability of failure will increase under very restricted conditions. In the rest of this section, we extend the pA-RMR tradeoff analysis between reliability and performance taking into account the impact of losing signals.

When a token is lost the associated bit information is not considered by the voter. It is actually equivalently to having one less replica available in the structure, that is, to reduce the redundancy factor by one unit. Thus, given a pA-RMR structure with R replicas, if a token is lost, the resulting structure is equivalent to a pA-RMR structure with $R-1$ replicas. However, since the classic RMR structure only uses odd redundancy factors in order to avoid ties, losing one replica in our generalized pA-RMR structure will give place to systems with even redundancy factors. But this is not generally a problem for our proposal. In fact, the pA-RMR structure can work with even number of replicas as long as the voting policy is not higher than $R/2$ since this particular case could lead to response failures; i.e. never achieving the required number of coherent input bits. Therefore, the result of Figure 6.6 can be extended for even redundancy factors. Figure 6.7 shows the reliability versus performance trade-off for pA-RMR structures with even and odd redundancy factors from $R = 1$ to 17. Filled markers are used to depict configurations with odd redundancy factors while empty markers correspond to even redundancy factors. In this figure, we can observe the impact of token losses, or

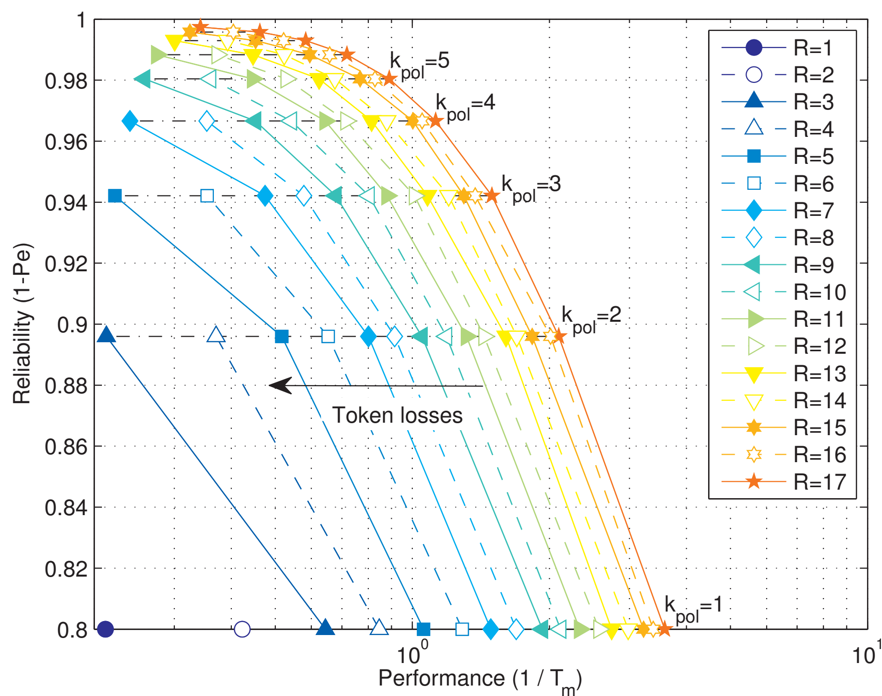


FIGURE 6.7: Reliability versus performance trade-off for different size pA-RMR from $R = 1$ to 17 with an input error probability of $\epsilon = 0.2$.

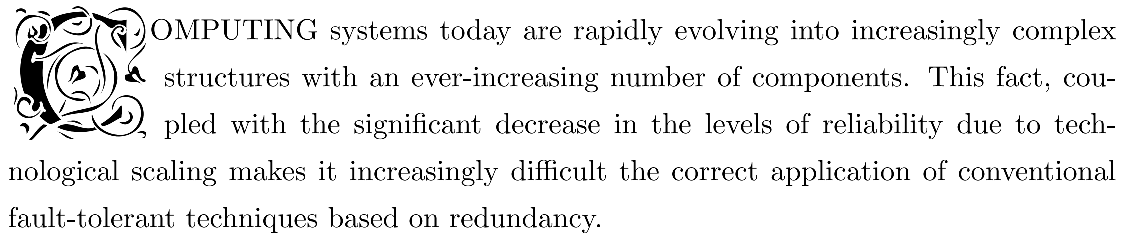
equivalently of reducing the redundancy factor, on the pA-RMR behavior. When a token is lost, if we can keep the same voting policy then the reliability level is maintained but the performance ($1/T_m$) diminishes. For example: if the chosen voting policy is $k_{pol} = 2$ and the redundancy factor is $R = 5$ then the reliability level is 0.896; in this case, losing one token is equivalent to having a pA-RMR with $R = 4$, the same reliability level but a 28.3% reduction in performance ($1/T_m$).

6.3 Conclusions

In this Chapter, we introduce the pA-RMR structure as an efficient partially-asynchronous fault-tolerant architecture based on hardware redundancy. We demonstrate the potential benefits of partially asynchronous architectures in terms of performance and reliability management. The analysis of the pA-RMR reliability allows to estimate its tolerance against the imperfections of individual replicas. For instance, consider a reliability requirement of an output error probability lower than $P_e < 0.01$. Then, if we use the voting policy $k_{pol} = 3$ we can tolerate a maximum error probability in the replicas of $\epsilon_{max} = 0.11$, but if we use the voting policy $k_{pol} = 6$ then we can tolerate up to $\epsilon_{max} = 0.19$. This enhancement of reliability together with the associated performance decrease becomes a choice for the system designer thanks to the use of pA-RMR. Regarding the influence on the pA-RMR performance, we also find the implications of using pA-RMRs with different sizes and voting policies. Combining the reliability and performance results we compare both dimensions together. There is a trade-off between reliability and performance in terms of the pA-RMR size and the applied voting policy. This structure provides a flexible way to adapt the system to specific requirements and achieve different combinations of reliability and performance. Because of this property we believe that the pA-RMR structure could be used in complex cross-layer resilient architectures. It provides an easy way of implementing most of the characteristic resilience tasks. Errors could be detected by checking the difference between the signaled result (from the configured policy) and the one achieved after the arrival of all the replicas. Every single pA-RMR cell could provide in this way additional information about the reliability of a particular part of the circuit and would help implementing the diagnosis task. Additionally, the capability of configuring each voter block to improve reliability or performance according to our interests could be used as the basic element to reconfigure and adapt the system.

Chapter 7

Multiple-layer Reliable Design



COMPUTING systems today are rapidly evolving into increasingly complex structures with an ever-increasing number of components. This fact, coupled with the significant decrease in the levels of reliability due to technological scaling makes it increasingly difficult the correct application of conventional fault-tolerant techniques based on redundancy.

In this Chapter, we present a comprehensive approach to the smart application of redundancy techniques in multiple-layer hierarchical systems. So far, many fault tolerant techniques based on redundancy have been proposed, implemented and tested by the scientific community. However, there is not much research on the possible distribution of redundancy effort through the system layers. System complexity grows with every new technology generation and so does the number of hierarchy layers. As a consequence, it is increasingly difficult to determine the optimum level or granularity at which to apply redundancy. On the other hand, it is also possible to consider distributing the hardware redundancy effort at different layers simultaneously. In the following sections we analyze this topic and provide valuable information for designers. The rest of the chapter is organized as follows: In Section 7.1, we present a model for cross-layer reliable architectures and show the possible application of static redundancy at multiple layers simultaneously. In Section 7.2, we introduce a fault model for the entire architecture. In Section 7.3, we develop the fault model using the Rent's Law in order to generate an analysis method which allow us to compare different distributions of redundancy in cross-layer architectures. In Section 7.4, we show the results of our analysis. We end this Chapter with some concluding remarks in Section 7.5.

7.1 Cross-Layer Reliable Architecture

In this section, we illustrate the hardware organization of current computing systems and present how static redundancy techniques can be applied at different layers simultaneously. Figure 7.1 depicts a schematic view of a cross-layer system and its multiple hierarchy layers. Each layer element embeds several elements of lower layers and therefore it incorporates a greater number of devices. In the figure we indicate a range for the approximate device count associated to the elements of each layer (D). In the rest of the paper we refer to particular layers using this device count parameter.

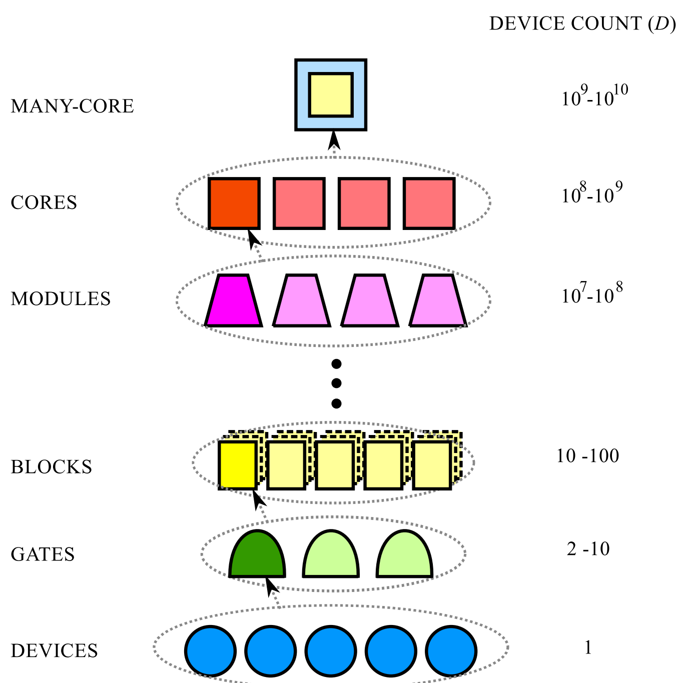


FIGURE 7.1: Schematic view of a cross-layer system and its multiple hierarchy layers. We characterize each layer with a parameter D that indicates the number of devices embedded in each element.

In the example of Figure 7.1 the first layer in the hierarchy ($D = 1$) corresponds to devices, which may be ultimate CMOS transistors or any other nanoscale device. The second depicted layer (with D between 2 and 10) corresponds to gates which are built out of a group of devices. The third layer corresponds to blocks composed of gates and so on. Each new layer integrates elements with a higher count of devices until we get to the entire computing system which encloses the total number of devices. The whole system is a composition of millions of devices organized in hierarchical groups of gates, blocks, modules and cores. Static redundancy in this context consists on the replication of all the elements in a particular layer D and then combining the resulting redundant information using a voting system. For example, Figure 7.1 represents a system in which the third depicted layer (with D between 10 and 100) is replicated 3 times. We can also

apply redundancy at multiple layers simultaneously but we will need a methodology to determine which is the best distribution of redundancy effort. In any case, the total redundancy effort corresponds to the product of all the redundancy factors applied to the architecture.

7.2 Fault Model

In this section we describe the fault model we use in the reliability analysis of cross-layer architectures. Our model takes into account the failure probability of both devices and interconnects. Indeed, instead of assuming perfect interconnects as usual in fault-tolerant analysis, we introduce a parameter δ that represents the probability of a interconnect being faulty. Under these conditions we can approximate the error probability of the whole system as the probability of at least one device or one interconnect being faulty:

$$P_e(D_s) = 1 - \underbrace{(1-\epsilon)^{D_s}}_{\text{devices}} \underbrace{(1-\delta)^{N_c(D_s)}}_{\text{interconnects}}. \tag{7.1}$$

Being D_s the total number of devices in the system and N_c the total number of interconnects. For convenience we express the number of interconnects and the system error probability as a function of the number of devices. The parameter ϵ stands for the failure probability of each device and δ stands for the failure probability of each interconnect.

Using this model we are implicitly assuming that each device have the same failure probability ϵ and it fails independently from the rest. This assumption is quite usual in fault-tolerant analysis. We also assume that all the interconnects have equivalent and independent failure probability δ. This assumption might seem unrealistic considering the wide variety of wires that a complex computing system have, such as shorter or longer wires needed to connect parts at different distances in the circuit and wider or narrower wires at different metal layers. However, we take into account that good designs use wider and more reliable wires for longer connections balancing this way the failure probability for all the interconnects. Moreover, some of the most important sources of imperfections in wires depend on critical points, corners or contacts that affect all interconnects in the same manner.

7.3 Analysis Method

The fault model described in the previous section can be used in any VLSI system but it is impractical in itself for our purpose. Our target is to analyze the impact

on reliability of applying different redundancy distributions in cross-layer systems. In order to take advantage of the proposed fault model we need to know at least how the number of interconnects and devices vary when we apply redundancy to any given hierarchy layer. To this end we use the Rent's Law [83, 84]. We use it as a mechanism to decompose the global error probability (7.1) at a given hierarchy layer and calculate the error probability of the elements at the considered layer. Afterwards, we can apply a redundancy technique and recompose again the global error probability. With the new reliability level we can make comparisons and extract useful information.

The Rent's Law, expressed in (7.3), establishes a relationship between the number of terminals (T) of a circuit area and the number of devices (D) it contains [83, 84]. In this formula, terminals of a circuit area (T) are defined as connections with the rest of the circuit.

$$T(D) = k \times D^r \tag{7.2}$$

It has been observed experimentally that the relationship between terminals and devices follows a power law with exponent r. This exponent depends on the design style and remains stable for different partitions of the circuit. Therefore, if we assume hierarchical consistency as previous work [85], we can use this property in cross-layer systems to estimate the amount of interconnects at any particular layer and decompose the global error probability to perform our analysis. Indeed, let us consider a cross-layer system of D_s devices, see Figure 7.2. If we want to analyze the impact of applying n-fold Modular Redundancy (n-MR) at layer D we can easily determine how many elements of layer D are needed to integrate the whole system (D_s/D). And by the Rent's Law we can determine the number of terminals associated to the complete system $T(D_s)$ and the ones associated to each element of the layer D: $T(D)$. With this data we can deduce the total number of interconnects at this layer, excluding the external connections. Notice that we represent the interconnects of layer D with slashed lines in Figure 7.2:

$$N(D) = \frac{k}{2}\left(D^{r-1}D_s - D_s^r\right) \tag{7.3}$$

Now, using the preceding information we can decompose the error probability of the whole system $P_{eo} \equiv P_e(D_s)$ and find the error probability of the elements of layer D:

$$\begin{aligned} P_{eo} = & 1 - (1 - P_e(D))^{\frac{D_s}{D}} (1-\delta)^{N(D)} \\ & \Rightarrow \; P_e(D) = 1 - \left(\frac{(1-P_{eo})}{(1-\delta)^{N(D)}}\right)^{\frac{D}{D_s}} \end{aligned} \tag{7.4}$$

Then, we can reconstruct the error probability of the whole system $P_e^*(D_s)$ applying the n-MR redundancy technique at layer D and then modifying the number of interconnects

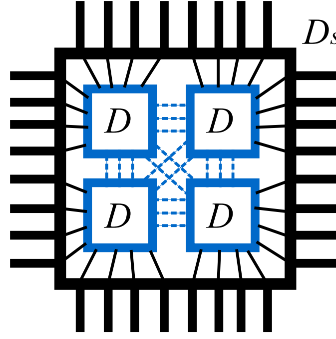


FIGURE 7.2: Schematic view of terminals and interconnects of a computing system with D_s devices. Interconnects of layer D are depicted with slashed lines.

at this level according to the extra interconnects required by the redundancy technique (we assume $n-1$ connections for each terminal to implement the n-MR technique):

$$P_e^*(D_s) = 1 - (1 - n\text{-MR}(P_e(D)))^{\frac{D_s}{D}} (1-\delta)^{N(D)+T(D)(n-1)} \tag{7.5}$$

With this analysis framework we can study different redundancy distributions and we can also try applying redundancy at multiple layers simultaneously.

7.4 Redundancy Distribution Simulation Results

In this section we present several analysis and simulation results of cross-layer systems reliability with different redundancy distributions applied across its layers. To perform the analysis we use the methodology described in the previous section. Our goal is to demonstrate the benefits of properly distributing the redundancy and propose a method to estimate the optimum redundancy configuration.

7.4.1 Redundancy at Different Layers

The first part of the analysis consists on studying the reliability improvement associated to the application of redundancy at only one particular hierarchy layer. For this reason, the simulation results in the next example show us the benefits of choosing properly the hierarchy layer but not of distributing redundancy. Figure 7.3 shows the global error probability (P_e^*) of a cross-layer system with $D_s = 10^{10}$ devices after applying redundancy n-MR at layer D with a redundancy factor n from 3 to 9. The Rent's Law parameters used in the simulation are $k = 2$ and $r = 0.6$. In the simulation we choose a global error probability before redundancy of $P_{eo} = 10^{-2}$ and the failure rate of interconnects $\delta = 10^{-15}$. In the figure we also depict the basic components of the

error probability in slashed lines, i.e.: P_e^{devices} error probability due to devices (ideal interconnects) and $P_e^{\text{interconnects}}$ error probability due to interconnects (ideal devices).

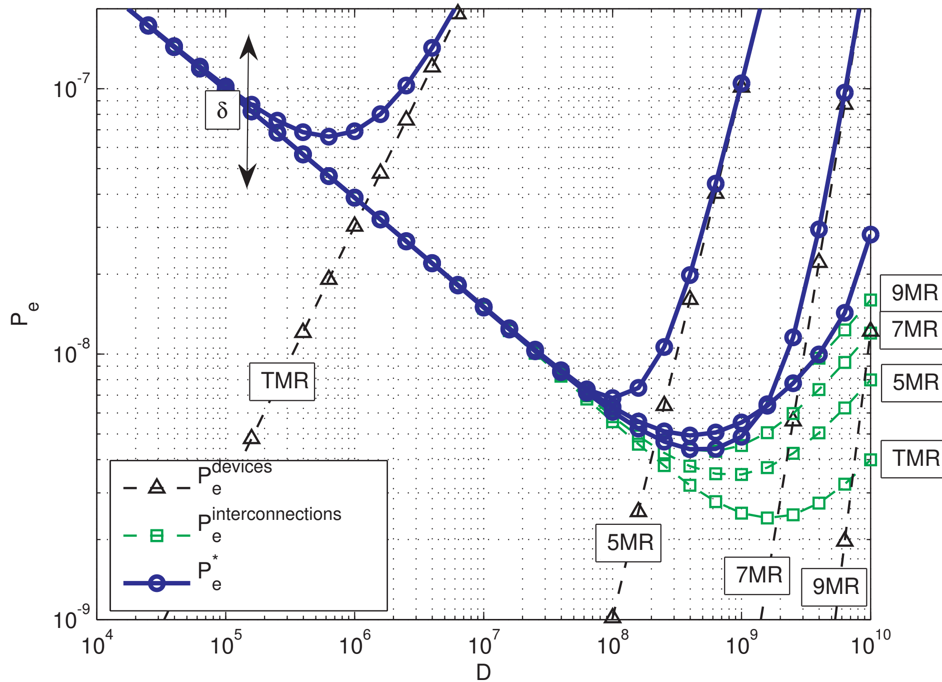


FIGURE 7.3: Global error probability P_e^* of a cross-layer system with $D_s = 10^{10}$ devices against redundancy layer D with n-MR technique. Slashed lines correspond to the global error probability components: black curves with triangular markers correspond to the error probability due to devices (ideal interconnects) and green curves with square markers correspond to interconnects (ideal devices). The levels of redundancy are $n = 3$, 5, 7 and 9. The rest of parameters are $k = 2$, $r = 0.6$, $P_{eo} = 10^{-2}$ and $\delta = 10^{-15}$.

In the figure we observe that assuming ideal interconnects (triangles, $\delta \to 0$) leads to the conclusion that the optimum layer is always the first one ($D = 1$) and the higher the redundancy factor the higher is the reliability improvement. In contrast, if the reliability is only conditioned by the interconnect failures (squares, $\delta \to 1/2$) then the optimum is close to the highest layers. Combining both components, in realistic situations, we get different optimum layers in which to apply each possible redundancy factor. If we apply redundancy at these optimum layers we can improve the system reliability several orders of magnitude. In particular, given the conditions of the previous simulation the best possible redundancy configuration at a single layer is 7-MR at layer $D = 4 \cdot 10^8$. With this configuration we reduce the global error probability seven orders of magnitude: from $P_{eo} = 10^{-2}$ to $P_e^* = 4.4 \cdot 10^{-9}$. However, the optimum layers vary with the interconnects failure probability δ. As δ grows, the optimum layers grow to higher levels because redundancy at lower layers requires adding more interconnects than at higher layers, and also the optimal redundancy factor decreases because the factors above are

progressively saturated by the global error probability component of the interconnects ($P_e^{\text{interconnects}} > P_e^{\text{devices}}$). Something similar happens but in the opposite direction with P_{eo} as it varies the devices component instead of the wires one. Therefore, in order to calculate the maximum reliability improvement in each particular case we have to take into account both δ and P_{eo} to deduce the optimum redundancy factor and the layer in which to apply it.

7.4.2 Redundancy at Multiple Layers Simultaneously

In this part we analyze the reliability of cross-layer systems when we apply redundancy at multiple layers simultaneously. Previous simulation demonstrates the relevance of choosing correctly the layer at which to apply redundancy. Now we want to see if we can get even better results by distributing redundancy at multiple layers (two layers).

In order to depict the results of global error probability against two different redundancy layers D_1 and D_2 we use colormap figures. Figure 7.4 shows the logarithm in base 10 of the global error probability ($\log_{10}(P_e^*)$) of a cross-layer system with $D_s = 10^{10}$ devices after applying redundancy TMR at two layers simultaneously D_1 and D_2. In the previous example we simulated redundancy factors from $n = 3$ to 9. However now, systems with redundancy distribution have at least redundancy factor $n = 9$ which comes from two simultaneous TMR techniques. The Rent's Law parameters, P_{eo} and δ are the same as previous simulation ($k = 2$, $r = 0.6$, $P_{eo} = 10^{-2}$ and $\delta = 10^{-15}$). In the figure cold colors mean low error probabilities and hot mean high error probabilities. First of all, notice that Figure 7.4 is symmetric with respect to $D_1 = D_2$. This is obvious since we are applying the same redundancy factor to both redundancy layers D_1 and D_2. In the following we focus only on the region $D_1 < D_2$ for clarity. If we look at the area with lower error probability, take for instance the region inside the yellow contour of $P_e^* = 1.1 \cdot P_{e\min}^*$, we realize that it is distributed along a strip located at redundancy layer D_1 close to $1.6 \cdot 10^9$ and the second redundancy layer ranging between 1 and 10^9. Indeed, if we are able to compute the characteristic layer $D_1 = 1.6 \cdot 10^9$ then it does not have much influence on the global error probability at which layer we apply the second TMR as long as it is below the layer $D_2 = 10^9$ (see slashed line in the figure). Given this result it seems very interesting to find a strategy to calculate the optimum value for D_1. Concentrating now on the optimum configuration, that is highlighted in the figure with a white dot, we observe that is located at $D_1 = 1.6 \cdot 10^9$ and $D_2 = 3 \cdot 10^5$ and it has associated the minimum global error probability $P_{e\min}^* = 2.4 \cdot 10^{-9}$. If we compare this optimum configuration with the one from the previous part for single redundancy layer we observe that we have improved the system reliability. In particular, for the simulated system we have reduced one half approximately the global error probability from $P_{e\min}^* = 4.4 \cdot 10^{-9}$

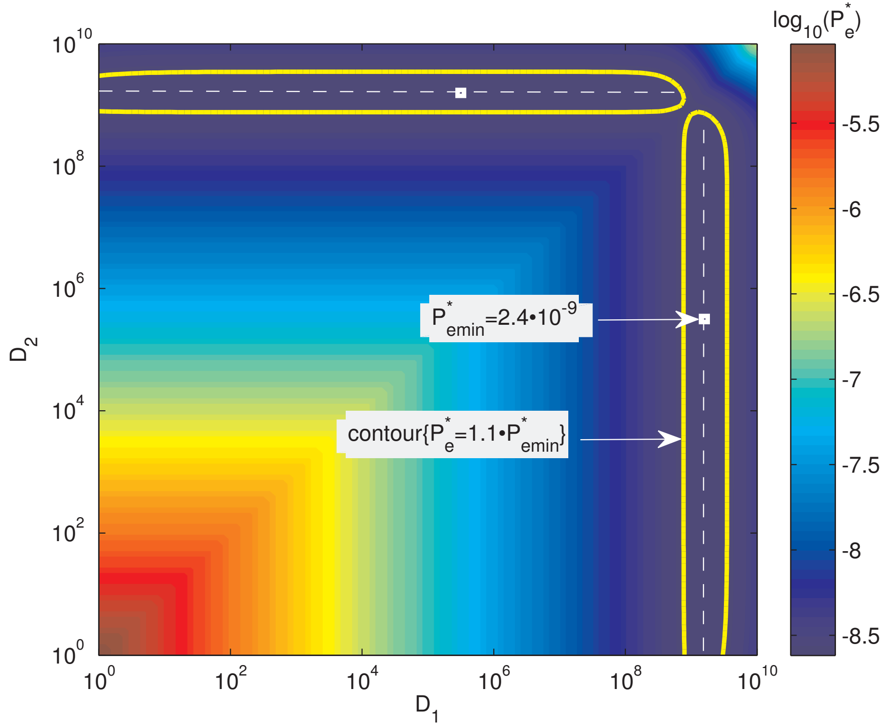


FIGURE 7.4: Global error probability P_e^* of a cross-layer system with $D_s = 10^{10}$ devices against redundancy layers D_1 and D_2. The optimum redundancy configurations are highlighted with white dot makers and it is labeled the associated minimum error probability (P_{emin}^*). The yellow contours enclose the regions with global error probabilities lower than $1.1 \cdot P_{emin}^*$. The levels of redundancy are $n = 3$ for both dimensions and the rest of parameters are $k = 2$, $r = 0.6$, $P_{eo} = 10^{-2}$ and $\delta = 10^{-15}$.

to $P_{emin}^* = 2.4 \cdot 10^{-9}$. This leads us to the conclusion that redundancy distribution through hierarchy layers in cross-layer systems is useful to improve reliability.

However, as in the previous part, we have to take into account that the location of the optimum redundancy configuration is influenced by the interconnects failure probability δ. Figure 7.5 depicts the impact of the interconnect failure probability δ in the optimum redundancy configuration of cross-layer systems with $D_s = 10^{10}$ devices and two layers with TMR redundancy (layers D_1 and D_2). The parameters of Rent's Law are the same as in previous simulations: $k = 2$ and $r = 0.6$. The global error probability without redundancy is $P_{eo} = 10^{-2}$. Observing the figure, we first notice how the contour of reliable redundancy configurations changes with δ. We refer to reliable configurations as the ones that produce a global error probability lower than $1.1 \cdot P_{emin}^*$. We see that as the interconnect failure probability δ increases the reliable contour reduces in area and moves towards configurations with higher D_1 and D_2. This shift towards higher layers can be understood by noting that redundancy at lower layers implies introducing more wires than at higher layers. For example, if we apply redundancy at layer 1 then we have to interconnect D_s times n-MR structures whereas at layer D_s we only have 1

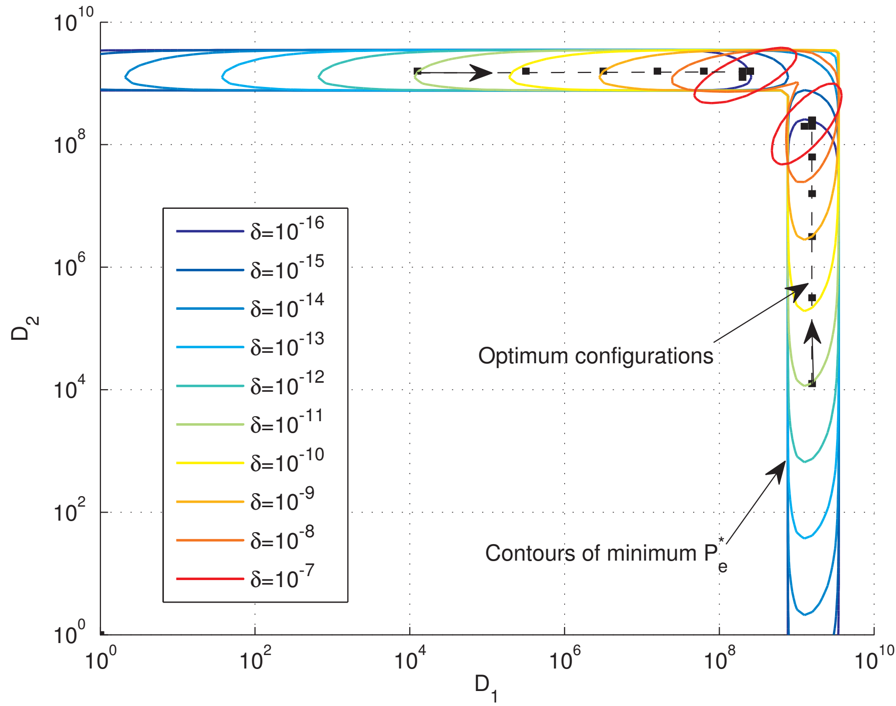


FIGURE 7.5: Impact of interconnect failure probability δ in the optimum redundancy distribution D_1 and D_2 in a cross-layer system with $D_s = 10^{10}$ devices. Both layers have the same redundancy $n = 3$ and the rest of parameters are $k = 2$, $r = 0.6$ and $P_{eo} = 10^{-2}$. Color lines show the contours of minimum error probability $P_e^* = 1.1 \cdot P_{e\min}^*$ for different values of δ. Black dots correspond to the corresponding optimum configurations.

n-MR structure to interconnect.

Another relevant aspect observed in the example focusing on the symmetric half $D_1 < D_2$ is that the optimum configurations have always the same redundancy layer D_1 regardless of δ. This fact highlights the relevance of finding a method to approximate this value. In order to do this, we analyze the equation (7.5) applied to the TMR technique in layer D_1. Since the condition $D_2 < D_1$ is met, Equation (7.5) can be used in its current form and we just have to be aware that the error probability of the replicated elements in layer D_1 depends on the redundancy layer D_2. However, since our interest is to analyze the contribution of δ in the global error probability we concentrate in the exponent of the term $(1 - \delta)$, which is:

$$D_1^{r-1} D_s - D_s^r + k D_1^r (n - 1). \tag{7.6}$$

If we optimize this exponent in terms of D_1 we find the layer D_1^{opt} that minimizes the exponent, and therefore it minimizes the global error probability since $(1 - \delta) < 1$:

$$D_1^{\text{opt}} = \frac{D_s(1 - r)}{2(n - 1)r} \tag{7.7}$$

If we apply this expression to the previous example it yields $D_1^{\text{opt}} = 10^{10}/6 \approx 1.67 \cdot 10^9$ which fits the simulation results. The influence of D_2 in the optimum value of D_1 is negligible as the simulation results demonstrate. We can use this approximation to decide the redundancy configuration of our cross-layer system.

7.5 Conclusions

In this Chapter, we introduce a new methodology to improve the efficiency of fault-tolerant techniques based on hardware redundancy. We focus our work in reliable cross-layer systems and analyze the benefits of distributing redundancy techniques across the hierarchy layers. For this we simulate computing systems under different reliability scenarios and redundancy configurations using a fault model that takes into account the failure probability of devices and interconnects using the Rent's Law. Our results show the major components of reliability in cross-layer systems and help discern the main trade-off. We note, for example, that the reliability of a cross-layer system with $D_s = 10^{10}$ devices and a failure probability of interconnects of $\delta = 10^{-15}$ can be optimized seven orders of magnitude from $P_e = 10^{-2}$ to $4.4 \cdot 10^{-9}$ by applying 7-MR at layer $D = 4 \cdot 10^8$ (single layer redundancy). Additionally, using two TMR layers simultaneously at layers $D_1 = 1.6 \cdot 10^9$ and $D_2 = 3 \cdot 10^5$ we can further improve the reliability to $P_e = 2.4 \cdot 10^{-9}$ (multiple layer redundancy). A correct distribution of redundancy through the hierarchy layers of complex systems can improve significantly the reliability benefits. Given the nature of the system we also observe the significant role of interconnects. We propose a method to estimate the optimum redundancy configuration based on the minimization of the error probability component associated to interconnects. Our conclusions may orient system designers to maximize the benefits of redundancy in cross-layer systems.

Chapter 8

Final Conclusions

This thesis has been motivated and inspired primarily by the work on reliability and redundancy developed by von Neumann in the 1950s. Grounded in the principles of hardware redundancy, our research line has delved into three main issues closely related to future computing technologies, namely: devices' heterogeneity, processing time asynchrony, and cross-layer redundancy design. The main objective has been to propose and analyze enhanced fault-tolerant methodologies based on redundancy to build robust computing architectures in the context of future nanotechnologies.

In the first place, we focused our attention on the problem of heterogeneity of compounding devices. Our initial proposal was to enhance the robustness of the averaging cell (AVG) architecture against heterogeneity by optimizing the values of the averaging weights. The idea was to counteract the uncertainty at the input signals by assigning weights inversely proportionally to the input variability levels. This approach gave place to the unbalanced averaging cell (U-AVG) structure. Significant improvements in terms of reliability and redundancy cost were demonstrated by means of Monte Carlo simulations.

Following up with the heterogeneity issue and our unbalanced averaging approach, we subsequently introduced the adaptive averaging cell (AD-AVG) in order to deal with the temporal variations in the input drift levels due to wear-out. The AD-AVG embeds a learning mechanism based on a variability monitor that allows for the on-line input weight adaptation such that the weight configuration properly reflects the aging status of the circuit at any time. Monte Carlo simulations proved that the AD-AVG considerably reduces the required amount of redundancy for a target reliability level when compared to conventional AVG and U-AVG techniques.

As a consequence of using a variability monitor, an important but counter-intuitive effect appeared in the context of the AD-AVG architecture. After running the Monte Carlo simulations for our adaptive AVG proposal and analyzing the results, we discovered that noise in the variability monitor causes a resonance effect in the yield versus degradation characteristic of the AD-AVG. Indeed, while in general the AD-AVG yield characteristic decreases with degradation, if we take into account a noisy variability monitor it increases under specific noise and degradation conditions up to a resonance peak (DSR peak) and then decreases as normal. In this thesis, we proposed to add controllable noise injectors to the AD-AVG inputs in order to virtually create the DSR peak conditions at every moment of the circuit lifetime and enhance the whole yield characteristic. Simulation results indicated that by applying the proper noise magnitude we can provide an optimum and nearly flat reliability level at any time before the DSR peak degradation level.

Our last proposal within the topic of heterogeneous-aware design was the combination of averaging cells with threshold logic gates resulting in what we call Averaging Cells Linear Threshold Gates (AC-LTG). The key idea behind this structure was the composition of a weighted average and a threshold operation that met the main purposes of both compounding structures, namely reliable computing and implementation of Boolean functions. In this thesis, we demonstrated that AC-LTG enables reliable computing at moderate redundancy levels. In the simulations, we took into account different sources of variability, such as input signal drift and parameter deviations from the manufacturing process.

In the second part of this thesis, we considered the statistical dispersion in the processing times that comes associated with the high variability levels of future nanotechnologies. In order to address this asynchrony issue, we included the time dimension as a design factor in our next redundancy reliable architecture proposal. Indeed, we introduced the partially-asynchronous R-fold Modular Redudnacy (pA-RMR), the main feature of which is the detection of input bit arrivals by means of token signals. Applying this small modification, we basically added a second degree of freedom to the RMR structure, which now not only has a configurable size (R replicas), but also allows modifying the number of tokens it waits before giving an output. Our analysis of pA-RMR reliability and performance resulted in a clear trade-off between both magnitudes and demonstrated the potential benefits of partially asynchronous redundant architectures in terms of performance and reliability management.

Finally, we concentrated on the problem of redundancy effort allocation in systems with many hierarchical layers. We performed several cross-layer reliability studies in order to provide meaningful orientations for the smart application of fault-tolerant techniques to

high transistor count systems. Basically, we compared the reliability of complex computing systems operating under different redundancy configurations and demonstrated that a correct distribution of redundancy through hardware hierarchy layers significantly improves the overall system reliability. Additionally, we also proved the key role of interconnects in the reliability of complex computing systems.

8.1 Thesis Contributions

The main contributions of this thesis can be summarized as follows:

- The fundamental error bounds for reliable computation have been extended for asymmetric error designs.
- A general variability-based fault model has been defined for the analysis of circuits and systems with heterogeneity and degradation effects.
- The optimum set of weights for averaging cell (AVG) architectures analyzed with the previous general fault model has been found and a method for its application has been proposed, namely the unbalanced averaging cell (U-AVG).
- An on-line self-adapting averaging architecture has been proposed to counteract the effects of technology degradation, specifically the adaptive averaging cell (AD-AVG).
- The so-called degradation stochastic resonance (DSR) effect has been discovered and analyzed in the context of AD-AVG architectures.
- A DSR-aware AD-AVG design based on noise injection has been proposed that provides an optimum and nearly flat reliability level over the lifetime of the system.
- A fruitful combination of averaging cells and linear threshold gates has been proposed (AC-LTG) to enhance the reliability of unate Boolean functions in an efficient way.
- A partially-asynchronous redundant architecture (pA-RMR) has been presented that extends the configuration possibilities to reach different trade-off between reliability and performance.
- Useful design guidelines have been provided for the smart application of redundancy techniques in complex multiple layer systems.

8.2 Future Work

During the development of this thesis, many lines of research were open that require further investigation. In the following list, we summarize some interesting ideas that we wish to expand upon in the future:

- Exploring the application of redundancy techniques in heterogeneous technology memory systems. We believe that the correct exploitation of heterogeneity in redundant memory structures has great potential for the development of high performance computing systems.
- Refining the final implementation of the AD-AVG architecture using a particular nanoscale technology to evaluate the reliability benefits and the associated overheads. In order to have a fair comparison between AD-AVG and other conventional fault-tolerant techniques, a more detailed review of their implementation needs to be done.
- Further developing our partially-asynchronous redundant structure (pA-RMR) and estimating its potential benefits at system level. Complex functional systems could be implemented combining multiple pA-RMR cells in series and parallel obtaining as a result the associated advantages of reliability and performance adjustment and error detection capabilities.

Appendix A

Publications related to this thesis

Most of the contents of this thesis have already been published in several journals and conference papers. Below we present a list of publications resulting from work in this thesis.

A.1 Journal papers

- **Nivard Aymerich** and Antonio Rubio, *"Reliability and Performance Tunable Architecture using Asynchronous R-fold Modular Redundancy"*, IEEE Transactions on Nanotechnology, 2013.
- **Nivard Aymerich**, Sorin Cotofana and Antonio Rubio, *"Controlled Degradation Stochastic Resonance in Adaptive Averaging Cell based Architectures"*, IEEE Transactions on Nanotechnology, June 2013.
- **Nivard Aymerich** and Antonio Rubio, *"Fault-tolerant nanoscale architecture based on linear threshold gates with redundancy"*, Microprocessors and Microsystems, July 2012.
- **Nivard Aymerich**, Shrikanth Ganapathy, Antonio Rubio, Ramon Canal and Antonio González, *"Impact of positive bias temperature instability (PBTI) on 3T1D-DRAM cells"*, Integration. The VLSI journal, June 2012.
- **Nivard Aymerich**, Sorin Cotofana and Antonio Rubio, *"Adaptive fault-tolerant architecture for unreliable technologies with heterogeneous variability"*, IEEE Transactions on Nanotechnology, May 2012.

A.2 Book chapters

- **Nivard Aymerich**, Sorin Cotofana and Antonio Rubio, *"Adaptive Fault-Tolerant Architecture for Unreliable Device Technologies"*, Nanoelectronic Device Applications Handbook, Chapter 46.

A.3 Conference papers

- **Nivard Aymerich** and Antonio Rubio, *"Study on the Optimal Distribution of Redundancy Effort in Cross-Layer Reliable Architectures"*, 13th IEEE International Conference on Nanotechnology, Beijing, China, 2013.
- **Nivard Aymerich** and Antonio Rubio, *"Extending the Fundamental Error Bounds for Asymmetric Error Reliable Computation"*, 9th ACM/IEEE International Symposium on Nanoscale Architectures (NANOARCH), New York City, USA, 2013.
- **Nivard Aymerich**, Sorin Cotofana and Antonio Rubio, *"Degradation stochastic resonance (DSR) in AD-AVG architectures"*, 12th IEEE International Conference on Nanotechnology, Birmingham, United Kingdom, 2012.
- **Nivard Aymerich**, Sorin Cotofana and Antonio Rubio, *"Adaptive fault-tolerant architecture for unreliable device technologies"*, 11th IEEE International Conference on Nanotechnology, Portland, Oregon, USA, 2011.
- **Nivard Aymerich**, Shrikanth Ganapathy, Antonio Rubio, Ramon Canal and Antonio González, *"Impact of positive bias temperature instability (PBTI) on 3T1D-DRAM cells"*, 21st edition of Great Lakes Symposium on VLSI, Lausanne, Switzerland, 2011.
- **Nivard Aymerich** and Antonio Rubio, *"Fault-tolerant nanoscale architecture based on linear threshold gates with redundancy"*, XXV Conference on Design of Circuits and Integrated Systems, Lanzarote (Canaries), Spain, 2010.

Bibliography

[1] K. Nikolic, A. Sadek, and M. Forshaw. Fault-tolerant techniques for nanocomputers. *Nanotechnology*, 13:357–362, 2002.

[2] John Von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. *Automata studies*, 34:43–98, 1956.

[3] W. Shockley. Circuit element utilizing semiconductive material, 1951. US patent US2569347.

[4] J. Bardeen and W.H. Brattain. Three-electrode circuit element utilizing semiconductive materials, 1950. US patent US2524035.

[5] J.S. Kilby. Miniaturized electronic circuits, 1964. US patent US3138743.

[6] J.S. Kilby. Method of making miniaturized electronic circuits, 1966. US Patent US3261081.

[7] G.E. Moore. Cramming more components onto integrated circuits. *Electronics Magazine*, 38(8), April 1965.

[8] Semiconductor industry association.

[9] S. Borkar. Designing reliable systems from unreliable components: the challenges of transistor variability and degradation. *IEEE Micro*, 25(6):10–16, Nov 2005.

[10] V Chandra. Dependable design in nanoscale cmos technologies: challenges and solutions. In *Workshop on Dependable and Secure Nanocomputing*, 2009.

[11] Gilbert Declerck. A look into the future of nanoelectronics. In *VLSI Technology, 2005. Digest of Technical Papers. 2005 Symposium on*, pages 6–10. IEEE, 2005.

[12] Kerry Bernstein, Ralph K Cavin, Wolfgang Porod, Alan Seabaugh, and Jeff Welser. Device and architecture outlook for beyond cmos switches. *Proceedings of the IEEE*, 98(12):2169–2184, 2010.

[13] M. Mishra and S.C. Goldstein. Scalable defect tolerance for molecular electronics. In *Proc. Workshop Nonsilicon Computation (NSC-1)*, pages 78–85, Feb 2002.

[14] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *Proceedings of the 40th annual Design Automation Conference*, pages 338–342. ACM, 2003.

[15] Pradip Bose. Designing reliable systems with unreliable components. *Micro, IEEE*, 26(5):5–6, 2006.

[16] International technology roadmap for semiconductors.

[17] Katsuhiro Tomioka, Masatoshi Yoshimura, and Takashi Fukui. A iii-v nanowire channel on silicon for high-performance vertical transistors. *Nature*, 488(7410):189–192, 2012.

[18] Xiang-Lei Han, Guilhem Larrieu, Pier-Francesco Fazzini, and Emmanuel Dubois. Realization of ultra dense arrays of vertical silicon nanowires with defect free surface and perfect anisotropy using a top-down approach. *Microelectronic Engineering*, 88(8):2622–2624, 2011.

[19] A Hellemans. Ring around the nanowire [news]. *Spectrum, IEEE*, 50(5):14–16, 2013.

[20] DDD Ma, CS Lee, FCK Au, SY Tong, and ST Lee. Small-diameter silicon nanowire surfaces. *Science*, 299(5614):1874–1877, 2003.

[21] Bonnie A Sheriff, Dunwei Wang, James R Heath, and Juanita N Kurtin. Complementary symmetry nanowire logic circuits: experimental demonstrations and in silico optimizations. *ACS nano*, 2(9):1789–1798, 2008.

[22] Qin Zhang, Wei Zhao, and Alan Seabaugh. Low-subthreshold-swing tunnel transistors. *Electron Device Letters, IEEE*, 27(4):297–300, 2006.

[23] Adrian M Ionescu and Heike Riel. Tunnel field-effect transistors as energy-efficient electronic switches. *Nature*, 479(7373):329–337, 2011.

[24] Siyuranga O Koswatta, Mark S Lundstrom, and Dmitri E Nikonov. Performance comparison between pin tunneling transistors and conventional mosfets. *Electron Devices, IEEE Transactions on*, 56(3):456–465, 2009.

[25] John P Shen, Wojciech Maly, and F Joel Ferguson. Inductive fault analysis of mos integrated circuits. *Design & Test of Computers, IEEE*, 2(6):13–26, 1985.

[26] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33, 2004.

[27] Andrew R Brown, Vincent Huard, and Asen Asenov. Statistical simulation of progressive nbti degradation in a 45-nm technology pmosfet. *Electron Devices, IEEE Transactions on*, 57(9):2320–2323, 2010.

[28] Martin Landau. Redundancy, rationality, and the problem of duplication and overlap. *Public Administration Review*, 29(4):346–358, 1969.

[29] Edward F Moore and Claude E Shannon. Reliable circuits using less reliable relays. *Journal of the Franklin Institute*, 262(3):191–208, 1956.

[30] Shmuel Winograd and Jack D Cowan. *Reliable computation in the presence of noise.* Number 22. MIT Press Cambridge, Mass., 1963.

[31] PG Depledge. Fault-tolerant computer systems. *Physical Science, Measurement and Instrumentation, Management and Education-Reviews, IEE Proceedings A*, 128(4):257–272, 1981.

[32] Frank P Mathur. Hardware reliability. 2003.

[33] A. Schmid and Y. Leblebici. Robust circuit and system design methodologies for nanometer-scale devices and single-electron transistors. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12(11):1156–1166, 2004.

[34] RE Lyons and W Vanderkulk. The use of triple-modular redundancy to improve computer reliability. *IBM Journal of Research and Development*, 6(2):200–209, 1962.

[35] Francis P Mathur and Algirdas Avižienis. Reliability analysis and architecture of a hybrid-redundant digital system: Generalized triple modular redundancy with self-repair. In *Proceedings of the May 5-7, 1970, spring joint computer conference*, pages 375–383. ACM, 1970.

[36] Jacob A Abraham and Daniel P Siewiorek. An algorithm for the accurate reliability evaluation of triple modular redundancy networks. *Computers, IEEE Transactions on*, 100(7):682–692, 1974.

[37] S Spagocci and T Fountain. Fault rates in nanochip devices. *Electrochem. Soc*, pages 354–368, 1999.

[38] K Nikolic, A Sadek, and M Forshaw. Architectures for reliable computing with unreliable nanodevices. In *Nanotechnology, 2001. IEEE-NANO 2001. Proceedings of the 2001 1st IEEE Conference on*, pages 254–259. IEEE, 2001.

[39] M. Stanisavljevic, A. Schmid, and Y. Leblebici. Optimization of nanoelectronic systems' reliability under massive defect density using cascaded R-fold modular redundancy. *Nanotechnology*, 19(46):1–9, Nov 2008.

[40] S. Roy and V. Beiu. Multiplexing schemes for cost-effective fault-tolerance. In *Nanotechnology, 2004. 4th IEEE Conference on*, pages 589–592. IEEE, 2004.

[41] S. Roy and V. Beiu. Majority multiplexing-economical redundant fault-tolerant designs for nanoarchitectures. *Nanotechnology, IEEE Transactions on*, 4(4):441–451, 2005.

[42] Akram S Sadek, Konstantin Nikolić, and Michael Forshaw. Parallel information and computation with restitution for noise-tolerant nanoscale logic networks. *Nanotechnology*, 15(1):192, 2004.

[43] Valeriu Beiu, Walid Ibrahim, and Sanja Lazarova-Molnar. A fresh look at majority multiplexing when devices get into the picture. In *Nanotechnology, 2007. IEEE-NANO 2007. 7th IEEE Conference on*, pages 883–888. IEEE, 2007.

[44] J. Han and P. Jonker. A system architecture solution for unreliable nanoelectronic devices. *IEEE Transactions on Nanotechnology*, 1(4):201–208, December 2002.

[45] Jie Han and Pieter Jonker. A defect- and fault-tolerant architecture for nanocomputers. *Nanotechnology*, 14(2):224–230, 2003.

[46] Gethin Norman, David Parker, Marta Kwiatkowska, and Sandeep Shukla. Evaluating the reliability of nand multiplexing with prism. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 24(10):1629–1637, 2005.

[47] George Roelke, Rusty Baldwin, and Dursun Bulutoglu. Analytical models for the performance of von neumann multiplexing. *Nanotechnology, IEEE Transactions on*, 6(1):75–89, 2007.

[48] James R Heath, Philip J Kuekes, Gregory S Snider, and R Stanley Williams. A defect-tolerant computer architecture: Opportunities for nanotechnology. *Science*, 280(5370):1716–1721, 1998.

[49] Daniel Mange, Moshe Sipper, Andre Stauffer, and Gianluca Tempesti. Toward robust integrated circuits: The embryonics approach. *Proceedings of the IEEE*, 88(4):516–543, 2000.

[50] T. Lehtonen, J. Plosila, and J. Isoaho. *On fault tolerance techniques towards nanoscale circuits and systems*, volume 708. Turku Centre for Computer Science, 2005.

[51] B. Hajek and T. Weller. On the maximum tolerable noise for reliable computation by formulas. *IEEE Transactions on Information Theory*, 37(2):388 – 391, March 1991.

[52] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.*, 5:115–133, 1943.

[53] W. Pitts and W. McCulloch. How we know universals: The perception of auditory and visual forms. *Bull. Math. Biophys.*, 9:127–147, 1947.

[54] M. Stanisavljevic, A. Schmid, and Y. Leblebici. Optimization of the averaging reliability technique using low redundancy factors for nanoscale technologies. *Nanotechnology, IEEE Transactions on*, 8(3):379–390, May 2009.

[55] F. Martorell, S.D. Cotofana, and A. Rubio. An analysis of internal parameter variations effects on nanoscaled gates. *IEEE Trans. Nanotechnol.*, 7(1):24–33, Jan 2008.

[56] Nicholas Pippenger. Reliable computation by formulas in the presence of noise. *Information Theory, IEEE Transactions on*, 34(2):194–197, 1988.

[57] Tomás Feder. Reliable computation by networks in the presence of noise. *Information Theory, IEEE Transactions on*, 35(3):569–571, 1989.

[58] Bruce Hajek and Timothy Weller. On the maximum tolerable noise for reliable computation by formulas. *Information Theory, IEEE Transactions on*, 37(2):388–391, 1991.

[59] William Evans and Nicholas Pippenger. On the maximum tolerable noise for reliable computation by formulas. *Information Theory, IEEE Transactions on*, 44(3):1299–1305, 1998.

[60] William S. Evans and Leonard J. Schulman. Signal propagation and noisy circuits. *Information Theory, IEEE Transactions on*, 45(7):2367–2373, 1999.

[61] William S Evans and Leonard J Schulman. On the maximum tolerable noise of k-input gates for reliable computation by formulas. *IEEE Transactions on Information Theory*, 49(11):3094–3098, 2003.

[62] JB Gao, Yan Qi, and José AB Fortes. Bifurcations and fundamental error bounds for fault-tolerant computations. *Nanotechnology, IEEE Transactions on*, 4(4):395–402, 2005.

[63] Falk Unger. Noise threshold for universality of two-input gates. *Information Theory, IEEE Transactions on*, 54(8):3693–3698, 2008.

[64] G. Snider. Computing with hysteretic resistor crossbars. *Applied Physics A*, 80(6):1165–1172, Mar 2005.

[65] J. Borghetti, Z. Li, J. Straznicky, X. Li, D.A.A. Ohlberg, W. Wu, D.R. Stewart, and R.S. Williams. A hybrid nanomemristor/transistor logic circuit capable of self-programming. *Proceedings of the National Academy of Sciences*, 106(6):1699–1703, Feb 2009.

[66] N. G. Stocks. Suprathreshold stochastic resonance in multilevel threshold systems. *Phys. Rev. Lett.*, 2000.

[67] O Oliaei. Stochastic resonance in sigma-delta modulators. *Electronics Letters*, 39(2):173–174, 2003.

[68] Thinh Nguyen. Robust data-optimized stochastic analog-to-digital converters. *Signal Processing, IEEE Transactions on*, 55(6):2735–2740, 2007.

[69] Ferran Martorell and Antonio Rubio. Defect and fault tolerant cell architecture for feasible nanoelectronic designs. In *Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006. International Conference on*, pages 244–249. IEEE, 2006.

[70] Cees Dekker. Solid-state nanopores. *Nature Nanotechnology*, 2(4):209–215, 2007.

[71] N. Aymerich, S. D. Cotofana, and A. Rubio. Adaptive fault-tolerant architecture for unreliable technologies with heterogenous variability. *Nanotechnology, IEEE Transactions on*, PP(99):1, 2012.

[72] RJ McIntyre. Multiplication noise in uniform avalanche diodes. *Electron Devices, IEEE Transactions on*, 13(1):164–168, 1966.

[73] S. Muroga. *Threshold logic and its applications.* John Wiley & Sons, 1971.

[74] M.K. Goparaju, A.K. Palaniswamy, and S. Tragoudas. A Fault Tolerance Aware Synthesis Methodology for Threshold Logic Gate Networks. In *IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems*, pages 176–183. IEEE, 2008.

[75] Tim Haifley and Atul Bhatt. Fault-tolerant ICs: The reliability of TMR yield-enhanced ICs. *Reliability, IEEE Transactions on*, R-36(2):224 –226, june 1987.

[76] B. Pratt, M. Caffrey, J.F. Carroll, P. Graham, K. Morgan, and M. Wirthlin. Fine-grain SEU mitigation for FPGAs using partial TMR. *Nuclear Science, IEEE Transactions on*, 55(4):2274–2280, 2008.

[77] Francis P. Mathur and Paulo T. de Sousa. Reliability models of NMR systems. *Reliability, IEEE Transactions on*, R-24(2):108 –113, june 1975.

[78] K. Nikolic, A. Sadek, and M. Forshaw. Fault-tolerant techniques for nanocomputers. *Nanotechnology*, 13:357–362, 2002.

[79] R Iris Bahar, Dan Hammerstrom, Justin Harlow, William H Joyner Jr, Clifford Lau, Diana Marculescu, Alex Orailoglu, and Massoud Pedram. Architectures for silicon nanoelectronics and beyond. *Computer*, 40(1):25–33, 2007.

[80] Alan F Murray and Anthony VW Smith. Asynchronous vlsi neural networks using pulse-stream arithmetic. *Solid-State Circuits, IEEE Journal of*, 23(3):688–697, 1988.

[81] Johan Lambie, Francesc Moll Echeto, José Luis Gonzalez, and Antonio Rubio. Asynchronous pulse logic cell for threshold logic and boolean networks. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 460–463. IEEE, 2005.

[82] N.P. Carter, H. Naeimi, and D.S. Gardner. Design techniques for cross-layer resilience. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1023–1028. European Design and Automation Association, 2010.

[83] B.S. Landman and R.L. Russo. On a pin versus block relationship for partitions of logic graphs. *Computers, IEEE Transactions on*, 100(12):1469–1479, 1971.

[84] MY Lanzerotti, G. Fiorenza, and RA Rand. Microminiature packaging and integrated circuitry: the work of ef rent, with an application to on-chip interconnection requirements. *IBM journal of research and development*, 49(4.5):777–803, 2005.

[85] A. Zykov and G. de Veciana. Exploring density-reliability tradeoffs on nanoscale substrates: When do smaller less reliable devices make sense? In *DFTVS'08*, pages 105–113. IEEE, 2008.