HAL
open science

Architectures of self-controllable digital operators

Ting An

To cite this version:

Ting An. Architectures of self-controllable digital operators. Micro and nanotechnologies/Microelectronics. Télécom ParisTech, 2014. English. NNT : 2014ENST0054 . tel-01366557

HAL Id: tel-01366557

https://pastel.hal.science/tel-01366557v1

Submitted on 14 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2014-ENST-0054

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Électronique et Communications »

présentée et soutenue publiquement par

Ting AN

le 30 Septembre 2014

Architectures d'opérateurs numériques auto-contrôlables

Directrice de thèse : **Lirida ALVES DE BARROS NAVINER**

Jury

M. Patrick GIRARD, Directeur de recherche CNRS, LIRMM	Rapporteur
M. Raoul VELAZCO, Directeur de recherche CNRS, TIMA	Rapporteur
M. Amara AMARA, Professeur, ISEP	Examinateur
Mme. Roselyne CHOTIN-AVOT, Maître de conférences, UPMC Paris VI	Examinateur
M. Weisheng ZHAO, Chercheur CNRS, IEF Université Paris-Sud	Examinateur
Mme. Lirida ALVES DE BARROS NAVINER, Professeur, Télécom ParisTech	Directrice de Thèse

T H È S E

TELECOM ParisTech
école de l'Institut Mines Télécom - membre de ParisTech
46 rue Barrault 75013 Paris - (+33) 1 45 81 77 77 - www.telecom-paristech.fr

Architectures of self-controllable digital operators

Ting AN

©

Le Département Communications et Electronique (COMELEC)
Institut Mines-Télécom, Télécom-ParisTech, LTCI-CNRS-UMR 5141
46 Rue Barrault, Paris CEDEX 13, 75634, France FRANCE

This thesis is set in Computer Modern 11pt,
with the LaTeX Documentation System

©Ting AN 2014

July 2014

"Genius is one percent inspiration,
ninety-nine percent perspiration"

by Thomas Edison

Acknowledgements

I wish to acknowledge all the people who helped and supported me during the thesis. Without them, this thesis will never be finished.

My deepest gratitude goes, first and foremost, to my advisor Professor Lirida Alves de Barros Naviner who provided me the entrance into this attractive domain, reliability of electronic devices. She helped me through all the stages of this thesis with so much patience. Her constant encouragement and guidance leaded to my growth and will light me in the future life as well. For me, she is more than an adviser, she is a lifelong mentor. The impressive experience to work with her is my precious for the whole life.

I would like to thank my rapporteurs Professor Patrick Girard and Professor Raoul Velazco for their reviews and constructive comments. My great thanks also belong to the jury members: Professor Amara Amara, Doctor Roselyne Chotin-Avot and Doctor Weisheng Zhao, for their participation and suggestions in my defense which will help me for the further improvement.

Moreover, I would like to express my heartfelt gratitude to a number of people who helped me directly or indirectly during my thesis. To my colleagues in COMELEC, Professor Hervé Petit and Professor Phillippe Matherat who reviewed my slides for many conferences and offered the valuable suggestions, and Florent Lozach, Tarik Graba and Yves Mathieu for their technical assistances. Also, I would like to take a time to thank Chantal Cadiat, Yvonne Bansimba and Karim Ben Kalaia for the support in administrative affaires and daily life.

More of my thanks go to the staffs of EDITE Florence Besnard and Fabienne Laussausaie for their miscellaneous assistances. An additional thanks to my Japanese teacher Atsuko Sasaki, she aroused my passion to this beautiful language.

I am very grateful to my colleagues in the office F305, Kaikai Liu, Arwa Ben Dia, Samuel Pagliarini and Marcia Costa, for the continuous care and love. It is my honor to work with the geniuses like you. Special thanks for Dr. Hao Cai and Dr. Tian Ban who helped me in my research of reliability and the paper drafts. For Elie Awwad, I am appreciated for the enthusiastic helps in travaux pratiques (TP) and administrative troubles. Furthermore, I would like to thank all my friends in Télécom ParisTech, Pietro Maris Ferreira, Mohamed Chaibi, Selma Belhadj Amor, Julia Vinogradova, Mariem Slimani, Omar Benzakour, Yimeng Zhao, Xiaoxing Yu, Youlong Wu, Yunfei Wei and so many others.

Finally, my biggest and the most special gratitude would belong to my parents Xiaoying An and Meijun Zhu, my husband Yao Li and my grandmother Shuhui Ma. Their love and support encouraged me to overcome all the difficulties to accomplish this thesis.

Abstract

The steady geometrical reduction of CMOS technology brought a great industry success and affected a lot the human life. However, the integrated circuits (ICs) are shrinking along with new challenges. The design and manufacturing are becoming more complex than before. ICs suffer from two major problems: the parametric variability in materials and limited precision processes, and the sensibility to environment noise. For instance, alpha particles or neutrons can randomly affect the electronic devices during their usage time. This phenomena was not observed in devices of previous generations.

With the increasing failure rate related to these two problems, the future ICs implemented with sub-micron CMOS technology are expected to be less reliable. New reliable ICs are highly desired in critical applications such as avionic, transport and biomedicine. Numerous solutions have been reported in literature covering the enhancement in different abstraction levels (i.e., system level, architecture level and electrical level). Among them, the improvement in architecture level benefits the independence from CMOS technology and the low latency of reaction. Expected architectural solutions will be self-controlled meaning that is able to either automatically indicate the occurrence of faults or directly mask the faults.

This thesis is devoted to the reliability analysis methodology and reliability enhancement approaches on architecture level. In particular, the reliability issues in usage time are discussed in details. Digital arithmetic operators for signal processing are taken as studied objects. In addition to the basic operators (i.e., binary adders), coordinate rotation digital computer (CORDIC) and advanced encryption standard (AES) processor are also covered in the scope of this work.

Concerning the reliability analysis, different assessment approaches are proposed. Analytical methods aim to evaluate the fault masking properties on combinational circuits. Based on proposed methods, it is possible to determinate the critical operator on an iterative processor. The reliability improved by self-checking circuits is estimated as well. Moreover, motivated by a realistic reliability analysis, a gate structure-aware fault modeling is introduced. It can be adopted in many existing analytical methods. On the other hand, the simulation method is applied to investigate the impact of two important aging mechanisms: NBTI and HCI. This methodology is demonstrated through the studies on different 65 nm CMOS adders.

The attempts on reliability improvement concentrate on effective design. This thesis firstly describes how to model and characterize the efficiency of reliability enhanced techniques considering the induced overcost (i.e., area, delay, power consumption). The most complex component in AES processors (i.e., S-Box) is taken as a case study. A reliable architecture for S-Boxes in AES processor is then proposed based on the obtained results. This architecture is capable of dealing with all single faults with a small area penalty.

Contents

Acknowledgements i

Abstract iii

List of Tables vii

List of Figures viii

List of Acronyms xii

Résumé français 1

1 Introduction 23
- 1.1 Motivation . . . 23
- 1.2 Report organization . . . 25

2 Basics on reliability 27
- 2.1 Reliability and faults . . . 27
  - 2.1.1 Metrics of reliability . . . 27
  - 2.1.2 Faults classification . . . 29
- 2.2 Fault sources . . . 30
  - 2.2.1 Aging effects: NBTI and HCI . . . 30
  - 2.2.2 Radiation related effects . . . 32
  - 2.2.3 Thermal noise . . . 34
- 2.3 Fault-tolerant techniques . . . 35
- 2.4 Conclusions . . . 38

3 Reliability evaluation methodologies 39
- 3.1 Fault masking . . . 39
  - 3.1.1 Electrical masking . . . 39
  - 3.1.2 Temporal masking . . . 39
  - 3.1.3 Logical masking . . . 40
  - 3.1.4 Logical masking assessment . . . 41
- 3.2 Masking analysis methods . . . 47
  - 3.2.1 Modeling of transient fault . . . 47
  - 3.2.2 Single fault on iterative circuits . . . 51
  - 3.2.3 Multiple faults on CED circuits . . . 58
- 3.3 Analysis of aging effects . . . 66
  - 3.3.1 Aging-aware design flow . . . 66
  - 3.3.2 Case study: 65 nm adders . . . 68

3.4 Conclusions . . . 73

4 Effective design for reliability **75**
4.1 Effective fault-tolerant design . . . 75
4.1.1 Introduction . . . 75
4.1.2 Case study: S-Box in AES . . . 76
4.2 Hybrid fault-tolerant S-Boxes . . . 81
4.2.1 Introduction . . . 81
4.2.2 Hybrid architecture for S-Boxes . . . 83
4.2.3 Cost evaluation . . . 85
4.2.4 Reliability assessment . . . 87
4.3 Conclusion . . . 89

5 Conclusions and prospectives **91**
5.1 Conclusions . . . 91
5.2 Prospectives . . . 93

A Binary adders **95**
A.1 One-bit adder . . . 95
A.2 Ripple carry adder . . . 96
A.3 Carry-select adder . . . 97
A.4 Carry look-ahead adder . . . 97
A.5 Carry-skip adder . . . 98
A.6 Parallel prefix adder . . . 100

B Finite field arithmetic operator **105**
B.1 Finite field . . . 105
B.2 Addition on $GF(2^m)$ . . . 106
B.3 Multiplication on $GF(2^m)$ . . . 106

Bibliography **108**

List of Tables

1 Cot en temps 11
2 Comparaison de la performance des additionneurs (1) 14
3 Comparaison de la performance des additionneurs (2) 14
4 Traitement de la reconfiguration 19
5 Le séquencement de l'exécution 20
6 Évaluation de performance du matériel (seulement S-boxes) 20
7 Performances en termes de surface et de délais pour les processeurs d'AES . 22

1.1 2013 ITRS - DRAMs, MPUs, and ASICs 23

3.1 Fault analysis in CMOS gate NAND2 48
3.2 Reliability evaluation based on different gate fault models 51
3.3 Operator impact coefficient of Conventional CORDIC (normalized) 57
3.4 Operator impact coefficient of MVSA CORDIC (normalized) 58
3.5 Eligibility rank of different implementations 58
3.6 Computational complexity to obtain PTM for a given level 64
3.7 Computational complexity to obtain PTM for a cascade of levels 64
3.8 Time consumptions (s) 65
3.9 Performance comparison of adders (1) 69
3.10 Performance comparison of adders (2) 69

4.1 Evaluation of the hardware performance 80
4.2 Reliability computation of Original S-Box (G=133) 81
4.3 Reconfiguration processing 84
4.4 The execution procedure 85
4.5 Evaluation of the hardware performance (only S-boxes) 87
4.6 Area and time performances of AES processor 90

A.1 One-bit addition rules 96
A.2 Comparisons of different adder architecture 103

List of Figures

1 La courbe de la baignoire . . . 2
2 Propagation de fautes/erreurs . . . 3
3 Mechanisms de a. HCI et b. NBTI . . . 3
4 a. Frappe d'une particule ionisante sur une jonction. b. Impulsion de courant cause par des particules ionisantes . . . 4
5 Schéma de TMR, b. Architecture générale d'un CED . . . 5
6 Masquage de fautes . . . 7
7 Schéma général de CED . . . 8
8 Porte logique OR . . . 9
9 ITM_{CED} pour une porte OR comme le F . . . 10
10 Flot de tâches de l'approche proposée . . . 11
11 Les évaluations des événements des sorties . . . 12
12 Flot de conception proposée . . . 13
13 Comparison des performances des additionneurs en termes de la dégradation de délais . . . 15
14 Entrées et le tableau des States . . . 16
15 Diagramme de chiffrement et déchiffrement AES . . . 17
16 Efficacité de la solution d'amélioration de fiabilité . . . 17
17 Architecture parallèle d'un seul tour de l'AES . . . 18
18 Diagramme du tableau configurable de S-Boxes (CSBA) . . . 19
19 L'architecture hybride pour les S-Boxes . . . 20
20 Comparaison des fiabilités pour le processeur d'AES . . . 21

2.1 System architecture examples and system reliabilities with respect to time . 28
2.2 Bathtub curve. Notice that the time axis is not scaled . . . 29
2.3 Fault/error propagation . . . 29
2.4 Fault sources for different fault types . . . 31
2.5 HCI and NBTI mechanisms at CMOS device level . . . 31
2.6 HCI and NBTI induced physical parameters degradation . . . 32
2.7 a. Ion particle hit on junction; b. Current pulse caused by ion particle . . . 33
2.8 Possible scenarios for charge injection in CMOS . . . 34
2.9 Normalized SRAM SER/bit trend . . . 34
2.10 Chronogram of AND gate with thermal noise . . . 35
2.11 a. Triple Module Redundancy; b. N-Modular Redundancy. . . . 36
2.12 General architecture of a CED scheme. . . . 37

3.1 Example of electrical attenuation on a logic circuit . . . 40
3.2 Example of a temporal masking . . . 40
3.3 Example of a logical masking . . . 41

3.4 Comparison of SRAM bit SER with latch SER 41
3.5 Logic gate OR 42
3.6 PTM computation of an example circuit 43
3.7 Modelin of permanent fault in PTM approache 43
3.8 Signal propagation on a OR gate 44
3.9 Reliability computation by signal probability propagation 45
3.10 The principle structure of FIFA platform. 46
3.11 a. A CMOS inverter; b. A CMOS NAND2 gate 48
3.12 Propagation of signal probabilities in C17 49
3.13 Different failure rates of constituent gates in C17. 50
3.14 Different gate failure probabilities 50
3.15 Iterative processor G 52
3.16 a. General structure to compute the impact coefficient; b. Injection of a fault 53
3.17 Elementary iteration of CORDIC 54
3.18 Elementary rotational iteration of Scaling-free CORDIC 55
3.19 Normalized fault impact coefficient of two different CORDICs 56
3.20 Fault probability in different bit positions 57
3.21 Concurrent error detection circuit 59
3.22 Framwork of the fault injection method. 60
3.23 ITM_{CED} for an OR gate as function circuit F. 60
3.24 The workflow of the proposed approach 63
3.25 Level partition of CED circuit 63
3.26 Structure of the studied adders 65
3.27 Run-time partition of improved method 66
3.28 a. Outputs events probabilities evaluation; b. Comparison of CED schemes 67
3.29 Aging-aware design flow 68
3.30 Power delay product versus year of 4-bit length RCA 70
3.31 NBTI and HCI Delay degradation over time of different adders, T=300K . 71
3.32 An example of the critical path changing with time under the HCI effect . . 71
3.33 Comparison of adders' performance on delay degradation 72

4.1 Inputs and State array 77
4.2 Diagram of AES encryption and decryption 77
4.3 Block diagram of S-Box 79
4.4 Reliability of fault-tolerant S-Boxes 81
4.5 a. Effective frequency; b. Effective power (50 MHz) 82
4.6 Reliability improvement efficiency 82
4.7 Parallel architecture of a single AES round 83
4.8 Diagram of Configurable S-Boxes Array (CSBA) 84
4.9 Permanent faults on CSBA 85
4.10 Hybrid S-Boxes architecture 85
4.11 Reliability comparison of AES processors 88

5.1 Reliability-aware front end design flow 94

A.1 Dependencies of arithmetic operations 95
A.2 Logic symbol and circuit architecture of full adder 96
A.3 Logic symbol and circuit architecture of Half Adder 96
A.4 Logic symbol and circuit architecture of n-bit RCA 97
A.5 Carry Select Adder architecture, $m = n/k$ 98
A.6 Logic implementation of: a. 4-bits CLA Adder; b. Carry lookahead chain . 99
A.7 Classic CSKA architecture 99

A.8 Architecture of a 8-bit fixed-size CSKA 100
A.9 An example of 6 bit prefix sum 101
A.10 Prefix sum operator 101
A.11 Structure of 16-bits SKA 102
A.12 Structure of 16-bits SKA 102
A.13 Structure of 16-bits BKA 103

List of Acronyms

AES Advanced Encryption Standard

ADP Area Delay Product

ASIC Application-Specific Integrated Circuit

BKA Bren-Kung Adder

BSIM Berkeley Short-channel IGFET Model

CA Cluster Array

CED Concurrent Error Detection

CLA Carry Lookahead Adder

CMOS Complementary Metal Oxide Semiconductor

CORDIC COordinate Rotation Digital Computer

CPM Conditional Prabobilistic Model

CPU Central Processing Unit

CSA Carry-Select Adder

CSBA Configurable S-Boxes Array

CSKA Carry-SKip Adder

DAC Duplication And Comparison

DGC Double Gate CMOS

DRAM Dynamic Read Access Memory

DSP Digital Signal Processing

DWAA Dynamic Weighted Averaging Algorithm

EM Electron Migration

FA Full Adder

FC Full Configuration

FIR Finite Impulse Response

FIT Failure-in-Time

FPGA Field-Programmable Gate Array

FSM Finite State Machine

FT Fault-Tolerant

GF Galois Field

HCI Hot Carrier Injection

HF Half Adder

IC Integrated Circuit

IEEE Institute of Electrical and Electronics Engineers

ITM Ideal Transfer Matrix

ITRS International Technology Roadmap of Semiconductor

KSA Kogge-Stone Adder

LET Linear Energy Transfer

LUT Lookup Tables

MVSA Modified Virtually Scaling-free Adaptative

MOSFET Metal Oxide Semiconductor Field Effect Transistor

MTTF Mean-Time-to-Failure

MPU Memory Protection Unit

NB Normal Basis

NBTI Negative Bias Temperature Instability

NMOS N-Channel Metal Oxide Semiconductor

NMR N-tuple Modular Redundancy

NP Nondeterministic Polynomial time

NTF Noise Transfer Function

PB Polynomial Basis

PBR Probabilistic Block Reliability

PBTI Positive Bias Temperature Instability

PDP Power Delay Product

PGM Probabilistic Gate Model

PM Phase Margin

PMOS P-Channel Metal Oxide Semiconductor

PPA Parallel Prefix Adder

PTM Probabilistic Transfer Matrix

QCA Quantum Cellular Automata

RCA Ripple Carry Adder

RMS Root-mean-square

RTL Register Transfer Level

S-Box SubBytes Box

SE Soft Error

SER Soft Error Rate

SET Single Event Transient

SEU Single Event Upsets

SKA SKlansky Adder

SNDR Signal-to-Noise-plus-Distortion Ratio

SNR Signal-to-Noise Ratio

SoCs Systems-on-Chip

SOI Silicon on Insulator

SPR Signal Probability Reliability

SPR-MP Signal Probability Reliability Multiple-Pass

TDDB Time-Dependent Dielectric Breakdown

TMR Triple Modular Redundancy

TTF Time-to-Failure

VHDL VHSIC Hardware Description Language

Résumé français

I Introduction

La réduction géométrique régulière des finesses de gravure en microélectronique a conduit à un grand succès dans l'industrie et a beaucoup changé la vie humaine. Cependant, cette évolution technologie continue apporte de nouveaux défis aux circuits intégrés (CIs). Leur conception et fabrication sont de plus en plus complexes qu'avant. Les CIs sont affectés par deux phénomènes majeurs: la variabilité paramétrique et les limites des procédés de fabrication, ainsi que la sensibilité aux conditions environnementales. Par exemple, les particules alpha ou/et les neutrons affectent de façon aléatoire les composants électroniques pendant leur durée d'utilisation. Des perturbations si significatives n'avaient pas été observées dans les générations précédentes.

Avec l'augmentation du taux de défaillance lié à ces deux phénomènes, les circuits basés sur les technologies nanoélectroniques sont censés être de moins en moins fiables. Le critère de fiabilit est exigé dans les applications critiques telles que l'avionique, des transports et de biomédecine. De nombreuses solutions ont été proposées dans la littérature comprenant l'amélioration de la fiabilité aux différents niveaux d'abstraction (par exemple, au niveau du système, au niveau de l'architecture ou au niveau électrique). Parmi ces techniques, l'amélioration au niveau de l'architecture profite de l'indépendance de la technologie et de la faible latence de réaction. Les solutions architecturales faisant l'objet de cette thse sont du type auto-contrôlables, c'est-à-dire qui ont capables d'indiquer automatiquement l'apparition de fautes ou de masquer les fautes directement.

Cette thèse est consacrée aux méthodes d'analyse et d'amélioration de la fiabilité au niveau de l'architecture. Les problèmes de fiabilité pendant la durée d'utilisation d'un circuit électronique sont décrits en détails. Les opérateurs arithmétiques numériques pour le traitement du signal sont pris comme des études de cas. Les opérateurs élémentaires (c-à-d additionneurs binaires), le calcul numérique par rotation de coordonnées (CORDIC) et le processeur du standard de chiffrement avanc (AES) sont également traités dans le cadre de ce travail.

Ce résumé en français est composé de cinq sections. La section II passe en revue les concepts de base de la fiabilité et des fautes. Elle présente les problèmes de fiabilité étudiés, y compris le vieillissement. Les techniques de tolérance aux fautes courantes sont aussi mentionnés dans cette section. La section III présente les méthodes proposées pour l'évaluation de la fiabilité. Ces approches sont basées sur la propriété de masquage logique des circuits combinatoires. Cette section traite également des deux mécanismes principaux de vieillissement: l'instabilité en temprature par polarisation négative (NBTI: Negative Bias Temperature Instability) et l'injection des porteurs chauds (HCI: Hot Carrier Injection). La Section IV est dédiée à la conception efficace de processeurs tolérants aux fautes. Tout d'abord, une méthode permettant de caractériser différentes solutions de tolérance aux fautes est proposée. Ensuite, une structure tolérant des fautes simples (qu'elles soient

transitoires ou permanentes) dans les modules S-Box des processeurs AES est proposée. Finalement, la Section V dresse les conclusions de ce travail.

II Fiabilité

Les circuits intégrés construits à partir de technologies fortement submicroniques devraient souffrir d'une réduction significative de la fiabilité. Ceci est devenu une préoccupation majeure et pourrait limiter l'évolution de la technologie. Les problèmes de fiabilité sont notamment le résultat de la variabilité paramétrique et les perturbations environnementales.

II.a La fiabilité et les fautes

La fiabilité est une caractéristique dépendant du temps qui reflète la continuation d'un fonctionnement correct. L'IEEE définit la fiabilité notée R ou $R(t)$ comme l'aptitude d'un système ou d'un composant à effectuer la fonction pour laquelle il a été conçu pendant une période déterminée $(0, t)$ étant donné les conditions spécifiées. Un système échoue lorsque l'opération prévue est différente de ce qui est attendu en raison de l'existence de fautes.

La courbe de la baignoire est largement acceptée pour décrire le taux de défaillance (voir la figure 1). Le taux de défaillance au début de la vie est essentiellement lié aux défauts de fabrication et peut être très élevè. Les composants défectueux sont normalement détectés par les tests. Une fois les composants défectueux rejetés, le taux de défaillance se maintient presque constante sur une période importante nommée la vie utile. L'obsolescence du composant est une conséquence du vieillissement et correspond à une période où le taux de défaillance recommence à augmenter de manière importante. Il faut noter que la manifestation des fautes aléatoires induites par l'environnement (interne ou externe) va encore augmenter le taux de défaillance pendant la période de la vie du composant.

Typiquement, une faute peut être le résultat d'un défaut au cours de la fabrication, d'une défaillance physique, d'une erreur de conception ou du bruit de l'environnement. Une faute peut se propager et produire un signal erroné. Une erreur est alors définie comme la manifestation d'une faute et peut provoquer successivement une autre erreur.

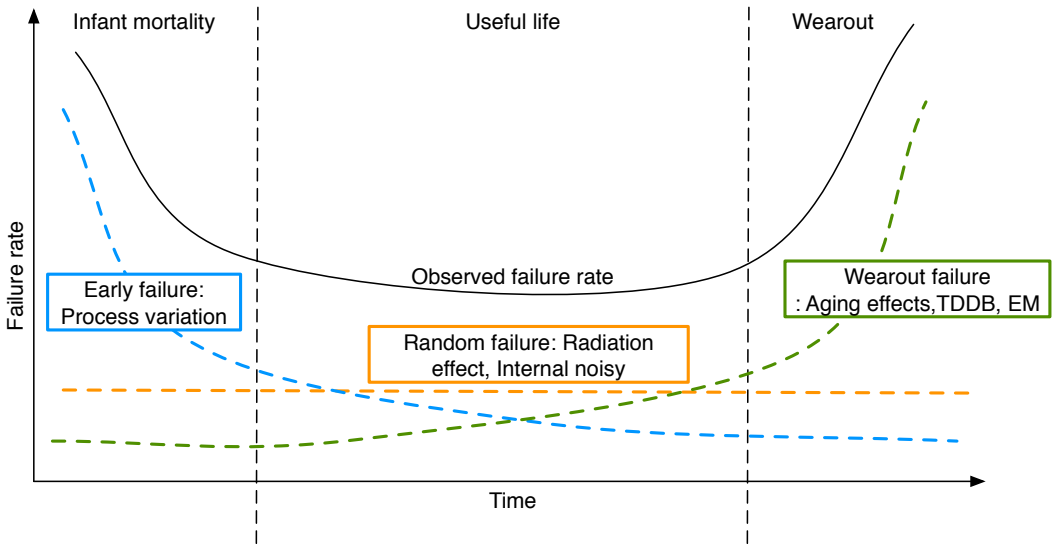


Figure 1: La courbe de la baignoire (l'axs de temps n'est pas mis en échelle).

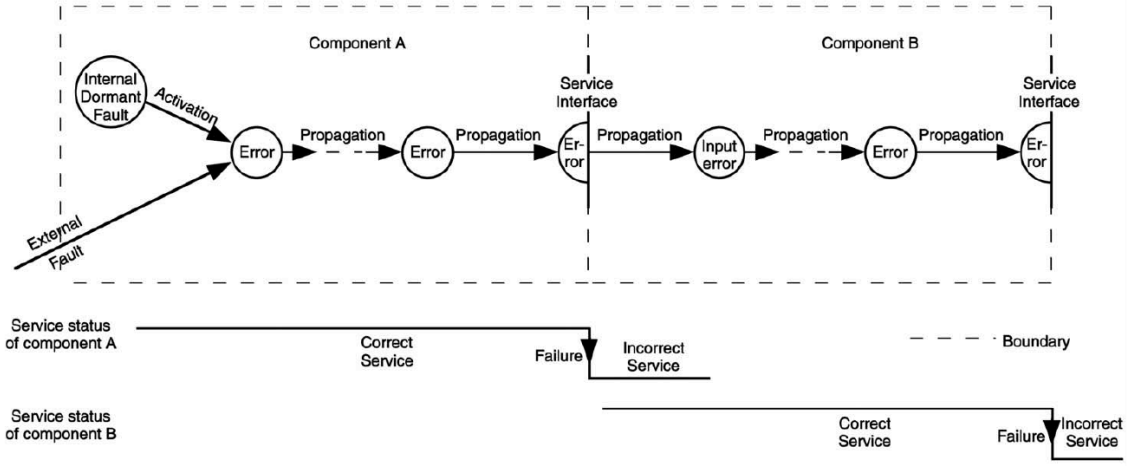


Figure 2: Propagation de fautes/erreurs.

La Fig. 2 illustre la propagation de fautes/erreurs.

D'un point de vue de son occurrence au cours du temps, une faute peut être classée en trois catégories: la faute permanent, la faute transitoire et la faute intermittente. Les fautes permanentes se produisent et restent jusqu'à une réparation du composant. Elles sont dues aux changements irréversibles dans la structure du circuit. Les fautes permanentes conséquentes du processus de fabrication sont en général bien traitées lors des tests post-fabrication. Néanmoins, ces fautes peuvent également se manifester pendant l'utilisation du dispositif à cause du vieillissement.

Les fautes intermittentes sont souvent induites par les variations de processus de fabrication. Ils sont caractérisées par une occurrence répétée et sporadiques, parfois avec le comportement d'éclatement. Les fautes intermittentes affectent les circuits/appareils avec des variations paramétriques, qui peuvent agir comme des courts circuits ou circuits ouverts sous certaines conditions (bruit, risques, vieillissement, etc.), et peuvent devenir des fautes permanentes.

Les fautes transitoires sont provoquées par les conditions environnementales temporaires comme les particules d'alphas et les neutrons, les décharges électroniques, le bruit thermique etc. Elles se produisent pendant une période courte et puis disparaissent. Les fautes transitoires sont normalement décrites comme des erreurs indépendants qui apparais qu'une fois. Une faute transitoire peut provoquer un dysfonctionnement des composants sans autant l'endommager. Le caractère aléatoire de ces fautes rend difficile l'étude de leur impact dans un composant électronique complexe.

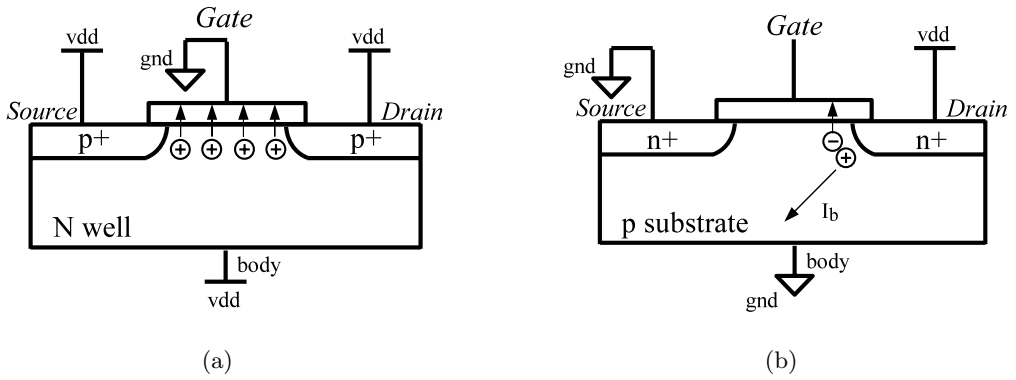
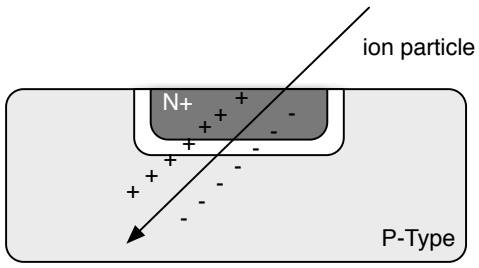
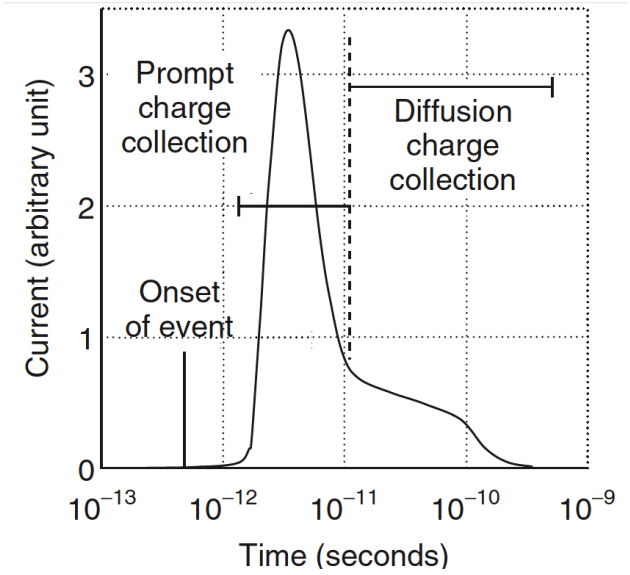


Figure 3: Mechanisms de a. HCI et b. NBTI.



(a)



(b)

Figure 4: a. Frappe d'une particule ionisante sur une jonction. b. Impulsion de courant cause par des particules ionisantes.

En fait, les fautes permanents liés au vieillissement et les faute transitoire liés aux rayonnements sont la source d'expériences des erreurs principaux soufferts par les CIs en phase de l'utilisation. Une autre source de menace importante est le bruit thermique due à la diminution continue de la tension de circuit, où une forte probabilité de fautes peut autre être prévu. Ainsi, les discussions dans cette résumé sont limitées aux effets du vieillissement, les effets liés aux rayonnements et le bruit thermique. D'autres mécanismes sont considérés comme des sources cumulatifs.

II.b Les techniques de tolérances aux fautes

La tolérence aux fautes qui est étroitement liée à la fiabilité représente la capacité d'un système à fonctionner correctement en présence de fautes. L'idée de base de la tolérance aux fautes est l'utilisation de la "redondance", sous une forme de spatiale ou temporelle. La redondance spatiale repose sur les composants supplémentaires, les fonctions ou les éléments de données. La redondance temporelle est basée sur la répétition des calculs et la comparaison avec les résultats déjà obtenus.

Les techniques de tolérances aux fautes sont mises en œuvre principalement par:

- Le masquage de fautes qui fournit le confinement des fautes
- La détection de fautes qui indique un fonctionnement erroné et est suivie par une action corrective dans la plupart des cas (par exemple, re-calcul).

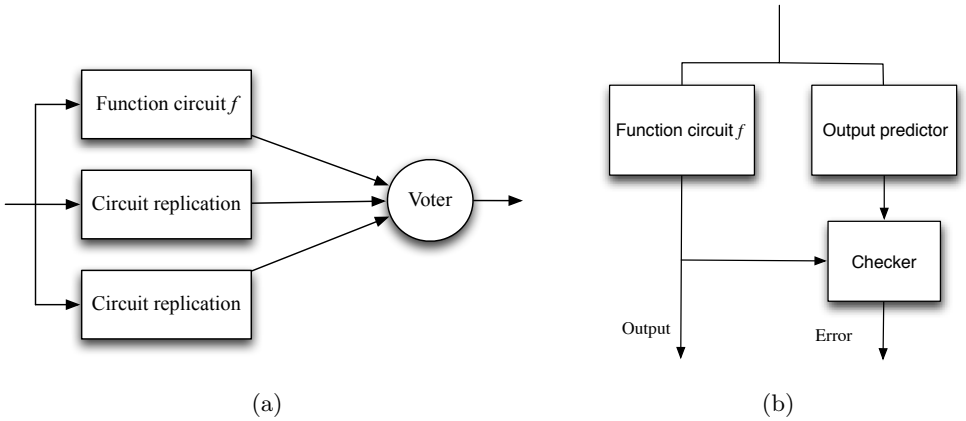


Figure 5: a. Schéma de la redondance modulaire triple (TMR), b. Architecture générale d'un CED.

Une des techniques du masquage de fautes classique est la *redondance modulaire triple* (TMR) montrée dans la Fig. 5(a).

Une structure TMR se compose de trois modules et un arbitre. Les modules exécutent la même opération et leurs résultats sont comparés par le voteur. Il choisit la sortie appropriée en fonction des critères prédéfinis. Dans la plupart des cas, l'arbitre applique un vote par majorité. Comme le TMR n'a pas besoin de changement des topologies d'opérateurs, il peut être facilement mis en œuvre. Le TMR classique exige une forte redondance matérielle. En revanche, la *redondances temporelle triple* (TTR) utilise la même ressource à mettre en œuvre la fonction trois fois et compare les résultats obtenus pour partir de différents tours de calcul. La surface supplémentaire est limitée mais cela implique le sacrifice de la vitesse de calcul. D'autre part, la redondance temporelle ne peut pas traiter les fautes permanentes, car ces ces fautes se retrouvent au même endroit lors de tous les calculs ne peuvent donc pas être détectées et les corrigées.

Une autre technique bien connue est la *détection d'erreur concurrente* (CED) (voir fig. 5(b)). Dans un schéma du CED, le circuit de fonction effectue le calcul de l'entrée I et donne les sorties $f(I)$. Une autre unité indépendante prévoit certaines caractéristiques particulières de la sortie $f(I)$ du système pour chaque séquence d'entrée I.

La *duplication avec la comparaison* (DAC) est la plus simple parmi les méthodes de CED, où l'unité de prédiction est mise en œuvre avec la même fonction. Un contrôleur est utilisé pour contrôler les deux sorties. Le circuit indique une erreur si les sorties ne sont pas identiques. Le surcoût matériel du DAC est de 100% sans tenir compte du contrôleur et peut se réveler trop important pour de nombreux cas. En fait, certains opérateurs ont une certaine redondance inhérente, par example, l'additionneur à sélection de retenue.(CSA). L'approche DAC peut bénéficier de cette redondance inhérente pour réduire le surcoût en surface. La *prédiction de parité* est largement utilisée dans les solutions du type CED pour son bas surcoût de matériel. La prédiction de parité pour les opérateurs arithmétiques calcule la parité de la donnée de sortie en fonction des parités des retenues internes et des parités des opérandes d'entrée. Les arbres de XORs sont utilisés pour générer les parités des entrées et les retenues. La structure de prédiction de parités souffre des fautes sur le chemin de propagation des retenues. Ces fautes affecteraient plusieurs bits à la sortie ainsi que le circuit de la vérification ou le contrôleur, et resteraient donc indétectables.

La prédiction dans les systèmes CED peut également être basée sur les codes de détection d'erreurs, comme les *codes de M-out-of-n codes*, les *codes de Berger* les *codes d'AN* (A et N indiquent deux entiers appliquées pour coder et décoder les mots de code, respectivement) et les *codes de résidus*.

Malgré la diversité des codes de détection d'erreur, ils ont tous une distance de Hamming qui caractérise la limitation de leur la capacité de détection. Une distance de Hamming d de codes de détection permet la détection de $d-1$ bits erronés. Par exemple, bien que les codes de Berger peuvent détecter toutes les erreurs unidirectionnelles, en raison de sa distance de Hamming de 2, la moitié des erreurs de 2 bits sont indétectables. Une faute simple sur des portes logiques peut produire plusieurs bits erronés sur les sorties, et peut rester indetectable si le nombre de bits erronés dépasse $d-1$. En outre, la mise en œuvre de codes de détection d'erreur requiert la conception de blocs en fonction de l'application cible. La redondance de matériel tels que TMR est toujours une solution intéressante pour faire face aux fautes permanents et transitoires. Par conséquent, nous nous concentrons sur la méthode TMR et les méthodes du CED, en particulier les CEDs avec la prédiction de parité.

III Méthodes d'évaluation de la fiabilité

La diminution de la fiabilité du fait de la réduction d'échelle de la technologie CMOS oblige les concepteurs à s'intéresser à l'évaluation de la fiabilité en plus des paramètres traditionnels que sont la surface, la vitesse et la consommation. Plusieurs travaux diverses ont été développés pour trouver un bon compromis entre la précision et la complexité de calcul de la fiabilité. Nous nous concentreront sur les techniques d'évaluation de la fiabilité au niveau de la porte logique pour caractériser les circuits logiques. Nous allons étudier des méthodes efficaces pour évaluer la capacité du masquage de fautes de circuits logiques. Nous étudierons aussi l'impact des mécanismes NBTI et HCI sur les circuits.

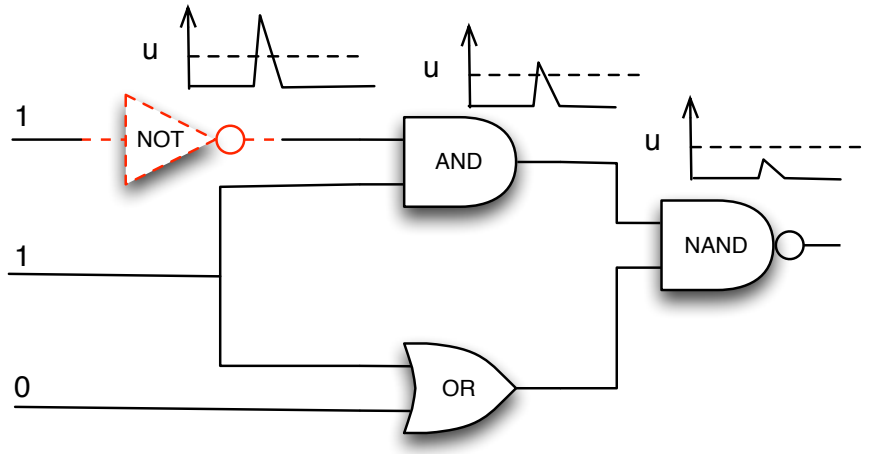
III.a Le masquage de fautes

Une faute transitoire dans un bloc logique peut se propager aux sorties primaires et être capturée par les cellules de mémoire lors qu'elle n'est pas filtrée par l'une des propriétés suivantes:

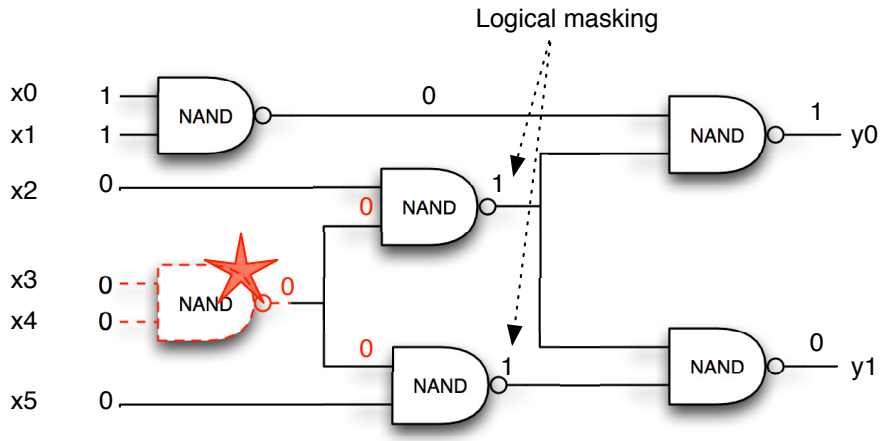
- *Masquage électrique* qui se produit quand une faute de glitch est atténuée par une série des portes logiques en raison des propriétés électriques (c-à-d, le glitch n'a pas assez de durée de temps ou d'amplitude pour se propager aux sorties).
- *Masquage temporel* qui indique qu'une erreur arrive en dehors de la fenêtre d'échantillonnage de l'élément de mémorisation.
- *Masquage logique* qui apparaît quand la sortie d'une porte logique est exacte quelle que soit la valeur de l'entre atteinte par la faute.

Le figure 6 montre les exemples des trois masquages. Pendant longtemps, les circuits logiques ont été considérés résistants aux fautes grâce aux effets de masquage, en particulier pour les fautes transitoires induites par des effets de rayonnement. Cependant, la vulnérabilité aux fautes des circuits combinatoires est devenue sévère avec la réduction continue de la dimension. En plus, en raison de la réduction des deux autres effets de masquage, le masquage logique est devenu l'effet principal de masquage dans les circuits logiques. Ainsi, notre étude ciblera donc la fiabilité du circuit logique par rapport au masquage logique.

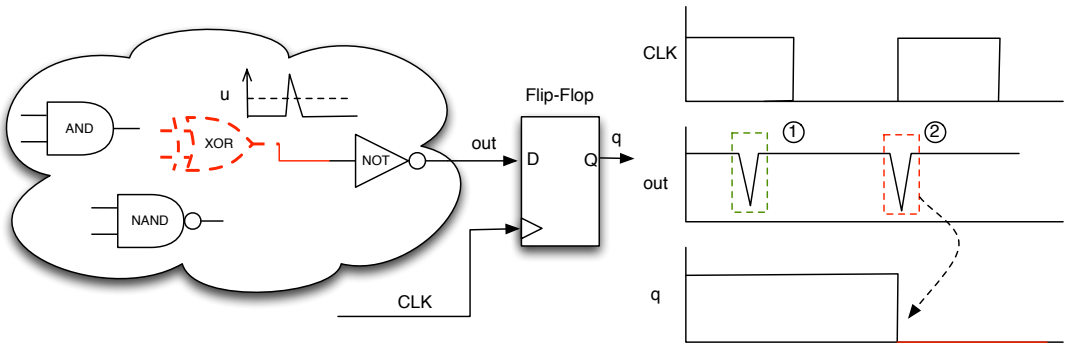
L'évaluation du masquage logique a été largement reportée dans le littérature. Les matrices de transfert probabilistes (PTM) sont considérées comme une méthode précise. Dans cette méthode, chaque bloc logique b_i est représenté par une matrice de base PTM_i. La PTM_c du circuit est obtenu en combinant tous les PTMs élémentaires. La fiabilité du circuit peut être directement extraite du PTM_c.



(a)



(b)



(c)

Figure 6: a. Masquage électrique, b. masquage logique, c. masquage temporel.

Le modèle de la probabilité du signal (SPR) se base sur le principe que chaque signal logique peut avoir quatre états: "0" correct (0_c), "0" incorrect (0_i), "1" correct (1_c) et "1" incorrect (1_i). La fiabilité d'un signal est donnée par sa probabilité d'avoir l'un des deux états corrects (0_c et 1_c). Cette méthode offre une complexité linéaire, mais souffre de problèmes de précision en cas de fanouts reconvergents.

Le modèle probabiliste binomial (PBR) analyse la fiabilité du circuit à partir de la fiabilité des nœuds dans le circuit. Il tien compte aussi bien de fautes simples que multiples. Un circuit combinatoire est modélisé comme une boîte noire avec l'entrée x et la sortie y. Cette méthode est basée sur l'hypothèse que la fiabilité du circuit exclusif est le sommation d'obtenir un y correcte étant donné un $x = x_i$, malgré la survenance de fautes.

Il existe aussi d'autres méthodes importantes comme le modèle de porte probabiliste (PGM), le modèle de probabilité conditionnelle (CPM). Récemment, une approche hybride nommée SNaP avec une complexité linéaire et pouvant profiter d'émulation par FPGA a été proposée. La base de SNaP comprend deux concepts: la source de fautes et la propagation de fautes. La fiabilité du circuit est obtenue à partir de l'estimation du nombre de portes permettant la propagation d'une faute vers l'une des sorties primaires. Bien que le SNaP permet l'évaluation de circuits complexes, sa complexité matérielle n'est pas négligeable pour la mise en œuvre de FPGA.

III.b L'analyse des fautes multiples sur les circuits du CED

Les techniques du CED sont largement utilisés pour détecter les erreurs survenant au cours de l'opération d'un circuit. Les schémas de CED sont traditionnellement basés sur l'hypothèse de faute simple. Cependant, l'influence de fautes multiples n'est plus négligeable avec la réduction d'échelle de circuits jusque aux dimensions nanométriques. Peu d'analyse des schémas CED sont reportées dans la littérature et, souvent, celles-ci supposent que les fautes n'affectent pas la partie de contrôle.

Les approches analytiques comme le SPR, le PTM etc. ont démontré être utiles pour faire face aux fautes multiples. Pour les schémas CED, ce qui nous intéresse est la fiabilité fonctionnelle. Mais les approches analytiques se concentrent sur la *probabilité du signal* définie comme la probabilité que toutes les sorties du circuits soient correctes et ne peuvent pas être directement appliquées à l'analyse des schémas CED.

Soit un circuit du CED comme donné dans la Fig. 7, composé de trois blocs: une fonction cible F, une prédicteur F_p et une contrôleur F_c. La vérification de F_c peut entraîner une sortie d'un bit e. Dans le reste de ce résumé, il est supposé que $e = 0$ lorsque aucune erreur est détectée et $e = 1$ autrement. L'analyse de ce circuit montre qu'il peut produire quatre événements exclusifs:

E_1 : le contrôleur indique un fonctionnement correct et la sortie du circuit est correcte;

E_2 : le contrôleur indique une opération incorrecte et la sortie du circuit est incorrecte;

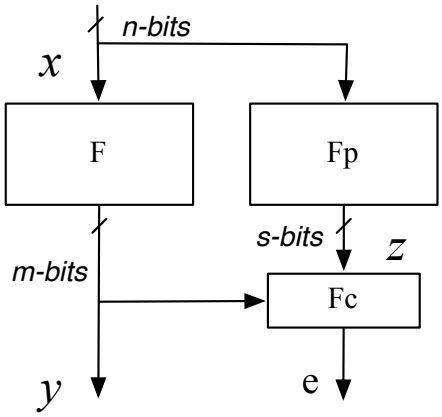
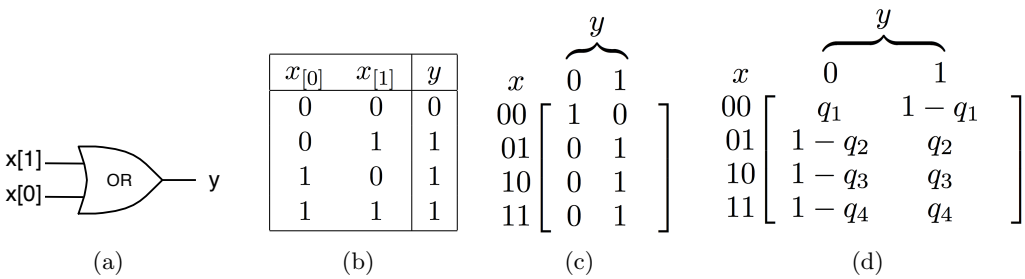


Figure 7: Schéma général de CED



$x_{[0]}$	$x_{[1]}$	y
0	0	0
0	1	1
1	0	1
1	1	1

$$\begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} y \\ \begin{array}{cc} 0 & 1 \end{array} \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array} \qquad \begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} y \\ \begin{array}{cc} 0 & 1 \end{array} \\ \begin{bmatrix} q_1 & 1-q_1 \\ 1-q_2 & q_2 \\ 1-q_3 & q_3 \\ 1-q_4 & q_4 \end{bmatrix} \end{array}$$

Figure 8: Porte logique OR: a. structure, b. table de vrit, c. ITM et d. PTM.

E_3 : le contrôleur indique une opération incorrecte et la sortie du circuit est correcte;

E_4 : le contrôleur indique un fonctionnement correct et la sortie du circuit est incorrecte;

La fiabilité fonctionnelle est définie comme la capacité d'un système ou d'un composant d'effectuer ses fonctions requises dans les conditions données pour une période de temps spécifiée, même en présence de fautes. Par conséquent, la fiabilité fonctionnelle d'un circuit du CED peut être comprise comme la somme des probabilités de produire uniquement les événements E_1 ou E_2. Par conséquent, la fiabilité fonctionnelle est exprimée en tant que:

$$R_{CED} = p(E_1) + p(E_2) \tag{1}$$

Nous proposons une méthode basée sur la technique PTM à évaluer cette fiabilité fonctionnelle. Considérons un bloc logique b avec l'entrée x et la sortie y, où $x \in \{x_0, x_1, \cdots, x_i, \cdots, x_{2^n-1}\}$ et $y \in \{Y_0, y_1, \cdots, y_j, \cdots, y_{2^m-1}\}$, sont des vecteurs binaires. La PTM du bloc b, noté PTM_b possède $2^n \times 2^m$ éléments et chaque élément (i, j) est la probabilité d'obtenir la sortie $y = y_j$ compte tenu de la occurrence de l'entrée $x = x_i$, notée $p(j|i)$. Dans le cas d'un bloc idéal (c-à-d sans faute), la PTM ne contient que des zéros et des uns, et est noté comme ITM (matrice de transfert idéale).

La Fig.8 montre la PTM d'une porte logique OR avec la probabilité d'erreur de $1 - q$, où q représente la probabilité d'obtenir une sortie correcte. Nous pouvons dire que l'ITM est définie pour $q = 1$. Pour calculer la PTM d'un circuit entier C, nous combinons la PTM de ses blocs de base en utilisant des produits

La fiabilité d'un bloc ou d'un circuit est directement extraite de ses PTM et ITM correspondantes, selon (2), où $p(i)$ représente la probabilité que l'entrée x est x_i.

$$R = \sum_{ITM(i,j)=1} p(j|i) \cdot p(i) \tag{2}$$

Afin d'appliquer l'approche PTM classique en analyse des circuits du CED, nous allons d'abord réécrire l'expression (1) pour montrer la probabilité d'occurrence des entrées et les sorties telle que définies précédemment. L'expression (3) est produite en faisant cela, où $p(i)$ signifie la probabilité de $x = x_i$ et $p(jE_1|i) + p(j \in E_2|i)$ et est liée à un bon fonctionnement du CED pour $x = x_i$. En d'autres termes, il représente la probabilité d'avoir une bonne $[e, y]_j$ étant donné que l'entrée est x_i.

$$\begin{aligned} R &= \sum_j \sum_i p(i) \left[p(j \in E_1|i)) + p(j \in E_2|i) \right] \\ &= \sum_j \sum_i p(i) \cdot PTM_{CED}(i, j) \cdot ITM_{CED}(i, j) \end{aligned} \tag{3}$$

Pour décrire cette idée, une porte OR est considérée comme la fonction F. Le ITM_{CED} correspondant est présenté dans la Fig. 9. Nous remarquons que la moitié gauche de

la matrice ITM_{CED} (colonnes pour $e = 0$) est le ITM_{OR}, tandis que la moitié droite (colonnes pour $e = 1$) est le complément logique de ITM_{OR}.

$$\begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} \overbrace{00 \quad 01}^{e=0} & \overbrace{10 \quad 11}^{e=1} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \end{array} = [ITM_{OR}, \overline{ITM_{OR}}]$$

Figure 9: ITM_{CED} pour une porte OR comme le F.

En analysant la ITM_{CED}, nous pouvons obtenir la nouvelle équation (4) de la fiabilité fonctionnelle comme suit.

$$\begin{aligned} R = & \sum_{j=0}^{(2^m-1)} \sum_{i=0}^{(2^n-1)} p(i) \cdot PTM_{CED}(i,j) \cdot ITM_F(i,j) \\ + & \sum_{j=2^n}^{(2^{m+1}-1)} \sum_{i=0}^{(2^n-1)} p(i) \cdot PTM_{CED}(i,j) \cdot \overline{ITM_F(i,j)} \end{aligned} \tag{4}$$

La complexité de de calcul peut être réduite par une analyse progressive de l'ensemble du circuit grâce aux probabailités conditionnelles. Cela conduit aux expressions (5) et (6).

$$p(E_1) = \sum_{j=0}^{2^m-1} \Big(\sum_{k=0}^{2^s-1} PTM_{F_c}((m \times j + k), 0) \cdot \sum_{i=0}^{2^n-1} \big(PTM_F(i,j) \cdot PTM_{F_p}(i,k) \cdot ITM_F(i,j) \cdot p(i) \big) \Big) \tag{5}$$

$$p(E_2) = \sum_{j=0}^{2^m-1} \Big(\sum_{k=0}^{2^s-1} PTM_{F_c}((m \times j + k), 1) \cdot \sum_{i=0}^{2^n-1} \big(PTM_F(i,j) \cdot PTM_{F_p}(i,k) \cdot \overline{ITM_F}(i,j) \cdot p(i) \big) \Big) \tag{6}$$

Le flot correspondant à l'approche proposée est présenté dans la Fig. 10. La première tâche de ce flot (*PTM core*) consiste à calculer les PTMs des sous-circuits F, F_p et F_c à partir de leurs netlists et les PTMs et ITMs élémentaires. Les PTMs de ces trois sous-circuits sont envoyées au *CED reliability estimator*. Il calcule la probabilité des événements E_1 et E_2 pour un y_j donné et doit être exécuté 2^m fois. Notez que le résultat de chaque itération est accumulé par l'*Accumulater* .

Le Tableau 1 montre la comparaison en termes d temps de l'exécution. Trois circuit différentes pour CED adoptés dans un additionneur parallèle à propagation de retenue (RCA) sont considérés: un additionneur de la duplication identique (noté RCA_dup), un additionneur diversifiée de la duplication (noté RCA_Div_dup) et un additionneur de prédiction de parité (noté que RCA_Parity). Nous avons comparé l'approche proposée avec l'injections de fautes qui est basée sur le PBR. Cette méthode peut aussi analyser les circuits du CED exposés à des fautes multiples. Les résultats obtenus démontrent la supérieurité de la méthode proposée, en particulier pour le circuit de duplication RCA_Dup.

La méthode proposée permet également d'étudier les événements E_3 et E_4. E_3 représente la sous-estimation d'un circuit lié à une pénalité de temps. E_4 concerne les erreurs silencieuses. D'un point de vue de la sécurité de systèmes, les erreurs silencieuses sont

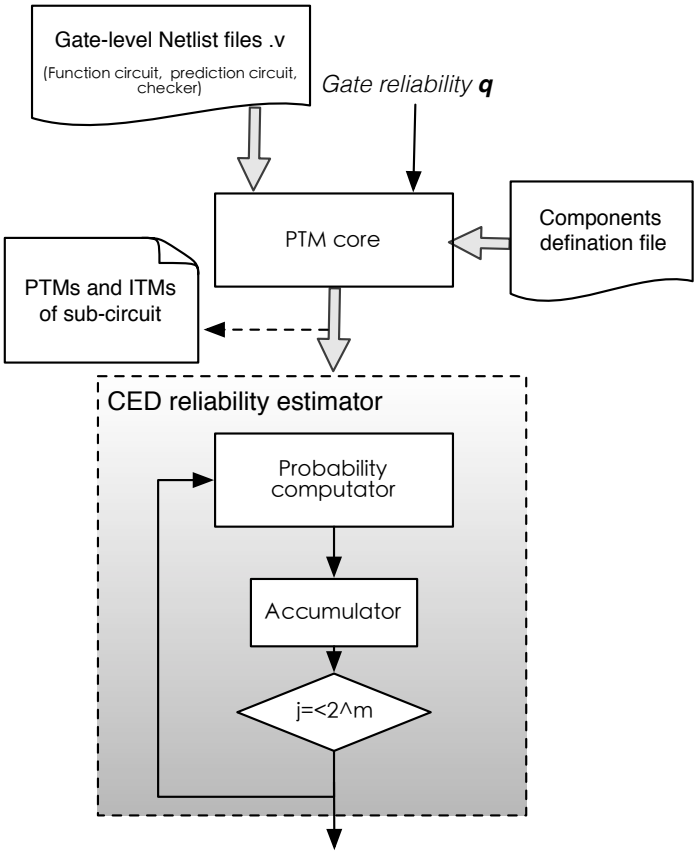
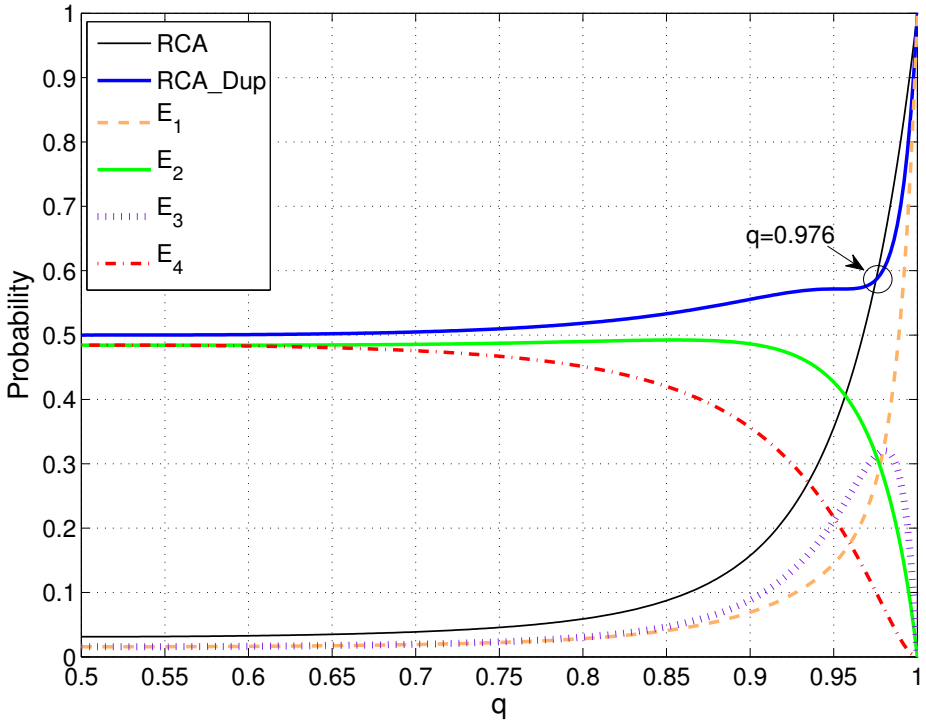


Figure 10: Flot de tâches de l'approache proposée.

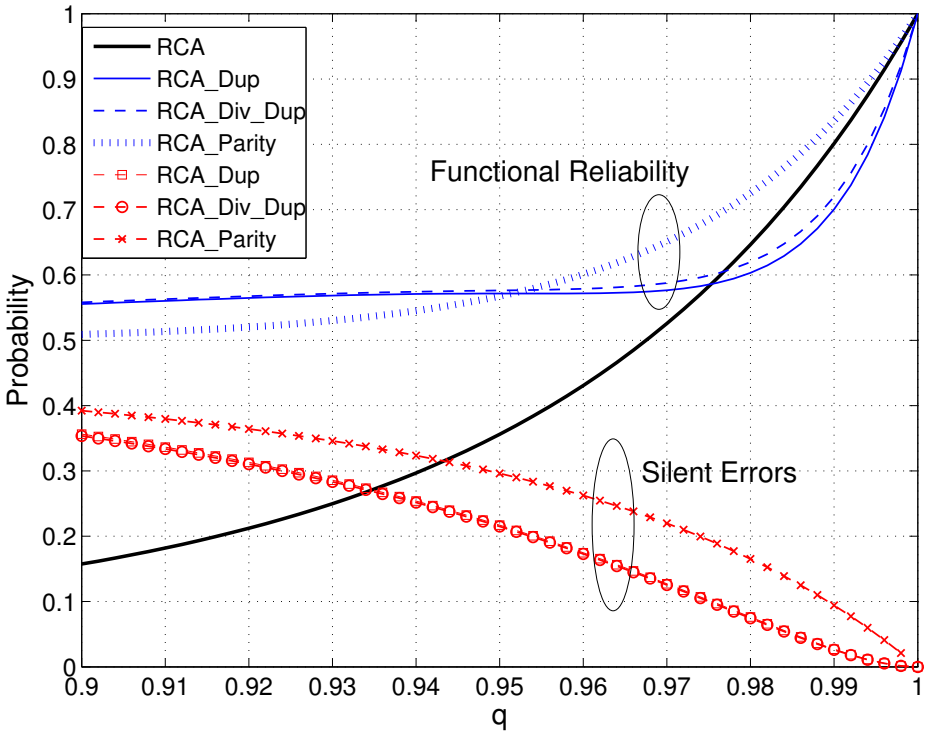
Table 1: Cot en temps (s) sur une station de travail Linux avec microprocesseur 2GHz AMD Athlon et 2GB RAM.

Type du circuit	**Méthode de l'évaluation**		**Nb. de portes**
	Injection de fautes	Proposé	
RCA_Dup	7398.1	18.6	86
RCA_Div_Dup	6345.7	19.3	78
RCA_Parity	367.2	4.7	32

un problème serieux car ils signifient défaillances potentielles sans l'avertissement. La Fig. 11(a) présente la probabilité des événements des sorties dans un circuit RCA_dup, où la ligne noire et la ligne bleu représentent les fiabilités de RCA et RC_dup, respectivement. La Fig. 11(b) est l'évaluation de la fiabilité fonctionnelle et la probabilité des erreurs silencieuses. La méthode de prédiction de parité montre une performance remarquable. Néanmoins, elle est très sujette aux erreurs silencieuses. Au contraire, les méthodes de la duplication permettent de faire face à de fautes multiples. Pour cette raison, elles sont des solutions utiles pour les applications nécessitant une haute sécurité, en dépit d'un surcoût de surface important.



(a)



(b)

Figure 11: a Evaluation des probabilités des événements des sorties en fonction de la fiabilité de la porte, b. Comparaison des méthodes de CED appliquées au RCA.

III.c L'analyse des effets de vieillissement

Les tâches de conception traditionnelles de CIs numériques comprennent la description comportementale (VHDL / Verilog), la simulation fonctionnelle, la synthèse de RTL et la simulation de post-layout. Afin d'étudier la dégradation des performances due la réduction des dimensions des composants CMOS, la fiabilité devrait être incluse dans ce flot de conception. Comme un chemin non critique peut devenir un chemin critique avec lle changement des temps de propagation en raison de l'effet NBTI, nous nous concentrons en particulier sur les effets du vieillissement sur le chemin critique et à proximité de chemin critique des circuits.

Nous proposons un flot de conception qui comprend l'estimation vieillissement tenant compte des deux mécanismes NBTI et HCI. Le flot de conception proposé est basé sur la génération de netlist de porte logique par la synthèse traditionnelle, comme illustré dans la Fig. 12. Un synthétiseur RTL traditionnel peut produire une description structurale de HDL pour un bloc numérique donné.

Selon les spécifications de conception, le *Behavioral modeling language* pour les circuits numériques (par exemple, VHDL et Verilog) est utilisé pour décrire les fonctions des circuits logiques. La conformité de la fonction est vérifiée par la *Functional simulation*. Le *Library statement* contient des fichiers de technologie nécessaires. Une liste d'interconnexions au niveau de grille est générée par le *RTL synthesis* et re-simulée par le simulateur de vieillissement, le *Aging simulator*.

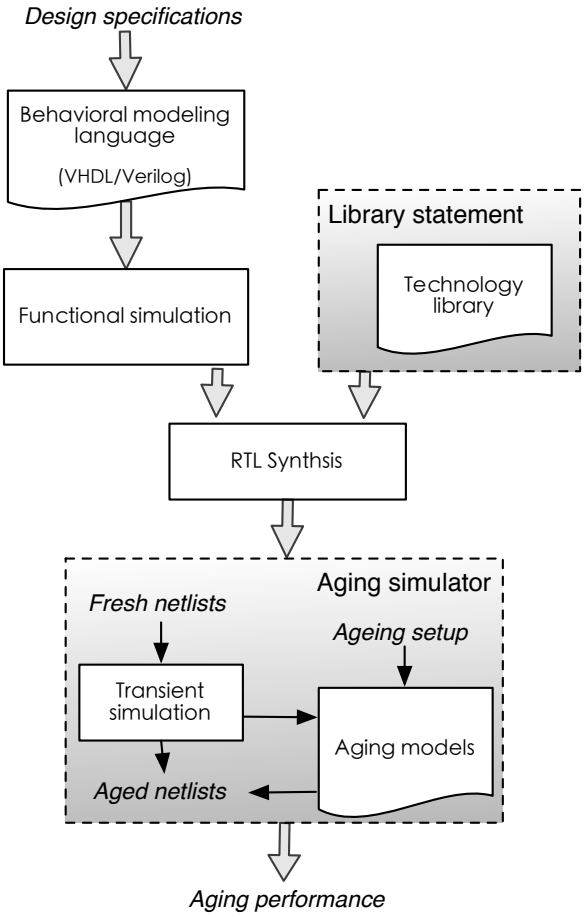


Figure 12: Flot de conception proposée.

Table 2: Comparaison de la performance des additionneurs (1)

Additionneur	**Surface (μm^2)**			**Délais (ps)**			**Puissance**		
Architecture	4	8	16	4	8	16	4	8	16
RCA	49.9	99.8	199.7	524.8	992.4	1926.6	2.71	5.42	11.08
CSA	96.7	193.4	386.9	372.7	600.7	1056.7	7.24	14.48	28.96
CLA	64.5	130,0	257.9	348.2	651.4	1257.8	4.71	9.41	18.82
KSA	64.0	160.2	402.5	350.4	532.5	601.2	3.87	9.15	21.96
SKA	57.7	132.1	302.6	416.6	550.2	819.6	3.59	8.03	17.75
BKA	57.7	125.8	268.3	417.3	687.7	911.6	3.96	8.79	19.42

Table 3: Comparaison de la performance des additionneurs (2)

Additionneur	**ADP (Normalizé)**			**PDP (Normalizé)**			**Dominant**
Architecture	4	8	16	4	8	16	**effect**
RCA	1.17	1.36	1.59	1.05	1.22	1.62	NBTI
CSA	1.61	1.60	1.69	1.80	1.97	2.25	NBTI
CLA	1.00	1.16	1.34	1.21	1.39	1.79	NBTI
KSA	1.00	1.17	1.00	1.00	1.10	1.00	HCI
SKA	1.07	1.00	1.03	1.10	1.00	1.10	HCI
BKA	1.07	1.19	1.01	1.22	1.37	1.34	HCI

Les modèles de vieillissement (NBTI et HCI) sont considérés disponibles sur le simulateur Eldo. Avec une initialisation de la configuration de vieillissement, une simulation transitoire est effectuée pour évaluer le stress sur chaque transistor. Puis, la netlist qui contient des informations relatives au composant “dégradé” est générée et peut encore être appliquée aux simulations de post-layout. Enfin, les rapports de synthèse relatifs aux synthèses “idéal” et “vieilli” peuvent être analysés.

Six additionneurs binaires ont été analysés: l’additionneur parallèle à la propagation de retenue (RCA), l’additionneur à la sélection de retenue (CSA), l’additionneur parallèle à la retenue anticipée (CLA), l’additionneur de Kogge-stone (KSA), l’additionneur de Sklansky (SKA) ,l’additionneur de Bren-kung (BKA). Les Tableaux 2 et 3 présentent les résultats de performance, coût et vieillissement. La technologie de 65nm de CMOS a été utilisée avec une temprature de 27°C et l’alimentation de 1.2 V. Comme on peut le voir dans le tableau, les impacts de NBTI et HCI sont différents selon l’architecture de l’additionneur. Les effets du vieillissement peuvent conduire à la dégradation de délais et diminuer le produit surface et délais (ADP). D’autre part, ils ont une influence positive sur la puissance et donc augmentent le produit délai-puissance (PDP).

La Fig 13 présente la comparaison des performances des différents additionneurs sous les effets du vieillissement en termes dégradation de délais. Dans cette figure, nous constatons que l’effet dominant dépend fortement de l’architecture. Les additionneurs rapides (c-à-d, KSA, SKA et BKA) sont plus vulnérables au HCI que NBTI alors que l’inverse est vrai pour les autre additionneurs. En comparaison avec les autres additionneurs, les additionneurs rapides peuvent être le meilleur compromis tenant compte à la fois de NBTI et HCl sous les conditons étudiées.

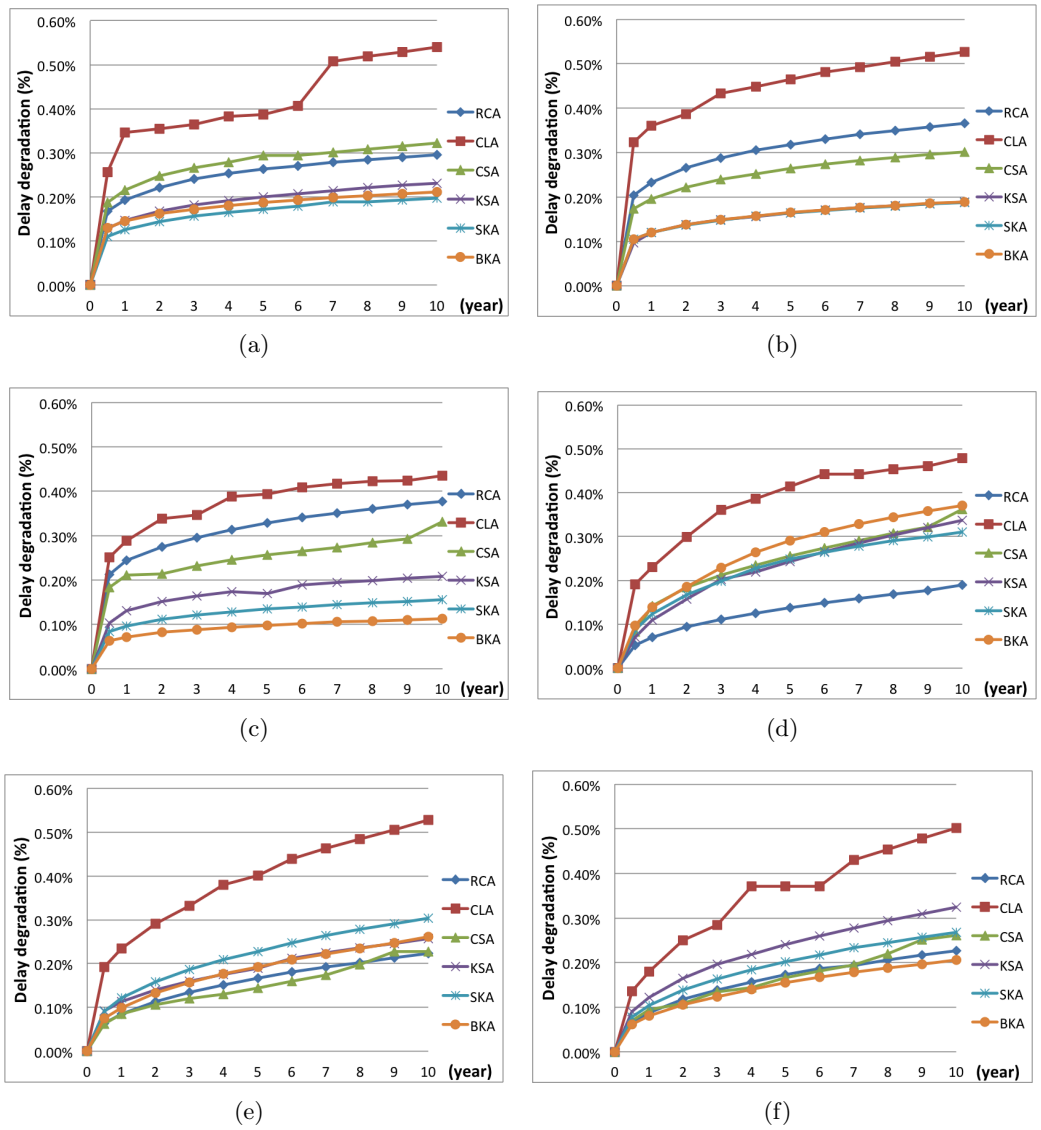


Figure 13: Comparison des performances des additionneurs en terme de la dégradation de délais pour différentes conditions: (a) 4 Bits, NBTI; (b) 8 Bits, NBTI; (c) 16 Bits, NBTI; (d) 4 Bits, HCI; (e) 8 Bits, HCI; (f) 16 Bits, HCI.

IV Conception efficace pour la fiabilité

Pendant les dernières décennies, des techniques de conception en vue de la tolérance aux fautes ont été largement rapportés. L'utilisation de la redondance produit un surcoût en termes de matériel ou de temps. Une conception devrait atteindre un bon compromis entre l'amélioration de la fiabilité et des surcoûts. D'autre part, bien qu'il existe des techniques génériques, des solutions ad hoc des applications spécifiques sont devenues de plus en plus nécessaires, du fait des contraintes en coût et en performance imposées par ces applications.

Le standard de chiffrement AES a été largement adopté dans les diverses applications critiques où une bonne fiabilité est exigée. Malheureusement, les problèmes de fiabilité des composants CMOS nanométriques peuvent limiter les performances de AES. Nous avons donc étudié la conception efficace de ces processeurs en nous concentrant sur l'amélioration de la S-Box, qui occupent les trois quarts de la surface dans un processeur d'AES.

IV.a. Le standard de chiffrement AES

Dans l'algorithme d'AES, les entrées et sorties sont constituées d'une séquence de 128 bits. Les données sont traitées sur l'unité de **Byte** (soit 8 bits). Chaque octet est représenté dans le corps de Galois $GF(2^8)$ avec un polynôme spécifié $(x_8 + x_4 + x_3 + x + 1)$. Les opérations internes sont exécutées sur des tableaux 4×4 d'octets appelés **State**. Un octet dans le State de S est noté $S_{r,c}$, où $0 \leq r, c \leq 3$. Les quatre octets de chaque colonne de State désignés comme un **Word** de 32 bits doté w_c. Sachant que la séquence d'octets d'entrée est $IN_0, IN_1, \ldots, IN_{15}$, les relations entre l'entrée Octets, Word et le State sont dans la Fig 14.

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

$\Rightarrow$

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$
w_0	w_1	w_2	w_3

Figure 14: Entrées et le tableau des States

L'algorithme d'AES est une norme de cryptographie symétrique itérative et chaque tour de chiffrement se compose de quatre transformations individuelles: *SubBytes* (S-Box), *ShiftRows*, *MixColumes* et *AddRoudKey*. Le décryptage est réalisé en inversant directement les transformations de cryptage comme le montre la figure 15, où Nr représente l'indicateur de tour. Le *roundKey* (i) est généré par un bloc appelé *key expansion*.

IV.b. L'efficacité d'une conception de tolérance aux fautes

Afin de caractériser les différentes techniques de tolérance aux fautes, nous définissons l'efficacité de l'amélioration de la fiabilité (noté η_t) par rapport au surcoût (7), où R_o est la fiabilité du circuit original, R_T représente la fiabilité du circuit enrichi d'une technique de tolérance aux fautes t et ΔC_T est le surcoût lié à cette technique.

$$\eta_t = \frac{R_t - R_o}{\Delta C_t} \tag{7}$$

Notez que ΔC_T peut inclure différents critères, en fonction de la stratégie de conception. L'équation (8) présente un exemple d'une mesure des surcoûts combinant le matériel, le temps et la consommation d'énergie. K_A, K_T, et K_P sont les facteurs de poids choisis par le concepteur en fonction des priorités de conception.

$$\Delta C_t = \Delta C_{A_t}^{K_A} \times \Delta C_{T_t}^{K_T} \times \Delta C_{P_t}^{K_P} \tag{8}$$

Nous nous concentrons sur le S-Box dans l'AES. La Fig. 16 montre l'efficacité de deux S-Boxes tolérants aux fautes étudiés: le TMR et le CED avec la prédiction de partie. La fiabilité des circuits est évaluée par l'approche PBR sous l'hypothèse des fautes transitoire. Deux cas de S-Boxes tolérants au fautes sont été considérés:

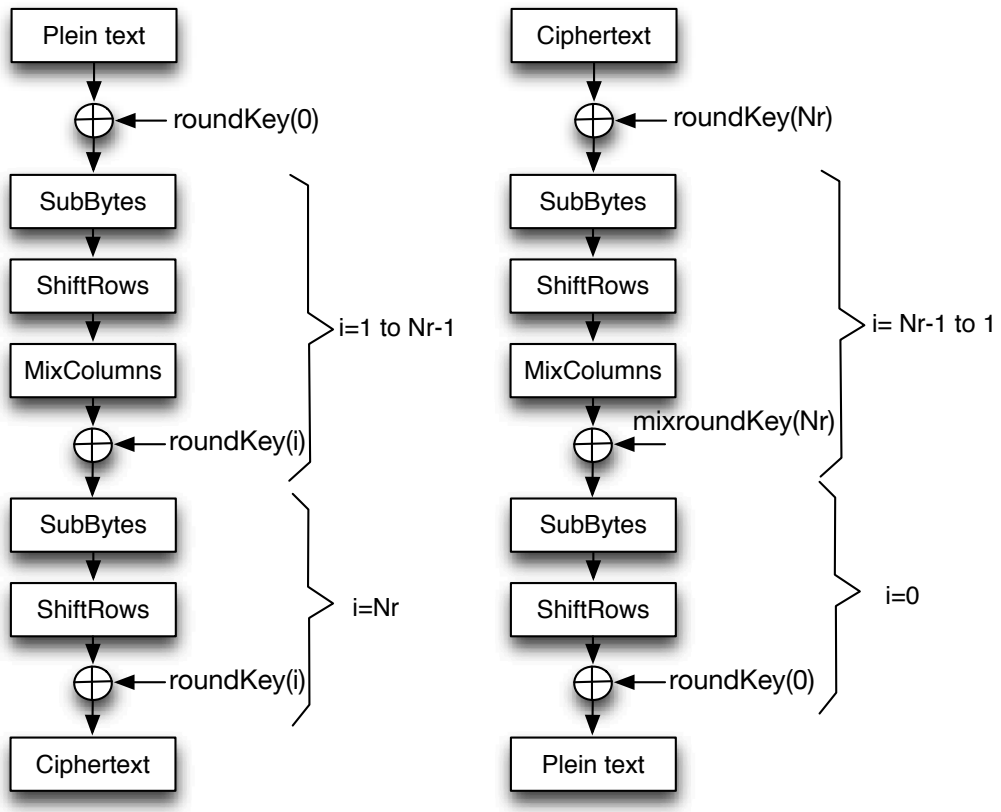


Figure 15: Diagramme de chiffrement et déchiffrement AES.

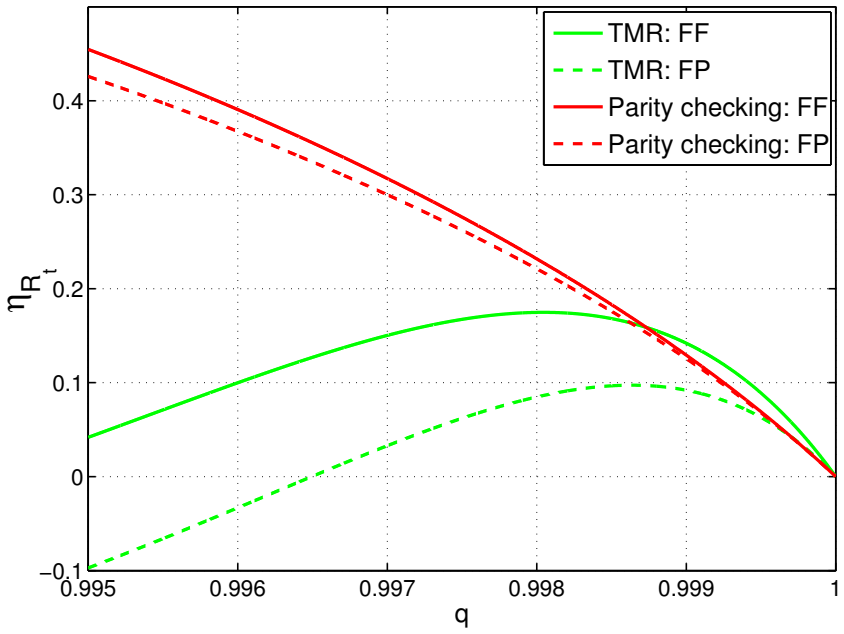


Figure 16: Efficacité de la solution d'amélioration de fiabilité($K_A = K_T = K_P = 0.5$).

- FF: Le voteur et le contrôle de circuit ne sont pas sujets aux de fautes.
- FP: Le voteur et le contrôle de circuit peuvent aussi être affectés par les les fautes

Les résultats de simulation montrent que le CED avec la prédiction de partie est une

approche très intéressante contre les fautes transitoires sur S-Box. Cette technique est efficace pour la plupart des cas étudiés. Toutefois, le CED avec la prédiction de partie est plus vulnérable aux fautes simples. Un faute transitoire simple sur une porte logique peut produire plusieurs bits faux à la sortie. Le CED avec la prédiction de partie peut détecter ces fautes si leur nombre est impair, mais les fautes en nombre pair restent indétectables.

D'autre part, le TMR reste toujours une bonne solution pour une application où la haute fiabilité est requise. Bien que le TMR soit conçu pour le masquage de fautes simples, les fautes multiples (si elles ont lieu sur le même module) sont également masquées par cette solution qui, en outre, traite aussi bien les fautes transitoires et les fautes permanentes.

IV.b.L'architecture fiable avec un surcoût faible pour les S-Boxes

Dans cette section, nous nous concentrons sur l'architecture compacte parallèle des processeurs d'AES, qui emploie 16 modules S-Box de 8-bits pour les calculs de *SubBytes*, comme présenté dans la Fig. 17. Les S-Boxes étudiés sont basés sur les portes logiques et requièrent une surface faible. Notre architecture proposée est applicable pour le chiffrement et le déchiffrement AES, indépendamment de la structure du S-Box.

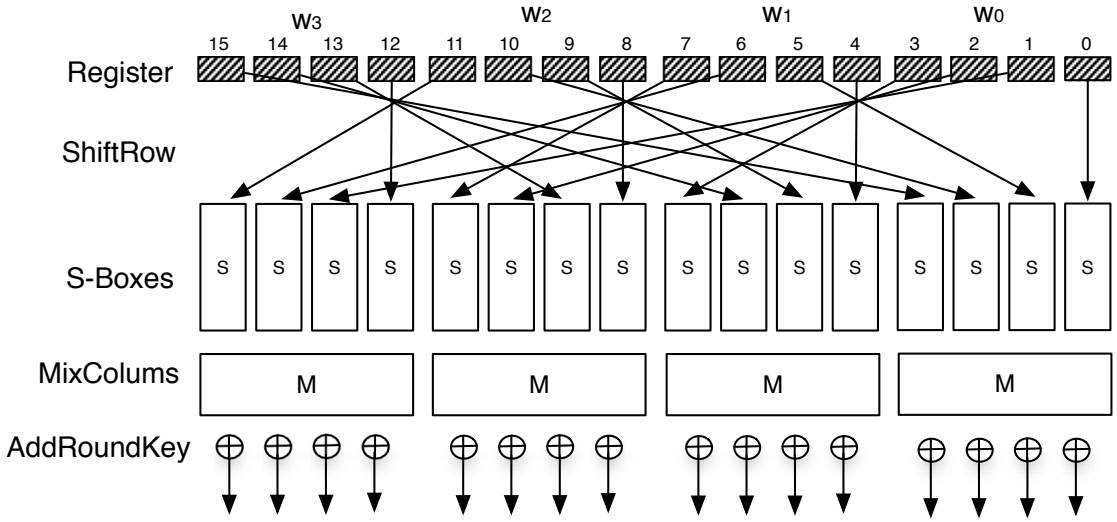


Figure 17: Architecture parallèle d'un seul tour de l'AES.

La base de l'architecture proposé est le tableau configurable de S-Boxes (CSBA) montré dans la Fig. 18. Cette partie contient quatre S-Boxes, deux blocs de configuration, 32 voteurs de la majorité d'un bit et certains registres. Le chemin de données d'entrée est un Word (c-à-d, 32 bits). Trois périodes d'horloge sont nécessaires pour effectuer un traitement complet. Comme on le voit dans le Tableau 4, les mêmes données d'octets sont traitées sur trois S-Boxes différents pendant trois périodes d'horloge. Prenez la donnée $S_{0,0}$ comme une exemple, sa transformation est effectuée sur des S-Boxes **A**, **B** et **C**. La configuration des S-Boxes à utiliser est accomplie en deux blocs qui comprennent uniquement des multiplexeurs, où les signaux de contrôle sont générés par une machine à états finis (FSM). Les sorties relatives aux différents cycles sont stockées dans les registres, et ensuite comparées par les voteurs. Cette architecture assure le masquage de fautes simples transitoires et permanents.

Pour un processeur d'AES avec chemin de données de 128 bits, quatre CSBAs sont nécessaires, donnant lieu à une architecture de configuration complète (FC).. Un inconvénient de l'architecture FC est son surcoût en material dû aux blocs de configuration supplémentaires et les registres.

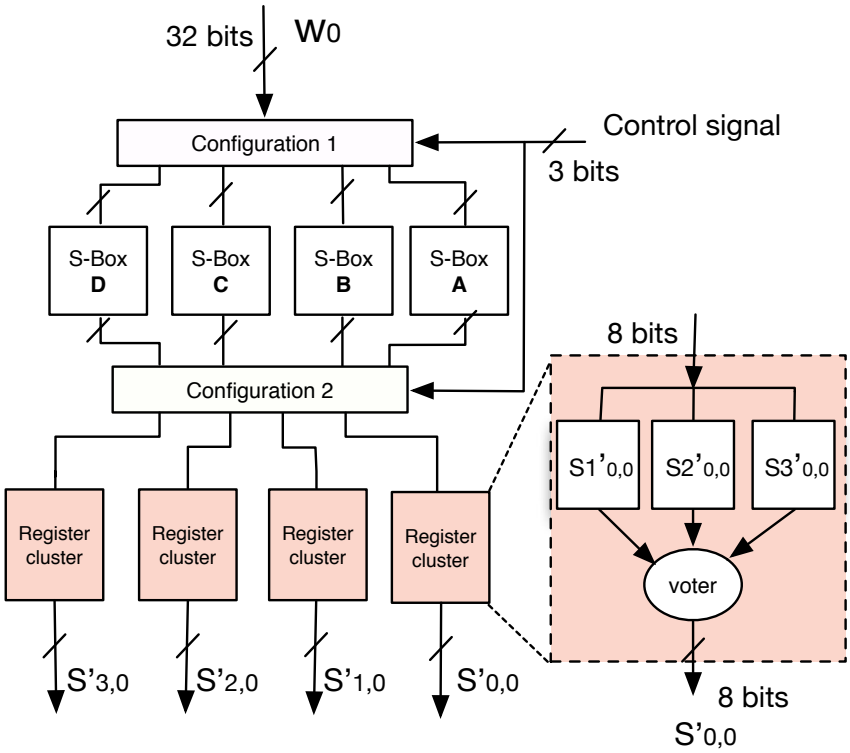


Figure 18: Diagramme du tableau configurable de S-Boxes (CSBA).

Table 4: Traitement de la reconfiguration

S-Box No.	*clock period*		
	1	2	3
A	$S_{0,0}$	$S_{3,0}$	$S_{2,0}$
B	$S_{1,0}$	$S_{0,0}$	$S_{3,0}$
C	$S_{2,0}$	$S_{1,0}$	$S_{0,0}$
D	$S_{3,0}$	$S_{2,0}$	$S_{1,0}$

Nous proposons donc une architecture hybride en profitant des redondances inhérentes aux S-Boxes identiques. L'idée est d'appliquer deux approches en même temps: le la configurabilité et le TMR. L'architecture hybride proposée (voir la Fig. 19) comprend deux blocs principaux: un tableau de cluster (CA) comprenant quatre groupes S-Boxes et un CSBA. Comme dans l'architecture FC, trois périodes d'horloge sont nécessaires pour accomplir un calcul complet de l'entrée $W_3W_2W_1W_0$. La procédure est indiquée dans le Tableau 5.

Comme on peut le constater dans le Tableau 5, le CSBA traite W_0 dans toutes les trois périodes d'horloge, tandis que W_1, W_2 et W_3 sont respectivement traitées par le bloc CA dans des périodes d'horloge différentes. En adoptant la TMR, la triplication de registres pour W_1, W_2 et W_3 n'est plus nécessaire et le matériel est donc économisé . Il faut noter que les multiplexeurs sont utilisés pour sélectionner le Word traité sur le bloc CA, et les registres sont encore nécessaires pour temporellement tenir la sortie.

Le Tableau 6 montre l'évaluation de la performance du matériel en utilisant la technologie du ST 65-nm CMOS. Deux S-Boxes distincts sont pris dans le cas d'étude: le S-Box de

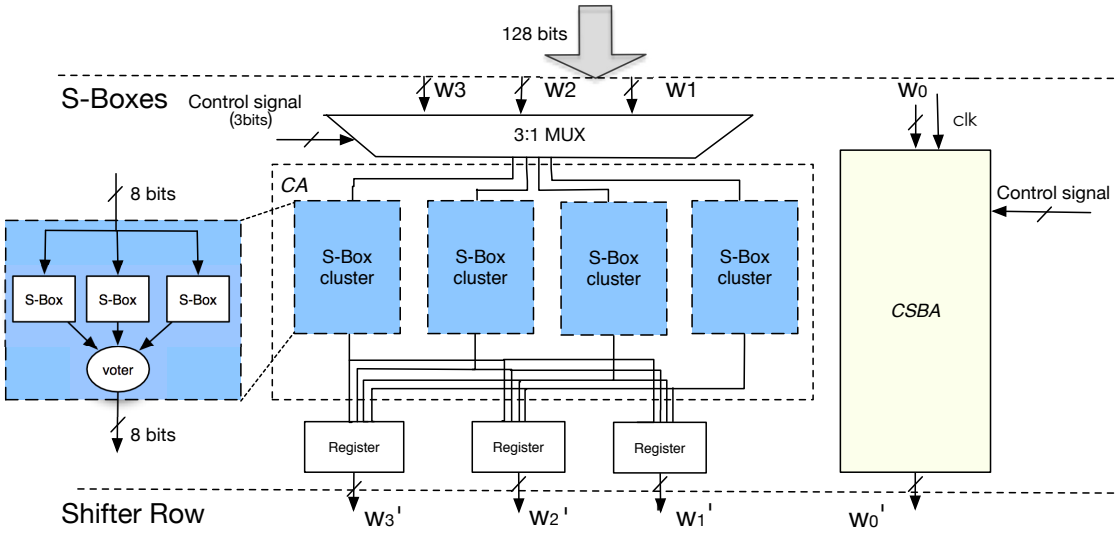


Figure 19: L'architecture hybride pour les S-Boxes.

Table 5: Le séquencement de l'exécution

Executed block	*clock period*		
	1	2	3
CSBA	W_0	W_0	W_0
CA	W_1	W_2	W_3

Table 6: Évaluation de performance du matériel (seulement S-boxes)

Architecture	**Base polynomiale**		**Base normale**	
	(μm^2)	(%)	(μm^2)	(%)
Original	6456.3	-	6647.7	-
TMR	19968.5	+209.3	20542.1	+209.0
TTR	10896.6	+68.8	11088.0	+66.8
Pédiction	7977.8	+23.56	8169.2	+22.9
FC	11709.9	+81.4	11892.9	+78.9
Hybrid	**8997.6**	**+39.4**	**9171.4**	**+37.9**

la base polynomiale (PBS-Box) et le S-Box de la base normale (NBS-Box). Nous pouvons remarquer que l'approche hybride proposée introduit un surcoût en surface qui ne dépasse pas 40%, ce qui est inférieur aux autres stratégies de tolérance au fautes mentionnées..

Le Tableau 7 (voir la fin de ce chapitre) montre les résultats obtenus avec l'application de l'architecture hybride des S-Boxes proposée pour le processeur AES. Nous pouvons constater que le surcoût en surface est de seulement 15%. Ces résultats montrent également que cette architecture est particulièrement adaptée pour le S-Box de la base polynomiale.

La Fig. 20 montre comment la fiabilité globale du S-Box change en fonction de la fiabilité de S-Box. Nous pouvons reconnaître que sans tenir compte de l'effet des multiplexeurs et des registres, la solution proposée est encore meilleure que l'approche TMR.

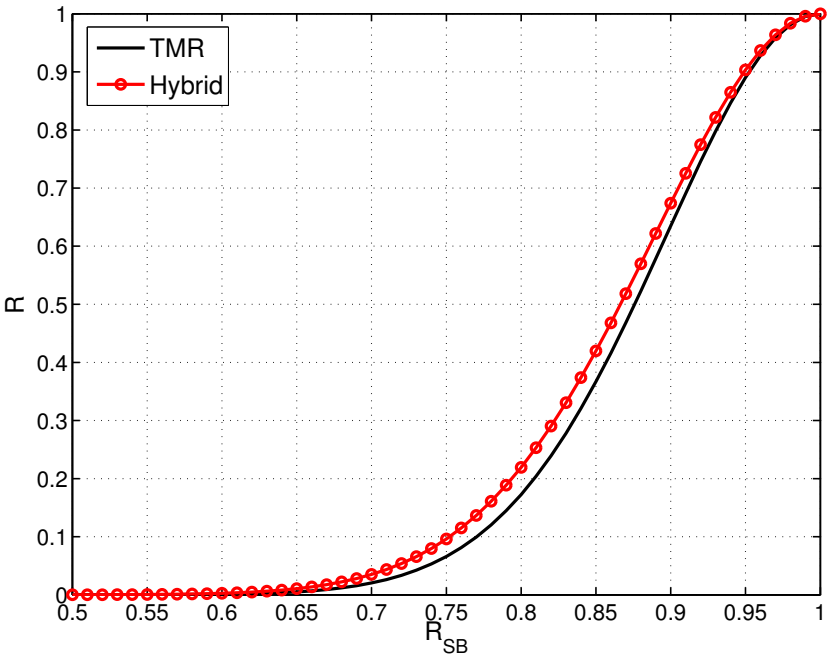


Figure 20: Comparaison des fiabilités pour le processeur d'AES.

V Conclusions

Cette thèse a traité les problèmes de la fiabilité induites par la réduction géométrique continue de la technologie de CMOS. Nos discussions ont porté sur les mécanismes dominants de vieillissement NBTI et HCI, les effets liés aux rayonnements ainsi que le bruit thermique. Les travaux ont été effectués pour atteindre une bonne amélioration de la fiabilité tout en limitant les pénalités (c-à-d, la surface, le délais et la puissance).

Une grande partie de ce travail a été dédiée à l'analyse de la fiabilité en termes de capacité de tolérance aux fautes. Considérant la propriété du masquage logique sur les circuits combinatoires, deux méthodes de l'évaluation ont été proposées. En plus de l'analyse de la tolérance aux fautes, la méthodologie pour l'étude des effets du vieillissement a été aussi discutée. D'autres efforts ont visé l'évaluation de l'efficacité des solutions d'amélioration de la tolérance aux fautes. Nous avons introduit une méthode générale pour caractériser les différents systèmes tolérants aux fautes tenant compte de l'amélioration de la fiabilité et les surcoûts engendrés par cette amélioration. Une architecture hybride intégrant la redondance matérielle et temporelle a ensuite été proposée pour améliorer les processeur d'AES.

Table 7: Performances en termes de surface et de délais pour les processeurs d'AES appliqués la structure proposé

S-Box	**Surface** (μm^2)			F_{Max} (MHz)			**Throughput** ($Mbps$)			**Optimization**
	Original	Proposé	(%)	Original	Proposé	(%)	Original	Proposé	(%)	
Base polynomiale	9632.0	11031.8	**+14.5**	307.4	255.6	-16.84	3933.9	1090.5	-72.3	surface
	11432.2	24341.2	+121.9	393.1	409.2	**+4.1**	5031.5	1745.8	**-65.3**	vitesse
	10254.4	13401.4	+30.7	328.0	277.6	-15.4	4198.1	1184.2	-71.8	default
Base normale	9482.2	11158.7	+17.7	276.7	239.4	-13.5	3515.3	1021.2	-70.9	surface
	14473.2	23718.2	+63.9	500.0	379.4	-24.1	6400.0	1618.6	-74.7	vitesse
	11440.0	13576.7	+18.7	271.7	247.3	-9.0	3477.6	1055.3	-69.7	default

Chapter 1

Introduction

1.1 Motivation

After 50 years of Moore prediction [1], CMOS technology contributes to every human activity from personal cell phone to satellite. The continuous decreasing of CMOS geometrical dimension enables the increasing number of transistors on a single chip. Integrated circuits (ICs) today are faster, smaller, more powerful and cheaper to be fabricated than ever before [2]. International technology roadmap for semiconductors (ITRS) predicts that the gate length of electronic devices can reach beyond 10 nm in 2025 [2]. Meanwhile, on-chip clock frequency will increase to higher than 6 GHz along with the supply voltage equaling to 0.65V, as shown in Table. 1.1 [2].

Although the CMOS scaling drives an enormous growth of semiconductor industry in past years, CMOS technology approaches to the physical limits. The design and manufacture are becoming more complex and ICs will be very hard to be fabricated exactly as expected. Even worser, as the supply voltage is reduced by a large margin, ICs, in particular, digital ICs (i.e., CPU, Memory, Logic) are more sensible to the environment noise. The suffering from low reliability is foreseeable for future ICs due to permanent or transient faults. Permanent faults mainly arise from the limitations induced by fabrication technologies such as defects and variations in the manufacturing procedure. Transient faults mainly arise from the sensibility to environment noise like thermal noise and radiation-related effects that can randomly interrupt the correct operation.

Table 1.1: 2013 ITRS - DRAMs, MPUs, and ASICs [2]

Parameter	**Year of production**							
	2014	2016	2018	2020	2022	2024	2026	2028
Flash $\frac{1}{2}$ pitch(nm)	17	14.2	11.9	11.9	11.9	11.9	11.9	11.9
DRAM $\frac{1}{2}$ pitch(nm)	26	22	18	15	13	11	9.2	7.7
MPU/ASIC mental 1 $\frac{1}{2}$pitch (nm)	31.8	28.3	22.5	17.9	14.2	11.3	8.9	7.1
MPU physical Gate Length (nm)	18.4	15.3	12.8	10.65	8.87	7.39	6.16	5.13
ASIC physical Gate length(nm)	21	17.5	14.6	12.1	10.1	8.43	7.03	5.86
V_{dd} High performance circuits (V)	0.85	0.81	0.78	0.75	0.72	0.69	0.66	0.64
On-Chip clock (GHz)	5.72	6.19	6.69	7.24	7.83	8.47	9.16	9.91

Reliability related issues during usage phase derive from any one of the following sources:

- **Time dependent effects**, also called aging-effects, involve hot carrier injection (HCI) [3], negative bias temperature instability (NBTI) [4], time-dependent dielectric breakdown (TDDB) [5], electromigration (EM) [6], etc. They can result in physical change of transistor parameters and make the circuit malfunction during its usage phase. HCI and NBTI manifest themselves mainly as the variation of threshold voltage in n-MOSFET and p-MOSFET, respectively. These variations can accumulate through logic gates and degrade the device performances. Furthermore, NBTI effects will be intensified at high temperature environment [4]. TDDB effects reflect a temporary random oxide damage induced by defects generated in gate dielectric, while EM effects are found in interconnect and caused by the excessive current density stress which finally induce the open connections or short circuits [7].

- **Environment dependent effects** have become a new challenge due to the increasing sensibility of ICs to environment noise. The noise margin of nano devices is continuously reducing with the supply voltage scaling down [8]. The source of noise can be internal or external. Internal noise such as thermal noise and cross talk noise between lines makes the transistors and devices working in a noisy signal environment. In most cases, external noise refers to radiation-related effects which have been considered only in critical applications (e.g., space or nuclear) for a long time. In recent years, the observation of radiation induced transient faults at sea level motivates more and more researches on these effects [9, 10]. The main reason is that high density of integrated transistor can increase the probability of occurring the particle hits. Moreover, the low threshold of deep sub-micron devices reduces the critical charge required to change the transistor state. Consequently, a radiation hit is more easier to affect the function of transistors and cause bit-flips.

As defined by ITRS, the methodologies targeting to these challenges can be divided in two groups, that are well known as *More Moore* and *More than More* [11]. The first group aims to search new materials with low-resistivity conductor, strained Si and low-K dielectrics [12]. Some attempts are also carried out to modify the transistor structure, including transport-enhanced MOSFETs [13], silicon on insulator (SOI) [14] and double gate CMOS (DGC) [15]. The other group aims to interact with outside world. It means to investigate and incorporate with new devices that do not necessarily scale with Moore's Law. Carbon nanotubes [16], nano wires [17], single electron transistor based new devices are wildly reported[18]. Quantum cellular automata (QCA) is also discussed as a new technique [19]. Different methodologies strongly affect the fabrication processing, and require high manufacturing techniques.

Another well-known solution is the use of fault-tolerant approaches based on redundancy [20]. The reported literature covers fault-tolerant mechanisms in various levels of abstraction from electrical level (i.e., set of transistors) to system level (i.e., set of processors). The improvements on transistors locally address the problem thereby can achieve an outstanding performance in reaction time. However, this represents a significant relevance with technology. Moreover, the devices may be over-protected where the insensitive parts are hardened as well. On the other hand, the enhancement on system level (e.g., software recovery) exhibits a great independence from the technology and manufacturing. Since the application has been specified in system level, the designers can concentrate on critical parts. Whereas, the improvement on system level can be very consuming in the reaction time and complexity due to a great quantity of procedures.

A compromise is to enhance on architectural level. Concentrating on basic operators, designers have a big flexibility to propose suitable solutions according to the problems to

be addressed. The fault-tolerant mechanisms on this level must be self-controlled. In other words, they should be capable of either indicating the fault when the fault takes place, or masking the fault locally and giving the correct output. In the first case, it can be achieved by the error detection, while error masking is used in the second case. A good solution must result in a small hardware overhead and slight performance penalty.

The objective of this thesis consists of two aspects: reliability analysis and reliability-aware methodologies for digital IC designs on architecture level. The scope of the thesis focuses on reliability issues in usage phase, such as aging effects, radiation related effects as well as thermal noise. The variability-induced concerns are not involved in the discussions. We devote to explore the digital arithmetic operators, particularly adder/subtractor which are fundamental elements in computer science and widely adopted in digital ICs such as microprocessors, digital signal processors (DSP), and data-processing application-specific integrated circuits (ASICs). These arithmetic operators play a crucial role directly relevant to system speed, area, power dissipation as well as the reliability. Besides the basic arithmetic operators, processors implementing the Advanced Encryption Standard (AES) are also covered in this thesis.

1.2 Report organization

The organization of this report is as follows.

Section 2 reviews the basic concepts of reliability and faults. The classification of faults and the relevant sources are introduced. It presents the sort of reliability issues studied in this thesis including radiation related effect, NBTI, HCI and thermal noise. Well-known fault-tolerant techniques are mentioned in this chapter as well.

Section 3 presents the methodologies for reliably assessment. A state of the art of methods dealing with fault masking property on combinational circuits is reviewed. Both single and multiple transient faults impacts are discussed through the proposed analytical methods. Concerning aging effects, an aging-aware digital design flow that allows to study the two main aging mechanisms (NBTI and HCI) is introduced.

Section 4 contributes to the effective designs for reliability where the fault-tolerant design are studied. Firstly, it shows a method to characterize different fault-tolerant techniques with respect to reliability. S-Boxes on AES processor are taken as a case study. After that, a fault-tolerant structure is designed to deal with all single transient and permeant faults on S-Boxes. The proposed structure is compared with classical fault-tolerant schemes in terms of hardware overcost and reliability improvement.

Section 5 concludes this report and presents the perspectives and future works, where the contributions of this thesis are also mentioned.

Appendix A details the studied binary adders, while a review of finite field operations used in AES processors is available in Appendix B.

Chapter 2

Basics on reliability

Despite the continuous increase in performance, emerging ICs built from advanced technologies are expected to suffer from a dramatical reduction in reliability. This has became a new concern that limits the technology evolution in future and should be considered as an important factor in IC designs henceforth. The reliability issues in modern design are root either in process variation or environment noise. They can be quite time-dependent. In order to cope with the reliability challenge in ICs, researches and developments of new electronic devices are worldwide reported [21–23]. Among existing methods, fault-tolerant techniques are well known for dealing with the presence of uncertainties [20, 24].

This chapter briefly recalls the preliminaries on reliability including the related concepts and metrics. Different sources of reliability related issues are illustrated in detail. In addition, we highlight some famous fault-tolerant schemes that will be further studied in this thesis. Our scope comprises the aging effects, radiation-related effects and thermal noise, which are considered as important challenges during the usage of circuits [25].

2.1 Reliability and faults

2.1.1 Metrics of reliability

Reliability is a time dependent characteristic that reflects the continuation of a correct operation. IEEE defines the reliability as the ability that a system or component performs its proper function during a specified period $(0, t)$ under a specified conditions, denoted as R or $R(t)$ [26]. A system fails when the provided operation is different from what is expected due to the existence of faults. Let time T be as a random time that a system failure occurs. Define $F_T(t)$ as the probability of a failure occurring in the interval $[0, t]$, such that:

$$\begin{aligned} F_T(t) &= P(T \leq t) \\ &= \int_{-\infty}^{t} f(t)dt \end{aligned} \tag{2.1}$$

The function $f(t)$ is the probability density of the random variable representing the time to failure. Reliability which indicates the probability of no failure in $[0, t]$ is then expressed as follows [27]:

$$\begin{aligned} R =& P(T > t) \\ &= 1 - F_T(T \leq t) \end{aligned} \tag{2.2}$$

The *failure rate* represents the probability of system failure in a given time interval. In most of device lifetime, failure rate is regarded as constant. Under this assumption,

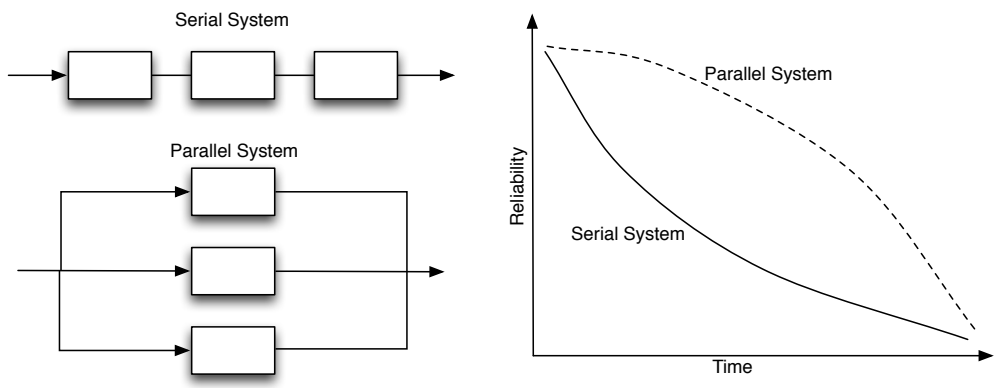


Figure 2.1: System architecture examples and system reliabilities with respect to time.

the reliability is modeled as an exponential distribution based on the failure rate λ, as expressed in (2.3).

$$R = e^{-\lambda t} \tag{2.3}$$

The reliability of a system can be obtained by combining the reliability of basic components. Indeed, many systems have the similar architecture configuration: serial or parallel. In a serial system, any malfunction of a single component will cause a system failure. Whereas, in a parallel system, the system fails only when all the components do not work. The reliability of a system can be abstracted from each component according to the architecture configuration as expressed in (2.4) and (2.5), where R_i represents the reliability of component i. As can be seen from Fig. 2.1, the serial system is less reliable than that in parallel when the same constituted components for both are assumed.

$$R_{parallel} = 1 - \prod_{i}^{n}(1 - R_i) \tag{2.4}$$

$$R_{serial} = \prod_{i}^{n} R_i \tag{2.5}$$

ITRS mentions other two traditional measures [2]:

- **Mean time to failure** (MTTF) is described as the expected time to failure, in other words, the expected lifetime of device, as expressed in (2.6). It can be also driven from failure rate as can be seen in (2.7).

$$MTTF = \int_0^\infty R(t)\ dt \tag{2.6}$$

$$MTTF = \frac{1}{\lambda} \tag{2.7}$$

- **Failure in time** (FIT) represents the number of failures in 10^9 device hours as shown in (2.8).

$$FIT = \frac{10^9}{MTTF} \tag{2.8}$$

The bathtub curve is widely accepted for describing the failure rate (see Fig. 2.2). The failure rate in early life period is mainly related to defects during manufacturing and can be very high. The defective devices are normally detected by tests. Once defective devices

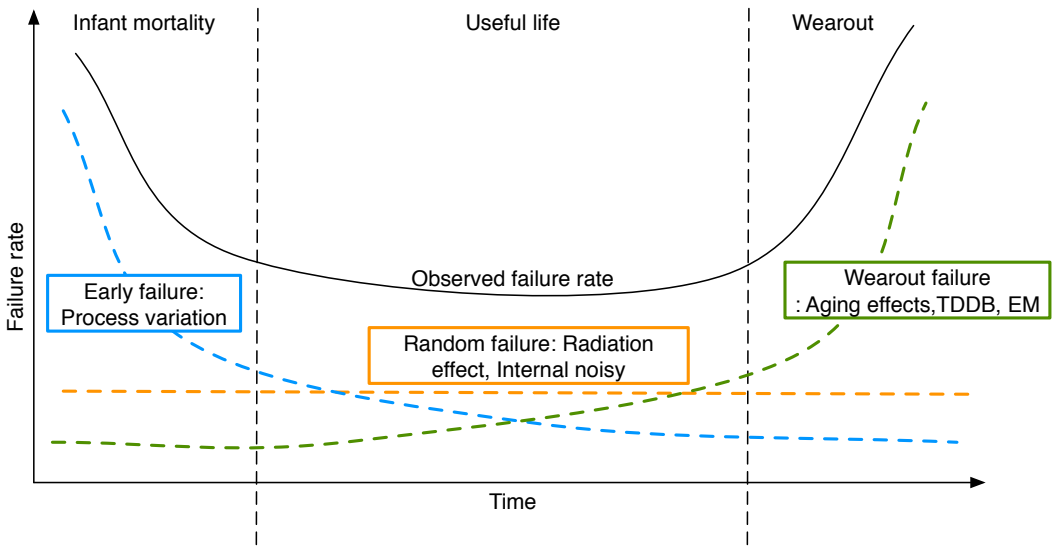


Figure 2.2: Bathtub curve. Notice that the time axis is not scaled.

are discarded, the failure rate will keep constant over major life period referred as working life. The device wearout is due to aging effects which dramatically increase the failure rate. Furthermore, the manifestation of random faults induced by environment (internal or external) will further raise the failure rate during device life period.

2.1.2 Faults classification

Typically, the cause of a *fault* can be a defect during manufacturing, a physical failure, a design mistake or environment noise. A fault may propagate and generate an internal erroneous signal. An *error* then is defined as the manifestation of a fault and may successively cause another error.

Faults potentially threaten the system and may result in the system failure if they reach the primary outputs of system. We can say that a fault first results in an error in internal state of the systems and then leads to an error in external state that finally leads to the deviation from correct service. A fault is "active" when it causes an error, otherwise, it is "dormant" [28]. Fig. 2.3 describes the flowchart of fault/error propagation.

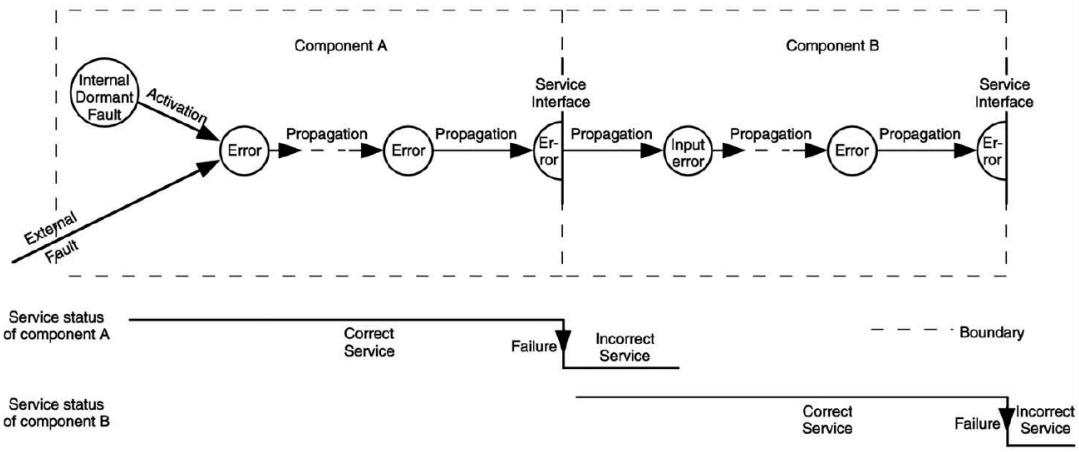


Figure 2.3: Fault/error propagation [28].

The basic classification of faults includes [20, 29]:

- **Permanent faults** which are due to irreversible changes in circuit structure, occur and remain stable until a repair is undertaken. These faults mainly arise from defects or the variation in manufacturing process. Permanent faults induced by fabrication process are usually well eliminated by appropriate design and manufacturing methods and can thus be ignored in usage time [29]. But sometimes, permanent faults also occur during the usage of device, such as aging-effects leading to device wearout.

- **Transient faults** or *soft errors* (SEs) which are caused by temporary environmental conditions like alpha particles and neutron, electrostatic discharge, thermal noise, crosstalk, etc. They occur for a short period of time and then disappear. In other words, transient faults are described as independent one-time errors. A transient fault can cause component malfunction without damaging the component itself. In memory cells, transient faults are referred as *single event upsets* (SEUs) while in a logic circuit are *single even transients* (SETs). The random occurrence and complex modeling character make transient faults representing a new different challenge.

- **Intermittent faults** which are induced by variations in manufacturing procedure. They are characterized as repeated and sporadic occurrence, sometimes with burst behavior. Intermittent faults affect circuits/devices with parametric variations, that can act as shorts or opens under certain conditions (noise, hazards, aging, etc), and may become permanent faults. Some intermittent faults arise only for certain input patterns and the circuit works properly in most of inputs cases. In order to repair the intermittent faults, the way is to remove or replace the faulty circuits [29].

Fig. 2.4 summarizes the source of faults according to fault type. Among these fault sources, the aging-related permanent faults and radiation-related effects induced transient faults are the main source of error experiences suffered by ICs in usage phase [29]. Another important threat source is the thermal noise due to the continuous shrinking in circuit voltage, where high fault probability can be foreseen [25]. Thereby, the discussions in this thesis are restricted to aging effects, radiation-related effects and thermal noise. Other mechanisms are considered as cumulative sources.

Fault modeling is an abstract description of fault's properties caused by physical phenomena. They can be modeled in different levels basically including: physical level, electrical level, gate level, register transfer level (RTL) and system level [30]. In this thesis, our discussions focus on gate level modeling.

2.2 Fault sources

2.2.1 Aging effects: NBTI and HCI

NBTI and HCI are two major aging degradation mechanisms in nanometer CMOS technology [3, 4]. First reported in [31], NBTI refers to the generation of positive oxide charge and interface traps at the Si-SiO_2 interface in CMOS transistor under negative gate bias, in particular at elevated temperature. NBTI impacts only PMOS transistors because only PMOS is under a uniform negative bias during CMOS operation [32]. Figure 2.5(a) depicts the channel holes interactions according to the passive hydrogen bonds in the dielectric, that leads to the generation of positive oxide charge and interface traps.

The impact of NBTI is always considered as a threshold voltage increasing after a negative bias is applied. The feature of NBTI-induced degradation could be modeled as expressed in (2.9) [32], where E_0 and E_a indicate process-dependent constants. In this equation, k and E_{ox} are Boltzmann constant and oxide field, respectively.

$$\triangle V_{th} \sim exp(\frac{E_{ox}}{E_0}) \cdot exp(\frac{-E_a}{kT}) \cdot t^n \tag{2.9}$$

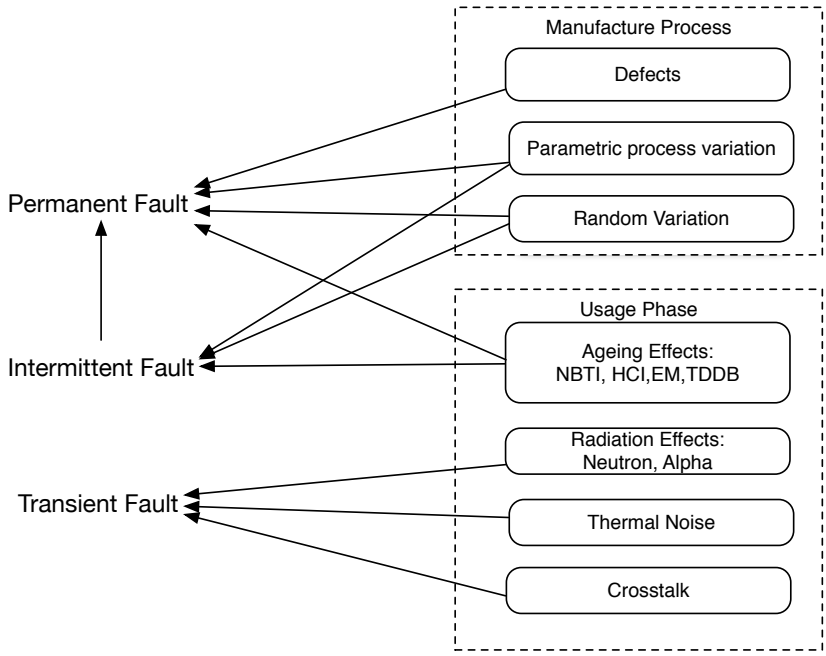


Figure 2.4: Fault sources for different fault types.

HCI mainly appears on NMOS type transistor, when hot electrons overcome the potential barrier and inject into gate oxide due to high lateral electric field near the drain as described in Fig. 2.5(b). HCI impact includes the threshold shift and degradation of carrier mobility as well as a change of output resistance. The impact of HCI is still very important due to the higher electric filed in the gate oxide, higher operating temperature and voltage amplitude [33].

As shown in Fig. 2.6(a), HCI can influence sub-threshold swing coefficient (n_{fac}), intrinsic threshold voltage (v_{th0}), intrinsic mobility (μ_0), saturation velocity (v_{sat}), drain source resistance per width (r_{dsw}), subthreshold region DIBL coefficient (eta_0) and threshold voltage offset (v_{off}). Besides, the only BSIM4 [34] parameter affected by NBTI is v_{th0} of PMOS transistor (see Fig. 2.6(b)). More severe v_{th0} degradation is observed when PMOS transistor works at high temperature ($150°C$) [35]. These physical parameter degradations can lead to circuit level degradation. In particular, this thesis focuses on delay degradation.

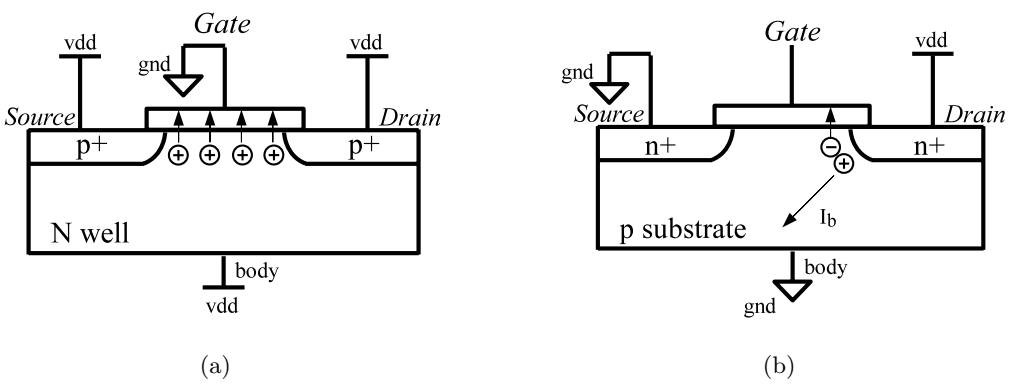
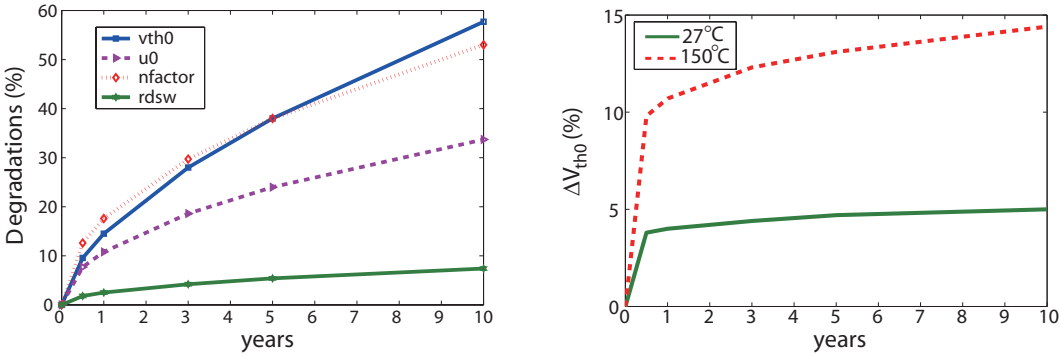


Figure 2.5: HCI and NBTI mechanisms at CMOS device level: a. Hot carriers inject into the dielectric at the drain end of NMOS type transistors; b. NBTI causes positive oxide charge and interface traps in PMOS type transistors.



(a) (b)

Figure 2.6: HCI and NBTI induced physical parameters degradation in: a. NMOS and b. PMOS transistor [36].

2.2.2 Radiation related effects

For a very long while, radiation effects have been considered as a reliability concern only in harsh environment applications as in space or in nuclear. The work of Binder *et al.* demonstrated that the radiation is capable to cause soft errors on electronic devices at sea level [37]. Today, many of works are carried on in order to investigate the impact of radiation effects at sea level. These researches show that radiation-related effects mainly associate to two sources:

- **High energy neutrons** generated from cosmic rays interacting with the earth's atmosphere.
- **Alpha particles** emitted by radiative impurities in low concentration in packing and interconnect materials.

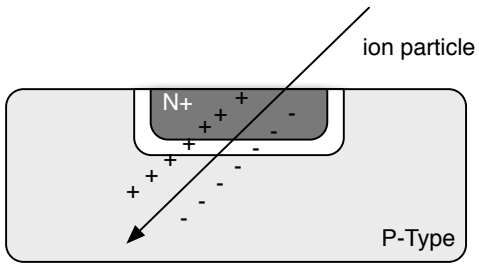
When an ion particle hits on the sensitive part of an electronic device (mainly the reverse biased junction), the particle will free hole-electron pairs, and loss its energy. This phenomena is called *linear energy transfer* (LET) and is followed by charge collection that generally takes place within two junctions. The collected charge, noted as Q_{coll}, will generate a short time pulse of current I_{in}. Collected charge is expressed by current I_{in} as in (2.10).

$$Q_{coll} = \int I_{in}(t)dt \tag{2.10}$$

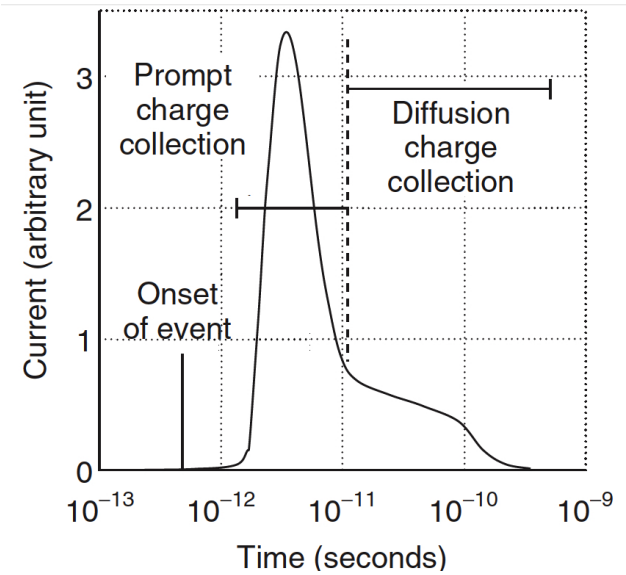
Take a reverse biased junction shown in Fig. 2.7(a) for example, when an ion particle hits on it, three phases of charge collection convectively take place [38]. In the first phase, the ionizing particle arouses to form a cylindrical track electro-hole pairs and high carrier concentration. Then, the carries are quickly collected by the electric filed when this ionization track closes to the depletion region. This collection of carriers generates a glitch of current/voltage at the node and will last tens of picoseconds. In the third phase another charge collection takes place when electrons diffuse into the depletion. This charge will continue for hundreds of nanoseconds. According to these phases, I_{in} is described in Fig. 2.7(b).

I_{in} depends on several process-related factors such as device's size. Modeling the shape of I_{in} has been discussed in lots of works [39–41]. As proposed in [41], I_{in} can be represented as double exponential pulse as given in (2.11), where τ_α and τ_β denote the collection time constant of junction and ion-track establishment time constant, respectively.

$$i(t) = \frac{Q_{coll}}{\tau_\alpha - \tau_\beta}(e^{\frac{t}{\tau_\alpha}} - e^{\frac{t}{\tau_\beta}}) \tag{2.11}$$



(a)



(b)

Figure 2.7: a. Ion particle hit on junction; b. Current pulse caused by ion particle [38].

The radiation hit can lead to a transistor state change once Q_{coll} is greater than the critical charge Q_{crit} which implies the sensitivity of transistor to ion onset. Considering the current pulse, we can represent this by (2.12). The device critical charge depends on the node capacitance, the operating voltage and the strength of feedback transistor. Since both supply voltage and capacitance are known to decrease with CMOS scaling down, Q_{crit} is also expected to decrease.

$$Q_{crit} \geq \int I_{in}(t)dt \tag{2.12}$$

Four scenarios of charge injection can take place as given in Fig. 2.8 [42]. The conducting transistors are represented by the resistors and the rectangle describes the drain regions. The big difference of four cases is the direction of current flow. In the first two cases, the voltage on P^+ node is raised while the voltage on N^+ node is declined in the other two cases. This way, the cases 1 and 3, the logic value on the node are temporally changed.

The most common transient effect induced by radiation-related effects is the *single transient effect* (SEE). It is normally measured by the *soft error rate* (SER) which represents the occurrence probability rate of SE or SE-induced functional failures in a given working environment. Fig. 2.9 depicts the single-bit SRAM SER/bit trend under usage conditions. The SER/bit peaks at the 130nm technology node and since then decreases. According to the authors, the neutron data reflects the measured data whereas alpha particle results are taken from simulations [9].

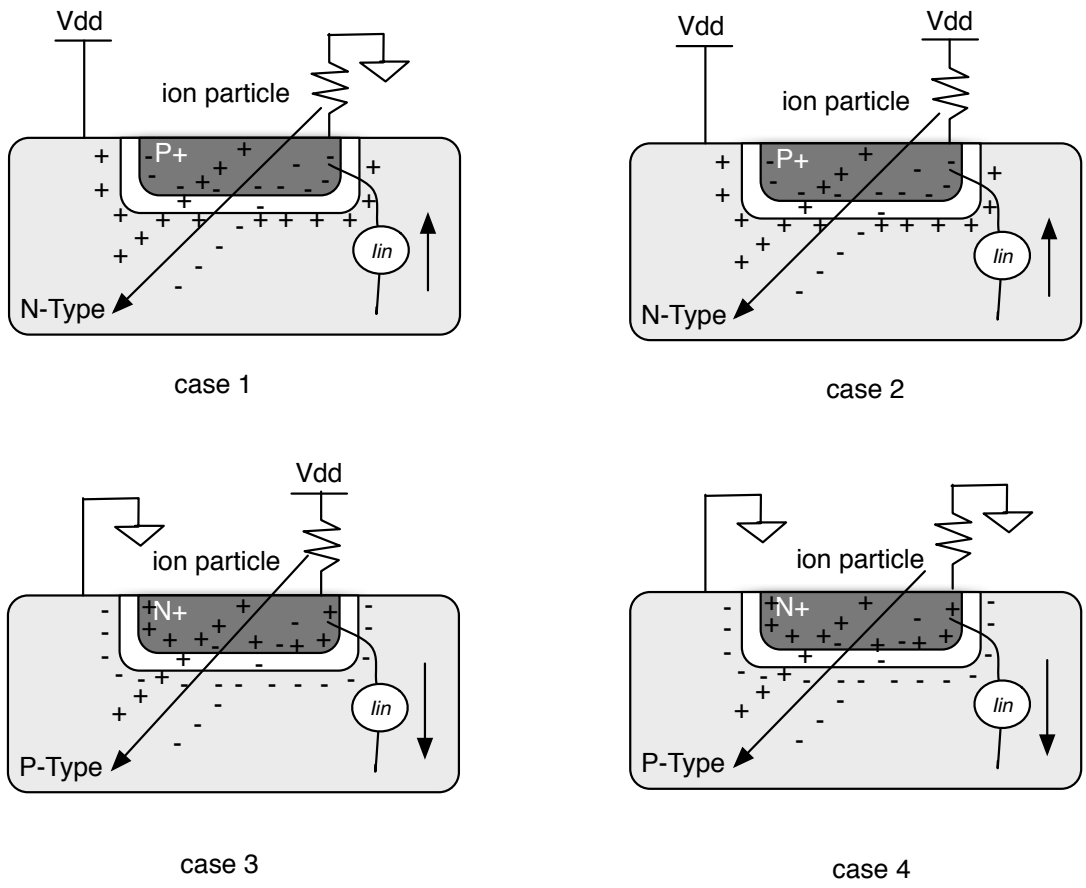


Figure 2.8: Possible scenarios for charge injection in CMOS [42].

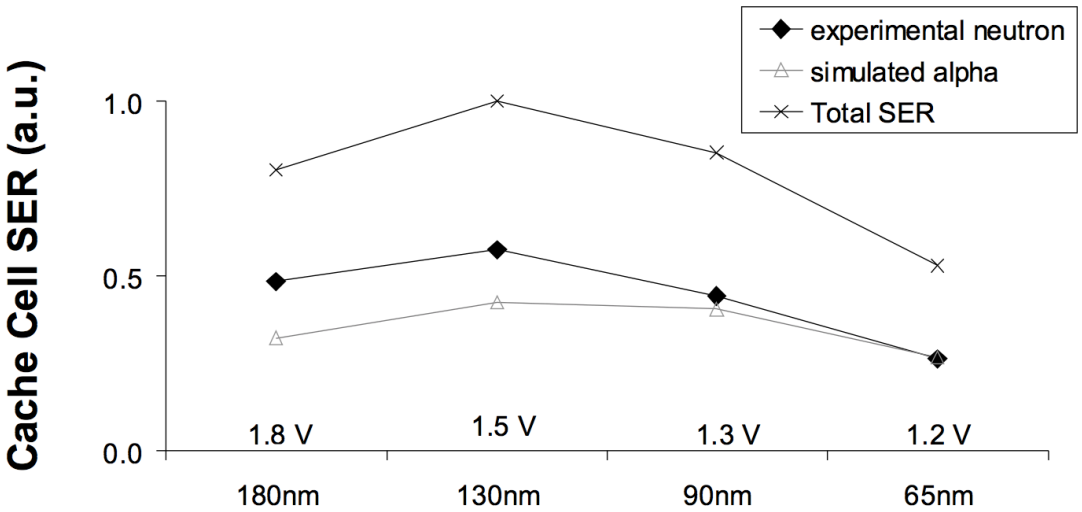


Figure 2.9: Normalized SRAM SER/bit trend [9].

2.2.3 Thermal noise

Thermal noise, also nominated as Johnson-Nyquist noise, is an intrinsic type of noise in electronic devices. This noise is generated by thermal fluctuations of the electrons inside resistances, capacitors and any other conductors, which can not be avoided. Also, the features of thermal noise are independent on neither the operational frequency nor the operational voltage. Mostly, thermal noise is characterized as a stationary gaussian

stochastic voltage fluctuation process which has a zero mean value [43]. For example, in a circuit modeled as capacitor C, the total Root-Mean-Square (RMS) thermal noise voltage on this capacitor can be written as (2.13), where T represents the temperature and k is the Boltzman constant.

$$U_n = \sqrt{\frac{kT}{C}} \tag{2.13}$$

In order to keep the power dissipation constant, the supply voltage is scaled linearly with the size in sub-micron CMOS that results in the noise margin reduction [43]. Consequently, the thermal noise more easily causes the crossing of logic threshold voltage and leads to bit flips. Moreover, as predicted by ITRS, the supply voltage will be lower than 0.65 V in 8 nm CMOS technology [2]. This decreasing trend makes the noise-tolerant design much severe than ever before and dramatically increases the SER. Figure 2.10 shows the chronogram of a two inputs AND gate affected by thermal noise. It is observed that due to the noise, two bit-flips occur at outputs. Actually, with signal-to-noisy ratio (SNR) getting worse, more and more bit-flips will randomly take place.

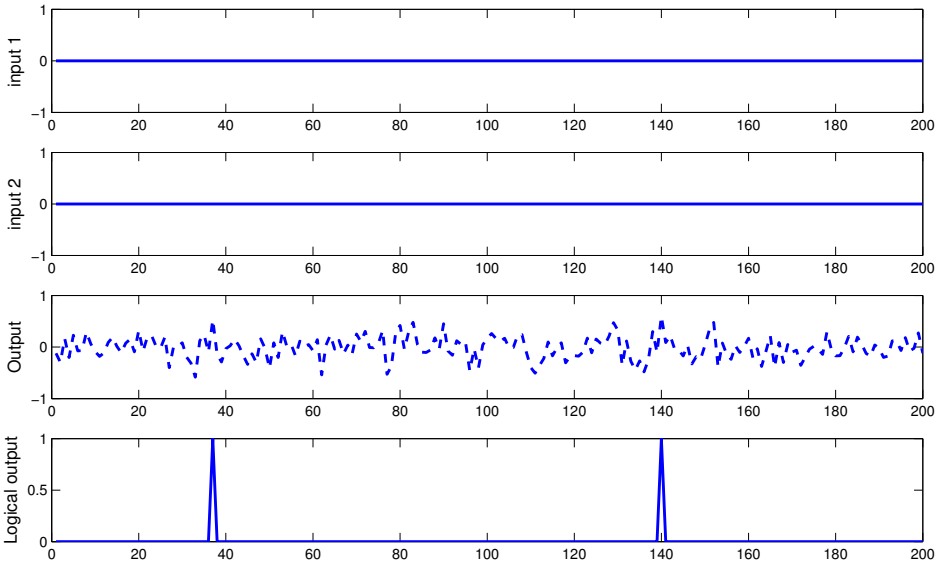
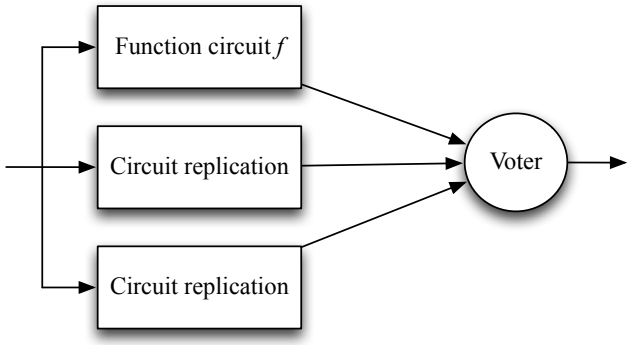


Figure 2.10: Chronogram of AND gate with thermal noise modeled as Gaussian noise (SNR=13.1).

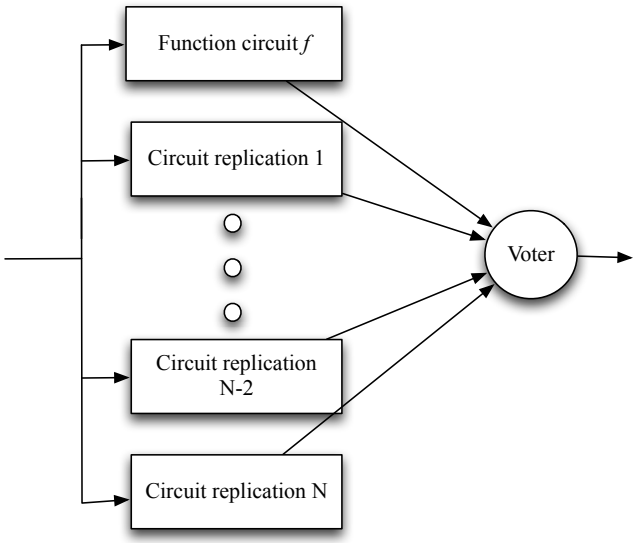
2.3 Fault-tolerant techniques

Fault tolerance which is closely related to the reliability represents the capability of a system to function properly in presence of faults [26]. The basic idea of fault tolerance is the use of "redundancy", either spatial or temporal. Space redundancy provides additional components, functions, or data items. Time redundancy repeats the same computation for several times and the result is compared to the previous result. Fault-tolerant techniques are performed mainly by: fault masking or fault detection. Fault masking provides the containment of faults, while fault detection informs the occurrence of erroneous operation which is followed by a corrective action in most of cases (e.g., re-computation).

Fig. 2.11(a) shows the general scheme of *triple modular redundancy* (TMR). It is a particular case of *N-modular redundancy* (NMR) which is the classical fault masking and concurrent fault correction technique as shown in Fig. 2.11(b). TMR consists of three



(a)



(b)

Figure 2.11: a. Triple Module Redundancy; b. N-Modular Redundancy.

modules and a voter. These modules perform the same operation and their results are compared by the voter. It chooses the proper output according to the predefined criteria, where in most cases the majority result is defined as the final output. As TMR does not require any change in operators' topologies, it can be easily implemented. The classical TMR requires high hardware redundancy. *Triple time redundancy* (TTR) uses the same resource to implement the function three times and compares the results gained from different rounds of computation. Extra area is saved by sacrificing the computation speed. However, the method of repeating the same calculation is powerless to permanent faults, since they occur at the same place during all calculations therefore can not be detected and corrected.

Another well-known technique is *concurrent error detection* (CED). It copes with the faults occurring during device operation (see Fig. 2.12). In a CED scheme, the function circuit performs the computation of input I and gives the outputs $f(I)$. Another independent unit, meanwhile, predicts some special characteristics of the system-output $f(I)$ for every input sequence I.

Duplication and comparison (DAC) is the simplest among CED schemes, where the prediction unit is implemented with the same function [45]. A comparator is used to check the two outputs. The circuit indicates an error if outputs are not identical. The hardware

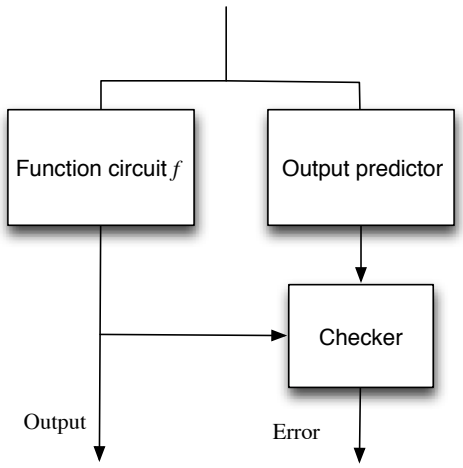


Figure 2.12: General architecture of a CED scheme [44].

overcost of DAC is 100% without considering the checker. Due to this overhead, DAC is excessively expensive for many cases. In fact, certain operators have inherent duplication as the carry-select adder (CSA). DAC can benefit from this inherent redundancy to save the area overcost [46].

Parity prediction is widely used in CED circuit for its lower extra area [45, 47]. Parity prediction for arithmetic operators computes the output operand's parity as a function of the operand's internal carries and input operands' parities. XOR trees are used to generate parities from inputs and carry signals. The parity prediction structure suffers from the faults on the carry circuit. These faults would affect multiple bits at output as well as the checking circuit, and stay therefore undetectable. An improved parity prediction approach has been described in [47]. In this approach, duplicated carry is obtained by independent circuits and two-rail checker is applied to generate the parity of carry. More studies on parity prediction can be found in [48–51].

The prediction in CED schemes can be also based on error detection codes. *M-out-of-n codes* are able to detect not only all single errors but also multiple unidirectional errors. Every code word has m "1s" and $n-m$ "0s" for a total bit length of codewords equaling n [52]. *Berger codes* have been first introduced in [53]. The main idea is to add extra binary string representing the number of "0s" in the given information word. A word of n-bit length requires $\lceil log_2 n \rceil$ extra bits. Berger codes have the capability of detecting all unidirectional errors. Arithmetic codes are the specific codes used for arithmetic operations including two basic codes: *AN codes* (A and N indicate two integers applied to encode and decode the codewords, respectively) and *residue codes*. AN codes are applied to addition and subtraction [54, 55], while Residue Codes are suitable for all arithmetic operations [56, 57].

Despite the variety of the error detection codes, they aim to cope with the errors on circuit's output and have a limited hamming distance that implies the detection ability. A hamming distance d of detection codes enables the detection of $d-1$ erroneous bits. For example, although Berger codes can detect all unidirectional errors, due to its hamming distance as 2, the half of 2-bit errors are undetectable [58]. However, a unique fault on logic gate may produce multiple erroneous bits at the outputs, and may stay undetectable if the number of faulty bits exceeds $d-1$. Furthermore, the implementation of error detection codes changes with target application where dedicated designs are required. The hardware redundancy such as TMR is always an attractive solution for coping with both permanent and transient faults. In this thesis, we concentrate on TMR method and CED methods,

in particular CED with parity prediction.

2.4 Conclusions

This chapter reviewed the basics on reliability while the differences among failure, error and fault were distinguished. Fault classification was presented along with the corresponding sources, including aging effects, radiations-related effects and thermal noise. A good understanding of these sources can help us to better define the fault model in reliability analysis. For instance, a fault induced by radiation-related effects can be assumed as a bit inversion on the output of logic gates continuing for no more than one clock cycle.

Chapter 3

Reliability evaluation methodologies

The decreasing of reliability due to the downscaling of CMOS technology forces the designer to adopt reliability assessment in addition to the traditional characterization for electronic devices. Efficient methodologies for reliability assessment are able to help designers to evaluate the sensitivity of their circuit within an acceptable impact on design time of product. Various works have been developed to find a good trade-off between the accuracy and computational complexity [59–61].

This chapter introduces reliability evaluation techniques at gate level for the estimation of logic design. We focus on efficiently evaluate the fault masking ability of logic circuits, particularly the logical masking which is the most intractable masking property to be analyzed. Other attempts on reliability analysis are devoted to study the impact of aging effects on digital ICs considering two main mechanisms: NBTI and HCI.

3.1 Fault masking

3.1.1 Electrical masking

Electrical masking occurs when a glitch fault is attenuated by subsequent logic gates due to the electrical properties and it may not have enough duration or amplitude to propagate to outputs, as seen in Fig. 3.1. The attenuation is caused by two reasons [62]:

- **Gate delay** induced by switching time of transistor causes the increase of rise and fall time of the glitch.
- **Short duration** of a pulse may lead to the decline of its amplitude, since the gate may begin to turning off before the output reaches its full amplitude.

It is possible to model the electrical masking in different ways, according to the source. For instance, the transient glitch caused by radiation effects is modeled as simple trapezoidal or triangle waveform in many works [63–65].

The electrical masking plays an important role for fault masking. Reported by Wang *et al.*, even for a small circuit with a logic depth of 5 stages, 138% overestimation has been observed without considering the electrical masking [66]. However, with the continuous scaling down of CMOS, the reduction in node capacitance and supply voltage reduces the effects of electrical masking.

3.1.2 Temporal masking

Temporal masking, also referred as latching-window masking, indicates the fact that an fault arrives at the output latch at the time of latching rather than the clock transition

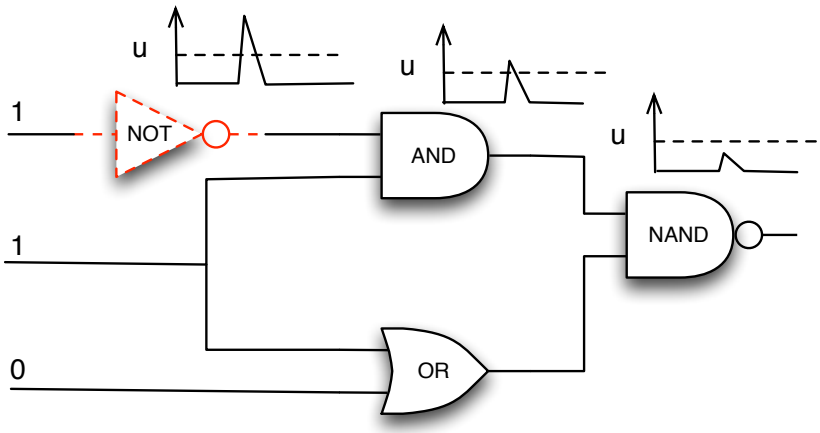


Figure 3.1: Example of electrical attenuation on a logic circuit.

(where the input value of a latch is captured). A glitch will cause a soft error when it is captured. The capture can occur when the glitch presents through out the whole latching window or overlap the latching window [62]. As shown in Fig. 3.2, the first glitch is perfectly eliminated by temporal masking, whereas the second one overlaps the clock transition time and leads to a soft error.

In practice, temporal masking can be evaluated as the probability of a glitch being captured, that associates to summation of the glitch width according to primary inputs [63]. As the temporal masking strongly depends on the clock frequency, the increase in operational frequency makes temporal masking contribute less and less to masking effects.

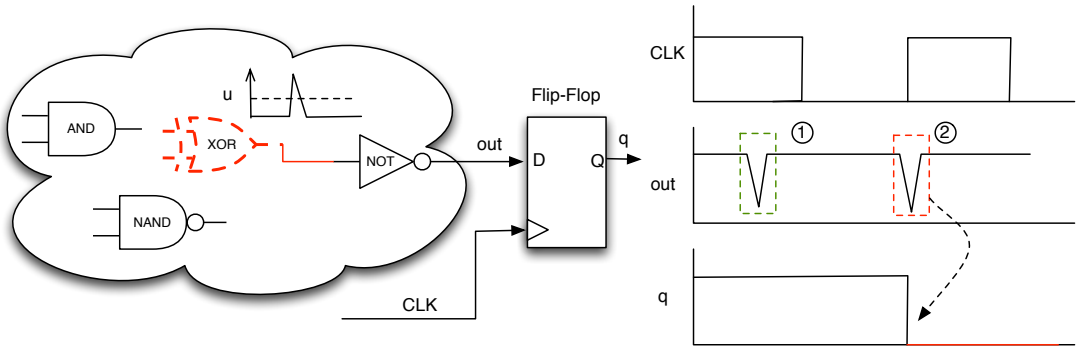


Figure 3.2: Example of a temporal masking.

3.1.3 Logical masking

Unlike the previously mentioned maskings, logical masking will keep stable with the technology scaling down. Logical masking appears when a fault occurs on non-sensitized paths of a circuit. Figure. 3.3 presents a simple logical masking on C17 circuit [67]. A fault occurs on a gate connecting to the fanout, which is masked before reach to the primary output. The strong dependence on circuit topology makes logical masking very hard to estimate. When the circuit contains lots of reconvergent fan-out, the estimation will become a NP problem [68].

For a long time, logic circuits were assumed to be more fault resistant compared with memory cells with these masking effects, especially for the transient faults induced by radiation effects. However, the vulnerability to faults of combinational circuit have become

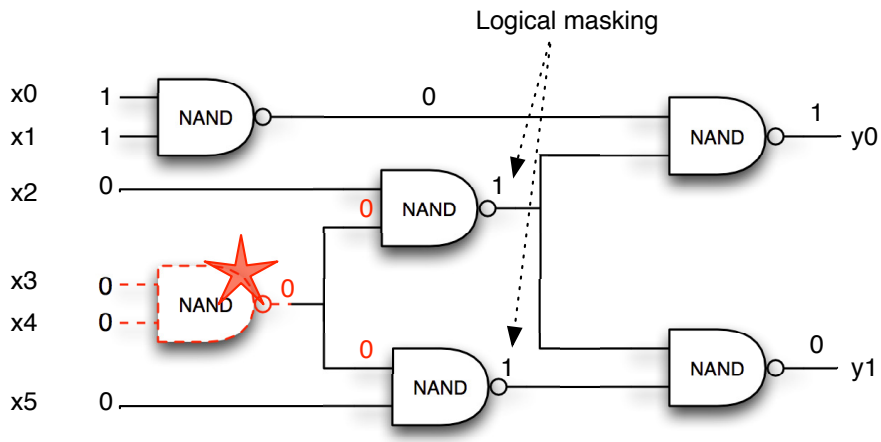


Figure 3.3: Example of a logical masking.

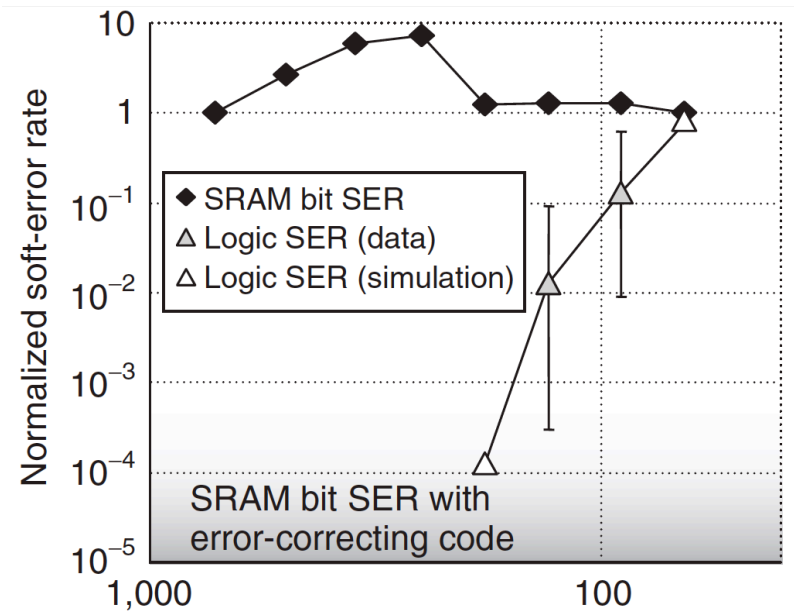


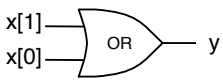
Figure 3.4: Comparison of SRAM bit SER with latch SER [38].

severe with the dimension scaling down (see Fig. 3.4). Moreover, due to the decline of two other masking effects caused by technology innovation, logical masking has became the major masking effect in logic circuits. Thus, this thesis devotes therefore to the evaluation of logic circuit reliability with respect to logical masking.

3.1.4 Logical masking assessment

3.1.4.1 PTM

The *probabilistic transfer matrices* (PTMs) approach is based on the probabilistic model presented in [69]. In PTM approach, the circuit reliability is determined by a matrix containing all output probability information with corresponding input pattern. The principle is to model each logic block by a probability matrix. Let us consider a logic block b with input x and output y, where $x \in \{x_0, x_1, \cdots, x_i, \cdots, x_{2^n-1}\}$ and $y \in \{y_0, y_1, \cdots, y_j, \cdots, y_{2^m-1}\}$. Suppose that x and y have binary representations. For example, x_{15} is noted as "1111" when n is "4". The PTM of block b, denoted as PTM_b, has $2^n \times 2^m$ elements where each (i, j) element is the probability of getting output $y = y_j$



(a)

$x_{[0]}$	$x_{[1]}$	y
0	0	0
0	1	1
1	0	1
1	1	1

(b)

$$\begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \overbrace{\begin{array}{cc} 0 & 1 \end{array}}^{y} \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array}$$

(c)

$$\begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \overbrace{\begin{array}{cc} 0 & 1 \end{array}}^{y} \\ \begin{bmatrix} q_1 & 1-q_1 \\ 1-q_2 & q_2 \\ 1-q_3 & q_3 \\ 1-q_4 & q_4 \end{bmatrix} \end{array}$$

(d)

Figure 3.5: Logic gate OR: a. Structure; b. Truth table; c. Corresponding ITM and d. PTM.

given the occurrence of input $x = x_i$, noted by $p(j|i)$. In an ideal (i.e., fault free) block, the PTM contains only zeros and ones and is referred as an ideal transfer matrix (ITM).

Fig. 3.5 presents an example of a logic OR gate with error probability of $1 - q_i$, where $q_i \in \{1, 2, 3, 4\}$ denotes the probability of getting a correct output given a input $x = x_{i-1}$. Taking the element in position $\{1, 1\}$ as instance, since the input "00" corresponds a correct output "0" , the probability of y equaling "1" represents the probability of getting an incorrect output, that is $\mathrm{P}(y = 1|x = 00) = 1 - q_1$. It is noteworthy that ITM is defined for $q_i = 1$ and can be extracted directly form the truth table that indicates the fault-free operation of logic gate. The wire and fanout in circuit are assumed fault-free and represented by ITMs.

To calculate PTM of a whole circuit C, we combine PTMs of its basic blocks using matrix products and tensor products. Assume two blocks and respective matrices with PTM_1 ($n \times m$) and PTM_2 ($o \times p$), then:

- These two blocks in parallel result in a PTM given by the inner product of PTM_1 and PTM_2
- These two blocks in parallel result in a PTM given by the tensor product of PTM_1 and PTM_2 with dimension of $no \times mp$

Taking the circuit in Fig. 3.6 for instance, we temporally define "$\cdot$" and "$\otimes$" as inter product operator and tensor product operator, respectively. For the consideration of simplicity, the next discussions are under the assumption that gates have identical error probabilities. The PTM of the circuit is computed as follows:

$$PTM_{circuit} = (ITM_{wire} \otimes ITM_{fanout}) \cdot (ITM_{wire} \otimes PTM_{or}) \cdot PTM_{and} \tag{3.1}$$

The reliability of a block or a circuit is directly extracted from the respective PTM and ITM, according to (3.2), where $p(i)$ denotes the probability that input x is x_i.

$$R = \sum_{ITM(i,j)=1} p(j|i) \cdot p(i) \tag{3.2}$$

The expression above gives the sum of probabilities corresponding to correct outputs. That is, the element in position (i, j) of PTM matrix will contribute to the sum only if the respective (i, j) element of the ITM has a value "1". Since the PTM gives the exact probabilities of getting correct outputs, this expression provides accurate reliability assessment. A fault can be modeled as a bit-flip for some type of transient faults as presented previously but can also be modeled as a stuck-at-0 or stuck-at-1 for permanent faults as depicted in Fig. 3.7 [70, 71].

The computational complexity of PTM approach depends on the number of logic gates as well as the circuit topology. In [72], authors applied *algebraic decision diagrame* (ADD)

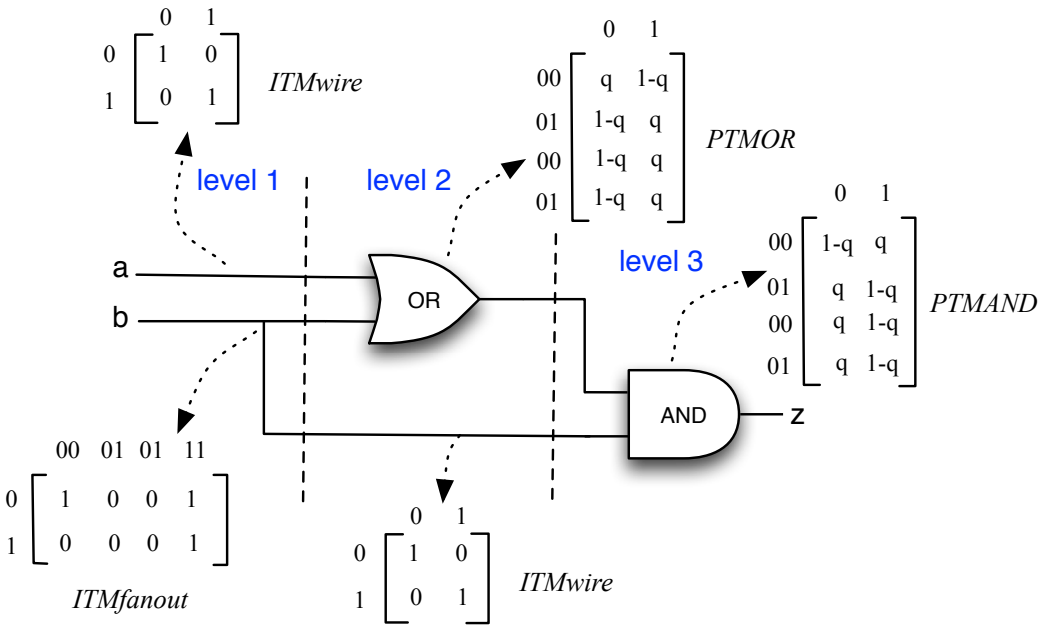


Figure 3.6: PTM computation of an example circuit (q is set identical for all gates).

$$\begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \overbrace{\begin{array}{cc} 0 & 1 \\ \hline 1 & 0 \\ 1-q & q \\ 1-q & q \\ 1-q & q \end{array}}^{y} \qquad \begin{array}{c} \\ x \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \overbrace{\begin{array}{cc} 0 & 1 \\ \hline q & 1-q \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{array}}^{y}$$

(a) (b)

Figure 3.7: a. Stuck-at-0 and b. stuck-at-1 on OR gate.

for efficient PTM computation without the loss of accuracy. Another solution has been reported by Naviner *et al.* through the reduction of redundancy information in matrices and the computation of useful submatrices from tensor products [73].

3.1.4.2 SPR

The basis of signal probability reliability (SPR) algorithm is the assumption that each logic signal has four states: correct 0 (0_c), incorrect 0 (0_i), correct 1 (1_c) and incorrect 1 (1_i) [59]. These states are represented by a 2×2 matrix as given in (3.3) with a convention for probabilities presented in (3.4). The reliability of a signal is given by its probability of having correct states (0_c and 1_c), expressed in (3.5).

$$signal = \begin{bmatrix} 0_c & 1_i \\ 0_i & 1_c \end{bmatrix} \tag{3.3}$$

$$P_{2\times 2}(signal) = \begin{bmatrix} P(0_c) & P(1_i) \\ P(0_i) & P(1_c) \end{bmatrix} \tag{3.4}$$

$$R_{signal} = P(0_c) + P(1_c) \tag{3.5}$$

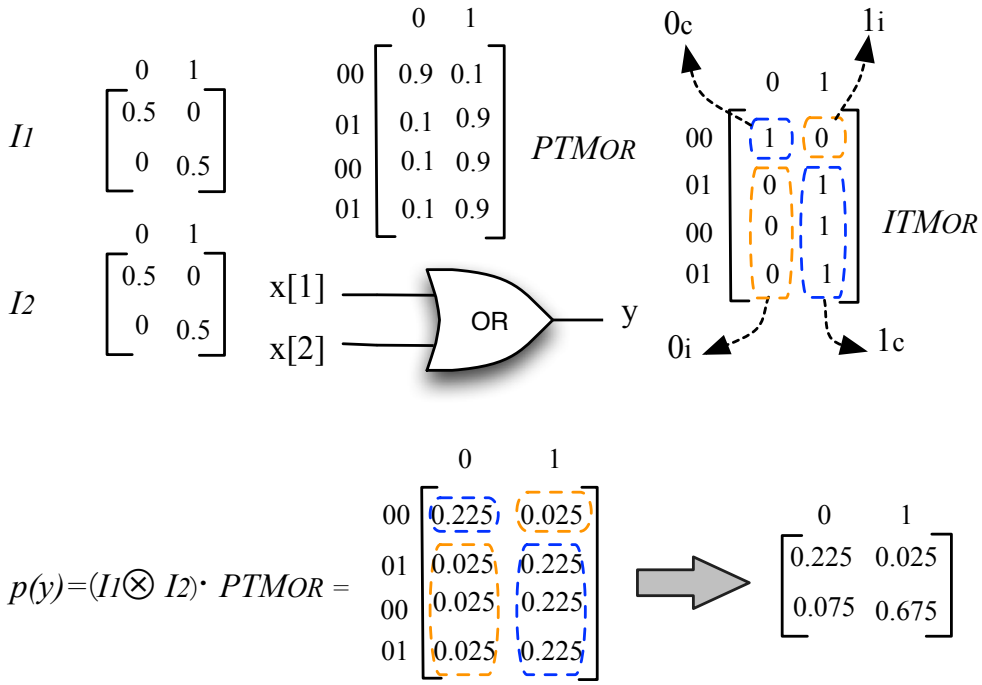


Figure 3.8: Signal propagation on a OR gate, $q = 0.9$.

In SPR algorithm, the reliability of a logic circuit is obtained by signal propagation from primary inputs to the primary output. The propagation of a fault in a logic circuit is evaluated similarly as that in PTM approach. Each fault-prone gate is represented by its corresponding PTM. Let us consider a OR gate, we assume input x fault-free with signal probability matrices I_1 and I_2. The signal propagation on such a gate can be described as in Fig. 3.8. The reliability of y is thus $0.225 + 0.675 = 0.9$. It implies when input are fault-free, the output of a gate has the same reliability as the gate, as expected. The obtained signal probability of one gate's output will be saved in a 2×2 matrix and used as input for next level, as depicted in Fig. 3.9. When the circuit has multiple outputs, the circuit reliability is computed through the multiplication of each output signal.

SPR algorithm efficiently reduces the storage requirement and computational complexity between matrices. Therefore, SPR is considered as a fast analysis method. However, basic SPR makes the assumption that signals are independent and ignores the correlation effects. As a result, this algorithm underestimates the circuit's reliability. Dynamic weighted averaging algorithm (DWAA) and SPR multiple-pass (SPR-MP) algorithms are applied to deal with correlation effects [59]. The DWAA relies on the adjustment of signal probabilities on the fanout cone of a reconvergent node. Whereas, the SPR-MP method repeats the propagation pass for every possible signal states on fanout signals, where one state is performed in a pass. Consequently, the complexity of SPR-MP method strongly depends on fanout signals number.

3.1.4.3 PBR

Probabilistic binomial reliability (PBR) analyzes the circuit's signal probability which is described as the probability of getting a correct output taking into account single and multiple faults [74]. A combinational circuit is modeled as a black box with input x and output y. This method is based on the assumption that the circuit reliability is the exclusive summation of getting correct y given a certain $x = x_i$ despite the faults occurrence. Taking

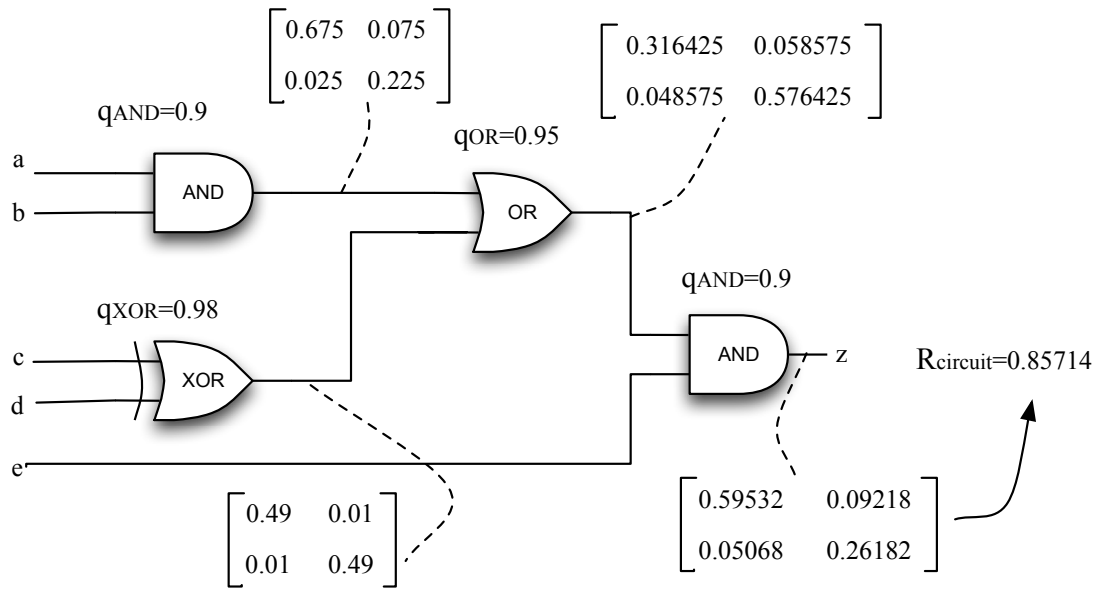


Figure 3.9: Reliability computation by signal probability propagation [59].

a circuit C with m-bit input and n-bit output for example, its reliability is expressed as follows:

$$R = \sum_{i=0}^{2^m-1} p(x_j)p(y = correct|x_i) \tag{3.6}$$

This assumption includes the reliability of fault-free operation as well as the masking ability of faults presences. PBR approach thus computes R as the sum of all R_k representing the reliability of circuit under k simultaneous faults, where G indicates the number of logic gates in the circuit. Each R_k is obtained from the probability of occurrence of k faults in a circuit ($F(q,k)$) and the percentage of fault masking considering all input patterns and k simultaneous faults (c_k), as shown in (3.7)

$$R = \sum_{k=0}^{G} R_k = \sum_{k=0}^{G} F(q,k)c_k \tag{3.7}$$

Assume that the gates are identical with a probability q of producing a correct output, $F(q,k)$ can be expressed by (3.8). The fault masking ability c_k considers all possible k faults for all input combinations as given in (3.9), where $\overline{y(x_j, e_{G:0}) \oplus y(x_j, e_{G:k}(l))}$ represents the occurrence of fault masking.

$$F(q,k) = (1-q)^k q^{G-k} \tag{3.8}$$

$$c_k = \frac{1}{2^m} \sum_{l=1}^{C_k^G} \sum_{j=0}^{2^{m-1}} \overline{y(x_j, e_{G:0}) \oplus y(x_j, e_{G:k}(l))} \tag{3.9}$$

The computation of fault masking coefficient c_k is determined by fault injection and simulation. The accuracy in PBR analysis is ensured by performing an exhaustive functional simulation that leads to a computational complexity exponential with the number of binary inputs and the number of gates in the circuit. The optimization can be achieved by reducing the number of simulations considered. Consequently, in this case accuracy can

not be ensured. A hardware implementation named FIFA has been presented to accelerate the PBR method via the use of FPGA [75]. FIFA tool is based on the structure shown in 3.10, in which a fault-free circuit works simultaneously as a reference for fault-prone design.

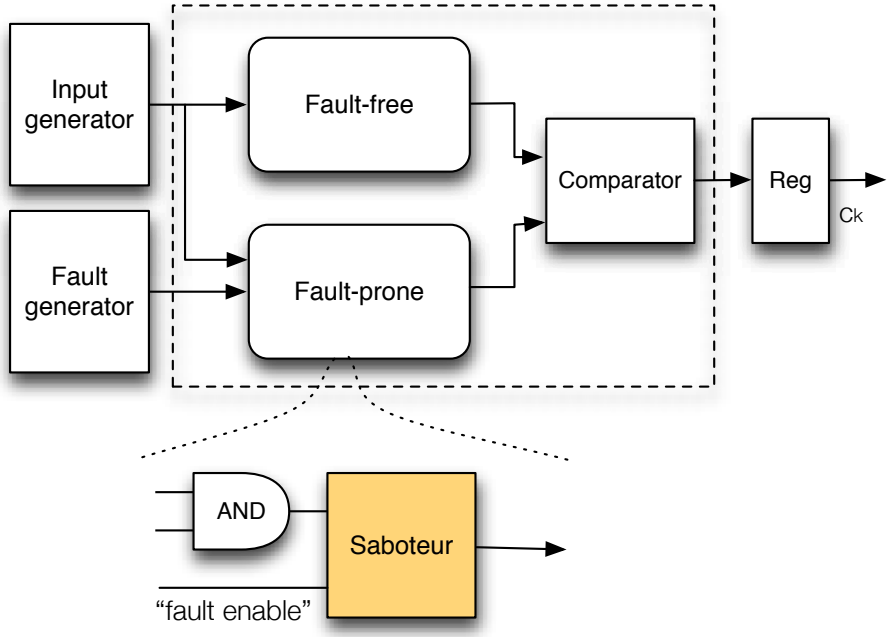


Figure 3.10: The principle structure of FIFA platform.

3.1.4.4 Other Important Works

The *probabilistic gate model* (PGM) method describes the signal probability of an input or output of a gate as the probability that the signal is a logical "1" [76, 77]. Similarly as PTMs and SPR approaches, this method expresses each logic gate as a PGM formulation. The formulation can be iteratively applied to compute the whole circuit reliability. PGM approach assumes that signals in circuits are independent of each other. The overall reliability is therefore obtained by multiplying the individual outputs. This method can be used for any type of gates and fault models. PGM provides a high reliability accuracy but still suffers from runtime problem. Soon afterwords, authors presented a modular approach based on PGM for large circuits [60].

Another approach deals with correlation effects is proposed, refereed as CPM [78]. This approach is based on SPR algorithm but can achieve a good compromise between accuracy and time complexity. It is possible to adopt this approach in analysis of large circuit due to the fact that the time complexity depends on number of reconvergences rather than circuit size.

The use of boolean difference calculus is reported in boolean difference-based error calculator (BDEC) that takes into account primary input error probabilities as well as gate error probabilities [79]. BDEC assumes that the error of signals on gate's output can come either from gate or from input gate error. Local reconvergent fanout is considered as a single circuit and evaluated individually. One advantage of this calculator is its linear-time complexity with the number of gates when the accuracy is ensured.

In [80], an alternative method is described relying on stochastic computation that enables the implementation of probabilistic analyses performed on PGM approach. The method encodes each gate in the statistic random binary bitstream. The length of bitstreams directly relates to the accuracy of estimation and computational complexity.

Recently, an hybrid approach named SNaP with a linear time complexity benefiting form FPGA emulation has been proposed [61]. The basis of SNaP includes two concepts: fault sources and fault propagation. Circuit reliability is obtained from the estimation of the number of gates that enables the propagation of a fault to any of primary outputs. Although SNaP allows the evaluation of large circuits, its hardware complexity is not negligible for the FPGA implementation.

3.2 Masking analysis methods

3.2.1 Gate structure-aware modeling of transient fault

An efficient and accurate assessment of reliability should be always done at the early stage of the circuit design process. To this, we already presented many methods including PTM, CPM, SPR, PGM and etc. Nevertheless, they consider the hypothesis of individual gate failure. Actually, faults affect individual gates at transistor level, and these vulnerable constituent transistors of different gates varies in numbers, constitutions and other factors. Aiming to achieve more accurate reliability estimations, researchers have realized that the identical gate failure rate is not accurate in reliability evaluation process. For example, an alternative metric was proposed to relate the reliability of the transistors to that of a gate [81]. As a matter of fact, the gate failure probability depends in part on the topology of a logic gate as well as its input vector. Consequently, even the identical gates may exhibit different failure probabilities. This section presents a gate topologie-aware

3.2.1.1 Fault analysis of CMOS basic gates

In digital logic designs, a MOS transistor can be modeled as a switch. In other words, its working resistance is either very high (the transistor is considered as "OFF") or very low (the transistor is considered as "ON"). Assume that the probability of failure for PMOS and NMOS transistor is ε_P and ε_N, respectively. The exact value of ε_P and ε_N can be calculated as in [82]. Figure 3.11(a) shows a CMOS inverter. As the input vector A will not affect the probability of failure in PMOS and NMOS transistors, the probability of NOT gate failure could be expressed as in (3.10).

$$\varepsilon_{NOT} = 1 - (1 - \varepsilon_P)(1 - \varepsilon_N) \tag{3.10}$$

The two-input NAND gate is also analyzed as an example (see Figure 3.11(b)). A radiation strike can upset one or more of its constituent transistors, leading the output to exhibit a transient error like flip-to-1 or flip-to-0. The failure in transistors behaves differently from the NOT gate, and the combination of transistors that is susceptible to errors also depends on the input pattern when the strike occurs, as explained below:

- Input AB = 00: output flips from 1 to 0 if both NMOS transistors are upset simultaneously, so do the two PMOS transistors.
- Input AB = 01: output flips from 1 to 0 when one NMOS and one PMOS transistor are upset.
- Input AB = 10: similar with the situation when "Input AB = 01".
- Input AB = 11: output flips from 0 to 1 if one of the four constituent MOS transistors is upset.

The aforementioned analysis can be illustrated by the corresponding switch mode. Table 3.1 illustrates the cases in details when one transistor or more transistors are upset

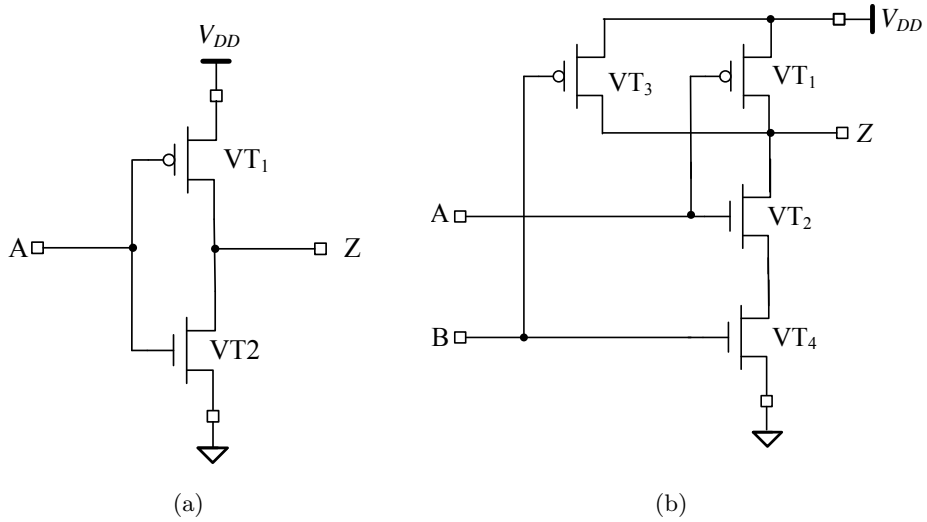


Figure 3.11: a. A CMOS inverter; b. A CMOS NAND2 gate.

Table 3.1: Fault analysis in CMOS gate NAND2

Inputs (A,B)	VT_1	VT_2	VT_3	VT_4	Flips	Probability of Failure
00	ON	OFF	ON	OFF	$1 \to 0$	$Pr_0 = 1 - (1 - \varepsilon_P^2)(1 - \varepsilon_N^2)$
01	OFF	OFF	ON	ON	$1 \to 0$	$Pr_1 = 1 - (1 - \varepsilon_P)(1 - \varepsilon_N)$
10	ON	ON	OFF	OFF	$1 \to 0$	$Pr_2 = 1 - (1 - \varepsilon_N)(1 - \varepsilon_P)$
11	OFF	ON	OFF	ON	$0 \to 1$	$Pr_3 = 1 - (1 - \varepsilon_P)^2(1 - \varepsilon_N)^2$

to produce an error at the primary output. The probability of gate failure is also presented. Transient faults analysis in this section reveals the relevance of logic structures and probabilities of input signals of each basic logic gate to assessing the failure rate of logic gates.

3.2.1.2 Mathematical modeling for transient faults

Actual faults in circuits can not be directly predicted in the design and validation procedures, therefore special fault models are required. Let $\{B_n : n = 1, 2, 3, \ldots\}$ be a finite or countable infinite partition of a sample space and each event B_n is measurable. The law of total probability states that for any event A of the same probability space:

$$\Pr(A) = \sum_n \Pr(A \mid B_n) \Pr(B_n) \tag{3.11}$$

Based on (3.11), the failure probability of basic logic gates could be expressed as in (3.12), where Pr (input vector$_n$) is the probability of input signals (also known as "signal probability").

$$\Pr(\text{gate failure}) = \sum_n \Pr(\text{gate failure} \mid \text{input vector}_n) \Pr(\text{input vector}_n) \tag{3.12}$$

Signal probability of logic gate input/output is defined as the probability that the signal is logical "1". Parker et al. proposed an algorithm for general combinational logic circuits which allows the formulation of an output probability by the given input probabilities [83]. Assume equiprobable primary inputs, that is:

$$Pr(00) = Pr(01) = Pr(10) = Pr(11) = 0.25 \tag{3.13}$$

The gate failure probability of NAND2 can be expressed as in (3.14) with the hypothesis that $\varepsilon_P = \varepsilon_N = \varepsilon$ for simplicity.

$$\varepsilon_{\text{NAND2}} = 0.25\{1-(1-\varepsilon^2)^2\} + 0.5\{1-(1-\varepsilon)^2\} + 0.25\{1-(1-\varepsilon^4)\} \tag{3.14}$$

3.2.1.3 Fault model in reliability evaluation

Given independent inputs, all kinds of boolean functions can be mapped into algebraic expressions of signal probabilities, according to the following three rules.

- Rule 1: Boolean "NOT", i.e., $Z = \bar{A}$, corresponds to $z = 1 - a$
- Rule 2: Boolean "AND", i.e., $Z = AB$, corresponds to $z = a \cdot b$
- Rule 3: Boolean "OR", i.e., $Z = A + B$, corresponds to $z = a + b - a \cdot b$

For the two-input NAND gate $Z = \overline{AB}$, output signal probability can be expressed as in (3.15).

$$\mathbf{z} = \begin{bmatrix} 1-ab & ab \end{bmatrix} \cdot \begin{bmatrix} 1-\varepsilon_{NAND2} \\ \varepsilon_{NAND2} \end{bmatrix} \tag{3.15}$$

Taking the benchmark circuit C17 [67] as an example, for equiprobable primary inputs, the propagations of signal probability are shown in Figure 3.12.

Figure 3.13 illustrates the different gate failure rates of constituent gates in C17. We observe that although NAND5 and NAND6 have the same position in the circuit (the third level), they exhibit different failure rates since they have different input signal probabilities.

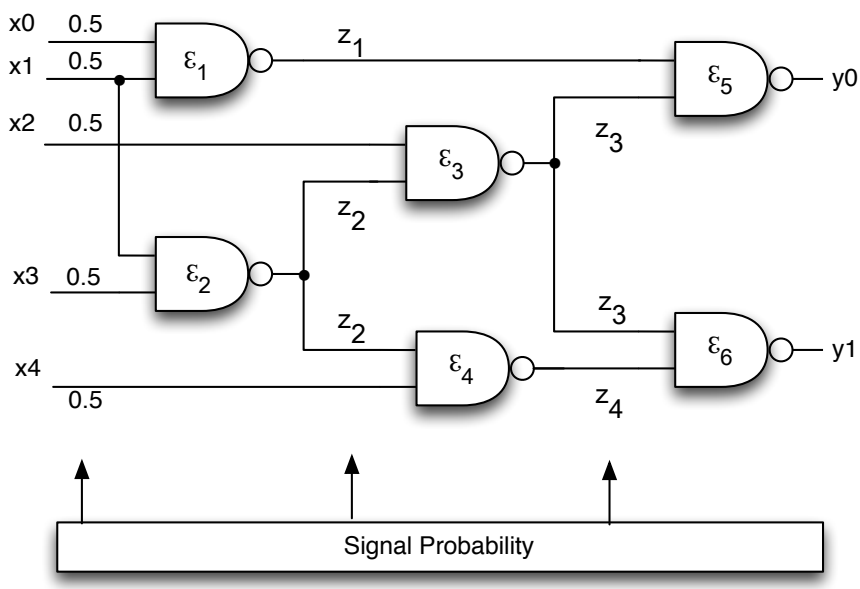


Figure 3.12: Propagation of signal probabilities in C17.

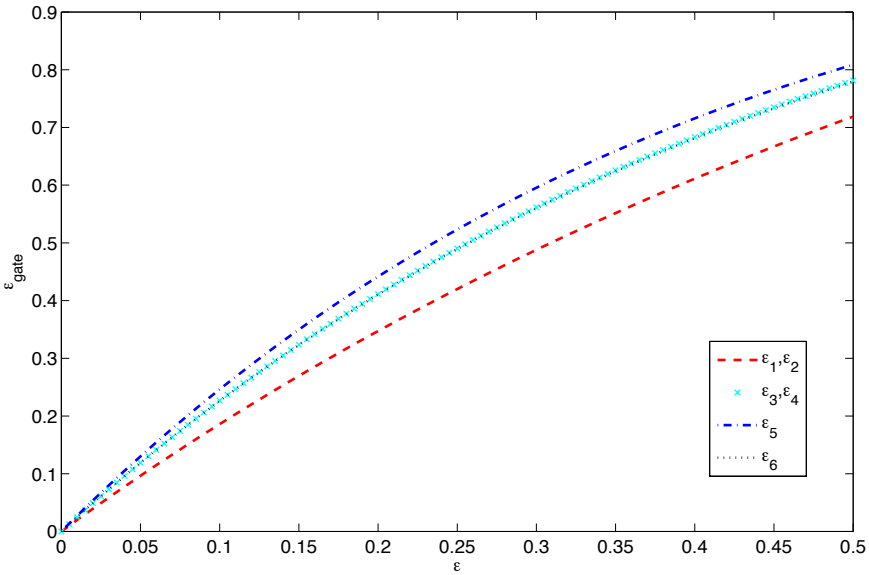


Figure 3.13: Different failure rates of constituent gates in C17.

In [81], a similar gate failure rate of NAND2 gate was proposed. It assumed that the correct functioning of a gate requires the correct functioning of all its transistors. The equation used to relate the reliability of the transistors to that of a gate is:

$$\varepsilon_{scm} = 1 - (1 - \varepsilon_{transistor})^n \tag{3.16}$$

where n is the number of transistors in the gate. It considered the identical gate failure probability of the same kind of gates , therefore it is not accurate. For primary inputs in circuit C17 of Figure 3.12, Figure 3.14 points out the differences between these two approaches for calculating gate failure rate. The proposed model has lower gate failure rate thus it can avoid overprotection to logic circuits in the further fault tolerant design.

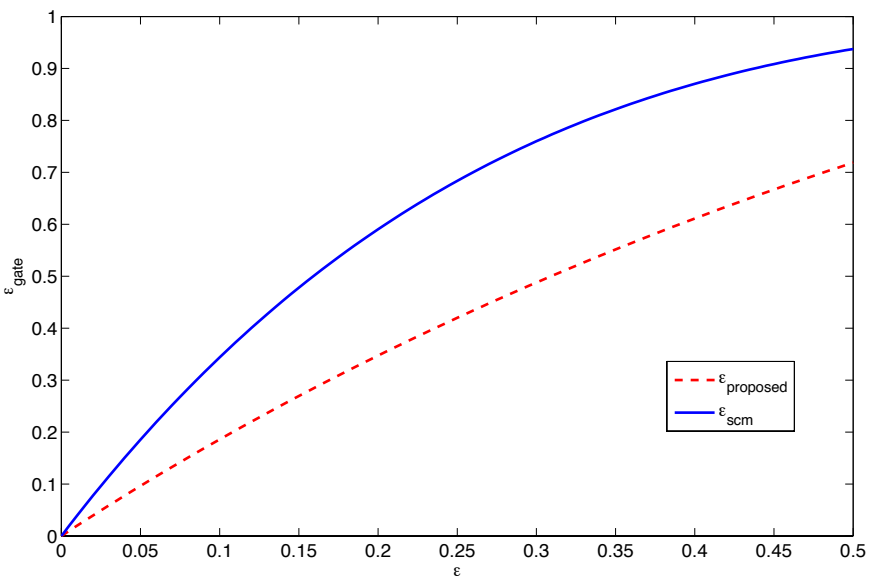


Figure 3.14: Different gate failure probabilities.

As transistor failure rates in emerging nanoelectronics are projected to be in the order of 10^{-2} to 10^{-1} depending on their size and frequency characterizations [84], we assume $\varepsilon_{transistor} = \varepsilon = 0.01$, the corresponding gate failure probabilities of six NAND2 gates in C17 are shown in Table 3.2. Reliability evaluation results (Tool: SPRMP [59]) based on the proposed fault model and a similar metric in [81] are also presented in Table 3.2 for comparison. We find that a higher reliability is obtained by applying our proposed model, which avoids the underestimation of circuits.

Table 3.2: Reliability evaluation based on different gate fault models

ε_{gate}	Method [81]	Proposed
$NAND1$	0.0394	0.0199
$NAND2$	0.0394	0.0199
$NAND3$	0.0394	0.0248
$NAND4$	0.0394	0.0248
$NAND5$	0.0394	0.0272
$NAND6$	0.0394	0.0247
$\mathbf{R_{Circuit}}$	**82.4617%**	**88.9199%**

3.2.2 SETs on arithmetic operator in iterative processors

Although transient faults have been discussed in various works [59, 69], authors focus more on the analysis of combinational circuits. Aiming to explore the impact of transient faults occurring on iterative processors, *fault impact coefficient* that takes into account both the spatial and temporal location of the fault is defined in this section. It also embeds the probability of occurring such a fault. The work shows how the impact varies with the bit position, the iteration and the faulty operator. The influences of different operator implementations are also discussed in order to identify the most reliable architecture.

3.2.2.1 SET in a generic processor

The fault model considered is the temporary inversion of the value of a bit on a logic gate output. The goal is to study the impact of a fault that would cross the logical operators and be sampled by a register. A single fault is assumed to occur during the complete execution of the algorithm. Consider a processor G as presented in Fig. 3.15 with inputs I and implementing an iterative algorithm to produce outputs O, that is $O = G(I)$. Assume G performs N iterations and consists of K operators that deal with operands coded on L bits. Suppose that the impact that a fault in G can have on the outputs O, depends on three factors: iteration number ($n = 1, 2, \ldots, N$), faulty operator ($k = 1, 2, \ldots, K$) and bit position in the faulty operator ($l = 0, 1, \ldots, L-1$).

Let $f_{(n,k,l)}$ denote a fault, and the number of possible inputs as N_I. The impact of such a fault is defined as the standard deviation given in (3.17), where G and $\widehat{G}$ correspond to the fault-free and fault-prone version of the processor, respectively.

$$\varepsilon_{(n,k,l)} = \sqrt{\frac{1}{N_I}\sum_{\boldsymbol{I}}[G(I) - \widehat{G}(I, f_{(n,k,l)})]^2} \tag{3.17}$$

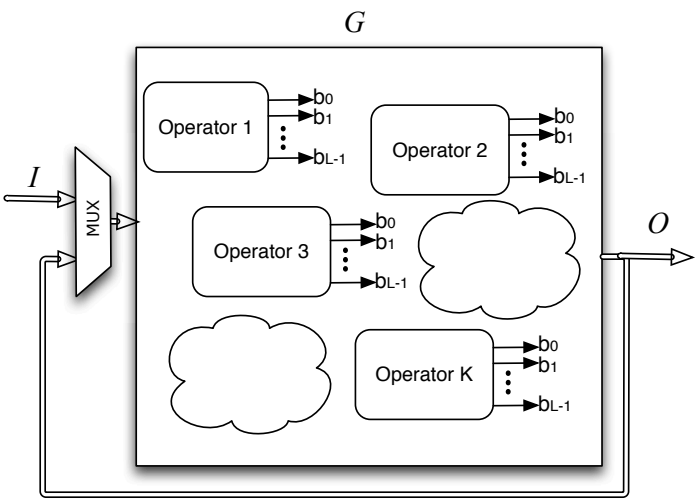


Figure 3.15: Iterative processor G

Denoting $p_{(n,k,l)}$ the probability that a fault $f_{(n,k,l)}$ occurs, the *fault impact coefficient* and its normalized expression are defined in (3.18) and (3.19), respectively. This parameter takes into account the combined effect of the fault impact and the probability that this fault really takes place. The greater the value of $\phi'_{(n,k,l)}$, the greater the relevance of the fault $f_{(n,k,l)}$.

$$\phi_{(n,k,l)} = p_{(n,k,l)} \times \varepsilon_{(n,k,l)} \tag{3.18}$$

$$\phi'_{(n,k,l)} = \frac{\phi_{(n,k,l)}}{\max\{\phi_{(n,k,l)}\}} \tag{3.19}$$

We define the *operator impact coefficient* of an operator k, Ψ_k, as (3.20). This coefficient shows how the operator k impacts the processor behavior.

$$\Psi_k = \sum_{n=1}^{N} \sum_{l=0}^{L-1} \phi_{(n,k,l)} \tag{3.20}$$

The coefficient Ψ_k can be useful in a selective hardening approach to better define which operator should be replaced by a more fault tolerant version. A possible eligibility metric in such a selective hardening approach is given in (3.21), where c_Ψ, c_A, and c_T are weight factors chosen by the designer according to the design priorities. A_k and T_k stand for area and propagation delay of the operator k, respectively.

$$\xi_k = \frac{1}{\Psi_k^{c_\Psi} \times A_k^{c_A} \times T_k^{c_T}} \tag{3.21}$$

3.2.2.2 Modeling approach

MATLAB models for studied structures are developed that follow the general scheme presented in Fig. 3.16(a). For each input I the algorithm is performed on a fault-free and a fault-prone version of the processor, then the obtained results $O = G(I)$ and $\widehat{O} = \widehat{G}(I)$ are compared. The simulations covered all faults $f_{(n,k,l)}$.

Suppose that T is the time required for a complete execution of the algorithm and P is the probability of occurring a fault on a logic gate during T. As no special condition is specified, we assume that the probability of faults is uniformly distributed over T (i.e.,

over all N iterations) and over the K operators. The probability one fault $f_{(n,k,l)}$ occurs can be expressed as:

$$p_{(n,k,l)} = \frac{P}{N \times K \times L} \tag{3.22}$$

In fact, for fault-prone version of the processor, the faults are injected by inverting the value of one bit, see Fig. 3.16(b), as described in Section 3.2.2.1, without taking into account the logical masking of the arithmetic operators. In other words, the occurrence of a fault in an operator results in the occurrence of a fault in one of its L output bits (referred as **Consider 1**).

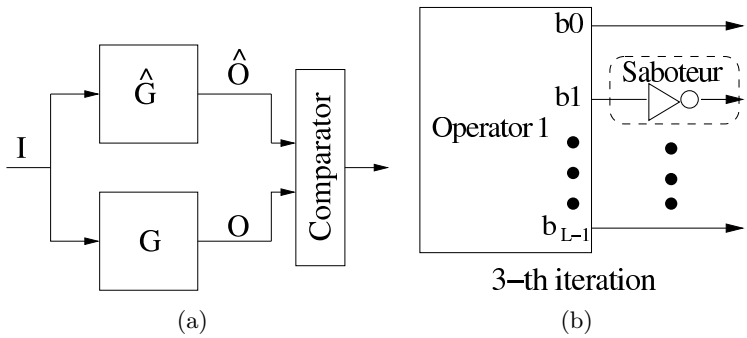


Figure 3.16: a. General structure to compute the impact coefficient; b. Injection of fault $f_{(3,1,1)}$.

The aforementioned analysis is then extended to consider the ability of fault masking that arithmetic operators can have (referred as **Consider 2**). Indeed, a fault can occur on a logic gate without producing the faulty bit on the operator's outputs. On the other hand, a unique fault in logic gate can also produce several faulty bits on the operator's outputs.

The behavior of arithmetic operators with respect to the faults is very dependent on the implemented architecture. In this work, the effect on the operator's outputs, that a fault on a gate has, is estimated by using the fault-propagation analysis algorithm *Signal Probability Reliability* (SPR) described in Section 3.1.4. Let N_g be the total number of logic gates in each one of the K operators. By assuming the probability of faults uniformly distributed on iterations and operators, we obtain the the probability of a fault in a given logic gate as:

$$p_g = \frac{P}{N \times K \times N_g} \tag{3.23}$$

3.2.2.3 Case study on CORDIC

This work takes COordinate Rotation Digital Computer (CORDIC) algorithm as an instance for iterative processor. The CORDIC algorithm is an effective iterative technique for evaluating complex functions [85]. Calculation of transcendental functions such as sine or cosine can be achieved by using only shifter and adder/subtractors. This simplicity makes CORDIC particularly suitable for hardware implementations [86].

The basics of CORDIC rely on that in the Cartesian coordinate system, the rotation of a vector $[x_0, y_0]$ by an angle θ can be described as:

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{3.24}$$

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = cos\theta \begin{bmatrix} 1 & -tan\theta \\ tan\theta & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{3.25}$$

CORDIC algorithms implement such operation through the iterative rotation by decomposing the angle of rotation θ into a sequence of prior known angles α_i, where

$$\theta = \sum_{i=0}^{n-1} d_i \alpha_i \quad and\ d_i = \pm 1 \tag{3.26}$$

With chosen α_i satisfying $tan\alpha_i = 2^{-i}$ [85], the i-th iteration of CORDIC is defined by (3.27), where d_i and z_i denote the rotation direction and the angle respectively, and K_i is considered as a gain factor during the i-th micro-rotation.

$$\begin{aligned} x_{i+1} &= x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} &= y_i + x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} &= z_i - d_i \cdot tan^{-1}(2^{-i}) \\ K_i &= 1/\sqrt{1+2^{-2i}} \end{aligned} \tag{3.27}$$

There are two modes of CORDIC: *rotation* and *vectoring*. In the rotation mode, initial vector (x_0, y_0) starts aligned with x-axis and is rotated till the desired angle. The rotation direction d_i is defined by $sign(z_i)$. In the vectoring mode, initial vector (x_0, y_0) is rotated until y_0 converges to zero, and the direction d_i is given by $sign(y_i)$. Fig. 3.17 describes the hardware implementation of a CORDIC elementary stage (i-th iteration). This implementation requires three adders/subtractors and two shifters. A conventional CORDIC needs N iterations to get the N word-bit precision.

Two architectures of CORDIC are considered. The first one is the conventional CORDIC presented in Fig. 3.17 and the second one is the modified virtually scaling-free adaptative (MVSA) CORDIC described in [87] (see Fig. 3.18). In MVSA CORDIC, the final angle is achieved by rotating the vector in single direction. As the angle is approximated as a pure summation of the elementary rotation angles, this approach carries high performance CORDIC implementations. Notice that arithmetic operators constituting MVSC and conventional structures are identical. Indeed, they consist of N-bits adders/subtractors based on the same structure, thus have the same area and same number of gates.

The simulation have been performed considering different architectures of adders/ subtractors used in both conventional and MVSA CORDIC processors. Ripple carry adder (RCA) is the simplest carry propagation adder leading to a low operation speed. Carry-skip adder(CSKA) accelerates the speed thanks to redundancy [88]. Carry look-ahead adder (CLA), also referred as parallel prefix adders (PPAs) is another fast adder suitable for high-speed application. Some advanced PPAs are also explored like Kogge-stone adder

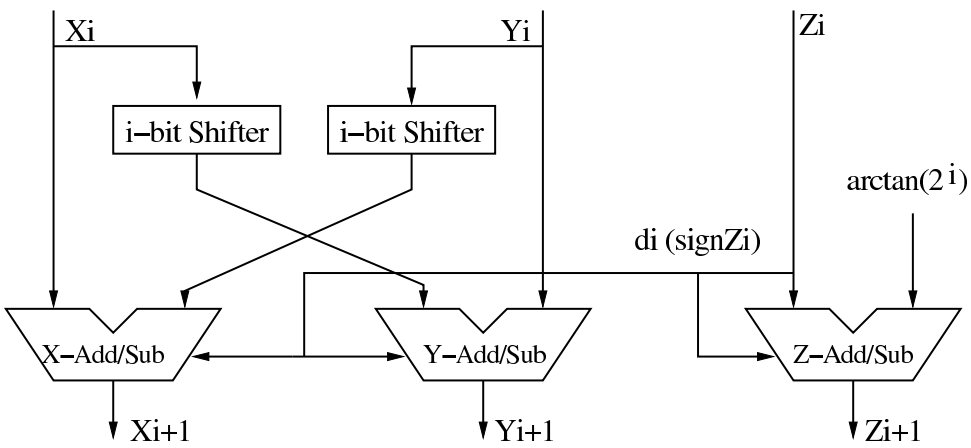


Figure 3.17: Elementary iteration of CORDIC.

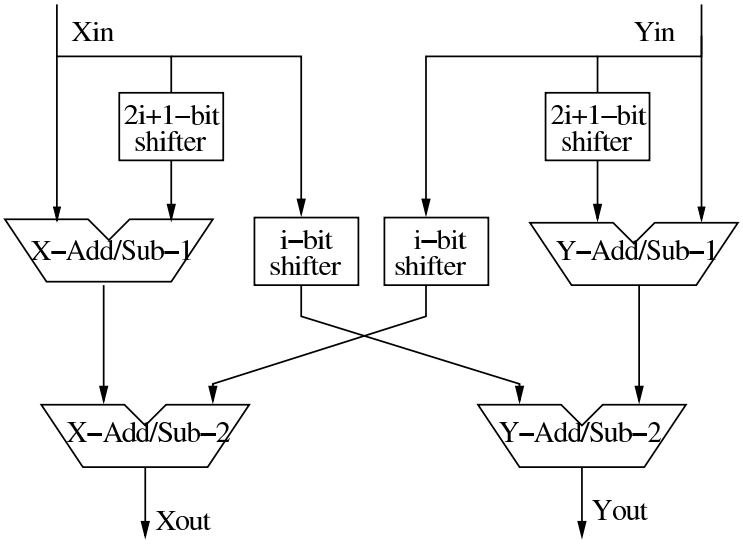


Figure 3.18: Elementary rotational iteration of Scaling-free CORDIC.

(KSA) [89], Sklansky adder (SKA) [90], Bren-kung adder (BKA) [91]. More details of these adders are presented in Appendix A.

The next paragraphs present the results obtained with the proposed approach. The simulations have been done with fault probability $P = 0.05$ [80], but the method applies to any probability value. Since CORDIC processors are completely symmetric, only the analysis of the datapath y is presented. Notice that, given that we concern about transient faults occurring on arithmetic operators, the shifters are assumed being fault-free.

The fault impact coefficient in **Consider 1** is illustrated in Fig. 3.19, which shows the contributions of three factors: iteration (1 to N), bit position (0 to $L-1$) and faulty operator (X-Add/Sub, Y-Add/Sub, Z-Add/Sub, etc). We unfold every iteration in range of bit position starting with MSB. To compare the two CORDICs used, the fault impact coefficient is normalized by the biggest value among two models in (3.19).

It can be observed that in conventional CORDIC:

- Impact keeps steady in terms of iterations if the fault takes place in Y-Add/Sub, but attenuates quickly with the increasing of iterations if the fault occurs in two others arithmetic operators;
- For faults occurring in X-Add/Sub and Z-Add/Sub, iteration influence is stronger than bit position.
- As expected, faults in MSBs produce more remarkable deviations than those in LSBs;
- Z-Add/Sub for angle computation has the smallest influence, while Y-Add/Sub has the most significant.

While in MVSA CORDIC:

- Iteration powerfully influences the impact of faults taking place in Y-Add/Sub-1,2;
- Same as conventional CORDIC, the biggest impact is carried by a fault in MSB;
- Fault in Y-Add/Sub-1,2 always brings in the worst result;
- X-Add/Sub-1 and Y-Add/Sub-1 are no longer a constraint after the 8th-iteration;

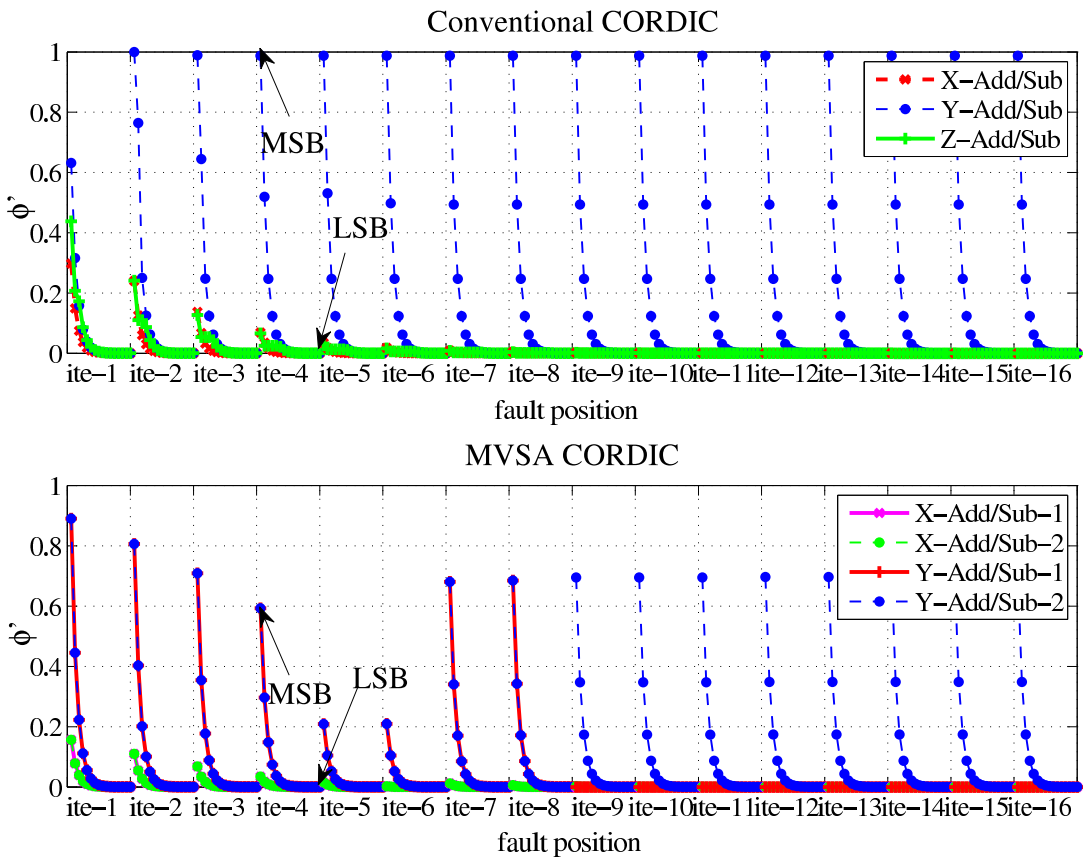


Figure 3.19: Normalized fault impact coefficient of two different CORDICs: Conventional CORDIC and MVSA CORDIC (datapath y).

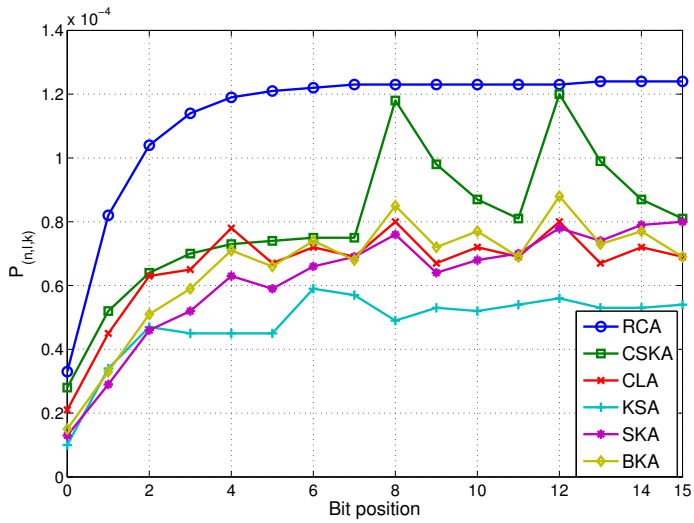
It is noted that the operators do not have the same impact on the fault tolerance of the CORDIC processor. These observations can be used for a selective hardening. The operators can be classified by producing an "eligibility rank" for the fault tolerance improvement. Such a rank allows a better trade-off between the penalty (area overhead) and the gain (improvement of fault tolerance).

Moreover, we also remark that MVSA CORDIC is less vulnerable to faults. In fact, in conventional CORDIC all N iterations are required. Instead of this, the amount of effective iterations depends on input angles for MVSA CORDIC. A fault in one iteration can affect the final result only when this iteration is "active". As a consequence, MVSA CORDIC is more tolerant to faults than conventional CORDIC.

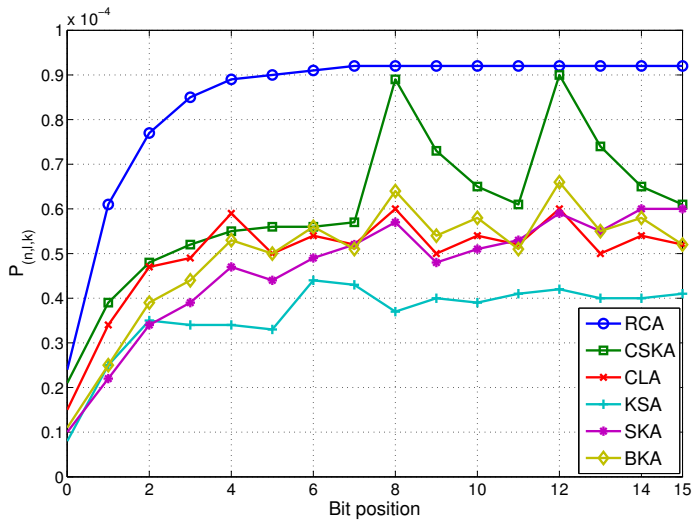
Figure 3.20 shows the probability of a fault occurrence applied in **Consider 2**, which is unconstrained by iterations and faulty operators. Observe that the probability of faults in LSB is always inferior to those of more significant bits. This is because, given the architecture, errors in less significant bits can be propagated and therefore they affect more significant bits while error propagation in opposite direction is impossible. This figure also confirms that the probability $p_{(n,k,l)}$ is strongly dependent on the implemented architecture.

The fault impact coefficients related to the operators for different architectures are listed in Table 3.3 and 3.4 . On one hand, the impact of a given architecture varies with the operator. On the other hand, the impact of a given operator changes with implemented architecture. The two tables show that RCA is the worst case for all operators, while KSA is the best one.

Table 3.5 presents the eligibility rank given in (3.21) assuming that c_Ψ, c_A, and c_T are set "1". The column entitled "Complexity" contains the Gate-count × gate-delay product



(a)



(b)

Figure 3.20: Fault probability in different bit position ($P = 0.05$): a. Conventional CORDIC; b. MVSA CORDIC.

Table 3.3: Operator impact coefficient of Conventional CORDIC (normalized)

Implemented architecture	*Operator*		
	X-Add/Sub	Y-Add/Sub	Z-Add/Sub
RCA	0.0519	1.0000	0.0784
CSKA	0.0367	0.7066	0.0573
CLA	0.0295	0.5675	0.0448
KSA	0.0225	0.4334	0.0340
SKA	0.0328	0.6314	0.0486
BKA	0.0306	0.5819	0.0468

Table 3.4: Operator impact coefficient of MVSA CORDIC (normalized)

Implemented architecture	*Operator*			
	X-Add/Sub-1	X-Add/Sub-2	Y-Add/Sub-1	Y-Add/Sub-2
RCA	0.0191	0.0194	0.2262	0.4896
CSKA	0.0137	0.0139	0.1615	0.3496
CLA	0.0110	0.0111	0.1298	0.2810
KSA	0.0085	0.0086	0.0999	0.2163
SKA	0.0122	0.0124	0.1446	0.3130
BKA	0.0114	0.0116	0.1346	0.2912

Table 3.5: Eligibility rank of different implementations

Implemented architecture	Complexity	*Operator*
		X-Add/Sub
RCA	1.92	6
CSKA	1.23	4
CLA	1.72	5
KSA	1.37	1
SKA	1.04	2
BKA	1.24	3

(normalized) extracted from [92]. The last column contains the rank of architectures for a given operator X-Add/Sub. This rank result stays the same for other operators. It is obvious that although SKA shows attractive performance in area and delay, it will not be the best choice due to its weakness in fault tolerance.

3.2.3 Multiple faults on CED circuits

CED techniques are widely applied to detect errors occurring during normal circuit operation. Various CED schemes have been presented in last decades [45, 47, 48, 93]. Traditionally, CED schemes are characterized under the hypothesis of single fault [44, 94]. However, the influence of multiple faults is no more negligible as circuits scale down to nanometer dimensions [95]. Multiple faults analysis of CED schemes have been only reported in few literature and most of them assume that faults do not affect fault-check parts [96]. Vasconcelos *et al.* proposed an analysis method based on fault injection and functional simulation approach [97]. The method assumes that faults can occur independently on all components in CED circuits. Nevertheless, the time complexity of this approach is exponential with the number of binary inputs and the number of gates in circuit. The following sections present how to apply PTM for CED circuit and an alternative method is presented for reducing the computational complexity.

3.2.3.1 Functional reliability of CED circuits

The general scheme of a CED structure is composed of three blocks: a target function F, a function predictor F_p (which predicts some special characteristics z of F's output y) and

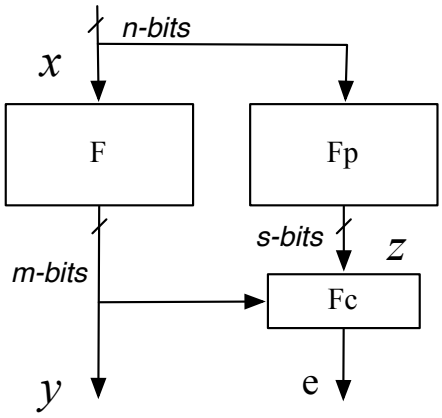


Figure 3.21: Concurrent error detection circuit.

a function checker F_c (which checks if these characteristics are satisfied by F's output), as seen in Fig. 3.21. The checking of F_c results in an one-bit output e. In the remainder of this paper, it is assumed that $e = 0$ when no error is detected and $e = 1$ otherwise. This is solely a convention and does not mean a restriction. The analysis of such a CED shows that it can produce four exclusive events:

E_1: when the checker indicates a correct operation and the circuit output is correct;

E_2: when the checker indicates an incorrect operation and the circuit output is incorrect;

E_3: when the checker indicates an incorrect operation and the circuit output is correct;

E_4: when the checker indicates a correct operation and the circuit output is incorrect;

Functional reliability is defined as the ability of a system or component to perform its required functions under stated conditions for a specified period of time, even in presence of faults [98]. Therefore, the functional reliability of a CED circuit can be stated to relate to the probability of producing only the events E_1 or E_2. Consequently, the functional reliability is expressed as:

$$R_{CED} = p(E_1) + p(E_2) \tag{3.28}$$

3.2.3.2 CED analysis by fault injection

The principle of the approach described by Vasconcelos *et al.* is to analyze the CED circuit under all possible fault configurations for each possible input vector. The framework shown in Fig. 3.22 is applied to examine the aforementioned four events. As illustrated in Fig. 3.22, a fault free circuit works simultaneously as a reference for fault-prone CED circuit. The correctness of output is indicated by f_i in which the value "1" means a correct y, and vice versa. Each event is related to a combination of $\{e, f_i\}$. For example, E_1 is considered occurring when both f_i and e are "1".

The computation of each event's probability is based on *Probabilistic Binomial Reliability* (PBR)(see Section 3.1.4.3) as follows:

$$P(E_i) = \sum_{k=0}^{G} p(k)c_k(E_i), \quad i \in 1, 2, 3, 4 \tag{3.29}$$

where $c_k(E_i)$ represents the percentage of occurring event E_i. Consequently, such a method suffers from the the same challenge as in PBR approach.

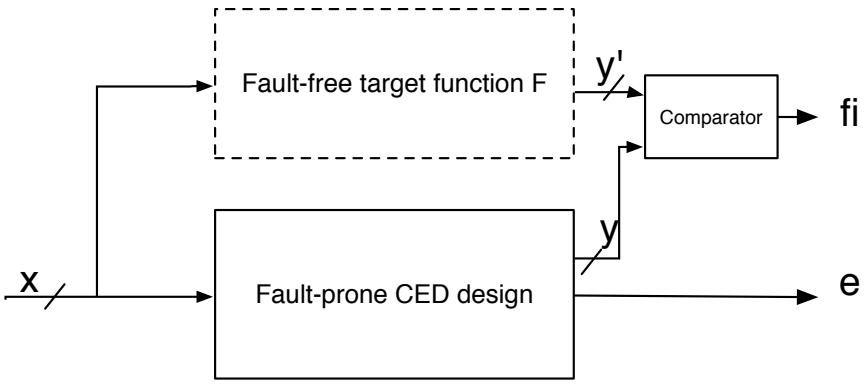


Figure 3.22: Framwork proposed in method [97].

3.2.3.3 PTM based approach

In order to apply the basic PTM approach in CED circuit analysis, let us first consider rewriting the expression (3.28) to show the probability of occurrence of inputs and outputs as previously defined in Section 3.1.4.1. The expression (3.30) is produced by doing this, where $p(i)$ stands for the probability of $x = x_i$ and $p(j \in E_1|i) + p(j \in E_2|i)$ is related to a proper functioning of the CED for $x = x_i$. In other words, it represents the probability of having a proper $[e, y]_j$ given that the input is x_i.

$$\begin{aligned} R &= \sum_j \sum_i p(i)\,[p(j \in E_1|i)) + p(j \in E_2|i)] \\ &= \sum_j \sum_i p(i) \cdot PTM_{CED}(i,j) \cdot ITM_{CED}(i,j) \end{aligned} \tag{3.30}$$

In order to illustrate this idea, an OR gate is taken as the function F. The corresponding ITM_{CED} is presented in Fig. 3.23. We remark that the left half of matrix ITM_{CED} (columns for $e = 0$) is the ITM_{OR}, while the right half (columns for $e = 1$) is the logic complement of ITM_{OR}.

$$\begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} \overbrace{00 \quad 01}^{e=0} & \overbrace{10 \quad 11}^{e=1} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \end{array} = [ITM_{OR}, \overline{ITM_{OR}}]$$

Figure 3.23: ITM_{CED} for an OR gate as function circuit F.

This can be generalized in a straightforward way for any CED circuit. Therefore, assuming F with n-bits input and m-bits output, the functional reliability can be expressed as follows:

$$\begin{aligned} R = &\sum_{j=0}^{(2^m-1)} \sum_{i=0}^{(2^n-1)} p(i) \cdot PTM_{CED}(i,j) \cdot ITM_F(i,j) \\ &+ \sum_{j=2^n}^{(2^{m+1}-1)} \sum_{i=0}^{(2^n-1)} p(i) \cdot PTM_{CED}(i,j) \cdot \overline{ITM_F(i,j)} \end{aligned} \tag{3.31}$$

3.2.3.4 An alternative approach

The following paragraphs present an alternative approach for estimating R. The objective is to reduce the computational complexity. For this, we define the events A and B:

Definition 1 *The event A is the occurrence of a correct circuit output y.*

Definition 2 *The event B is the occurrence of a checker output e indicating no error.*

Let $p(y_j^c)$ be as the probability of $y_j \in A$. According to the joint probability theory, the probability of E_1 is represented in (3.32), where $\coprod$ depicts the disjoint union.

$$\begin{aligned} p(A \cap B) &= p\Big(\coprod_j y_j^c \cap B\Big) \\ &= \sum_j \big(p(y_j^c) \cap B\big) \\ &= \sum_j p(y_j^c) \cdot p\big(B|y_j^c\big) \end{aligned} \tag{3.32}$$

The following equalities can be noticed, where $p(z_k)$ is the probability that $z = z_k$.

$$p(y_j^c) \cdot p\big(B|y_j^c\big) = \sum_k p(y_j^c) \cdot p(B|y_j^c \cap z_k) \cdot p(z_k|y_j^c) \tag{3.33}$$

We now analyze the dependency of z with respect to y_j^c. Based on the law of conditional probability and the law of total probability, (3.34) and (3.35) are obtained.

$$p(z|y_j^c) = \frac{p(z \cap y_j^c)}{p(y_j^c)} \tag{3.34}$$

$$p(y_j^c) = \sum_i p(x_i) \cdot p(y_j^c|x_i) \tag{3.35}$$

Since faults are considered to occur independently on F or F_p, y and z are independent for a given x_i:

$$\begin{aligned} &p(z \cap y|x_i) = p(z|x_i) \cdot p(y|x_i) \\ &\Rightarrow p(z \cap y) = \sum_i p(x_i) \cdot p(z|x_i) \cdot p(y|x_i) \end{aligned} \tag{3.36}$$

Applying (3.36) into (3.34) we have :

$$\begin{aligned} p(z|y_j^c) &= \frac{\sum_i p(x_i) \cdot p(y_j^c|x_i) \cdot p(z|x_i)}{p(y_j^c)} \\ &= \sum_i p(y_j^c|x_i) \cdot \frac{p(x_i)}{p(y_j^c)} \cdot p(z|x_i) \end{aligned} \tag{3.37}$$

Rewritting (3.32) by replacing $p(z_k|y_j^c)$ in (3.33) with that given in (3.37), the probability of E_1 can be expressed as shown in equation (3.38).

$$p(A \cap B) = \sum_j \Big(\sum_k p\left(B|y_j^c \cap z_k\right) \cdot \sum_i \big(p(y_j^c|x_i) \cdot p(z_k|x_i) \cdot p(x_i)\big)\Big) \tag{3.38}$$

The probabilities in the equation (3.38) can be obtained from PTMs of F, F_p and F_c as the following:

$$p\left(B|y_j^c \cap z_k\right) = PTM_{F_c}((m \times j + k), 0) \tag{3.39}$$

$$p(y_j^c|x_i) = PTM_F(i,j) \cdot ITM_F(i,j) \tag{3.40}$$

$$p(z_k|x_i) = PTM_{F_p}(i,k) \tag{3.41}$$

Applying these equalities to (3.38) allows us to rewrite E_1 in (3.42). In an equivalent manner, we can get E_2 as given in (3.43). We notice that only PTMs of sub-circuits are required instead of that of the overall CED circuit.

$$p(E_1) = \sum_{j=0}^{2^m-1}\left(\sum_{k=0}^{2^s-1} PTM_{F_c}((m \times j + k), 0) \cdot \sum_{i=0}^{2^n-1}\left(PTM_F(i,j) \cdot PTM_{F_p}(i,k) \cdot ITM_F(i,j) \cdot p(i)\right)\right) \tag{3.42}$$

$$p(E_2) = \sum_{j=0}^{2^m-1}\left(\sum_{k=0}^{2^s-1} PTM_{F_c}((m \times j + k), 1) \cdot \sum_{i=0}^{2^n-1}\left(PTM_F(i,j) \cdot PTM_{F_p}(i,k) \cdot \overline{ITM_F}(i,j) \cdot p(i)\right)\right) \tag{3.43}$$

Based on the aforementioned mathematical developments, the workflow of the alternative approach is described in Fig. 3.24. The first task in this flow (*PTM core*) consists in calculating PTMs of the subcircuits F, F_p and F_c from their netlists and the respective elementary PTMs and ITMs.

The PTMs of these three sub-circuits are sent to *CED reliability estimator*. It computes the probability of events E_1 and E_2 for a given y_j and must be executed 2^m times, according to equations (3.42) and (3.43). Notice that the result of every iteration is accumulated by *Accumulator*.

3.2.3.5 Comparison with the basic PTM based approach

Suppose PTMs and ITMs of sub-circuits F, F_p and F_c are already available. Meanwhile, denote $\oplus$ as the computational complexity for adding two numbers and $\otimes$ as the computational complexity for multiplying two numbers.

In the case of method described in Section 3.2.3.1, the complexity is that of obtaining the global matrix PTM_{CED} and adding the appropriate elements in accordance with the equation (3.31). On the basis of PTMs approach, the PTM of two blocks b_1 and b_2 in cascade results from the inner product of PTM_{b_1} and PTM_{b_2}. In the case of parallel structure, the resulting PTM comes from the tensor product of PTM_{b_1} and PTM_{b_2}.

Figure 3.25 shows the partitioning of the CED circuit in order to calculate the overall PTM_{CED}. Utilizing the rules above on the blocks in a level u leads to the values T_u in Table 3.6. Similarly, the complexities related to the cascade involving different levels are shown in Table 3.7.

The total complexity T_a thus is:

$$T_a = 2^{n+m+1}(2 \times \otimes + \oplus) - \oplus + \sum_u T_u + \sum_v T_{1\to v} \tag{3.44}$$

In the case of method described in 3.2.3.4, the total complexity comes straight from the computation of (3.42) and (3.43):

$$T_b = 2^{m+n+s+1}(2^2 \times \otimes + \oplus) - \oplus \tag{3.45}$$

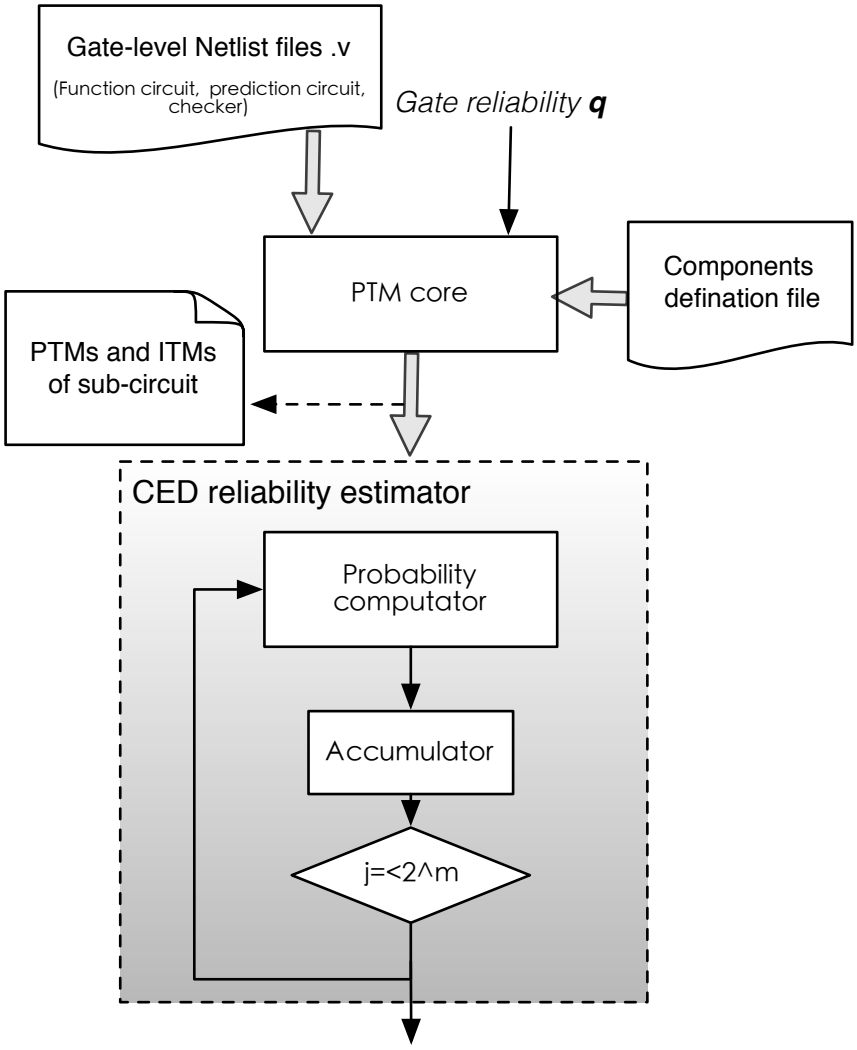


Figure 3.24: The workflow of the proposed approach.

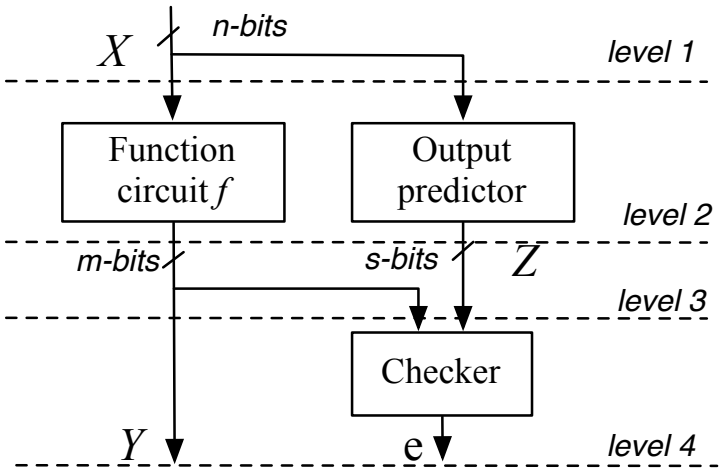


Figure 3.25: Level partition of CED circuit.

Table 3.6: Computational complexity to obtain PTM for a given level

T_1	$(2 \times 2^2)^n \times \otimes = 2^{3n} \times \otimes$
T_2	$2^{n+m} \times 2^{n+s} \times \otimes = 2^{2n+m+s} \times \otimes$
T_3	$(2 \times 2^2)^m \times (2^2)^s \times \otimes = 2^{3m+2s} \times \otimes$
T_4	$(2^2)^m \times (2^{m+s} \times 2) \times \otimes = 2^{3m+s+1} \times \otimes$

Table 3.7: Computational complexity to obtain PTM for a cascade of levels

$T_{1\to2}$	$2^{n+2n+(m+s)}(\oplus + \otimes) = 2^{3n+m+s}(\oplus + \otimes)$
$T_{1\to3}$	$2^{2n+(m+s)+(2m+s)}(\oplus + \otimes) = 2^{2n+3m+2s}(\oplus + \otimes)$
$T_{1\to4}$	$2^{(m+s)+(2m+s)+(m+1)}(\oplus + \otimes) = 2^{4m+2s+1}(\oplus + \otimes)$

We can easily conclude that $T_b \ll T_a$, which confirms a significant gain in computation efficiency.

3.2.3.6 Case Study: Adders

The proposed methodologies have been implemented with Matlab [99], while Modelsim [100] has been used as the simulation platform for a fault injection approach [97]. Simulation results have been obtained on a Linux workstation with a 2GHz AMD Athlon microprocessor and 2GB RAM. The values of gate reliabilities are assumed identical in simulations. This assumption is taken only as an example and dose not impact the evaluation of proposed method. Of course, different values for gates reliabilities can be used for better representation of real context [101]. It is therefore noteworthy that the following obtained results will change if more realistic gate models are applied.

Remind that in fault injection approach, the result accuracy can be improved by increasing the maximum number of considered simultaneous faults, but the computational complexity would be dramatically increased as well. In order to optimize the procedure of simulations, this work assume the maximum number of simultaneous faults to be 3 in the fault injection approach.

Three different CED schemes adopted in a ripple carry adder (RCA) operator are considered: an identical duplication adder (noted as RCA_Dup), a diverse duplication adder (noted as RCA_Div_Dup) and a parity checking adder (noted as RCA_Parity). The RCA_Dup implements the FA as presented in Fig. 3.26(b) for both target function F and predictor F_p, while the FA in Fig. 3.26(b) is applied to predictor F_p in RCA_Div_Dup. The checker used in these adders is two-rail checker which is a classical totally self-testing checker with respect to all single internal faults [102].

Table 3.8 shows the run-time comparison. It is noteworthy that conventional PTMs based method is impracticable for RCA_Dup and RCA_Div_Dup due to excessive memory requirements. The obtained results verifies the remarkable performance of the solution described in 3.2.3.4, particularly for identical duplication circuit.

Fig. 3.27 depicts a detailed evaluation of proposed solution's run-time. We recognize that PTM calculations are the most time-consuming task, while *CED reliability estimator* task (noted **Estimator** in the figure) contributes very little to the implementation cost. Indeed, the computational complexity of task **Estimator** depends on the PTM matrix size of each circuit part. However, these matrices are obtained through PTM approach, thus the computational complexity strongly depends on the circuit tolopogy (the number

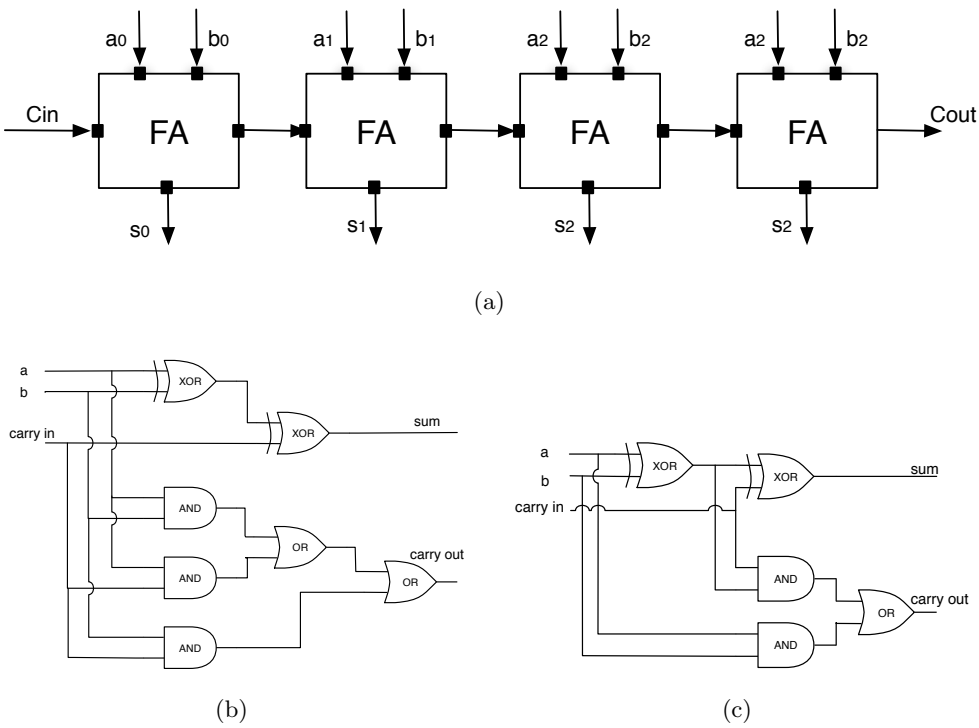


Figure 3.26: a. Structure of 4-bits RCA; b. Implementation of full adder by b. 7 gates and c. 5 gates.

Table 3.8: Time consumptions (s)

Circuit type	Evaluation method			Gate
	Fault injection [97]	Basic PTM	Alternative	Nb.
RCA_Dup	7398.1	NA	18.6	86
RCA_Div_Dup	6345.7	NA	19.3	78
RCA_Parity	367.2	50.3	4.7	32

of gates and how the gates are interconnected). In other words, a more effective *PTM core* will significantly decrease the run-time [103].

The proposed method also allows to study events E_3 and E_4 mentioned in Section 3.2.3.1. Fig. 3.28(a) presents the probability of outputs events in a RCA_Dup scheme, where the black line and blue line represent the reliabilities of RCA and RCA_Dup, respectively. We can see this scheme does not harden original circuit if $q > 0.976$ where the probability of E_3 is very high, over 30%. Indeed, a large number of correct operations are mistakenly assumed to be erroneous due to faults on predictor or checker. The underestimation performed by this scheme may lead to important time penalty. Meanwhile, the figure points out that the probability of occurring silent errors (i.e. occurrence of event E_4) is highly increased when multiple faults become the majority case, as expected. From a system security point of view, silent errors are an issue as they mean potential failures without any cautions. Figure 3.28(b) shows the evaluation of functional reliability and silent errors probability. We focus on the gate reliability q higher than 0.9 considering the actual design. RCA_Div_Dup scheme slightly outperforms RCA_Dup scheme both on both area and reliability. However, these schemes seem to be powerless in case of high reliable gates. This phenomenon does not suggest they are useless, the outputs are assured any-

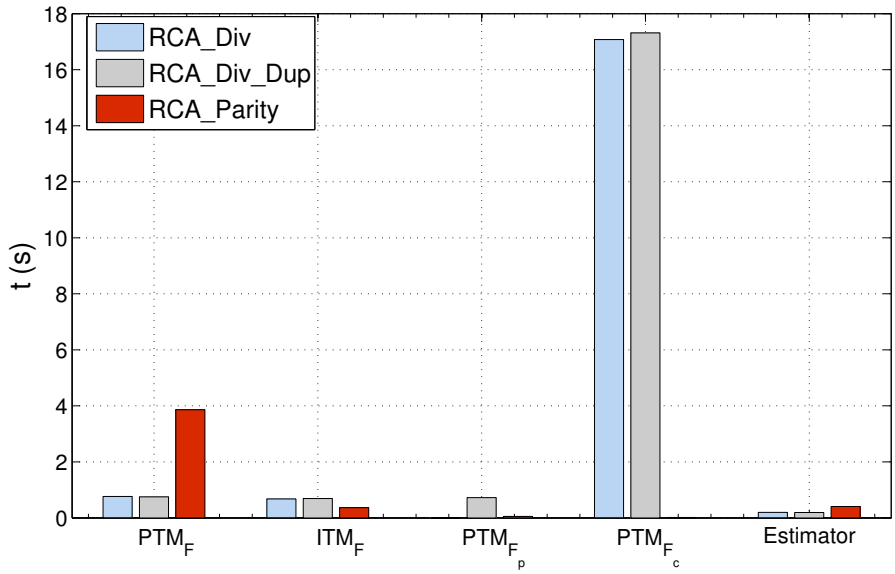


Figure 3.27: Run-time partition of improved method.

way. The induced time penalty must be carefully discussed for high frequency applications. Parity checking scheme shows a remarkable performance and overcomes when $q > 0.95$. Nevertheless, it is very prone to silent errors, more than twice compared to duplication schemes when $q > 0.97$.

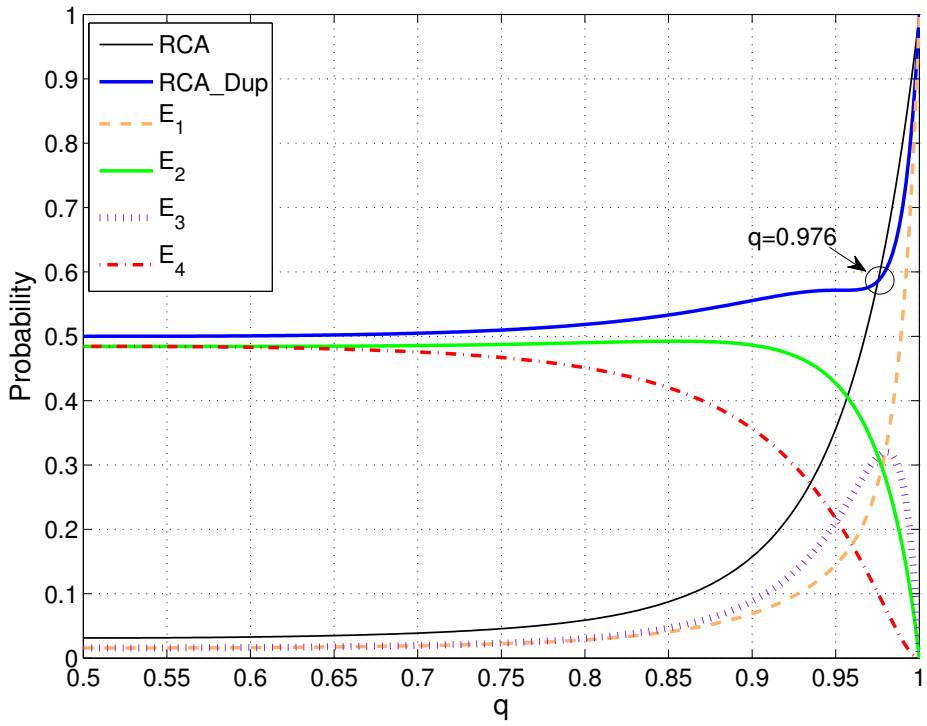
A unique transient fault on a logic gate may produce multiple faults at the output of a logic block. Parity checking can deal with all odd faults, but even faults stay undetectable. Conversely, duplication schemes allow to cope with multiple faults. For this reason, they are useful solutions for applications requiring high security, despite of the high area overhead.

3.3 Analysis of aging effects

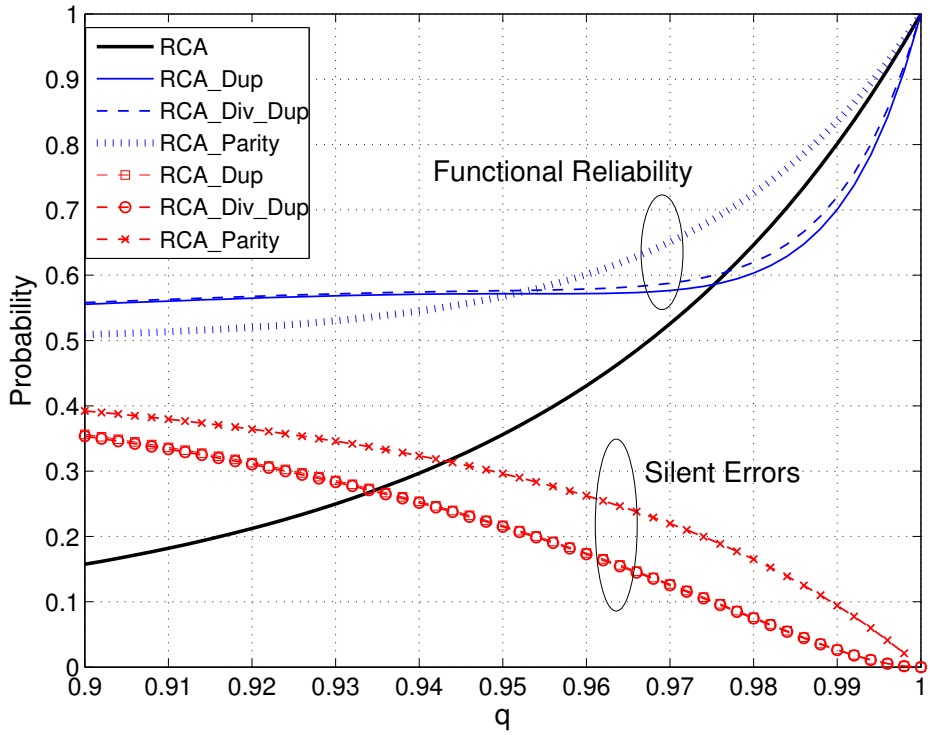
Traditional digital IC's design tasks include behavioral description (VHDL/Verilog), functional simulation, RTL synthesis and post-layout simulation. In order to consider performance degradation due to scaling down, the reliability should be included in this design flow. Since a non-critical path can become a critical path with the time progress due to NBTI effect [104], we particularly concentrate on the aging effects on critical path and near critical path of circuits [105]. We propose a design flow that includes aging estimation taking into account both NBTI and HCI mechanisms. With the help of this flow, it is possible to aging-aware simulate the circuit netlist generated from RTL synthesis. Binary adders with 65 nm CMOS in different architectures are studied in order to demonstrate this flow. We show a comprehensive simulation study of aging effects in binary adders with 65 nm CMOS.

3.3.1 Aging-aware design flow

SPICE simulators are widely used to evaluate timing performance at gate level. Nevertheless, they are not suitable to directly simulate aging induced timing degradations in complex circuits due to low computation efficiency [106]. The proposed aging-aware design flow is based on the gate-level netlist generation of traditional synthesis, as illustrated in Figure 3.29. Traditional RTL synthesizers can produce a structural (i.e. logic or transistor level) description based on a HDL description for a given digital block.



(a)



(b)

Figure 3.28: a. Outputs events probabilities evaluation as a function of gate reliability; b. Comparison of CED schemes applied to RCA.

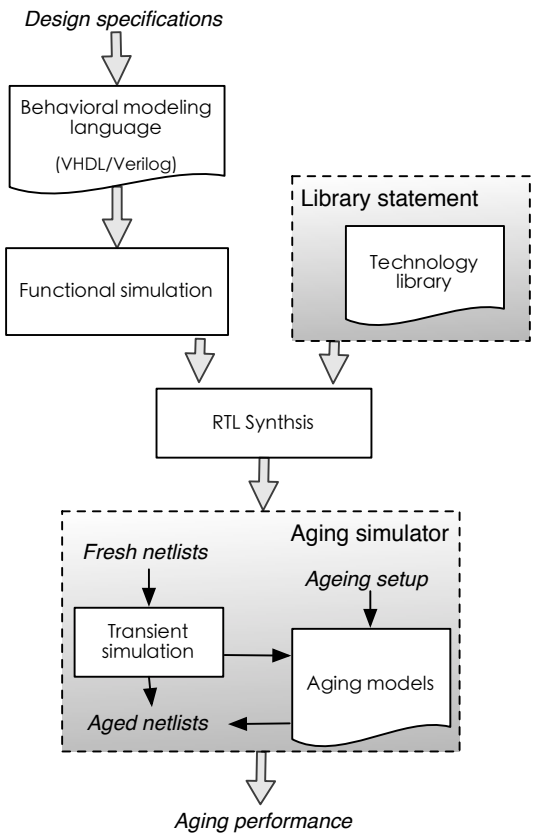


Figure 3.29: Aging-aware design flow.

According to design specifications, the *Behavioral modeling language* for digital circuits (e.g., VHDL, Verilog) is used to describe the functions of logic circuits. Their function correctness is verified by the *Functional simulation*. The *Library statement* includes necessary technology files. A gate-level netlist is generated by *RTL synthesis* and re-simulated by the *Aging simulator*.

In this thesis, NBTI and HCI aging models are considered available on Eldo simulator from Mentor Graphics [107]. With an initialization of aging setup (e.g., aging model selection, aging stress duration), a transient simulation is performed to evaluate the stress on each transistor [108]. Then, aged netlist which contains degraded information is generated and can be further applied to post-layout simulations. Finally, both ideal and aged synthesis report can be investigated by designers.

3.3.2 Case study: 65 nm adders

The following paragraphs present the ASIC implementation results as well as the aging effect performance of different adders. Six architectures are taken into account in this work (RCA, CSA, CLA, KSA, SKA, BKA presented in Appendix A). The 65 nm CMOS technology is utilized to demonstrate the proposed flow. Even the 65 nm technology is not highly susceptible to aging effects, our flow allows to explore how they affect adder circuits anyway. The simulation environment is characterized with room temperature of 27°C and 1.2 V supply voltage. The input duty cycle is set to 50%. Notice that the experimental results are obtained without the consideration of input probability. We conservatively assume that all the PMOS and NMOS transistors in the adders are affected equally by NBTI and HCI respectively.

Table 3.9: Performance comparison of adders (1)

Adder Architecture	Area (μm^2)			Delay (ps)			Power		
	4	8	16	4	8	16	4	8	16
RCA	49.9	99.8	199.7	524.8	992.4	1926.6	2.71	5.42	11.08
CSA	96.7	193.4	386.9	372.7	600.7	1056.7	7.24	14.48	28.96
CLA	64.5	130,0	257.9	348.2	651.4	1257.8	4.71	9.41	18.82
KSA	64.0	160.2	402.5	350.4	532.5	601.2	3.87	9.15	21.96
SKA	57.7	132.1	302.6	416.6	550.2	819.6	3.59	8.03	17.75
BKA	57.7	125.8	268.3	417.3	687.7	911.6	3.96	8.79	19.42

Table 3.10: Performance comparison of adders (2)

Adder Architecture	ADP (Normalized)			PDP (Normalized)			Dominant effect
	4	8	16	4	8	16	
RCA	1.17	1.36	1.59	1.05	1.22	1.62	NBTI
CSA	1.61	1.60	1.69	1.80	1.97	2.25	NBTI
CLA	1.00	1.16	1.34	1.21	1.39	1.79	NBTI
KSA	1.00	1.17	1.00	1.00	1.10	1.00	HCI
SKA	1.07	1.00	1.03	1.10	1.00	1.10	HCI
BKA	1.07	1.19	1.01	1.22	1.37	1.34	HCI

3.3.2.1 Synthesis results

Tables 3.9 and 3.10 show the performance comparison, where ADP and PDP indicate relative area delay product and power delay product, respectively. Different word length adders are reported (4 bits, 8 bits and 16 bits). From this table, PPAs are the best tradeoffs considering area, delay and power. It is noteworthy all three adders have an interesting delay performance, especially in the case of long word length. Thus, they are referred to fast adders in this work. In particular, BKA shows an attractive balance between area and delay.

As it can be seen from Table 3.10, the impacts of NBTI and HCI exhibit different feathers according to adder architectures. Aging effects can lead to delay degradation and decrease the ADP product. Instead, they have positive influences on power and thus increase the PDP. This is particularly true for HCI (see Fig. 3.30). We therefore concentrate on delay degradation. More details are shown in Fig. 3.31, where blue lines stand for the delay degradation induced by NBTI and red lines for HCI. The impacts of NBTI and HCI on adders will be further discussed in the next subsection.

3.3.2.2 Ageing consideration

As shown in Figure 3.31, both NBTI and HCI induced additional delays are time-dependent and monotonically increasing. The dominant aging effect changes with adder architecture and depends on PMOS and NMOS partitions. The fast adders are more sensitive to HCI, while no significant difference between two effects is observed for CSA. NBTI is the dominant aging effect for RCA and CLA. Particularly, CLA's delay is severely degraded by

NBTI (higher than 0.5% after 10 years stress). Remind that the simulations are performed at room temperature. NBTI degradation will get worse with temperature increasing [4], whereas HCI can not be intensified at high temperature [3].

Besides, we observe that the dominant aging effect varies with time in the case of faster adders. They become more sensible to HCI than NBTI after 2 years stress. The degradation speed caused by HCI is always faster than NBTI which makes the HCI as dominant effect in the later period, as expected according to Fig. 2.6.

Another interesting aspect is that the maximum delay degradation is not always appearing at critical path. Taking CLA as example, the most significant degradation occurs on "carry out" bit which is near the critical path (MSB of the sum). As aforementioned, NBTI can lead to the possible critical path changing. Our experimental results prove that HCI may also result in this changing (see Figure 3.32). Let us take a 8 bits KSA for instance, we notice the fresh circuit has bit "S[7]" as the critical path. However, the bit "Cout" is more sensible to HCI with the time progress, and will surpass the critical path after several years. Furthermore, the arise of delay do not always enhance the impact of aging effects. On the contrary, the impact could be reduced with increasing of delay. The 16 bits SKA suffers less degradation than the 4 bits SKA. This suggests that the aging effect is not a linear function of the impact on the individual logic gate.

Figure 3.33 depicts the performance comparison of different adders under aging effects. The observations are summarized as follows:

- Either NBTI or HCI, CLA is the most aging-sensitive adder among all architectures. For instance, a 8-bit length CLA is thrice sensible to HCI than a 8-bit length SKA. Moreover, CLA is much more vulnerable to HCI than others.
- BKA is the least NBTI-sensitive adder architectures (delay degradation$<$ 0.2%), while RCA is the most efficient architecture to resist HCI.
- Compared with the other adders, the fast adders can be the best trade-off considering both NBTI and HCI.
- The aging induced delay degradation in 65 nm CMOS adders is lower (0.4%-0.6 %) than that related to other reliability effects (e.g. process variation). In other words, this observation demonstrates that 65 nm CMOS adders are reliable respect to aging effects.

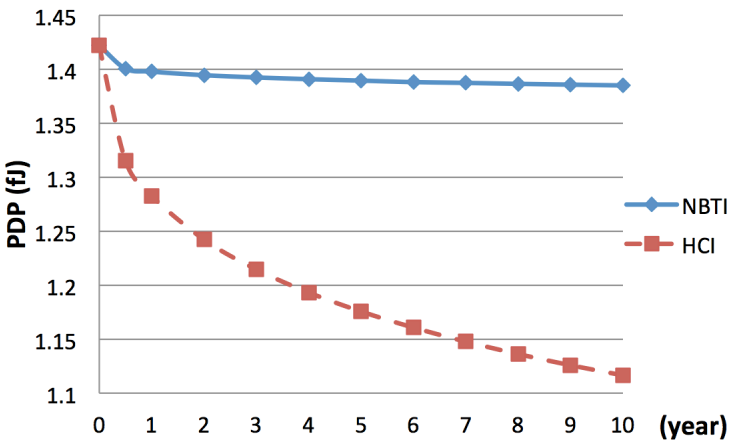


Figure 3.30: Power delay product versus year of 4-bit length RCA.

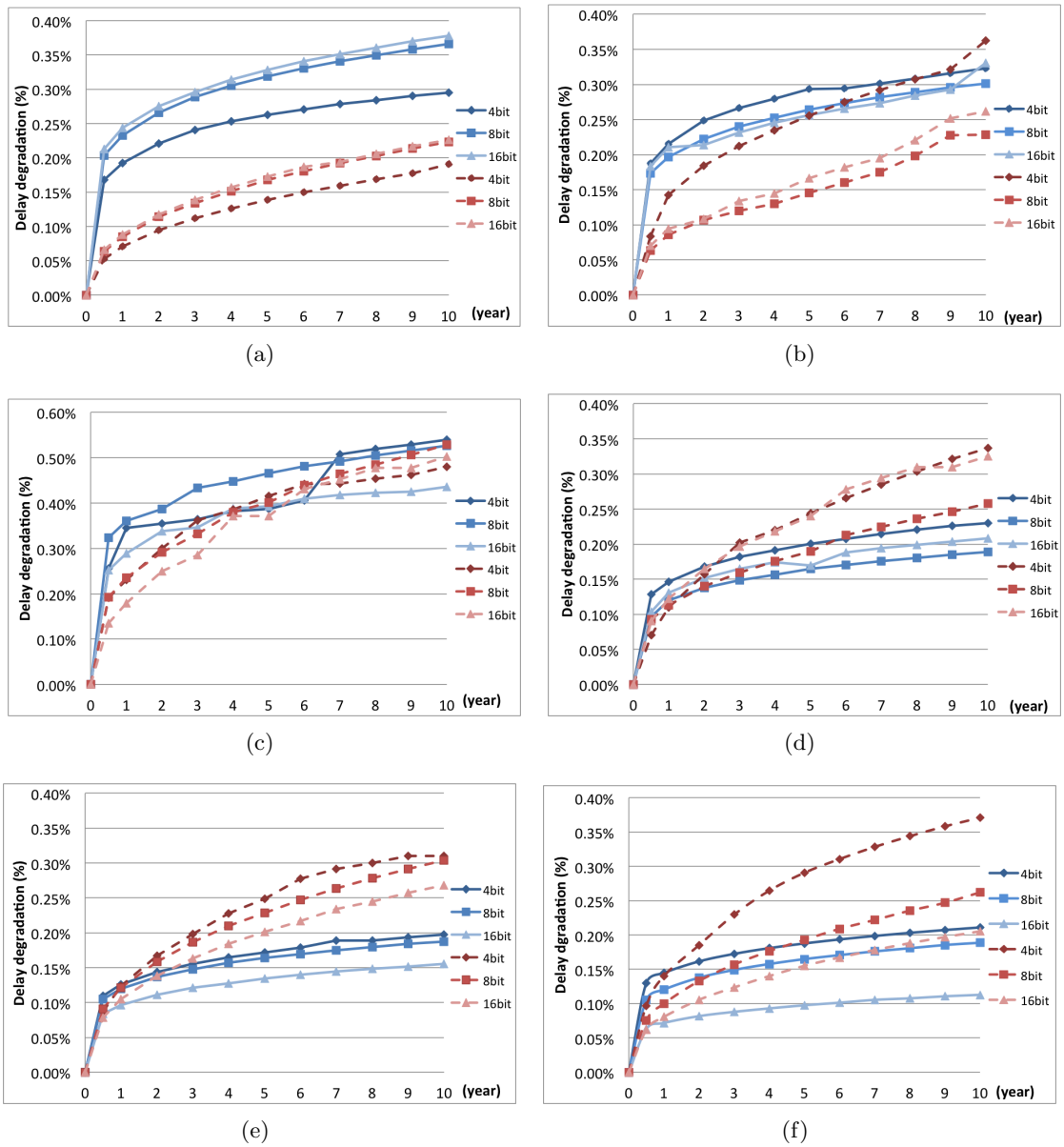


Figure 3.31: NBTI and HCI Delay degradation over time of different adders, T=300K: (a) RCA; (b) CSA; (c) CLA; (d) KSA; (e) SKA; (f) BKA; .

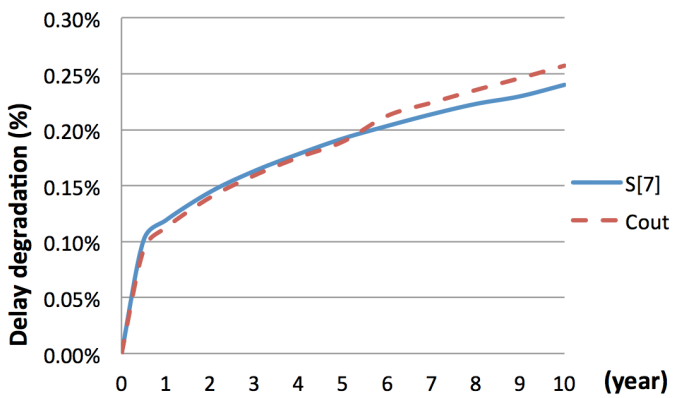


Figure 3.32: An example (8-bit length KSA) to demonstrate the critical path changing with time under the HCI effect.

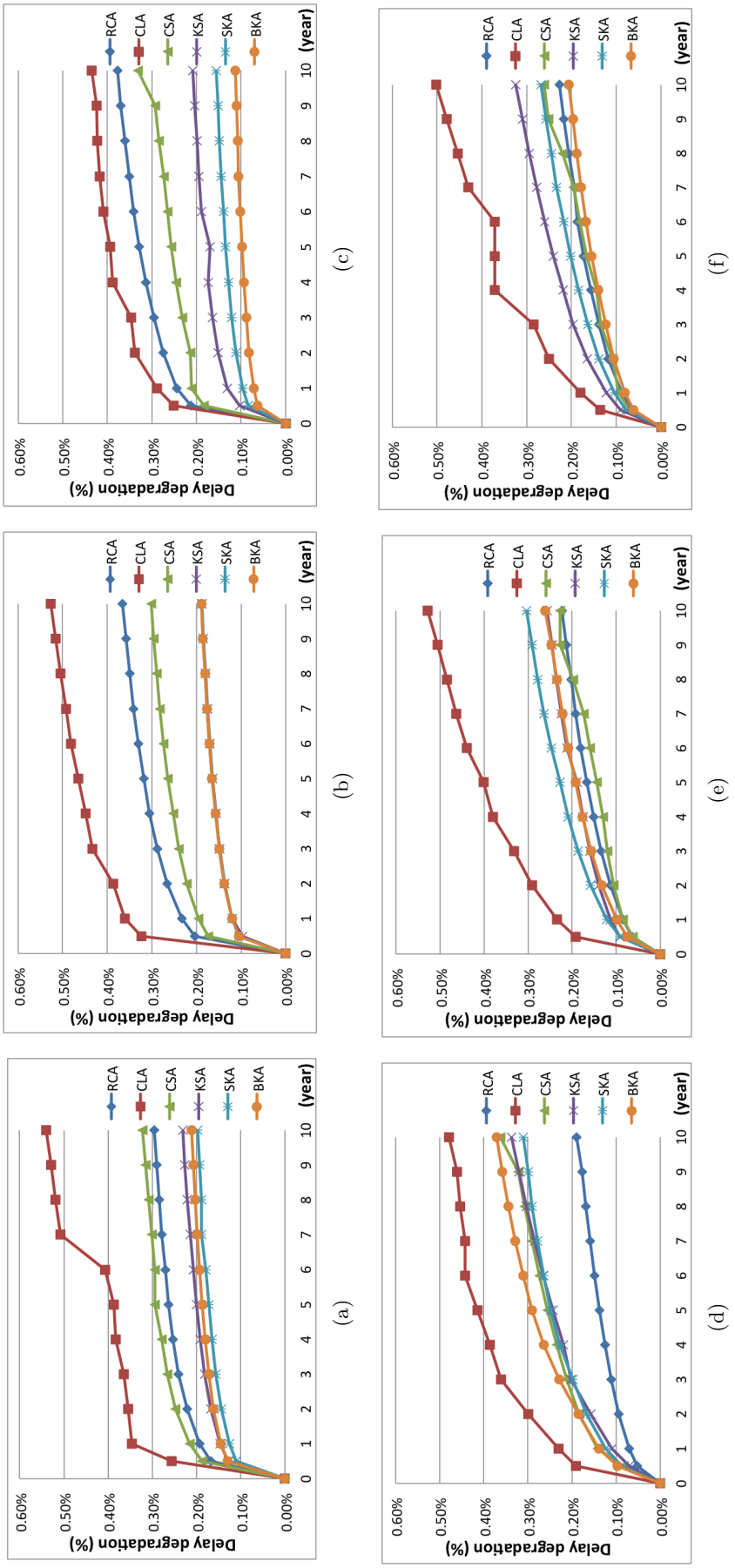


Figure 3.33: Comparison of adders' performance on delay degradation over different bit length (a) 4 Bits length, NBTI; (b) 8 Bits length, NBTI; (c) 16 Bits length, NBTI; (d) 4 Bits length, HCI; (e) 8 Bits length, HCI; (f) 16 Bits length, HCI.

3.4 Conclusions

This chapter dealt with the reliability assessment during the lifetime of digital circuits. It covered the reliability with respect to fault masking properties and the performance degradation under aging stress. The proposed method enables the evaluation of the faults' impact on arithmetic operators in iterative processor, where we noticed that MVSA CORDIC is more resistant than conventional CORDIC in considered cases. Another method based on PTM approach was presented to evaluate the circuits implementing self-checking schemes. Compared with the fault injection method, it is more efficient with an accurate analysis. Besides, we pointed out the relationship between the gate failure rate gate and its structure for a realistic modeling in reliability assessment. An aging-aware design flow was also proposed for the simulation study of aging. Binary adders were discussed and compared with respect to area, delay, power and timing performance under aging effect. The simulation results pointed out that PPAs, in particular, KSA and BKA can achieve a good trade-off among considered factors under assumptions in this thesis.

Chapter 4

Effective design for reliability

For the last decades, fault-tolerant designs have been vastly reported. The use of redundancy leads to a overcost in hardware or in time. An efficient fault-tolerant design should attain a good tradeoff between reliability improvement and cost overhead. On the other hand, although there exist various of techniques, application-specific solutions became more and more demanded for application used in harsh environment due to the limited budgets in hardware and severe performance requirements.

Advanced encryption standard (AES) [109] has been widely adopted in various critical applications in which high reliability is required [110, 111]. The existence of malicious injected faults makes the hardware processors implementing AES unable to guarantee their proper security [112]. However, despite the huge achievement against fault attacks [113, 114], low reliability of electronic devices can be foreseen even in the absence of fault attacks with the downscaling of CMOS technology [115]. This chapter explores the effective design for AES processors, in particular, the enhancement of S-Boxes that occupy three fourths of area in AES processor [116].

4.1 Effective fault-tolerant design

4.1.1 Introduction

In order to characterize different fault-tolerant techniques, let us define the reliability improvement efficiency (noted η_t) with respect to the overhead as (4.1), where R_o denotes the reliability of the original circuit, R_t represents the reliability of the circuit with a fault-tolerant technique t, and ΔC_t is the cost related to this technique.

$$\eta_t = \frac{R_t - R_o}{\Delta C_t} \tag{4.1}$$

Notice that ΔC_t can include different criteria, depending on design strategy. Equation (4.2) presents an example of a cost metric combining area, time and power consumption. K_A, K_T, and K_P are weight factors chosen by the designer according to the design priorities.

$$\Delta C_t = \Delta C_{A_t}^{K_A} \times \Delta C_{T_t}^{K_T} \times \Delta C_{P_t}^{K_P} \tag{4.2}$$

4.1.1.1 Area penalty

We define the area overhead as the ratio between extra area needed by fault tolerance and the area of original circuit A_o:

$$\Delta C_{A_t} = \frac{A_t - A_o}{A_o} \tag{4.3}$$

4.1.1.2 Time penalty

The *effective frequency* defines the frequency of getting correct outputs. Let the data throughput be 1 data per clock cycle, the *effective frequency* is then given by (4.7).

$$F_{eff} = \frac{1}{1+\Theta} \times F_t \tag{4.4}$$

F_t is the maximum frequency of the fault-tolerant circuit, and Θ is the operating time penalty. As TMR approach masks the fault and gives the correct output in one operation, thus Θ equals to zero. However, the CED scheme indicates the fault when the fault takes place and normally is followed by a correction operation. Consequently, one extra clock time is required to fix the fault. Considering this case, we define Θ of CED scheme as the probability of the detection of faults (see Section 3.2.3.1):

$$\Theta = p(E_2) + p(E_3) \tag{4.5}$$

The time overhead is therefore represented as the decrease in operating frequency as shown in (4.6).

$$\Delta C_{T_t} = \frac{F_o - F_{eff}}{F_o} \tag{4.6}$$

4.1.1.3 Power penalty

In order to take into account the operating time penalty, we define the *effective power* that represents the required power for getting correct outputs as follows:

$$P_{eff} = (1+\Theta) \times P_t \tag{4.7}$$

Then the power penalty can be described by the increase of power shown in equation (4.8).

$$\Delta C_{P_t} = \frac{P_{eff} - P_o}{P_o} \tag{4.8}$$

4.1.2 Case study: S-Box in AES

4.1.2.1 AES algorithm

In AES algorithm, inputs and outputs consist of sequences of 128 bits. The data is processed on the unit of **Byte** (i.e., 8 bits). Each Byte is represented in Galois Field $GF(2^8)$ with a specified polynomial $P = x^8 + x^4 + x^3 + x + 1$ (more details of Galois Field are presented in Appendix B). Take a Byte $b(x)$ as an example, the representation in $GF(2^8)$ is written as follows:

$$b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 \tag{4.9}$$

The internal operations are performed on 4×4 array of Bytes called **State**. One Byte in State S is denoted as $S_{r,c}$, where $0 \leq r, c \leq 3$. The four Bytes in each column of State are referred as a 32-bit **Word** W_c. Fig. 4.1 describes he relationship between input Byte, Word and State, assuming the input sequence of Bytes to be $in_0, in_1, \ldots, in_{15}$.

The AES algorithm is an iterative symmetric cryptographic standard and each round of encryption consists of four individual transformations: *SubBytes*, *ShiftRows*, *MixColumns* and *AddRoundKey*. The decryption is achieved by directly inverting the encryption transformations as shown in Fig. 4.2, where Nr represents the round number. The *roundKey*(i) is generated by a block called *key expansion*.

We now briefly explain these four transformations used in AES encryption/decryption round except the last round.

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

$\Rightarrow$

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$
w_0	w_1	w_2	w_3

Figure 4.1: Inputs and State array.

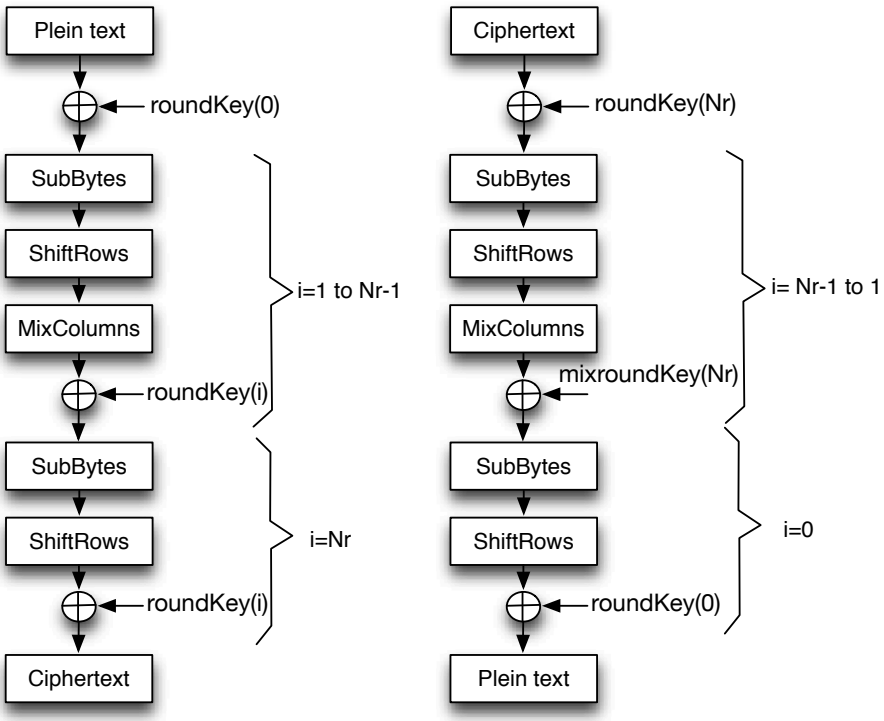


Figure 4.2: Diagram of AES encryption and decryption.

- **SubBytes**

  This nonlinear transformation performs a mapping (eight to eight bits) applied independently on each Byte of the State by utilizing a substitution table which is referred as S-Box in hardware implementation. In total, 16 identical 8×8 S-Boxes are required. This S-Box includes 16 Bytes (128 bits) values. Mathematically, this nonlinear transformation computes inverse multiplicative of each byte and is followed by an affine transformation over $GF(2)$ as shown in (4.10), where M is an 8×8 matrix and C is a 8-bit vector $\{01100011\}$.

$$S'_{r,c} = MS^{-1}_{r,c} + C \tag{4.10}$$

  The inversion multiplication is performed on $GF(2^8)$ modulo polynomial $P = x^8 + x^4 + x^3 + x + 1$. More details of implementation are mentioned in Section 4.1.2.2. The affine transformation of a byte in S-Box can be expressed in a matrix as (4.11)

and implemented by XOR and AND gates.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \tag{4.11}$$

- **ShiftRows**

  In the ShiftRows, the rows of State are cyclically shifted to left with different offset, where no change for the first row, and $r - bytes$ left shifts for the $r - th$ row as given in (4.12).

$$S'_{r,c} = S_{r,(c+shift(r,4))\ mod\ 4} \tag{4.12}$$

  This transformation has the effect of moving lower Bytes to higher and swapping the lowest with the highest as shown in (4.13).

$$\begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} => \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,1} & S_{1,2} & S_{1,3} & S_{1,0} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,3} & S_{3,0} & S_{3,1} & S_{3,2} \end{bmatrix} \tag{4.13}$$

- **MixColumns** In the third transformation, columns of State are considered to be multiplied over $GF(2^8)$ modulo $x^4 + 1$ by a constant matrix $C(x) = 03x^3 + 01x^2 + 01x + 02$ as described in (4.14).

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \tag{4.14}$$

- **AddRoundKey** The last step of AES is a simply bitwise XOR operation with a Round Key generated from *key schedule* and each Round Key consists of 4 words. These 4 words are added on each column of a State as follows, where W_{c_i} is the $i - th$ byte in $c - th$ word of a given Round Key.

$$S'_{i,c} = S_{i,c} \oplus W_{c_i} \quad i \in \{0, 1, 2, 3\} \tag{4.15}$$

4.1.2.2 S-Box implementation

S-Box has been proved the most critical design in processors embedded AES algorithm [117, 118]. The implementation can be either based on look up table (LUT) or on logic only. In LUT approach, each S-Box is independently implemented by a 256×8 bit LUT. As previously discussed, SubByte transformation uses sixteen S-Boxes performing in parallel. Consequently, 2048 bits are required, thereby lead to a high memory consumption. Logic only based approach applies composite field for S-Box and is particularly suitable for ASIC system thanks to zero memory usage and low area complexity.

In the following paragraphs, we briefly introduce the basics of composite field S-Box. The principle of logic based S-Box is that $GF(2^8)$ and the composite field $GF(((2^2)^2)^2)$

are isomorphic. $GF(((2^2)^2)^2)$ can be built from multiple extensions of degree 2:

$$\begin{cases} GF(2) \Rightarrow GF(2^2) & P_0 = x^2 + x + 1 \\ GF(2^2) \Rightarrow GF((2^2)^2) & P_1 = x^2 + x + \phi \\ GF((2^2)^2) \Rightarrow GF(((2^2)^2)^2) & P_2 = x^2 + x + \mu \end{cases}$$

This makes S-Box very easy to be implemented in hardware [117]. The complexity of S-Box depends on the choice of two coefficients $\phi \in GF(2^2)$ and $\mu \in GF(2^4)$. The proper ϕ and μ are selected to assure irreducible property of P_1 and P_2 in $GF(2^2)$ and $GF((2^2)^2)$, respectively.

Fig. 4.3 shows the structure of composite field implemented by using only logic gates. The inversion can be implemented based on normal basis or polynomial basis. The input data I is first mapped from $GF(2^8)$ to isomorphic representation I^{iso} in the field of $GF(((2^2)^2)^2)$, and then computed by the multiplication inverse over $GF(((2^2)^2)^2)$. The computation result is remapped after the affine transformation.The studied S-Box adopts the polynomial basis $\phi = 11$ and $\mu = 1010$.

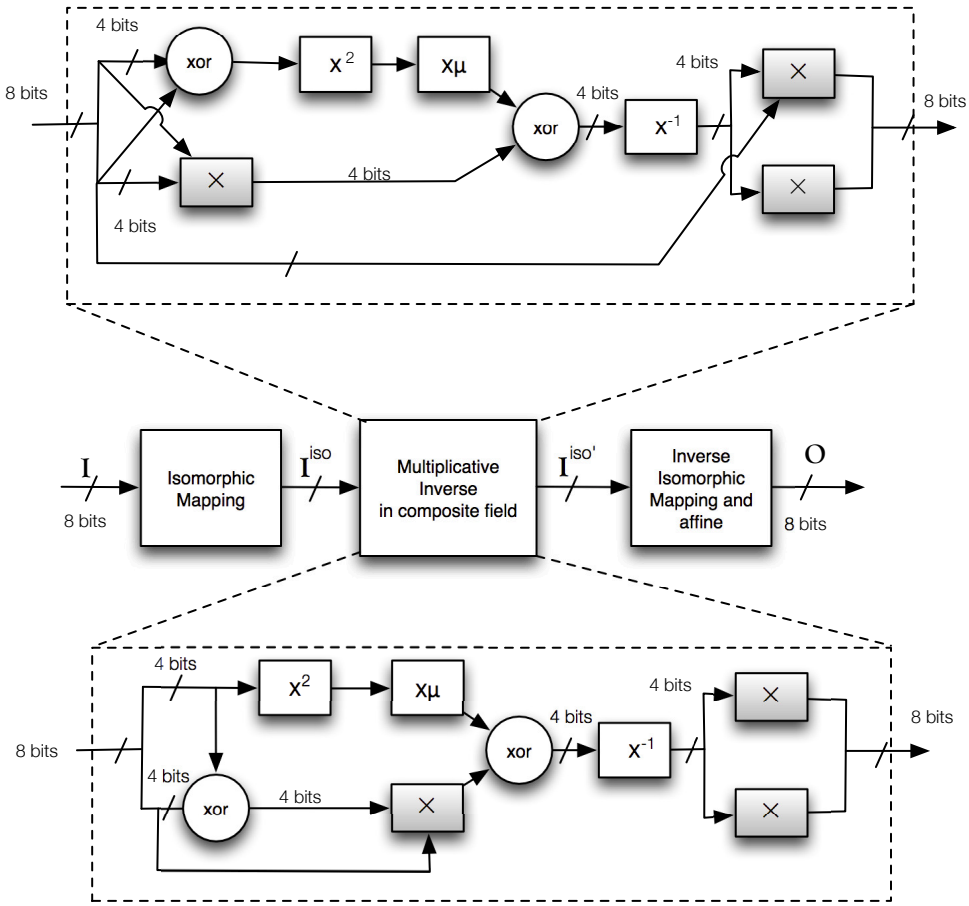


Figure 4.3: Block diagram of S-Box of normal basis (upper) or polynomial basis (below).

4.1.2.3 Analysis of fault-tolerant S-Box

In order to enable the comparison of different fault-tolerant techniques, 3 types of S-Box are discussed: the original S-Box, the TMR S-Box and the parity S-Box proposed in [51]. To evaluate the performances in area, time and power, we used the STM 65-nm CMOS

technology and CORE65LPSVT standard cell for synthesis [119]. Verilog has been used as the entry of Encounter RTL Compiler [120]. Notice that the presented results are post synthesis and do not consider the post layout routing. It's also noteworthy that the power consumption is evaluated for the working frequency of 50 MHz. As seen in Table 4.1, due to the triplication, TMR S-box results in 209.27% area overhead against only 23.2% for parity S-Box. Despite the lower area and power overhead compared with TMR, the maximum target frequency of parity S-Box has a significant decrease.

Table 4.1: Evaluation of the hardware performance

Structure	**Area (A)**		**Frequency (F)**		**Power (P)**	
	(μm^2)	(%)	(MHz)	(%)	(μw)	(%)
Original	403.52	-	459.56	-	52.08	-
TMR	1248.03	209.27	433.46	5.67	184.95	255.13
parity	497.12	23.20	323.26	29.66	103.08	97.91

Reliability assessment is based on the PBR approach described in Section 3.1.4.3. Table 4.2 shows the R_k terms of Original S-Box under different q that is the probability of getting correct output on a gate. Remarking that contribution of the R_3 term to the sum $R_{circuit}$ is negligible, we assume at most 3 simultaneous faults. Indeed, a complete PBR emulation is very time consuming. On the other hand, the probability that all gates malfunction at the same time is low. The same conclusion is acquired for parity S-Box (G=161). We also note the circuit reliability is undesirable (below 0.5) when $q = 0.99$. The following analysis has been accomplished under the consideration of $q > 0.995$. It is worth mentioning that this q value has been taken just as an example to illustrate our approach. Of course, different values of q will produce different results of reliability R.

Figure 4.4 depicts the evaluation of reliability. Two cases of fault-tolerant S-Boxes were taken into account:

- FF: The voter and checking circuit are fault-free.
- FP: The voter and checking circuit are fault-prone.

As seen in Fig. 4.4, none of the two techniques overcomes over all the gate reliability. Parity S-Box shows the similar behavior in both cases, while the faulty voter strongly affects the performance of TMR. In the fault-free case, parity checking approach overcomes TMR when $q > 0.9981$. In the fault-prone case, this range is shifted to $q > 0.9991$. We recognize that if considering the voter to be fault-prone, TMR will not enhance the circuit reliability for $q < 0.9965$.

Figure 4.5 presents the effective frequency and power of parity S-Box. It can be observed that the effective frequency increases with q, while the opposite is true for effect power as expected. Assuming that the reliability is more important than cost penalty, we set "0.5" for the weight factors in experiments. Figure 4.6 shows the efficiency of two studied fault-tolerant S-Boxes. The simulation results show that CED with parity checking is a very attractive approach against the transient faults on S-Box. This technique is efficient for most of the studied cases, especially when gates are less reliable ($q < 0.9987$). However, parity checking is vulnerable to SETs, which are the most probable faults when gates have high reliability. A unique transient fault on a logic gate may produce multiple faults at the output. Parity checking can deal with all odd faults, but even faults stay undetectable.

Actually, TMR could always be a good solution for high-reliability-required application. Although TMR strategy is designed for single fault masking, multiple faults on single

Table 4.2: Reliability computation of Original S-Box (G=133)

Fault number	Gate reliability q				
	0.99	0.995	0.999	0.9995	0.9999
$\mathbf{R_0}$	0.2627	0.5134	0.8754	0.9356	0.9868
$\mathbf{R_1}$	0.0302	0.0294	0.0100	0.0053	0.0011
$\mathbf{R_2}$	0.0070	0.0034	2.29e-04	6.12e-05	2.58e-06
$\mathbf{R_3}$	0.0013	3.00e-4	4.05e-06	5.41e-07	4.55e-09
$\mathbf{R_{circuit}}$	0.3042	0.5465	0.8856	0.9409	0.9879

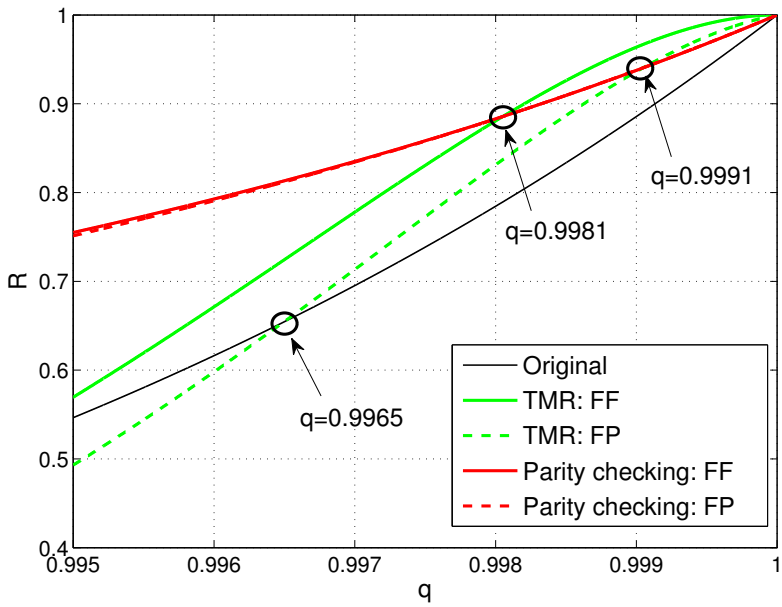


Figure 4.4: Reliability of circuits.

module can be tolerable as well. Moreover, this scheme copes with both transient faults and also permanent faults. Gvien the strong impact on fault tolerance, the highly reliable voter is necessary.

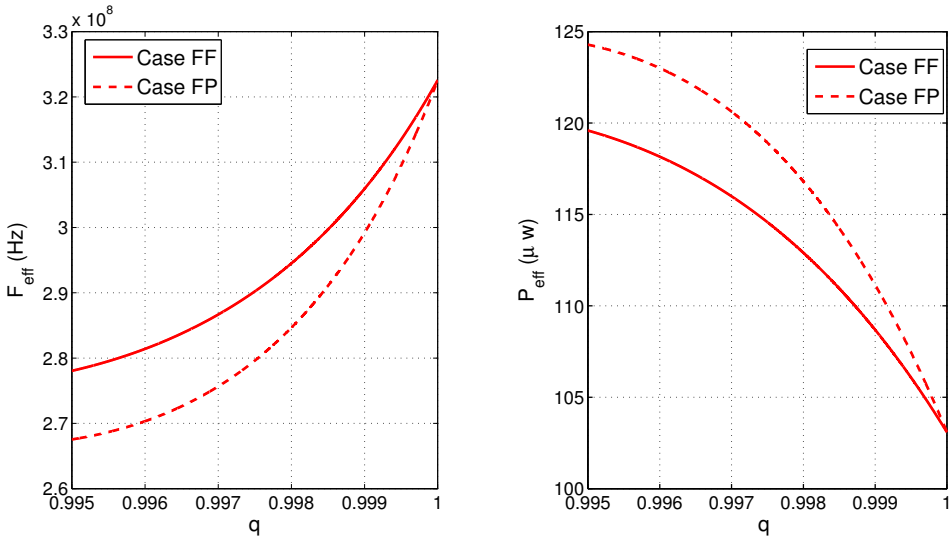
4.2 Low cost reliable architecture for S-Boxes

This section presents a reliable architecture that allows to mask all single transient and permanent faults. The proposed hybrid architecture exploits the inherent redundancy of AES processor's parallel implementation and embeds the redundancy in time and space. Experimental results show the low area and power overhead of this new solution. The proposed architecture's reliability is also analyzed and compared with well-known fault-tolerant approaches.

4.2.1 Introduction

4.2.1.1 Related works

In reported literature, Banu *et al.* applied Hamming codes to avoid the SEUs on AES processor [121]. The parity checking is another popular solution [51, 122, 123]. It increases



(a) (b)

Figure 4.5: a. Effective frequency; b. Effective power (50 MHz).

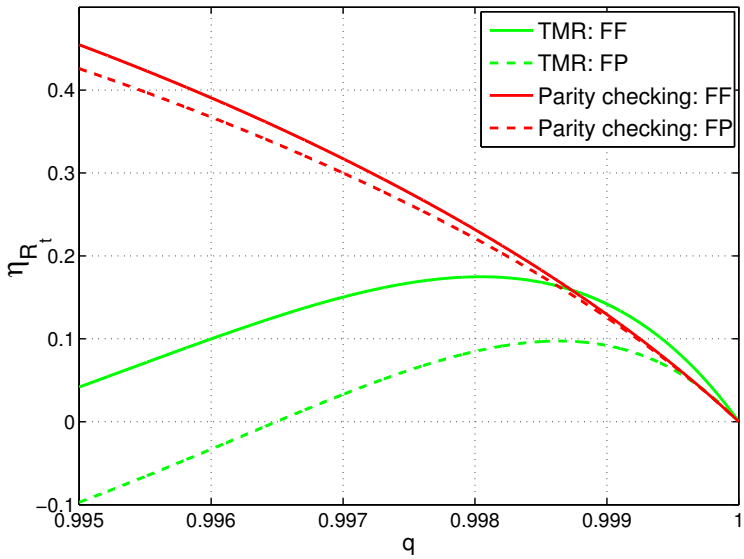


Figure 4.6: Reliability improvement efficiency ($K_A = K_T = K_P = 0.5$).

the fault-tolerance capability while keeping low area as well. But because of the non-linear transformation in S-Box, the prediction of the parity is not trivial and requires dedicated circuits. Moreover, these approaches concentrate on the faults captured by registers and are not sufficiently effective for the compact AES adopting logic-based S-Boxes, in which single faults on combinational part may generate multiple errors on registers.

Hardware redundancy is particularly attractive for compact S-Boxes. Yu *et al.* proposed a hybrid solution to detect all single faults on a compact AES processor [124]. The S-Boxes are duplicated while parity prediction is applied for other components. The use of additional S-Boxes is reported as well [125]. Satoh *et al.* presented an alternative method that merges datapath of encryption and decryption together without any additional arithmetic operators, and double clock cycles are required in one round time [126]. The drawback of these solutions is that extra corrective actions should be established.

4.2.1.2 Hardware implementation of AES

Different implementations of AES have been reported in literature [117, 127]. They can be classified into two groups: high-speed and compact implementations. High-speed implementations lead to the development of AES processors operating with the speeds of tens of Gigabits per second, while compact implementations are iterative designs based on only one round or quarter-round loop structure, optimized for the minimum area [128]. In this work, we concentrate on the compact parallel architecture of AES processor, which employs 16 S-Boxes of 8-bits for *SubBytes* computations (see Fig. 4.7). The studied S-Box implementation is based on logic gates and leads to a low area consumption [129]. Our proposed architecture is valid for all AES encryption and decryption, regardless of the S-Box structure.

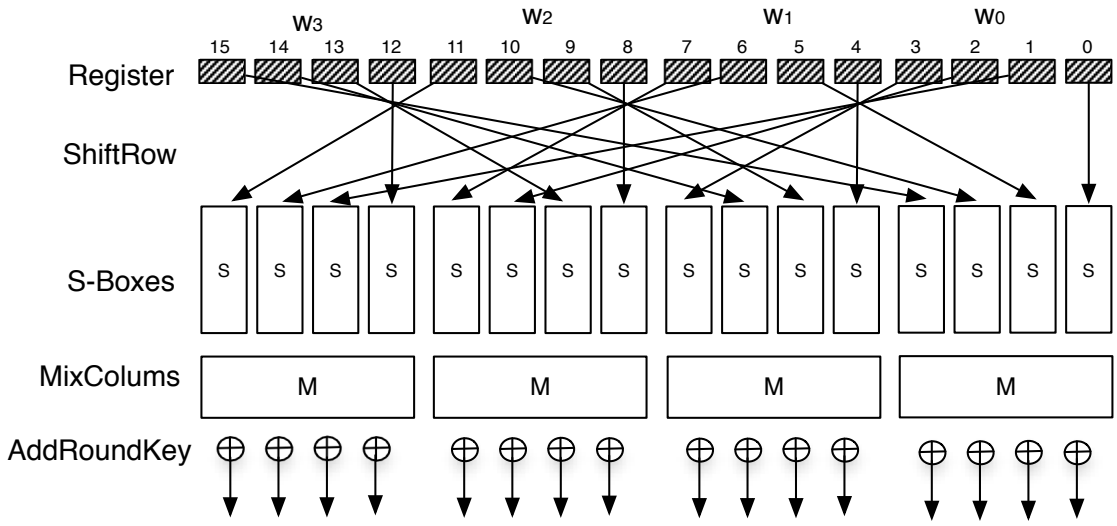


Figure 4.7: Parallel architecture of a single AES round.

4.2.2 Hybrid architecture for S-Boxes

4.2.2.1 Configurable S-Boxes array

The main idea behind the proposed architecture is the configurable S-Boxes array (CSBA) shown in Fig. 4.8. This part contains four S-Boxes, two configuration blocks, 32 one-bit majority voters and some registers. The input datapath is one word (i.e., 32 bits). Three clock periods are required to perform a complete processing. As seen in Table 4.3, the same byte data is processed on three different S-Box during three clock periods. Take data $S_{0,0}$ as an example, its transformation is performed on S-Boxes **A**, **B** and **C**.

The configuration of used S-Boxes is accomplished by two blocks that only consist of multiplexers, where *control signals* are generated by a finite state machine (FSM) block. The outputs related to different clocks are saved in registers, and then compared by voters. This architecture assures the single fault-masking of both transient faults and permanent faults. Let $S1'_{r,c}, S2'_{r,c}, S3'_{r,c}$ be the results after SubBytes transformation of $S_{r,c}$ in the first, second and third clock period, respectively.

- Supposing that a transient fault takes place in S-Box **C** in the second clock period, the register saves an erroneous $S2'_{0,1}$. However, as the $S1'_{0,1}$ and $S3'_{0,1}$ are correct, we get the correct final output $S'_{0,1}$.

Table 4.3: Reconfiguration processing

S-Box No.	*clock period*		
	1	2	3
A	$S_{0,0}$	$S_{3,0}$	$S_{2,0}$
B	$S_{1,0}$	$S_{0,0}$	$S_{3,0}$
C	$S_{2,0}$	$S_{1,0}$	$S_{0,0}$
D	$S_{3,0}$	$S_{2,0}$	$S_{1,0}$

- Supposing there is a permanent fault on S-Box **C**, we get the faulty $S1'_{2,0}$, $S2'_{1,0}$ and $S3'_{0,0}$ as depicted in Fig. 4.9. Thanks to the majority vote, the correct outputs are still obtained.

It is noteworthy that any single fault on registers can be also tolerated. For AES of 128 bits datapath, four CSBAs are needed, referred to as full configuration (FC) architecture.

4.2.2.2 Hybrid S-Boxes architecture

One drawback of the aforementioned FC architectures is the high overhead due to the extra configuration blocks and registers. Actually, AES is inherently redundant thanks to the identical S-Boxes. We thus combine each three S-Boxes with 8 one-bit voter as an S-Box TMR cluster which has the datapath of 8 bits. This cluster can tolerate transient and permanent faults on a single S-Box. A straightforward solution is to apply clusters to the overall SubByte transformation and proceed in two clock times (64 bits each clock time). Therefore, 8 clusters (24 S-Boxes) are required. Since 16 S-Boxes are implemented in a parallel architecture, we need 8 extra S-Boxes. This is nearly half area of the original SubByte transformation circuit, and obviously is not feasible.

The idea is to apply both configurable and TMR approaches. The proposed hybrid

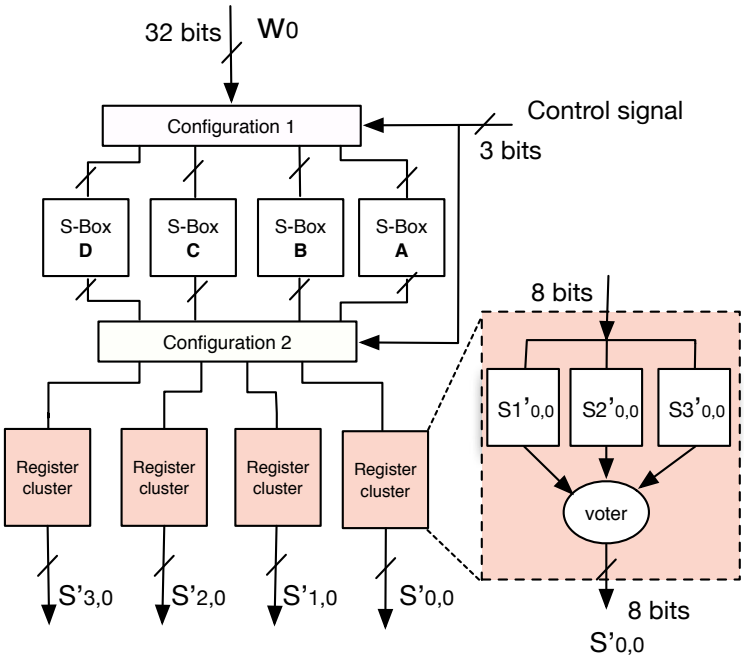


Figure 4.8: Diagram of Configurable S-Boxes Array (CSBA).

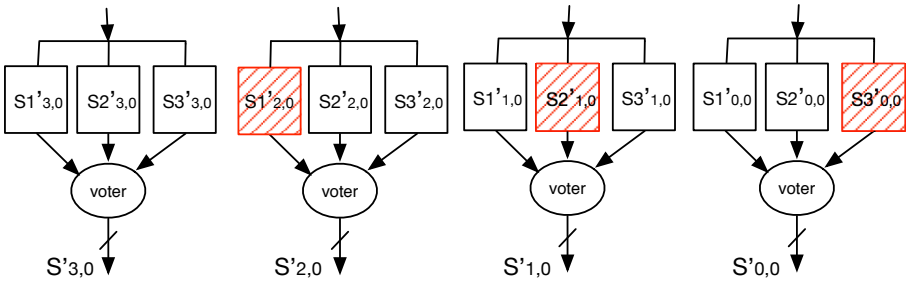


Figure 4.9: Permanent faults on CSBA.

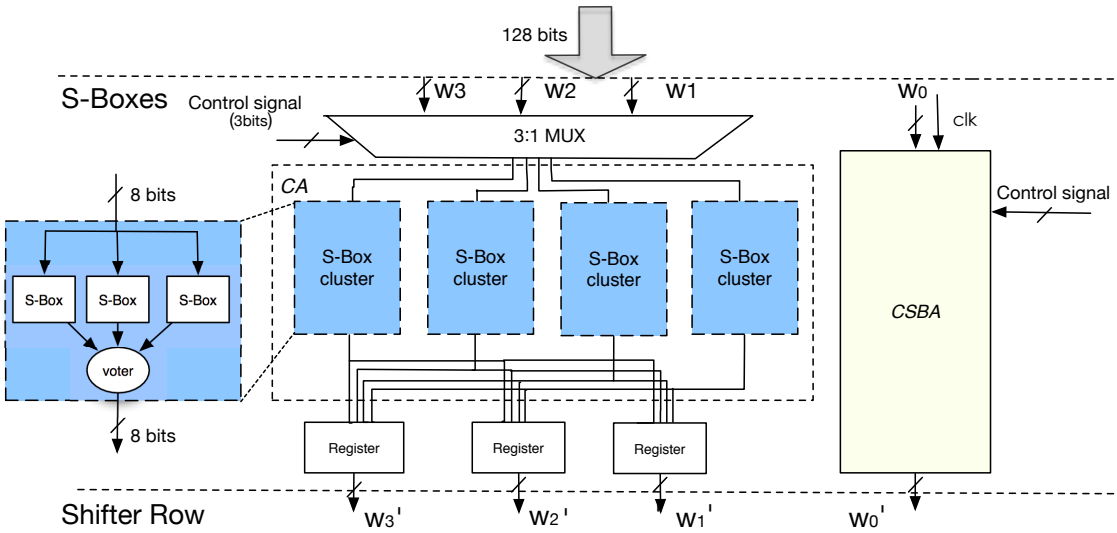


Figure 4.10: Hybrid S-Boxes architecture.

architecture (see Fig. 4.10) includes two main blocks: a cluster array (CA) including four S-Box clusters and a CSBA. Like in the FC architecture, three clock periods are required to accomplish a complete computation of input $W_3W_2W_1W_0$. The procedure is shown in Table 4.4.

As can be noticed in Table 4.4, the CSBA processes W_0 in all three clock periods, while W_1, W_2 and W_3 are respectively executed by CA block in different clock periods. By adopting the TMR, register triplication for W_1, W_2 and W_3 are no more necessary and the area is saved. It's noteworthy that multiplexers are utilized to select the processed word on CA block, and registers are still needed for holding the output.

Table 4.4: The execution procedure

Executed block	*clock period*		
	1	2	3
CSBA	W_0	W_0	W_0
CA	W_1	W_2	W_3

4.2.3 Cost evaluation

In order to evaluate the proposed S-Boxes structures, we compare them with the classical TMR and TTR strategies with respect to hardware overhead. The overhead is firstly evaluated by the transistor count without considering the S-Box implementation, and then by synthesis using the STM 65-nm CMOS technology and CORE65LPSVT standard

cell library [119]. The nominal junction temperature is 25°C and the presented results are post synthesis. Verilog is used as entry for Encounter RTL Compiler [120].

4.2.3.1 Transistor count

In this section, we assume all architectures use the same S-Box implementation with the transistor count of N_{SB}.

TMR: Let N_{vt} be the transistor count of one-bit voter, the transistor count of TMR architecture is given in (4.16).

$$N_{TMR} = 3 \times 16 \times N_{SB} + 128 \times N_{vt} \tag{4.16}$$

The voter implemented is the majority voter composed of three 2-input NAND gates and one 3-input NAND gate. We estimate its transistor count as 18. Consequently, the (4.16) can be rewritten as (4.17).

$$N_{TMR} = 48 \times N_{SB} + 2304 \tag{4.17}$$

TTR: Consisting of the original combinational part, the register part and voter array, the transistor count of ST-TMR structure can be calculated by (4.18).

$$N_{TTR} = 16 \times N_{SB} + N_{REG} + 128 \times N_{vt} \tag{4.18}$$

The considered register is the D flip-flip with an "*enable*"signal, consisting of 14 transistors. Equation (4.18) is rewritten as (4.19).

$$N_{TTR} = 16 \times N_{SB} + 7680 \tag{4.19}$$

Full configuration: Let us note the transistor count of configuration block 1 and 2 as N_{con_1} and N_{con_2}, respectively. One CSBA block requires N_{CSBA} transistors given in:

$$\begin{aligned} N_{CSBA} =& 4 \times N_{SB} + N_{con_1} + N_{con_1} \\ &+ 32N_{vt} + 3 \times 32 \times N_{DFF} \end{aligned} \tag{4.20}$$

Notice that the two configuration boxes can be implemented by 3:1 multiplexers for every 3 input bits. Each 3:1 multiplexer includes three 2-input NAND gates and one 3-input NAND gate. Finally, we obtain the transistor count of the FC architecture shown in (4.21).

$$\begin{aligned} N_{FC} =& 4 \times N_{CSBA} \\ =& 16 \times N_{SB} + 12288 \end{aligned} \tag{4.21}$$

Hybrid: As mentioned in Section 4.2.2.2, the proposed hybrid architecture contains four S-Box clusters, 32 multiplexers, registers of 96 bits and a CSBA block. We define the transistor count of the proposed hybrid architecture as (4.22).

$$N_{hybrid} = 16 \times N_{SB} + 5568 \tag{4.22}$$

4.2.3.2 ASIC implementation

The logic-based implementation of S-Boxes referred in Section 4.1.2.2 is compact S-Boxes established on *composite field arithmetic*. Two different composite field S-Boxes are taken as instances in experiments: polynomial basis S-Box (PBS-Box) [129] and normal basis S-Box (NBS-Box) [130]. Let C_O be the original structure cost (area, time, power, etc.),

Table 4.5: Evaluation of the hardware performance (only S-boxes)

Architecture	Polynomial basis[129]		Normal basis[130]	
	(μm^2)	(%)	(μm^2)	(%)
Original	6456.3	-	6647.7	-
TMR	19968.5	+209.3	20542.1	+209.0
TTR	10896.6	+68.8	11088.0	+66.8
Parity Checking	7977.8	+23.56	8169.2	+22.9
Full configuration	11709.9	+81.4	11892.9	+78.9
Hybrid	**8997.6**	**+39.4**	**9171.4**	**+37.9**

and C_{FT} the cost of the structure implemented fault-tolerant technique. We consider the overhead defined as (4.23).

$$Overhead = \frac{C_{FT} - C_O}{C_O} \times 100\% \tag{4.23}$$

Table 4.5 presents the area results of different S-Boxes' architectures, where the parity checking approach is reported as well. In this table, the "Original" denotes the SubByte transformation part with 16 S-Box repetitions, see Fig. 4.7. It can be recognized that the proposed hybrid approach introduces no more than 40% of area overhead, which is lower than other fault-tolerant strategies' overhead. Furthermore, even compared with parity checking approach which is famous for its low area overhead, the proposed technique's overhead exceeds by only 15%.

Table 4.6 (see the end of this chapter) shows the performance evaluations of AES processor applying the proposed hybrid S-Boxes architecture. In addition to default optimization, the size and speed optimizations are shown as well. We remark that the area overhead can reach 14.5%. Even without area optimization, this value is not greater than 30.7%.

The maximum frequency is decreased by at most 24.1% due to the extra voters, configuration and FSM blocks. The throughput is reduced by about two thirds by adopting the hybrid architecture in consequence of time redundancy, as expected. However, instead of lowering the maximum frequency, the proposed scheme raises the frequency by 4.09% in the case of applying PBS-Box under speed optimization. Consequently, the throughput is higher than other configurations, but with a high area overhead of 121.9%.

Actually, these large variations in logic synthesis are caused by the long combinational logic path in architecture's round function block. In this work, we concentrate on the compact implementations, the one with the area optimization is therefore selected. More effort in synthesis can be carried out in order to realize a better tradeoff between speed and area.

4.2.4 Reliability assessment

In the following paragraphs, we analyze the reliability of the proposed hybrid architecture. It is compared with TMR approach which is the most popular fault masking technique to cope with both permanent and transient faults. We focus on the impact of S-Box and the other blocks being assumed fault-free. The TTR approach is not considered due to its helplessness against permanent faults as mentioned in Section 2.3.

4.2.4.1 TMR

Classical TMR architecture allows to mask all faults on signal module. A TMR scheme with an ideal voter is effective if no more than one incorrect output is generated by the three equivalent modules. Let us note R_{SB} the reliability of a single S-Box, the reliability of S-Boxes applying TMR scheme is given by:

$$R_{TMR-SB} = 3 \times R_{SB}^2 - 2 \times R_{SB}^3 \tag{4.24}$$

As all S-Boxes are independent, the overall reliability of the TMR architecture can be represented by:

$$R_{TMR} = R_{TMR-SB}^{16} \tag{4.25}$$

4.2.4.2 Hybrid approach

For a CSBA block, the faults occurring on any single S-Box will be tolerated. We therefore obtain the reliability of CSBA defined by (4.26).

$$R_{CSBA} = 4 \times R_{SB}^3 - 3 \times R_{SB}^4 \tag{4.26}$$

On the other hand, any malfunction of CA during three clock periods will generate incorrect final outputs. The reliability of the hybrid architecture can be defined as:

$$R_{hybrid} = R_{CSBA} \times (R_{TMR-SB}^4)^3 \tag{4.27}$$

Figure 4.11 shows how the overall reliability of S-Boxes changes with a single S-Box's reliability. We can recognize that without considering the effect of multiplexers and registers, the proposed solution is even better than TMR approach.

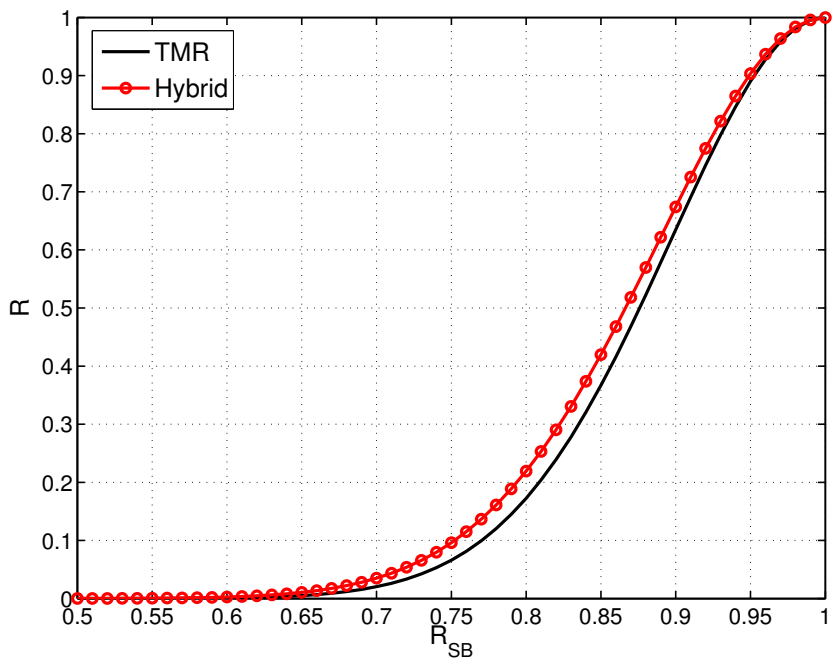


Figure 4.11: Reliability comparison.

4.3 Conclusion

In this chapter, we concentrated on effective fault-tolerant design. The processor implemented AES was taken as a specific application. In order to estimate the efficiency of various fault-tolerant techniques, we proposed a characterization matrix considering the main factors in ICs design (i.e., area, delay and power) and reliability improvement induced by the technique. After the analysis of feasible fault-tolerant approaches, a reliable architecture for S-Boxes with a low area overhead (14.5%) was presented. By exploiting the inherent redundancy of parallel implemented AES processors, we combined the configurable structure with TMR approach. This hybrid solution can mask all single faults on S-Boxes and registers. The multiple faults that take place on a single S-Box can be tolerated as well. It outperforms conventional fault masking techniques such as TMR and TTR strategies in hardware cost.

Table 4.6: Area and time performances of AES processor applying the proposed architecture

S-Box	**Area** (μm^2)			F_{Max} (MHz)			**Throughput** ($Mbps$)			**Optimization**
	Original	Proposed	(%)	Original	Proposed	(%)	Original	Proposed	(%)	
Polynomial basis[129]	9632.0	11031.8	**+14.5**	307.4	255.6	-16.84	3933.9	1090.5	-72.3	size
	11432.2	24341.2	+121.9	393.1	409.2	**+4.1**	5031.5	1745.8	**-65.3**	speed
	10254.4	13401.4	+30.7	328.0	277.6	-15.4	4198.1	1184.2	-71.8	default
Normal basis[130]	9482.2	11158.7	+17.7	276.7	239.4	-13.5	3515.3	1021.2	-70.9	size
	14473.2	23718.2	+63.9	500.0	379.4	-24.1	6400.0	1618.6	-74.7	speed
	11440.0	13576.7	+18.7	271.7	247.3	-9.0	3477.6	1055.3	-69.7	default

Chapter 5

Conclusions and prospectives

5.1 Conclusions

This thesis has dealt with reliability concerns induced by continuous geometrical downscaling of CMOS technology, particularly the issues during the usage of ICs. Our discussions covered the dominant aging mechanisms NBTI and HCI, radiation-related effects as well as thermal noise. The presented works were specified to attain a good reliability improvement with a small penalty in performance (i.e., area, delay and power). Binary adders, CORDIC processor and AES processor were taken as the studied targets.

A great portion of this work goes to the reliability analysis with respect to fault tolerance. Considering the logic masking property on combinational circuits, two assessment methods were proposed. One explored the impact of transient faults on iterative processors. The defined fault impact coefficient pointed out the significance of operators concerning the fault tolerance, which can be useful in a selective hardening strategy. Compared to traditional analysis methods, the impact of a fault was weighted by its produced deviation from what was expected at the primary output. This assumption is more realistic for a specified application in which the error caused by faults may be acceptable for some threshold, such as the DSP memory-core.

On the other hand, an efficient analysis approach was proposed to estimate the reliability of CED circuits under multiple faults which has not been sufficiently studied in reported works. Merging the PTMs with conditional probability, this approach can provide an accurate reliability analysis with a reduced time consumption compared to other methods. The comprehensive analysis indicated that silent error is another important aspect that should be considered in critical applications.

A new fault-modeling approach revealed the relevance between gate reliability and its topology. It provided the fault-tolerant design of ICs with significant information. The mathematical modeling enables its application in many analytical assessment methodologies (e.g., SPR, PGM). It is worth mentioning that in this work the logic gate and transistor were assumed to have the failure rate as q and ε, respectively. In fact, these informations can be obtained from the cell's library provided by foundries. The cell's library contains a large number of factors for modeling cell's reliability. Although an exact modeling is hard to be achieved, the assessment based on foundries' data are more realistic and reliable.

In addition to fault tolerance analysis, the methodology for studying aging effects was discussed as well. The delay degradation difference among the studied adder architectures can be up to three times according to the circuit size and time stress. The analysis of two major aging mechanisms NBTI and HCI demonstrated the insensitivity of 65 nm CMOS adders under aging stress. Indeed, the aging is not a dominant effect on 65 nm

CMOS technology, but this aging-aware design flow for digital ICs exhibited comprehensive exploration of aging effects on binary adder for further improvement of reliability.

Other attempts in the work focused on the reliability enhancement. Motived by an effective design, we introduced a general method to characterize different fault-tolerant schemes considering the reliability improvement and the relevant overhead in area, delay and power. It identified more effective techniques according to the gate reliability. S-Boxes adopting the representative fault-tolerant schemes were evaluated. Unfortunately, due to the limited time, the proposed efficient method in Section 3.2.3 could not be applied for the reliability evaluation in this thesis.

A hybrid architecture embedding hardware and temporal redundancy was then designed for enhancing S-Boxes in AES processor. The inherent redundancy in AES processor has been proved efficient in reducing the area overcost. Any encryption and decryption of AES processors with parallel implementation are able to deploy this architecture. Furthermore, although our goal is to cope with the faults due to CMOS technology decreasing, malicious attacks using fault injection can also be tolerated.

The major contributions of this work are summarized as: analysis methodologies of reliability, evaluation method for fault-tolerant designs and application-specific solution for reliability enhancement. In fact, given the reliability specification, the proposed methodologies involved different steps to improve the circuit reliability at architecture level. Additionally, the reliability enhancement in this work was gained through the use of fault tolerance. Despite the various fault-tolerant schemes reported in literature, the obtained results showed the difficulties to achieve the reliability requirement within a limited budget in performance overhead, in particular, under the assumption of multiple faults. The low cost design for S-Boxes suggested that the proposed hybrid approach merging different redundancy techniques is a suitable solution to face this challenge.

The research performed during the thesis resulted in several publications in international conferences as listed below:

- **Accurate Reliability Analysis of Concurrent Checking Circuits Employing An Efficient Analytical Method** [131], published in *Microelectronics Reliability*, January 2015.

- **Efficient implementation for accurate analysis of CED circuits against multiple faults** [132], presented at *21th Mixed Design of Integrated Circuits and Systems (MIXDES)*, Lublin, Poland, June 2014.

- **Simulation Study of Aging in CMOS Binary Adders** [133], presented at *MIPRO 37th International Convention/Microelectronics, Electronics and Electronic Technology (MEET)*, Opatija, Croatia, May 2014.

- **Analytical method for reliability assessment of concurrent checking circuits under multiple faults** [134], presented at *MIPRO 37th International Convention Microelectronics, Electronics and Electronic Technology (MEET)*, Opatija, Croatia, May 2014.

- **A Low Cost Reliable Architecture for S-Boxes in AES Processors** [135], presented at *IEEE Symp. Defect and Fault Tolerance (DFT)*, New York city, US, October 2013.

- **Reliability analysis of combinational circuits with the influences of noise and single-event transients** [136], presented at *IEEE Symp. Defect and Fault Tolerance (DFT)*, New York city, US, October 2013.

- **Modeling of transient faults and fault-tolerant design in nanoelectronics** [137], presented at *2013 IEEE 56th International Midwest Symposium (MWSCAS)*, Columbus Ohio, US, August 2013.
- **A general cost-effective design structure for probabilistic-based noise-tolerant logic functions in nanometer CMOS technology** [138], presented at *IEEE Eurocon*, Zagreb, Croatia, July 2013.
- **Exploring the Impact of Transient Faults on CORDIC Processor** [139], presented at *Journées Nationales du Réseau Doctoral en Micro-nanoélectronique (JNRDM)*, Grenoble, France, June 2013.
- **Evaluation of Fault-tolerant Composite Field AES S-Boxes under Multiple Transient Faults** [140], *IEEE International NEWCAS Conference*, Paris, France, May 2013.
- **Transient Fault Analysis of CORDIC Processor** [141], presented at *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, Seiville, Spain, December 2012.
- **Parallel Scaling-Free and Area-Time Efficient CORDIC Algorithm** [142], presented at *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, Seiville, Spain, December 2012 .

5.2 Prospectives

The prospectives of current works encompass four aspects:

- Automation of the analysis method for CED circuits
- Study of aging effects on more advanced technologies than 65 nm.
- Reliability enhancement for the overall AES processors.
- Integration of the proposed reliability considerations in digital design flow.

The presented analytical method for CED circuits have been implemented on Matlab where the 4-bit RCA was taken as a case study. In order to evaluate different CED circuits, the automation of this method could be performed so as to integrate the proposed method in a design flow.

The simulation study of aging confirmed the insensitivity of 65 nm CMOS technology. Some other reliability issues like process variation threatens more ICs than aging. However, the threaten of aging effects is predictable with the technology downscaling. Therefore, the researches on more advanced technologies such as 28 nm CMOS are required.

As an important part in many critical applications, the malfunction of AES processor can cause a serious system failure. The presented architecture for S-Boxes ignored the faults occurring on other linear transformations that should be enhanced in future work. Two solutions are possible: extend the configurable structure or adopt other fault-tolerant schemes for the linear parts (e.g., CED with parity prediction). Moreover, the voters in the proposed architecture were also supposed fault-free. In fact, the highly reliable voter is necessary due to its strong impact on the performance of TMR mechanism.

In addition, several approaches for reliability analysis as well as reliability improvement techniques were mentioned. There is not a design flow taking into account all these methods. The tasks in the present thesis were accomplished individually. Figure 5.1 illustrates a possible digital design flow for reliability with respect to fault tolerance, where FT

represents fault-tolerant. Last but not remains least, the lack of a universal methodology considering different reliability issues is always a challenge in future work. That is why the interaction among various issues need to be explored.

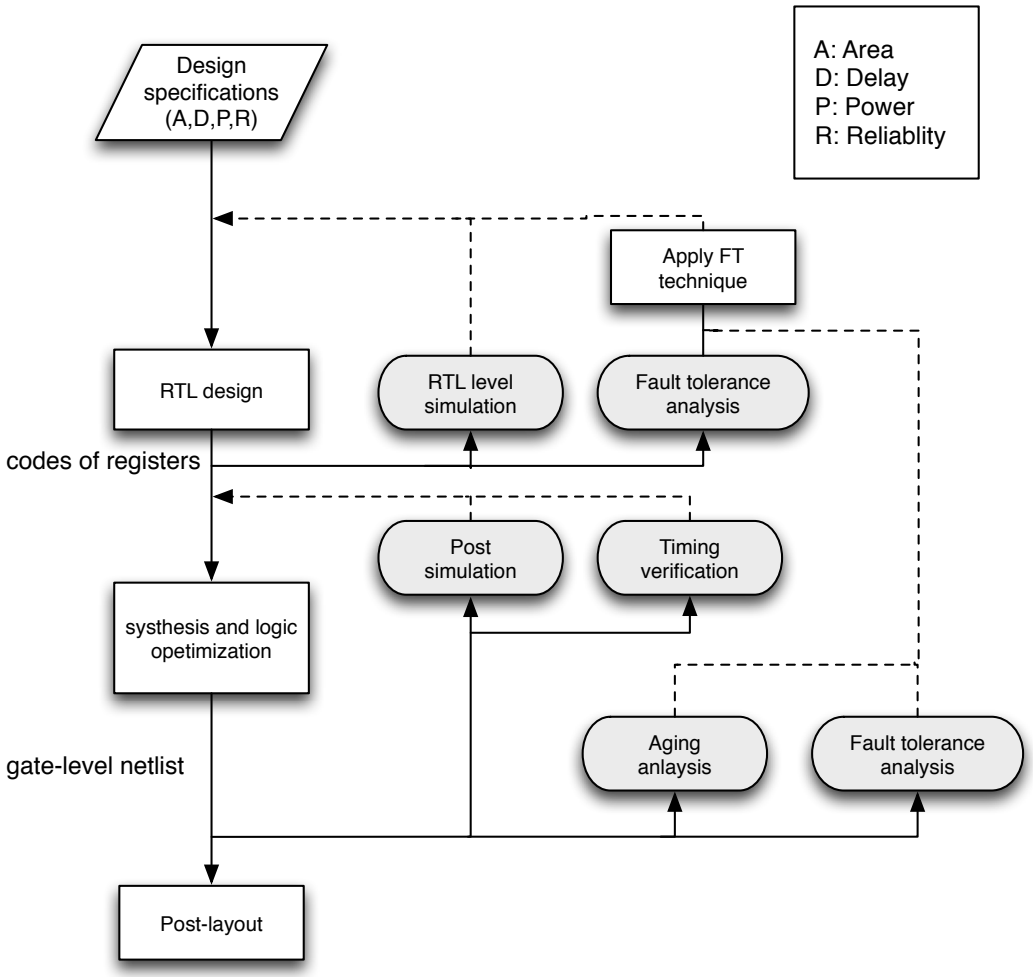


Figure 5.1: Reliability-aware front-end design flow.

Appendix A

Binary adders

As the fundamental and most applied operations in computer science, arithmetic operations attracted intense attentions in digital circuit design. Basic arithmetic operations includes shift, addition/subtraction, multiplication, division, square root and etc. Figure A.1 shows the dependencies between different operations. Among them addition/subtraction plays a remarkable role, particularly the binary addition/subtraction. It is the most often applied operation and is the essential component for constituting more complex operations such as multiplication and division.

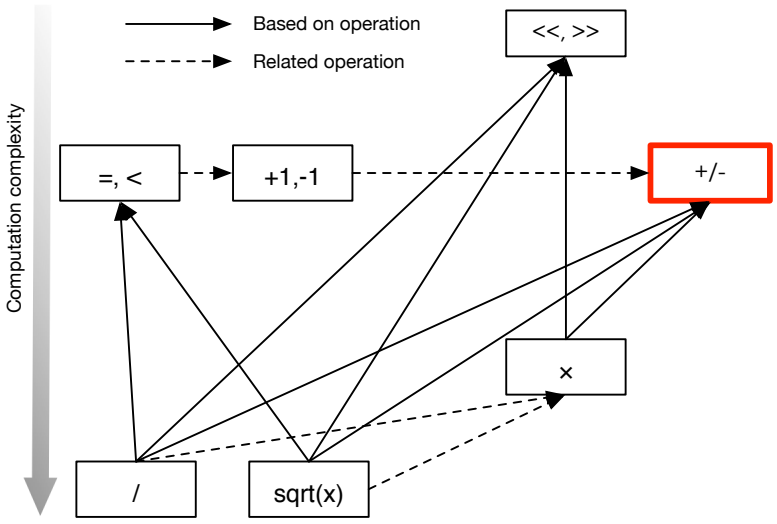


Figure A.1: Dependencies of arithmetic operations (extracted from [143]).

A.1 One-bit adder

One-bit addition using carry is the most simple and basic operation, referred as *full adder (FA)*. FA has three inputs (a carry-in and two operands) and two outputs (a sum and a carry-out). Table A.1 shows the truth table of FA, where Cin and $Cout$ represent carry-in and carry-out, respectively. The outputs Sum and $Cout$ can be obtained through Boolean expressions as expressed in (A.1), while the circuit implementation and the logic symbol are shown in Fig. A.2.

$$\begin{aligned} Sum =& A \oplus B \oplus Cin \\ Cout =& A \wedge B \vee Cin \wedge (A \oplus B) \end{aligned} \tag{A.1}$$

Table A.1: One-bit addition rules

A	B	Cin	Cout	Sum
0	0	0	0	0
1	0	0	0	1
0	1	0	0	1
1	1	0	1	0
0	0	1	0	1
1	0	1	1	0
0	1	1	1	0
1	1	1	1	1

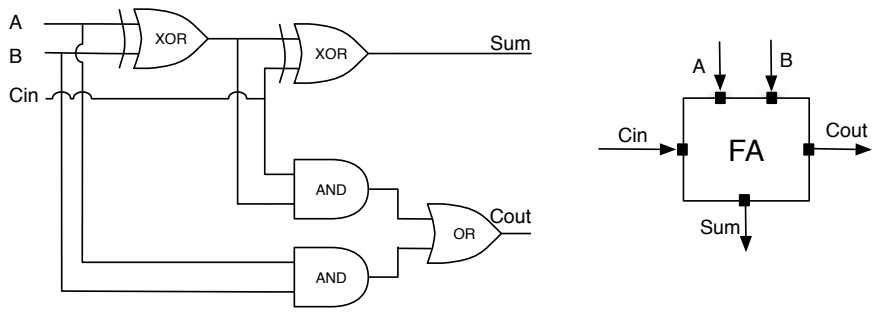


Figure A.2: Logic symbol and circuit architecture of full adder.

Half adder (HF) is considered as a special case of FA with two operands inputs as given in (A.2). Its implementation and the logic symbol are presented in Fig. A.3

$$\begin{aligned} Sum =& A \oplus B \\ Cout =& A \wedge B \end{aligned} \tag{A.2}$$

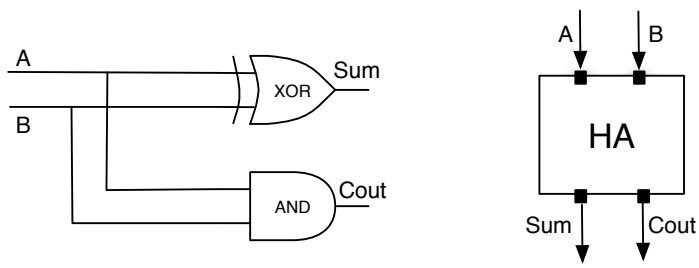


Figure A.3: Logic symbol and circuit architecture of Half Adder.

A.2 Ripple carry adder

Most of the n-bit adders based on carry-propagate chain implement FAs and HAs. A n-bit adder has $(2n+1)$-bit input and $(n+1)$-bit output. Let us define two binary number inputs operands A, B and their summation Sum, as expressed in (A.3):

$$\begin{aligned} A =& a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + a_{n-3}2^{n-3} + \cdots + a_1 2 + a_0 \\ B =& b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + b_{n-3}2^{n-3} + \cdots + a_1 2 + b_0 \\ Sum =& s_{n-1}2^{n-1} + s_{n-2}2^{n-2} + s_{n-3}2^{n-3} + \cdots + s_1 2 + s_0 \end{aligned} \tag{A.3}$$

The relationship between inputs and outputs is described as follows:

$$\begin{aligned} 2^n Cout + Sum &= A + B + Cin \\ 2^n Cout + \sum_{0}^{n-1} 2^i s_i &= \sum_{0}^{n-1} 2^i (a_i + b_i) + Cin \end{aligned} \tag{A.4}$$

Define c_i as the carry-in of i-th bit, (A.5) can be also written, where c_i is the carry-in bit of the i-th bit addition.

$$\begin{aligned} s_i &= a_i \oplus b_i \oplus c_i \\ c_{i+1} &= (a_i \wedge b_i) \vee c_i \wedge (a_i \oplus b_i) \end{aligned} \tag{A.5}$$

Figure A.4 shows the simplest carry propagation adders structure composed by a series of FAs which is mentioned as the ripple carry adder (RCA). We can notice that the delay of such an adder is determinated by the carry chain. Assume that each FA requires T_{fa} in order to obtain a stable carry-out, the delay of overall circuit is expressed as below:

$$T = n \times T_{fa} \tag{A.6}$$

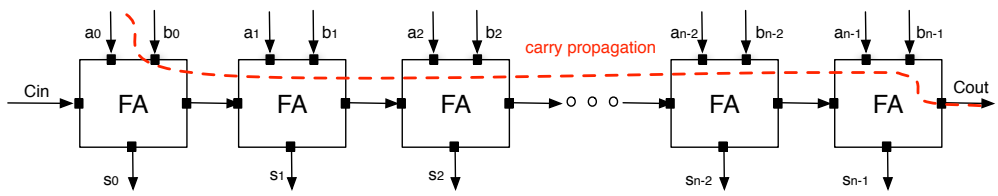


Figure A.4: Logic symbol and circuit architecture of n-bit RCA.

A.3 Carry-select adder

An alternative implementation of binary adder is to formulate it into carry select problem. Such an adder is named as *carry-select adder* (CSA) [144]. The $s(i:j)$ of stages i to j is computed by two possible input carries at the same time (i.e., c_i="1" and c_i="0") instead of waiting the actual carry-in from previous stages. For a n-bit RCA, the improvement is achieved by decomposing the overall FA chain into k small FA groups of size n/k (see Fig. A.5). k is the called reduced factor, where the optimal block size is $m = \lfloor\sqrt{n}\rfloor$. The multiplexer Mux is used to select the correct output of a group, that is, the results with assumption of c_i="1" are selected when actual carry-input is "1", and vice versa. The main difference between CSA and RCA is that in the latter the carry has to ripple through all full adders, but in the former the carry need only to pass through a group of m FA and a single multiplexer (MUX) [145]. Define T_{mux} as the delay on a MUX, the maximum delay for CSA adder is given in (A.7).

$$T_{CSA} = k \times T_{mux} + m \times T_{FA} \tag{A.7}$$

A.4 Carry look-ahead adder

The principle of carry look-ahead adder (CLA) is to parallel compute the carry bits by exploring the inherent properties of FAs and HAs. The adder computes some carry bits

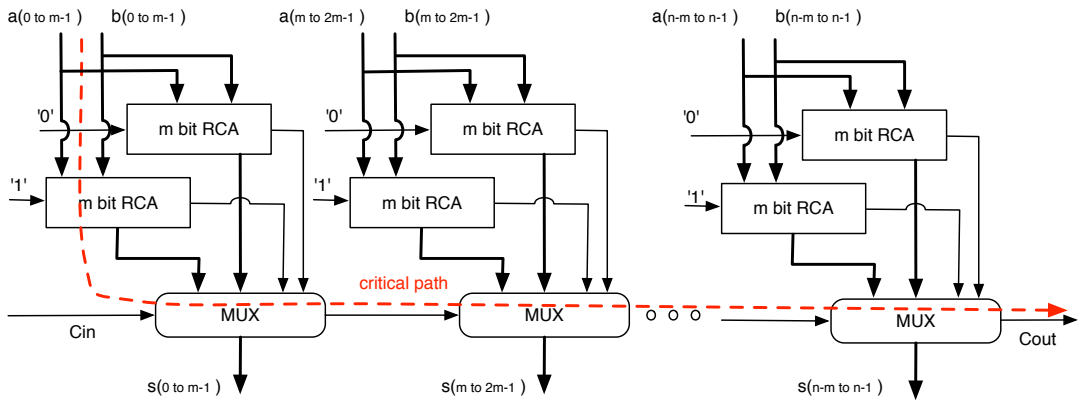


Figure A.5: Carry Select Adder architecture, $m = n/k$.

before the sum bits. Let g_i and p_i represent the *carry generation* and *carry propagation* in $i-$th stage, respectively, as expressed in (A.8)

$$\begin{aligned} g_i &= a_i \wedge b_i \\ p_i &= a_i \oplus b_i \end{aligned} \tag{A.8}$$

Taking a 4-bit adder for example, c_i of each stage is therefore rewritten as given in (A.9).

$$\begin{aligned} c_0 &= Cin \\ c_1 &= g_0 \wedge (c_0 \vee p_0) \\ c_2 &= g_1 \wedge (c_1 \vee p_1) \\ c_3 &= g_2 \wedge (c_2 \vee p_2) \\ c_3 &= g_3 \wedge (c_3 \vee p_3) \end{aligned} \tag{A.9}$$

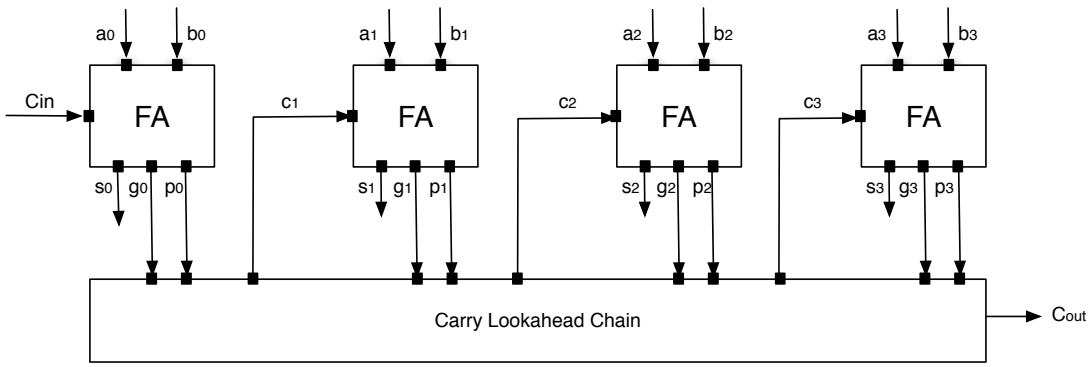
It is obvious that the carry-out can be either generated by its own carry generation or propagated by the carry bit from previous stage. By expanding each c_i on the right side of equations, we can express c_i by using p, g, and Cin, regardless of the carry-in of each stage:

$$\begin{aligned} c_1 &= g_0 \wedge (p_0 \vee Cin) \\ c_2 &= g_1 \wedge (p_1 \vee g_0) \wedge (p_1 \vee p_0 \vee Cin) \\ c_3 &= g_2 \wedge (p_2 \vee g_1) \wedge (p_2 \vee p_1 \vee g_0) \wedge (p_1 \vee p_0 \vee p_0 \vee Cin) \\ c_4 &= g_3 \wedge (p_3 \vee g_2) \wedge (p_3 \vee p_2 \vee g_1) \wedge (p_3 \vee p_2 \vee p_1 \vee p_0 \vee g_0 \vee Cin) \end{aligned} \tag{A.10}$$

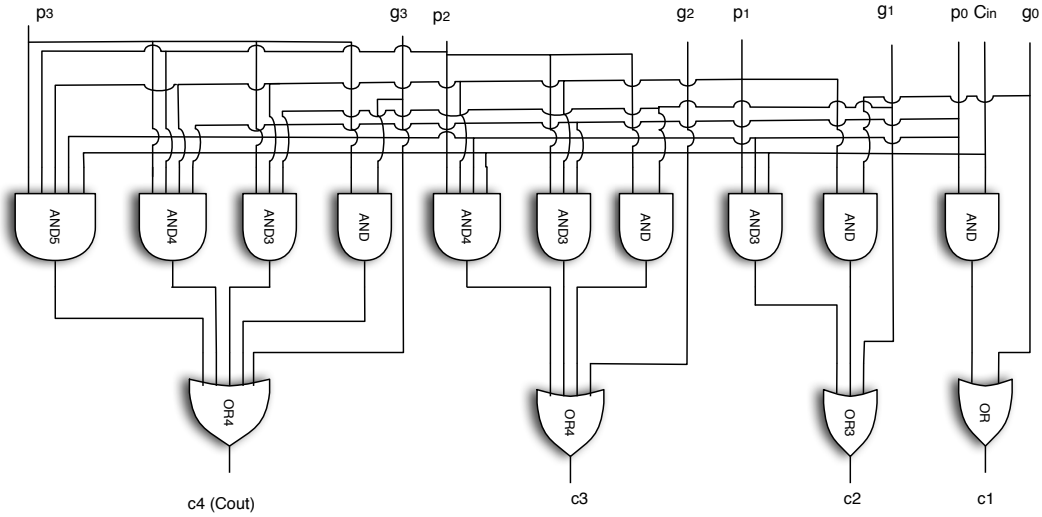
The block diagram of a 4-bit CLA is shown in Fig. A.6. The high fan-in and fan-out number are the inherent difficulties in CLA adder and could be mitigated by combining RCA and CLA. One possible solution is to group every s-bit addition as a CLA block and apply ripple carry technique between the n/s blocks.

A.5 Carry-skip adder

Carry-skip adder (CSKA) is another solution trading the hardware to achieve the speed. The basis is the observation that in addition of a_i and b_i, the carry-out only depends on the carry in bit when p_i is set. This can be extended to any bit length addition. A single CSKA contains a RCA chain, a group of propagation bit (p_i with $i \in \{0, 1, \ldots, n-1\}$) and a multiplexer, as presented in Fig.A.7. The signal Cin can skip all carry propagation



(a)



(b)

Figure A.6: Logic implementation of: a. 4-bits CLA Adder; b. Carry lookahead chain.

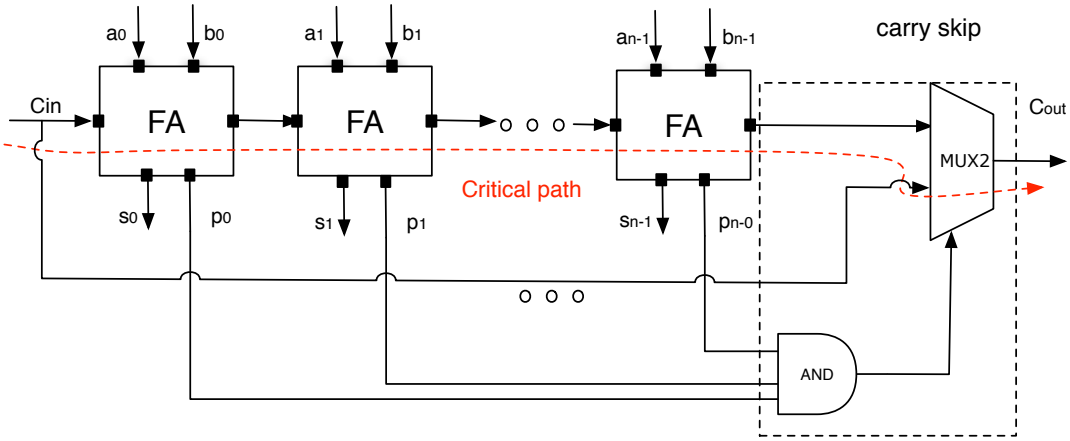


Figure A.7: Classic CSKA architecture.

stages and directly generate a carry-out when the group of p_i is 1. This is the best case of a single CSKA, where the delay is the time consuming on skip stage plus the delay on AND gate.

A single CSKA not really overcomes RCA in delay. In practice, a n-bit CSKA is

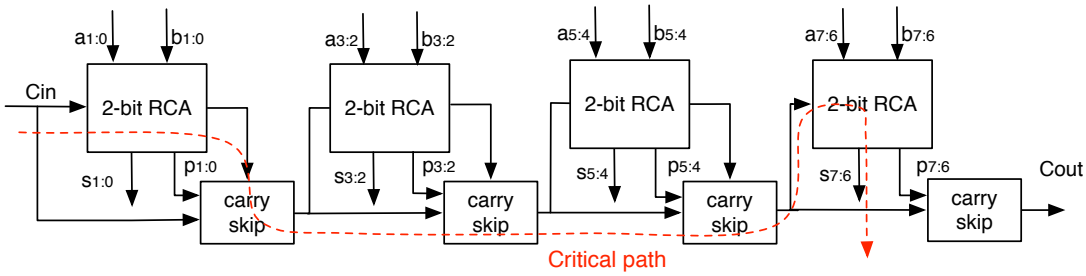


Figure A.8: Architecture of a 8-bit fixed-size CSKA.

composed by several m-bit CSKA blocks, refereed as fixed-size block CSKA. The optimal block size is $m^* = \sqrt{\frac{n}{2}}$. For instance, Fig. A.8 shows a 8-bit adder building from four 2-bit length CSKA. As carry propagation is performed in parallel on each block, the critical path of such an adder is expressed as in (A.11), where T_{RCA2} represents the delay on 2-bit RCA and T_{skip} stands for the delay on a skip block. k is the number of implemented blocks. Furthermore, it is possible to apply variable block-size length. For instance, a suitable solution for a 16-bit CSKA is 7 blocks with block size combination equaling $[1, 2, 3, 4, 3, 2, 1]$.

$$T_{CSKA} = 2 \times T_{RCA2} + (k-1) \times T_{skip} \tag{A.11}$$

A.6 Parallel prefix adder

Actually, binary addition can be considered as a prefix sum problem. The acceleration is achieved through the parallel prefix algorithms which have been intensively reported in literature [89–91]. The prefix sum is a cumulative sum of a sequence of numbers. Suppose n inputs are $x_{n-1}x_{n-2}x_{n-3}\ldots x_0$, the cumulative sum y of n-bit is presented as follows, where "•" denotes the prefix sum.

$$\begin{aligned} y_0 &= x_0 \\ y_1 &= x_0 \bullet x_1 \\ y_2 &= x_0 \bullet x_1 \bullet x_2 \\ y_3 &= x_0 \bullet x_1 \bullet x_2 \bullet x_3 \\ &\ldots \\ y_{n-1} &= x_0 \bullet x_1 \bullet x_2 \bullet x_3 \bullet x_{n-1} \end{aligned} \tag{A.12}$$

Such cumulative sum is expressed in (A.13).

$$y_i = y_{i-1} \bullet x_i \tag{A.13}$$

Due to the fact that the sum is associative, the computation can be implemented serial or parallel as given in Fig. A.10. We remark that the delay is efficiently reduced thanks to the parallel computation.

In parallel prefix adder (PPA) the prefix sum operator • has 4 bit input and 2 bit output (see Fig. A.10). Let us take the expressions in (A.9) as an instance. In this case, x_i is the pair of carry generation and propagation, $\{g_i, p_i\}$, the carry bit of each stage is expressed as follows:

$$\begin{aligned} c_1 &= g_0 \wedge (p_0 \vee Cin) \\ c_2 &= \{g_0, p_0\} \bullet \{g_1, p_1\} \\ c_3 &= \{g_0, p_0\} \bullet \{g_1, p_1\} \bullet \{g_2, p_2\} \\ c_4 &= \{g_0, p_0\} \bullet \{g_1, p_1\} \bullet \{g_2, p_2\} \bullet \{g_3, p_3\} \end{aligned} \tag{A.14}$$

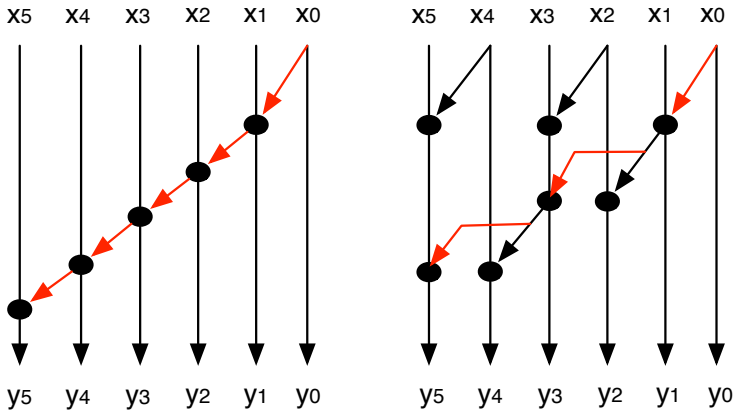


Figure A.9: An example of 6 bit prefix sum with serial implementation (left) and parallel implementation (right) with the critical path colored in red.

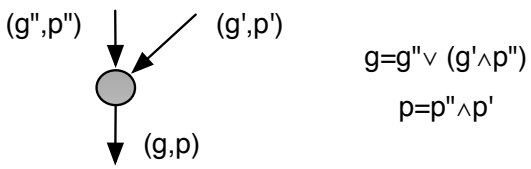


Figure A.10: Prefix sum operator.

Different prefix tree algorithms have been proposed to attain the best tradeoff between the area and delay speed. In this work, we focus on three of them:

- **Sklansky tree** is the first attempt of parallel prefix presented by Sklansky in 1960 [90]. The outputs are evaluated by simply overlay the prefix approach without any optimization. The author uses minimum prefix structure to obtain the signals required by output computation and leads to high fanout (see Fig. A.11). The white balls in the figure represent the HAs evaluating the value of p and g. Nevertheless, Sklansky tree has a short tree depth ($log\ n$) thus a small delay. For instance, 4 stages are required in a 16-bits adder.

- **Kogge-stone tree** Kogge *et al.* proposed a very different structure that contains minimum number of fan-out (no more than 2) [89]. Conversely, the required prefix nodes are remarkable increased and induced a high area overhead as shown in Fig. A.12. The depth of KAS is similar as SKA and equals to $log\ n$.

- **Bren-kung tree** is a compromise solution between prefix nodes and maximum fan-out number introduced by Bren and Kung [91]. As described in Fig. A.13, this structure is based on partially parallel propagation. The tree depth is nearly double of that in SKA and KSA, equals to $2log\ n - 2$. However, the area cost is efficiently reduced compared with KSA.

Table A.2 summarizes the characteristics of each architecture previously presented, according to area, delay, regularity (synthesis).

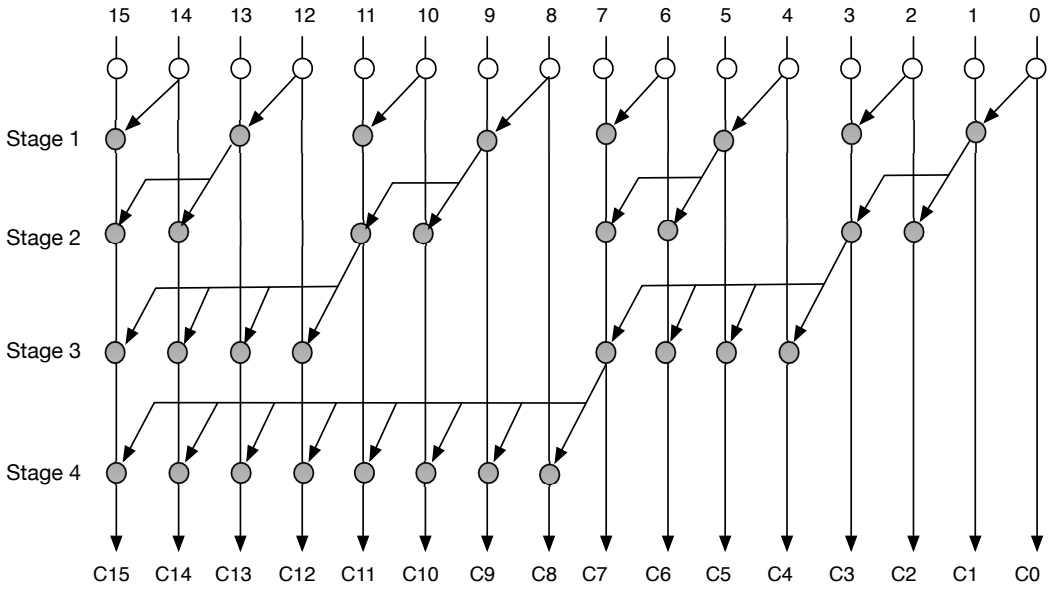


Figure A.11: Structure of 16-bits SKA.

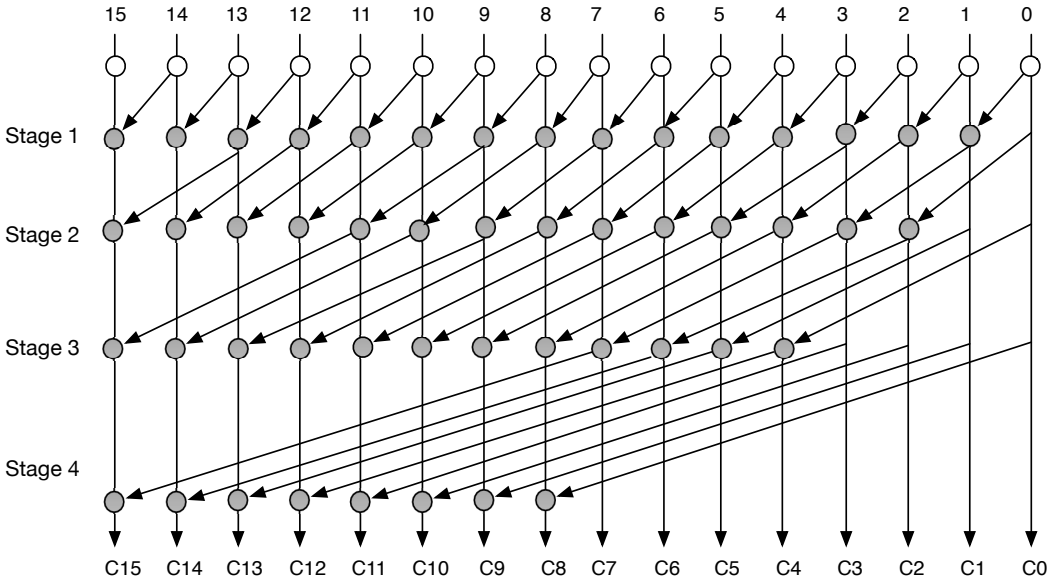


Figure A.12: Structure of 16-bits KSA.

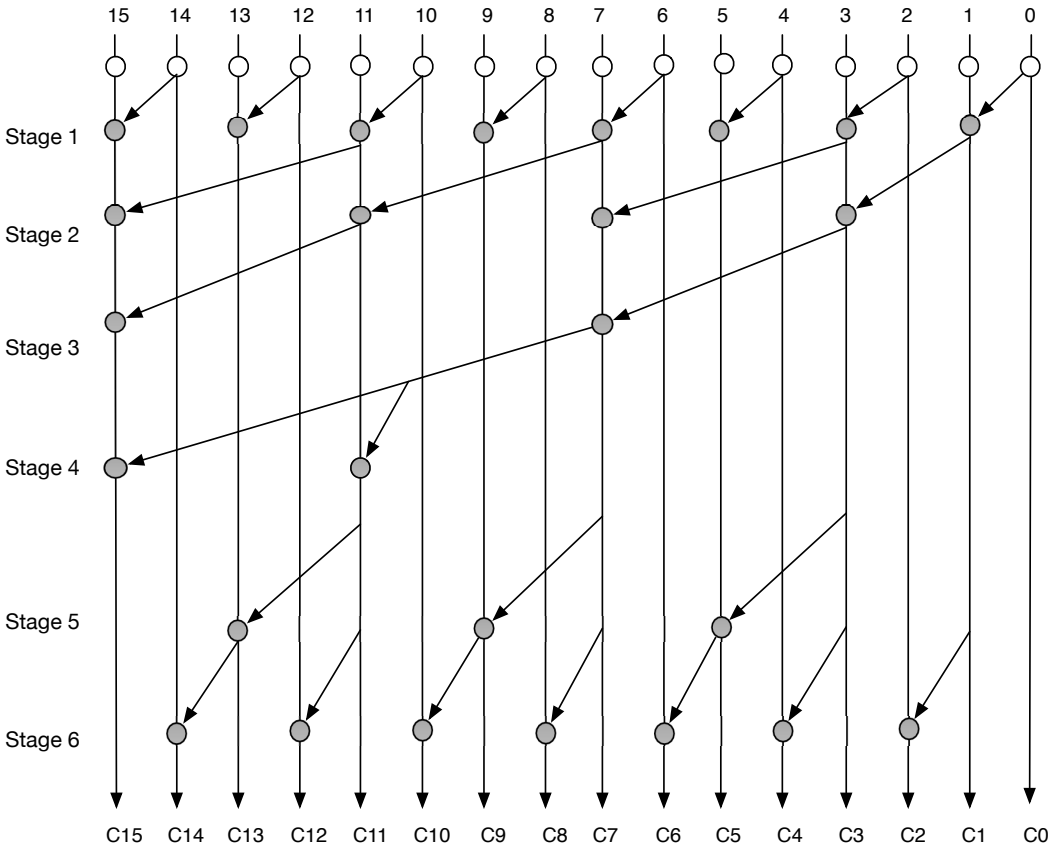


Figure A.13: Structure of 16-bits BKA.

Table A.2: Comparisons of different adder architecture [143]

Adder architecture	**Area**	**Delay**	**Regularity**
RCA	$O(n)$	$O(n)$	highest
CSA	$O(n)$	$O(n^{\frac{1}{2}})$	high
CSKA	$O(n)$	$O(n^{\frac{1}{2}})$	high
CLA	$O(n)$	$O(logn)$	medium
KSA	$O(n\ logn)$	$O(logn)$	medium
SKA	$O(n\ logn)$	$O(logn)$	medium
BKA	$O(n\ logn)$	$O(logn)$	medium

Appendix B

Finite field arithmetic operator

Finite field arithmetics are intensively applied in error correction codes such as Reed-Solomon codes, and cryptography systems like Rijndael encryption algorithm and Elliptic curve cryptography [146, 147]. Finite field is well known as *Galois field* and is always denoted as GF(q), here, q is the *order* of a finite field which indicates the possible element values. The property of limited element number discriminates Galois field arithmetic operations from integer arithmetic operations. This section presents the basic Galois field operations such as addition, subtraction, multiplication and inversion, in particular these of binary field GF(2^m) used in Rijndael algorithm.

B.1 Finite field

A field $\mathbb{F}$ indicates a set of objects equipped only two operations: addition (+) and multiplication ($\cdot$). Define $a, b, c \in \mathbb{F}$, the arithmetic operations on $\mathbb{F}$ respect the following properties:

- $\mathbb{F}$ is a closed set for its two operations:

$$\begin{array}{rcl} a+b & \in & \mathbb{F} \\ a \cdot b & \in & \mathbb{F} \end{array}$$

- $\mathbb{F}$ is a commutative group for + with the additive identity element denoted "0":

$$\begin{array}{rcl} a+b & = & b+a \\ a+0 & = & a \end{array}$$

- $\mathbb{F}$ is a commutative group for $\cdot$ with multiplicative identity element denoted "1":

$$\begin{array}{rcl} a \cdot b & = & b \cdot a \\ a \cdot 1 & = & a \end{array}$$

- Both operations on $\mathbb{F}$ are associative:

$$\begin{array}{rcl} (a+b)+c & = & a+(b+c) \\ (a \cdot b) \cdot c & = & a \cdot (b \cdot c) \end{array}$$

- Each element in $\mathbb{F}$ has an addictive negative element and a multiplicative inverse element ($a \neq 0$):

$$\begin{array}{rcl} a+c=0 & \Rightarrow & c=-a \\ a \cdot c=1 & \Rightarrow & c=a^{-1} \end{array}$$

The field that contains finite elements is named as *finite field*, always denoted as $GF(q)$ where q is the *order* of the field. If there exist $q = p^m$ and m as a integer number, then we nominate p as the *characteristic* of the field. Any two finite fields of the same order q are isomorphic such as $GF(2^8)$ and $GF((2^4)^2)$. In the case of $m = 1$, the field is a *primary field*, otherwise it is a extension field [148]

Let p be as a prime number, the prime field $GF(p)$ includes p integer elements from 0 to $p-1$ with two operations on $GF(p)$ performed modulo p. $GF(2)$ is the simplest prime filed and includes two values, 0 and 1. Addition is an logic "and" and multiplication is a logic "xor". Moreover, the additive inverse of each element is itself, the subtraction is then identical as addition. The field of $GF(2^m)$ where $m > 1$ is a extension field and can be constructed from $GF(2)$ through a primitive polynomial $q(x)$. Each element of $GF(2^m)$ is represented as a polynomial of degree $m-1$ over $GF(2)$.

There exist different representations for elements on $GF(2^m)$ according to the way of its construction, which can be produced from $GF(2)$ or sub-field of $GF(2^m)$. *Subfield* $GF(2^l)$ is defined as a set that respects all operations of $GF(2^m)$, $l \times n = m$ and n is a integer. An extension field which is not defined over the prime field but one of its subfields is known as a *composite field*.

B.2 Addition on $GF(2^m)$

Every element in $GF(2^m)$ is represented as a polynomial coefficient with a maximum degree of $m-1$. Define $a, b \in GF(2^m)$ presented in (B.1).

$$\begin{aligned} a =& a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + a_{m-3}x^{m-3} + \cdots + a_1x + a_0 \quad a_i \in \{0,1\} \\ b =& b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + b_{m-3}x^{m-3} + \cdots + b_1x + b_0 \quad b_i \in \{0,1\} \end{aligned} \tag{B.1}$$

Addition of two numbers on $GF(2^m)$ is achieved by modular adding each coefficient according with the corresponding power of a and b. In other words, such an addition is expressed as $(a+b)\ mod\ 2$ which is implemented by an XOR gate as shown in (B.2)

$$c = (a+b)\ mod\ 2 \Rightarrow c_i = a_i \oplus b_i \tag{B.2}$$

Take $a, b \in GF(2^8)$, $a = \{50\}$ and $b = \{a1\}$ for instance, the addition can be represented into a polynomial format as follows:

$$\begin{aligned} a + b &= \{50\} + \{a1\} \\ &= \{01010010\} + \{10100001\} \\ &= (x^5 + x^4 + z) + (x^7 + x^5 + 1) \\ &= x^7 + x^4 + z + 1 = \{92\} \end{aligned} \tag{B.3}$$

Considering $b \in GF$, $b + (-b) = 0$ where $-b$ is the negative of b and is a unique element in finite field. Then, subtraction of field elements can be defined in terms of addition.

$$a - b = a + (-b) \tag{B.4}$$

B.3 Multiplication on $GF(2^m)$

Unlike addition, multiplication of two numbers in polynomial representation contains two steps [149]:

- Integer multiply the two polynomials and obtain the intermediate polynomial P

- Reduce the result P by modulo $q(x)$ which is a irreducible polynomial of degree m .

Take the same a and b for instance as in addition, the product of them is shown as:

$$\begin{aligned} P &= (x^5 + x^4 + z) \cdot (x^7 + z^5 + 1) \\ &= x^{13} + x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x \end{aligned} \tag{B.5}$$

It is noticed that the maximum power is 13. A polynomial $q(x)$ is applied in order to reduce the degree. Take the $q(x)$ used in AES algorithm [146] as given in (B.6), thereby P can be rewritten in (B.7).

$$q(x) = x^8 + x^4 + x^3 + x + 1 \tag{B.6}$$

$$P \ mod \ q(x) = x^7 + x^5 + x^2 + x^1 = \{10100110\} = \{a6\} \tag{B.7}$$

Bibliography

[1] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, 1965.

[2] ITRS2013, "International technology roadmap for semiconductors." www.itrs.net/Links/2013ITRS/Home2011.htm, 2013.

[3] C. Hu, S. C. Tam, F.-C. Hsu, P.-K. Ko, T.-Y. Chan, and K. W. Terrill, "Hot-electron-induced MOSFET degradationmodel, monitor, and improvement," *IEEE Transactions on Electron Devices*, vol. 32, no. 2, pp. 375–385, 1985.

[4] K. O. Jeppson and C. M. Svensson, "Negative bias stress of MOS devices at high electric fields and degradation of mnos devices," *Journal of Applied Physics*, vol. 48, no. 5, pp. 2004–2014, 2008.

[5] J. Stathis, "Percolation models for gate oxide breakdown," *Journal of Applied Physics*, vol. 86, no. 10, pp. 5757–5766, 1999.

[6] J. Lienig, "Introduction to electromigration-aware physical design," in *Proceedings of the 2006 international symposium on Physical design*, pp. 39–46, ACM, 2006.

[7] G. Gielen, P. De Wit, E. Maricau, J. Loeckx, J. Martín-Martínez, B. Kaczer, G. Groeseneken, R. Rodríguez, and M. Nafría, "Emerging yield and reliability challenges in nanometer CMOS technologies," in *Proceedings of the conference on Design, automation and test in Europe*, pp. 1322–1327, ACM, 2008.

[8] L. B. Kish, "End of moore's law: Thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, pp. 144–149, Dec. 2002.

[9] N. Seifert, P. Slankard, M. Kirsch, B. Narasimham, V. Zia, C. Brookreson, A. Vo, S. Mitra, B. Gill, and J. Maiz, "Radiation-induced soft error rates of advanced CMOS bulk devices," in *Reliability Physics Symposium Proceedings, 2006. 44th Annual.*, pp. 217–225, 2006.

[10] N. Seifert, V. Ambrose, B. Gill, Q. Shi, R. Allmon, C. Recchia, S. Mukherjee, N. Nassif, J. Krause, J. Pickholtz, *et al.*, "On the radiation-induced soft error performance of hardened sequential elements in advanced bulk cmos technologies," in *Proceedings of IEEE International on Reliability Physics Symposium (IRPS)*, pp. 188–197, IEEE, 2010.

[11] W. Arden, "More than moore white paper by the IRC - ITRS." http://www.itrs.net/Links/2010ITRS/IRC-ITRS-MtM-v2%203.pdf, 2011.

[12] S. Dhar, M. Pattanaik, and P. Rajaram, "Advancement in nanoscale cmos device design en route to ultra-low-power applications," *VLSI Design*, vol. 2011, pp. 2:1–2:19, Jan. 2011.

[13] T. Skotnicki, J. A. Hutchby, T.-J. King, H.-S. Wong, and F. Boeuf, "The end of cmos scaling: toward the introduction of new materials and structural changes to improve mosfet performance," *IEEE Circuits and Devices Magazine*, vol. 21, no. 1, pp. 16–26, 2005.

[14] M. Bruel, "Silicon on insulator material technology," *Electronics letters*, vol. 31, no. 14, pp. 1201–1202, 1995.

[15] E. J. Nowak, I. Aller, T. Ludwig, K. Kim, R. V. Joshi, C.-T. Chuang, K. Bernstein, and R. Puri, "Turning silicon on its edge [double gate cmos/finfet technology]," *IEEE Circuits and Devices Magazine*, vol. 20, no. 1, pp. 20–31, 2004.

[16] K. Teo, R. Lacerda, M. Yang, A. Teh, L. Robinson, S. Dalal, N. Rupesinghe, M. Chhowalla, S.-B. Lee, D. Jefferson, *et al.*, "Carbon nanotube technology for solid state and vacuum electronics," in *IEE Proceedings-Circuits, Devices and Systems,*, vol. 151, pp. 443–451, IET, 2004.

[17] G. Bourianoff, "The future of nanocomputing," *Computer*, vol. 36, no. 8, pp. 44–53, 2003.

[18] M. T. Bohr, "Nanotechnology goals and challenges for electronic applications," *IEEE Transactions on Nanotechnology*, vol. 1, no. 1, pp. 56–62, 2002.

[19] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, p. 49, 1993.

[20] I. Koren and C. M. Krishna, *Fault-tolerant systems.* Morgan Kaufmann, 2010.

[21] F. Crupi, B. Kaczer, R. Degraeve, V. Subramanian, P. Srinivasan, E. Simoen, A. Dixit, M. Jurczak, and G. Groeseneken, "Reliability comparison of triple-gate versus planar soi fets," *IEEE Transactions on Electron Devices*, vol. 53, no. 9, pp. 2351–2357, 2006.

[22] M. Jurczak, N. Collaert, A. Veloso, T. Hoffmann, and S. Biesemans, "Review of finfet technology," in *Proceedings of IEEE International SOI Conference*, pp. 1–4, IEEE, 2009.

[23] S. Sugahara and J. Nitta, "Spin-transistor electronics: an overview and outlook," *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2124–2154, 2010.

[24] M. Goessel, V. Ocheretny, E. Sogomonyan, and D. Marienfeld, *New Methods of Concurrent Checking (Frontiers in Electronic Testing).* Springer Publishing Company, Incorporated, 2008.

[25] D. T. Franco, *Fiabilité du signal des circuits logiques combinatoires sous fautes simultanées multiples.* PhD thesis, Télécom ParisTech, 2008.

[26] M. Stanisavljevic, A. Schmid, and Y. Leblebici, *Reliability of nanoscale circuits and systems.* No. EPFL-BOOK-164975, Springer, 2011.

[27] R. E. Barlow and F. Proschan, *Mathematical theory of reliability*, vol. 17. Siam, 1996.

[28] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.

[29] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE micro*, vol. 23, no. 4, pp. 14–19, 2003.

[30] L. Anghel, R. Leveugle, and P. Vanhauwaert, "Evaluation of SET and SEU effects at multiple abstraction levels," *Proceedings of the 11th IEEE International On-Line Testing Symposium*, vol. 0, pp. 309–312, 2005.

[31] L. M. Terman, "An investigation of surface states at a silicon/silicon oxide interface employing metal-oxide-silicon diodes," *Solid-State Electronics*, vol. 5, no. 5, pp. 285–299, 1962.

[32] J. H. Stathis and S. Zafar, "The negative bias temperature instability in MOS devices: A review," *Microelectronics Reliability*, vol. 46, no. 2, pp. 270–286, 2006.

[33] J. W. McPherson, "Reliability challenges for 45nm and beyond," in *Proceedings of the 43rd annual Design Automation Conference*, pp. 176–181, ACM, 2006.

[34] BSIM, "BSIM4 homepage, version 4.7." http://www-device.eecs.berkeley.edu/bsim/bsim4.htm, 2012.

[35] H. Cai, H. Petit, and J.-F. Naviner, "A hierarchical reliability simulation methodology for AMS integrated circuits and systems," *Journal of Low Power Electronics*, vol. 8, no. 5, pp. 697–705, 2012.

[36] C. Hao, *Fiabilisation de Convertisseurs Analogique-Numérique à Modulation Sigma-Delta*. PhD thesis, Télécom ParisTech, 2013.

[37] D. Binder, E. Smith, and A. Holman, "Satellite anomalies from galactic cosmic rays," *IEEE Transactions on Nuclear Science*, vol. 22, no. 6, pp. 2675–2680, 1975.

[38] R. Baumann, "Soft errors in advanced computer systems," *Design & Test of Computers, IEEE*, vol. 22, no. 3, pp. 258–266, 2005.

[39] G. Srinivasan, P. Murley, and H. Tang, "Accurate, predictive modeling of soft error rate due to cosmic rays and chip alpha radiation," in *32nd IEEE International Annual Proceedings. on Reliability Physics Symposium*, pp. 12–16, IEEE, 1994.

[40] S. V. Walstra and C. Dai, "Circuit-level modeling of soft errors in integrated circuits," *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 358–364, 2005.

[41] S. Uznanski, G. Gasiot, P. Roche, C. Tavernier, and J. Autran, "Single event upset and multiple cell upset modeling in commercial bulk 65-nm CMOS SRAMs and flip-flops," *IEEE Transactions on Nuclear Science*, vol. 57, no. 4, pp. 1876–1883, 2010.

[42] H. Cha and J. H. Patel, "A logic-level model for α-particle hits in CMOS circuits," in *Computer Design: VLSI in Computers and Processors, 1993. ICCD'93. Proceedings., 1993 IEEE International Conference on*, pp. 538–542, IEEE, 1993.

[43] L. B. Kish, "End of moore's law: thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 3, pp. 144–149, 2002.

[44] S. Mitra and E. J. McCluskey, "Which concurrent error detection scheme to choose?," in *Test Conference, 2000. Proceedings. International*, pp. 985–994, 2000.

[45] G. D. Kraft and W. N. Toy, *Microprogrammed control and reliable design of small computers*. Englewood Cliffs, N.J. Prentice-Hall, 1981.

[46] B. K. Kumar and P. K. Lala, "On-line detection of faults in carry-select adders," in *Proceedings of IEEE International Test Conference (ITC)*, pp. 912–912, IEEE Computer Society, 2003.

[47] M. Nicolaidis, R. Duarte, S. Manich, and J. Figueras, "Fault-secure parity prediction arithmetic operators," *Design & Test of Computers, IEEE*, vol. 14, no. 2, pp. 60–71, 1997.

[48] M. Nicolaidis, "Carry checking/parity prediction adders and alus," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 1, pp. 121–128, 2003.

[49] G. Cardarilli, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, "Error detection in signed digit arithmetic circuit with parity checker [adder example]," in *Proceedings of the 18th IEEE International Symposium Defect and Fault Tolerance in VLSI Systems, 2003.*, pp. 401–408, IEEE, 2003.

[50] D. Marienfeld, E. Sogomonyan, V. Otcheretnij, and M. Gossel, "A new self-checking and code-disjoint non-restoring array divider," in *Proceedings of the 12th IEEE International On-Line Testing Symposium (IOLTS)*, pp. 23–30, IEEE, 2006.

[51] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Concurrent structure-independent fault detection schemes for the advanced encryption standard," *IEEE Transactions on Computers*, vol. 59, no. 5, pp. 608–622, 2010.

[52] P. Lala and A. Walker, "On-line error detectable carry-free adder design," in *Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 66–71, IEEE, 2001.

[53] J. Lo, S. Thanawastien, and T. Rao, "Concurrent error detection in arithmetic and logical operations using berger codes," in *Proceedings of the 9th Symposium on Computer Arithmetic*, pp. 233–240, IEEE, 1989.

[54] I. Proudler, "Idempotent AN codes," in *IEE Colloquium on Signal Processing Applications of Finite Field Mathematics*, pp. 8–1, IET, 1989.

[55] G. Gaubatz and B. Sunar, "Robust finite field arithmetic for fault-tolerant public-key cryptography," *Fault Diagnosis and Tolerance in Cryptography*, pp. 196–210, 2006.

[56] J. Biernat, "The complexity of fault-tolerant adder structures," in *Third International Conference on Dependability of Computer Systems, 2008. DepCos-RELCOMEX'08*, pp. 316–323, IEEE, 2008.

[57] R. Forsati, K. Faez, F. Moradi, and A. Rahbar, "A fault tolerant method for residue arithmetic circuits," in *Proceedings of International Conference on Information Management and Engineering (ICIME)*, pp. 59–63, IEEE, 2009.

[58] M. Medwed and J.-M. Schmidt, "Coding schemes for arithmetic and logic operations-how robust are they?," in *Information Security Applications*, pp. 51–65, Springer, 2009.

[59] D. Franco, M. Vasconcelos, L. Naviner, and J. Naviner, "Signal probability for reliability evaluation of logic circuits," *Microelectronics Reliability*, vol. 48, no. 8, pp. 1586–1591, 2008.

[60] J. Han, H. Chen, E. Boykin, and J. Fortes, "Reliability evaluation of logic circuits using probabilistic gate models," *Microelectronics Reliability*, vol. 51, no. 2, pp. 468–476, 2011.

[61] S. Pagliarini, A. Ben Dhia, L. d. B. Naviner, and J.-F. Naviner, "Snap: A novel hybrid method for circuit reliability assessment under multiple faults," *Microelectronics Reliability*, vol. 53, no. 9, pp. 1230–1234, 2013.

[62] P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proceedings of International Conference on Dependable Systems and Networks, 2002. DSN 2002*, pp. 389–398, IEEE, 2002.

[63] Y. S. Dhillon, A. U. Diril, and A. Chatterjee, "Soft-error tolerance analysis and optimization of nanometer circuits," in *Design, Automation, and Test in Europe*, pp. 389–400, Springer, 2008.

[64] B. Zhang, W.-S. Wang, and M. Orshansky, "FASER: Fast analysis of soft error susceptibility for cell-based designs," in *Proceedings of the 7th international symposium on quality electronic design*, pp. 755–760, IEEE Computer Society, 2006.

[65] M. Omana, G. Papasso, D. Rossi, and C. Metra, "A model for transient fault propagation in combinatorial logic," in *Proceedings of 9th IEEE On-Line Testing Symposium, 2003. IOLTS 2003*, pp. 111–115, IEEE, 2003.

[66] F. Wang and Y. Xie, "Soft error rate analysis for combinational logic using an accurate electrical masking model," *IEEE Transactions on Dependable and Secure Computing,*, vol. 8, no. 1, pp. 137–146, 2011.

[67] ISCAS85. http://www.pld.ttu.ee/~maksim/benchmarks/iscas85/verilog/. [Online: accessed 10-July-2014].

[68] H. Fujiwara and S. Toida, "The complexity of fault detection problems for combinational logic circuits," *IEEE Transactions on Computers*, vol. 100, no. 6, pp. 555–560, 1982.

[69] K. Patel, J. Hayes, and I. Markov, "Evaluating circuit reliability under probabilistic gate-level fault models," in *Proceedings of the International Workshop on Logic and Synthesis*, pp. 59–64, 2003.

[70] S. Krishnaswamy, I. L. Markov, and J. P. Hayes, *Design, analysis and test of logic circuits under uncertainty.* Springer, 2013.

[71] J. Galey, R. Norby, and J. P. Roth, "Techniques for the diagnosis of switching circuit failures," *IEEE Transactions on Communication and Electronics*, vol. 83, no. 74, pp. 509–514, 1964.

[72] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Accurate reliability evaluation and enhancement via probabilistic transfer matrices," in *Design, Automation and Test in Europe, 2005. Proceedings*, pp. 282–287, IEEE, 2005.

[73] L. A. Naviner, M. C. de Vasconcelos, D. T. Franco, and J.-F. Naviner, "Efficient computation of logic circuits reliability based on probabilistic transfer matrix," in *3rd International Conference on Design and Technology of Integrated Systems in Nanoscale Era, 2008. DTIS 2008.*, pp. 1–4, IEEE, 2008.

[74] M. de Vasconcelos, D. Franco, J.-F. Naviner, *et al.*, "Efficient computation of logic circuits reliability based on probabilistic transfer matrix," in *Design and Technology of Integrated Systems in Nanoscale Era, 2008. DTIS 2008. 3rd International Conference on*, pp. 1–4, 2008.

[75] L. A. Naviner, J.-F. Naviner, *et al.*, "FIFA: A fault-injection–fault-analysis-based tool for reliability assessment at RTL level," *Microelectronics Reliability*, vol. 51, no. 9, pp. 1459–1463, 2011.

[76] J. Han, E. Taylor, J. Gao, and J. Fortes, "Faults, error bounds and reliability of nanoelectronic circuits," in *Proceedings of IEEE International Conference on Application-Specific Systems, Architecture Processors*, ASAP '05, pp. 247–253, july 2005.

[77] E. Taylor, J. Han, and J. Fortes, "Towards accurate and efficient reliability modeling of nano-electronic circuits," in *Proceedings of the 6th IEEE Conference on Nanotechnology (IEEE-NANO)*, p. 395398, june 2006.

[78] J. T. Flaquer, J.-M. Daveau, L. Naviner, and P. Roche, "Fast reliability analysis of combinatorial logic circuits using conditional probabilities," *Microelectronics Reliability*, vol. 50, no. 9, pp. 1215–1218, 2010.

[79] N. Mohyuddin, E. Pakbaznia, and M. Pedram, "Probabilistic error propagation in a logic circuit using the boolean difference calculus," in *Advanced Techniques in Logic Synthesis, Optimizations and Applications*, pp. 359–381, Springer, 2011.

[80] H. Chen and J. Han, "Stochastic computational models for accurate reliability evaluation of logic circuits," in *Proceedings of the 20th symposium on Great lakes symposium on VLSI*, GLSVLSI '10, pp. 61–66, ACM, 2010.

[81] H. Chen, J. Han, and F. Lombardi, "A transistor-level stochastic approach for evaluating the reliability of digital nanometric cmos circuits," in *Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp. 60 –67, oct. 2011.

[82] W. Ibrahim and V. Beiu, "Reliability of nand-2 cmos gates from threshold voltage variations," in *International Conference on Innovations in Information Technology (IIT)*, pp. 135–139, Dec.

[83] K. Parker and E. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Transactions on Computers*, vol. C-24, pp. 668 – 670, june 1975.

[84] K. Nikolic, A. Sadek, and M. Forshaw, "Architectures for reliable computing with unreliable nanodevices," in *Proceedings of the 2001 1st IEEE Conference on Nanotechnology (IEEE-NANO)*, pp. 254–259, 2001.

[85] J. Volder, "The CORDIC trigonometric computing technique," *IRE Transactions on Electronic Computers*, no. 3, pp. 330–334, 1959.

[86] T. Adiono and R. Purba, "Scalable pipelined CORDIC architecture design and implementation in FPGA," in *Proceedings of the International Conference on Electrical Engineering and Informatics (ICEEI)*, pp. 646 – 649, IEEE, Sept. 2009.

[87] K. Maharatna, S. Banerjee, E. Grass, M. Krstic, and A.Troya, "Modified virtually scaling-free adaptive CORDIC rotator algorithm and architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, pp. 1463 – 1474, Oct. 2005.

[88] A. Guyot, B. Hochet, and J.-M. Muller, "A way to build efficient carry-skip adders," *IEEE Transactions on Computers*, vol. 36, no. 10, pp. 1144–1152, 1987.

[89] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 786–793, 1973.

[90] J. Sklansky, "Conditional-sum addition logic," *IRE Transactions on Electronic Computers*, no. 2, pp. 226–231, 1960.

[91] R. P. Brent and H. Kung, "A regular layout for parallel adders," *IEEE Transactions on Computers*, vol. 31, no. 3, pp. 260–264, 1982.

[92] R. Zimmermann, *Binary Adder Architectures for Cell-Based VLSI and their Synthesis*. PhD thesis, Swiss Federal Institute of Technology (ETH) Zurich, Hartung-Gorre Verlag, 1998.

[93] D. P. Vasudevan, P. K. Lala, and J. P. Parkerson, "Self-checking carry-select adder design based on two-rail encoding," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 12, pp. 2696–2705, 2007.

[94] S. Bayat-Sarmadi and M. A. Hasan, "Concurrent error detection in finite-field arithmetic operations using pipelined and systolic architectures," *IEEE Transactions on Computers*, vol. 58, no. 11, pp. 1553–1567, 2009.

[95] C. Rusu, A. Bougerol, L. Anghel, C. Weulerse, N. Buard, S. Benhammadi, N. Renaud, G. Hubert, F. Wrobel, T. Carrière, *et al.*, "Multiple event transient induced by nuclear reactions in CMOS logic cells," in *Proceedings of 13th IEEE International On-Line Testing Symposium*, pp. 137–145, IEEE, 2007.

[96] X. Guo and R. Karri, "Invariance-based concurrent error detection for advanced encryption standard," in *Proceedings of the 49th Annual Design Automation Conference*, pp. 573–578, 2012.

[97] M. C. de Vasconcelos, D. T. Franco, L. A. Naviner, and J.-F. Naviner, "Relevant metrics for evaluation of concurrent error detection schemes," *Microelectronics Reliability*, vol. 48, no. 8, pp. 1601–1603, 2008.

[98] W. Kuo, V. R. Prasad, F. A. Tillman, and C. L. Hwang, *Optimal Reliability Design: Fundamentals and Applications*. Cambridge University Press, 1999.

[99] Matlab. http://www.matlab.com/, 2014.

[100] ModelSim. http://www.mentor.com/, 2014.

[101] T. Ban, J. Wang, T. An, and L. Naviner, "Modeling of transient faults and fault-tolerant design in nanoelectronics," in *Proceedings of 56th IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 545–548, IEEE, 2013.

[102] W. C. Carter and P. R. Schneider, "Design of dynamically checked computers.," in *IFIP Congress (2)*, pp. 878–883, 1968.

[103] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Probabilistic transfer matrices in symbolic reliability analysis of logic circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 13, no. 1, p. 8, 2008.

[104] W. Wang, S. Yang, S. Bhardwaj, R. Vattikonda, S. Vrudhula, F. Liu, and Y. Cao, "The impact of NBTI on the performance of combinational and sequential circuits," in *Proceedings of the 44th annual Design Automation Conference*, pp. 364–369, ACM, 2007.

[105] R. Baranowski, A. Cook, M. E. Imhof, C. Liu, and H.-J. Wunderlich, "Synthesis of workload monitors for on-line stress prediction," in *Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp. 137–142, IEEE, 2013.

[106] D. Lorenz, M. Barke, and U. Schlichtmann, "Efficiently analyzing the impact of aging effects on large integrated circuits," *Microelectronics Reliability*, vol. 52, no. 8, pp. 1546–1552, 2012.

[107] Mentor Graphics Corporation, "Eldo user manual," 1999.

[108] H. Cai, H. Petit, and J.-F. Naviner, "Reliability aware design of low power continuous-time sigma–delta modulator," *Microelectronics Reliability*, vol. 51, no. 9, pp. 1449–1453, 2011.

[109] National Institute of Standards and Technology, *Announcing the Advanced Encryption Standard (AES) [electronic resource]*. No. 197, Nov. 2001.

[110] R. Banu and T. Vladimirova, "On-board encryption in earth observation small satellites," in *Proceedings of the 40th Annual IEEE International Carnahan Conferences Security Technology*, pp. 203–208, 2006.

[111] C. Herbst, E. Oswald, and S. Mangard, "An AES smart card implementation resistant to power analysis attacks," in *Applied Cryptography and Network Security*, pp. 239–252, Springer, 2006.

[112] C. Giraud, "DFA on AES," in *Advanced Encryption Standard - AES*, vol. 3373, pp. 27–41, Springer Berlin Heidelberg, 2005.

[113] G. Piret and J. Quisquater, "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD," *Cryptographic Hardware and Embedded Systems-CHES 2003*, pp. 77–88, 2003.

[114] P. Maistri, P. Vanhauwaert, and R. Leveugle, "A novel double-data-rate AES architecture resistant against fault injection," in *Fault Diagnosis and Tolerance in Cryptography*, pp. 54–61, 2007.

[115] A. Barenghi, G. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi, "Low voltage fault attacks to aes," in *Proceedings of IEEE International Symposium Hardware-Oriented Security and Trust (HOST)*, pp. 7–12, 2010.

[116] S. Morioka and A. Satoh, "An optimized s-box circuit architecture for low power AES design," *Cryptographic Hardware and Embedded Systems-CHES 2002*, pp. 271–295, 2003.

[117] X. Zhang and K. Parhi, "On the optimum constructions of composite field for the AES algorithm," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 10, pp. 1153–1157, 2006.

[118] N. Ahmad and S. Rezaul Hasan, "Low-power compact composite field AES S-Box/Inv S-Box design in 65nm CMOS using novel XOR gate," *the VLSI Journal on Integration*, 2012.

[119] STMicroelectronics . http://www.st.com/, 2014.

[120] CADENCE. http://www.cadence.com/, 2014.

[121] R. Banu and T. Vladimirova, "Fault-tolerant encryption for space applications," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, pp. 266–279, 2009.

[122] G. Bertoni, L. Breveglieri, I. Koren, P. Maistri, and V. Piuri, "Error analysis and detection procedures for a hardware implementation of the advanced encryption standard," *Computers, IEEE Transactions on*, vol. 52, no. 4, pp. 492–505, 2003.

[123] M. Czapski and M. Nikodem, "Error detection and error correction procedures for the advanced encryption standard," *Designs, Codes and Cryptography*, vol. 49, no. 1-3, pp. 217–232, 2008.

[124] N. Yu and H. M. Heys, "A hybrid approach to concurrent error detection for a compact ASIC implementation of the advanced encryption standard," 2007.

[125] G. Di Natale, M. Doulcier, M.-L. Flottes, and B. Rouzeyre, "A reliable architecture for parallel implementations of the advanced encryption standard," *Journal of Electronic Testing*, vol. 25, no. 4, pp. 269–278, 2009.

[126] A. Satoh, T. Sugawara, N. Homma, and T. Aoki, "High-performance concurrent error detection scheme for AES hardware," in *Cryptographic Hardware and Embedded Systems (CHES)*, pp. 100–112, Springer, 2008.

[127] F. Rodriguez-Henriquez, N. Saqib, and A. Diaz-Perez, "4.2 gbit/s single-chip FPGA implementation of AES algorithm," *Electronics Letters*, vol. 39, pp. 1115–1116, 2003.

[128] K. Gaj and P. Chodowiec, "FPGA and ASIC implementations of AES," *Cryptographic engineering*, pp. 235–294, 2009.

[129] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Efficient and high-performance parallel hardware architectures for the AES-GCM," *IEEE Transactions on Computers*, vol. 61, no. 8, pp. 1165–1178, 2012.

[130] D. Canright, "A very compact s-box for AES," in *Cryptographic Hardware and Embedded Systems–CHES 2005*, pp. 441–455, Springer, 2005.

[131] T. An, K. Liu, H. Cai, and L. Alves de Barros Naviner, "Accurate reliability analysis of concurrent checking circuits employing an efficient analytical method," *Microelectronics Reliability*, Jan. 2015.

[132] T. An, K. Liu, and L. Alves de Barros Naviner, "Efficient implementation for accurate analysis of CED circuits against multiple faults," in *21th Mixed Design of Integrated Circuits and Systems, MIXDES*, June 2014.

[133] T. An, H. Cai, and L. Alves de Barros Naviner, "Simulation study of aging in cmos binary adders," in *MIPRO 37th International Convention/Microelectronics, Electronics and Electronic Technology*, May 2014.

[134] T. An, K. Liu, and L. Alves de Barros Naviner, "Analytical method for reliability assessment of concurrent checking circuits under multiple faults," in *MIPRO 37th International Convention/Microelectronics, Electronics and Electronic Technology*, May 2014.

[135] T. An, L. Alves de Barros Naviner, and P. Matherat, "A low cost reliable architecture for S-Boxes in AES processors," in *IEEE Symp. Defect and Fault Tolerance (DFT)*, pp. 155–160, Oct. 2013.

[136] T. An. K. Liu, C. Hao, L. Alves de Barros Naviner, and P. Matherat, "Reliability analysis of combinational circuits with the influences of noise and single-event transients," in *IEEE Symp. Defect and Fault Tolerance (DFT)*, pp. 218–223, Oct. 2013.

[137] T. Ban, J. Wang, T. An, and L. Naviner, "Modeling of transient faults and fault-tolerant design in nanoelectronics," in *Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium on*, pp. 545–548, IEEE, 2013.

[138] T. An. K. Liu and L. Alves de Barros Naviner, "A general cost-effective design structure for probabilistic-based noise-tolerant logic functions in nanometer CMOS technology," in *IEEE Eurocon conference*, pp. 1829–1836, July 2013.

[139] T. An, L. Alves de Barros Naviner, and P. Matherat, "Exploring the impact of transient faults on CORDIC processor," in *Journées Nationales du Réseau Doctoral en Micro-nanoélectronique (JNRDM)*, June 2013.

[140] T. An, L. Alves de Barros Naviner, and P. Matherat, "Evaluation of fault-tolerant composite field AES S-Boxes under multiple transient faults," in *IEEE International NEWCAS Conference*, pp. 1–4, June 2013.

[141] T. An, M. Causo, L. Alves de Barros Naviner, and P. Matherat, "Transient fault analysis of CORDIC processor," in *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, pp. 757–760, Dec. 2012.

[142] M. Causo, T. An, L. Alves de Barros Naviner, and P. Matherat, "Parallel scaling-free and area-time efficient CORDIC algorithm," in *IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*, pp. 149–152, Dec. 2012.

[143] R. Zimmermann, "Computer arithmetic: Principles, architectures, and VLSI design, available:." http://www.iis.ee.ethz.ch/~zimmi/publications/comp_arith_notes.pdf, 1999. [Online: accessed 10-July-2014].

[144] O. Bedrij, "Carry-select adder," *IRE Transactions on Electronic Computers*, no. 3, pp. 340–346, 1962.

[145] D. P. Vasudevan and P. K. Lala, "A technique for modular design of self-checking carry-select adder," in *Proceedings of 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 325–333, IEEE, 2005.

[146] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," 1998.

[147] D. Husemöller, R. Lawrence, O. Forster, and S. Theisen, *Elliptic curves*, vol. 111. Springer, 2004.

[148] D. Hankerson, S. Vanstone, and A. J. Menezes, *Guide to elliptic curve cryptography*. Springer, 2004.

[149] J.-P. Deschamps, G. J. Bioul, and G. D. Sutter, *Synthesis of arithmetic circuits: FPGA, ASIC and embedded systems.* John Wiley & Sons, 2006.

Architectures d'opérateurs numériques auto-contrôlables

Ting AN

RESUME : La réduction géométrique régulière des finesses de gravure en microélectronique a conduit continue apporte de nouveaux défis aux circuits intégrés (CIs). Leur conception et fabrication sont de plus en plus complexes qu'avant. Les CIs sont affectés par deux phénomnes majeurs : la variabilité paramétrique et les limites des procédés de fabrication, ainsi que la sensibilité aux conditions environnementales. Avec l'augmentation du taux de défaillance lié à ces deux phénomènes, les circuits basés sur les technologies nanoélectroniques sont censés être de moins en moins fiables. Le critère de fiabilit est exigé dans les applications critiques telles que l'avionique, des transports et de biomédecine. Parmi les techniques existants, l'amélioration au niveau de l'architecture profite de l'indépendance de la technologie et de la faible latence de réaction. Les solutions architecturales faisant l'objet de cette thse sont du type auto-contrôlables, c'est-à-dire qui ont capables d'indiquer automatiquement l'apparition de fautes ou de masquer les fautes directement. Cette thèse est consacrée aux méthodes d'analyse et d'amélioration de la fiabilité au niveau de l'architecture. Les problèmes de fiabilité pendant la durée d'utilisation d'un circuit électronique sont décrits en détails. Les opérateurs arithmétiques numériques pour le traitement du signal sont pris comme des études de cas.

MOTS-CLEFS : Fiabilié, Auto-contrôlables, Effets des radiations, Vieillissement, Techniques de tolérance aux fautes, Analyse de la fiabilié, Conception efficace, Opérateurs arthmétiques

Architectures of self-controllable digital operators

ABSTRACT : The steady geometrical reduction of CMOS technology brought new challenges to the integrated circuits (ICs). The design and manufacturing are becoming more complex than before. ICs suffer from two major problems : the parametric variability in materials and limited precision processes, and the sensibility to environment noise. With the increasing failure rate related to these two problems, the future ICs implemented with sub-micron CMOS technology are expected to be less reliable. New reliable ICs are highly desired in critical applications such as avionic, transport and biomedicine. Numerous solutions have been reported in literature covering the enhancement in different abstraction levels (i.e., system level, architecture level and electrical level). Among different techniques, the improvement in architecture level benefits the independence from CMOS technology and the low latency of reaction. Expected architectural solutions will be self-controlled meaning that is able to either automatically indicate the occurrence of faults or directly mask the faults. This thesis is devoted to the reliability analysis methodology and reliability enhancement approaches on architecture level. In particular, the reliability issues in usage time are discussed in details. Digital arithmetic operators for signal processing are taken as studied objects.

KEY-WORDS : Reliability, Self-controllable, Radiation related effects, Aging effects, Fault-tolerant, Reliability analysis, Effective design, arithmetic operators

TELECOM ParisTech

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY