Hardware / Software Architectural and Technological Exploration for Energy-Efficient and Reliable Biomedical Devices

THÈSE N^{O} 8917 (2018)

PRÉSENTÉE LE 30 NOVEMBRE 2018

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE DES SYSTÈMES EMBARQUÉS

PROGRAMME DOCTORAL EN MICROSYSTÈMES ET MICROÉLECTRONIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Loris Gérard DUCH

acceptée sur proposition du jury:

Prof. G. De Micheli, président du jury
Prof. D. Atienza Alonso, directeur de thèse
Prof. F. Catthoor, rapporteur
Prof. D. Sciuto, rapporteuse
Prof. A. P. Burg, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2018

With engineering, I view this year's failure as next year's opportunity to try it again. Failures are not something to be avoided. You want to have them happen as quickly as you can so you can make progress rapidly.
— Gordon Moore

To my family

Acknowledgements

Before my first day at the Embedded Systems Laboratory (ESL), the 1st of September 2014, I did not realize how lucky I was to join the research team of *Prof. David Atienza*. Since then, as a remarkable thesis advisor, he always knew how to enrich my mind and cultivate my curiosity to turn myself into an accomplished researcher. His enthusiasm, kindness, commitment and faith in my abilities have been extremely helpful and motivating to carry out my research work under the best possible conditions. For all of these reasons, I am deeply grateful to him. Nevertheless, he is not the only person who strongly contributed to my doctoral studies. In the following paragraphs, I will try to express my sincerest gratitude to all those who helped me to produce this manuscript.

First of all, I greatly thank my thesis jury members *Prof. Giovanni De Micheli*, *Prof. Francky Catthoor*, *Prof. Donatella Sciuto* and *Prof. Andreas Burg*, for taking their precious time to review and provide valuable insights on my thesis.

Then, I am grateful to the postdoctoral researchers *Giovanni Ansaloni*, *Pablo Garcia del Valle*, *Rubén Braojos*, *Miguel Peón-Quirós* and *Alexandre Levisse*, who carefully oversaw my work and helped me face the avalanche of difficulties encountered during the elaboration of our various publications.

Furthermore, I would like to express my heartfelt thanks to *Prof. Laura Pozzi* for her outstanding support, guidance and encouragement throughout our instructive and fruitful collaboration.

Without knowledge transfer and experience sharing, my work as a PhD student would have been far more difficult and complex. In the following lines, I would like to seize this opportunity to thank former ESL members, current postdocs and scientists whom I had the privilege to meet and who contributed directly or indirectly to the successful completion of my doctoral studies: *Hossein Mamaghanian, Ivan Beretta, Francisco Rincon, Srinivasan Murali, Martino Ruggiero, Marina Zapater, Leila Cammoun, Amir Aminifar, Adriana Arza, Shrikanth Ganapathy, Simone Corbetta*, and *Pieter Weckx*.

Furthermore, I would particularly like to thank my colleague and friend *Soumya Basu* for our tight collaboration and mutual support on our joint research project E4Bio.

In addition, I would like to deeply thank my officemates: *Karim Kanoun, Grégoire Surrel* and *Yasir Qureshi* for sharing so many good moments together and for their contribution to the peaceful and warm atmosphere of our cozy office, ELG 132.

A special thanks also to *Ali Pahlevan, Artem Andreev, Fabio Dell'Agnola, Dionisije Sopic* and *Elisabetta De Giovanni* for creating such a friendly and great working environment, since our

earliest days at ESL.
Furthermore, I wish all the best to the new generation of ESL members and PhD students who will sustain and contribute to the reputation of our laboratory: *Arman Iranfar, Farnaz Forooghifar, Halima Najibi, Benoit Denkinger, William Simon, Kawsar Haghshenas, Lara Orlandic, Szabolcs Balási, Wellington Silva De Souza, Damián Pascual, José Herruzo Ruiz, Tomas Teijeiro* and *Renato Zanetti*.
Behind the scenes of my research projects, I want to thank the administrative and technical staff of EPFL. In particular, I appreciated the work of the secretaries *Homeira Salimi* and *Francine Eglese* for having organized all of the social events of our laboratory and for their generosity regarding the repayment of my wonderful trips to conferences in Europe and Asia. I would also like to sincerely thank our IT technician *Rodolphe Buret* for providing fast and daily assistance with a big smile.
Outside of my professional environment, I would like to thank my entire family including my parents, brothers, aunts, uncles, and cousins, without whose support I certainly could not have done this thesis. Since the early days of my life, they have played the most important role in the accomplishment of my education. Because of the fundamental importance that they represent to me and our mutual love, I am eternally grateful to them.
Last but by no means least, I am thankful to all my friends who helped me take a breath along the steep path of my doctoral studies. Finally, I wish to very warmly thank *Aurore Delachat*, who literally held my hand during the difficult times of our PhD student lives, and with whom an intense scientific, emotional, and sentimental connection has never been interrupted.

Lausanne, 1st July 2018 Loris Duch

Abstract

NOWADAYS, the ubiquity of smart appliances in our everyday lives is increasingly strengthening the links between humans and machines. Beyond making our lives easier and more convenient, smart devices are now playing an important role in personalized healthcare delivery. This technological breakthrough is particularly relevant in a world where population aging and unhealthy habits have made non-communicable diseases the first leading cause of death worldwide according to international public health organizations. In this context, smart health monitoring systems termed **Wireless Body Sensor Nodes (WBSNs)**, represent a paradigm shift in the healthcare landscape by greatly lowering the cost of long-term monitoring of chronic diseases such as cardiovascular disorders, as well as improving patients' lifestyles. WBSNs are able to autonomously acquire biological signals with different modalities and embed on-node Digital Signal Processing (DSP) capabilities to deliver clinically-accurate health diagnoses in real-time, even outside of a hospital environment. Energy efficiency and reliability are fundamental requirements for WBSNs, since they must operate for extended periods of time, while relying on compact batteries to reduce patients' discomfort. These constraints, in turn, impose carefully designed hardware and software architectures for hosting the execution of complex biomedical applications.

In this thesis, I develop and explore novel solutions at the architectural and technological level of the integrated circuit design domain, to enhance the energy efficiency and reliability of current WBSNs.

Firstly, following a top-down approach driven by the characteristics of biomedical algorithms, I perform an architectural exploration of **a heterogeneous and reconfigurable computing platform** devoted to bio-signal analysis. By interfacing a shared Coarse-Grained Reconfigurable Array (CGRA) accelerator, this domain-specific platform can achieve higher performance and energy savings, beyond the capabilities offered by a baseline multi-processor system. More precisely, I propose three architectural versions of the CGRA, each contributing differently to the maximization of the application parallelization. The proposed *Single, Multi* and *Interleaved-Datapath* CGRA designs allow the developed platform to achieve substantial energy savings of up to 37 %, when executing complex biomedical applications, with respect to a multi-core-only platform.

Secondly, I investigate how the modeling of technology reliability issues in logic and memory components can be exploited to adequately adjust the frequency and supply voltage of a circuit, with the aim of optimizing its computing performance and energy efficiency.

To this end, I propose **a novel framework for workload-dependent Bias Temperature In-**

stability (BTI) impact analysis on biomedical application results quality. Remarkably, the framework is able to determine the range of safe circuit operating frequencies without introducing worst-case guard bands. Experiments highlight the possibility to safely raise the frequency up to 101 % above the maximum obtained with the classical static timing analysis. Then, through the study of several well-known biomedical algorithms, I propose **an approach allowing energy savings by dynamically and unequally protecting an under-powered data memory** in a new way compared to regular error protection schemes. This solution relies on the *Dynamic eRror compEnsation And Masking* (DREAM) technique that reduces by approximately 21 % the energy consumed by traditional error correction codes.
By demonstrating the ability to produce acceptable application results under extreme operating and reliability conditions, this work paves the way for the development of efficient wearable biomedical devices with reduced energy consumptions.

Keywords: Energy-efficient Biomedical Devices, Heterogeneous and Reconfigurable Bio-signal Processing Architectures, Multi-core Systems, Hardware Accelerators, Coarse-Grained Reconfigurable Arrays, Technology-level Reliability Exploration, Bias Temperature Instability, Voltage and Frequency Scaling, Error Protection Techniques.

Résumé

Aujourd'hui, l'omniprésence des appareils intelligents dans notre vie quotidienne renforce de plus en plus les liens que l'être humain entretient avec les machines. Au-delà de rendre notre vie plus facile et confortable, les appareils intelligents jouent désormais un rôle important dans la délivrance de soins de santé personnalisés. Cette avancée technologique est particulièrement nécessaire dans un monde où le vieillissement de la population et les habitudes de vie malsaines ont placé les maladies non transmissibles à la première place des causes de décès, d'après les organismes de santé internationaux. Dans ce contexte, les systèmes intelligents de surveillance médicale, plus communément appelés **Capteurs Corporels Sans-Fil (WBSN en anglais)**, représentent un changement de paradigme dans le domaine de la santé, en diminuant considérablement les coûts du suivi long-terme des maladies chroniques, telles que cardiovasculaires, ainsi qu'en apportant des améliorations notables aux conditions de vie des patients. Les WBSNs sont capables de faire l'acquisition de signaux biologiques avec différentes caractéristiques et de façon autonome. Par ailleurs, ils intègrent des capacités de traitement de signaux numériques afin de fournir en temps réel des diagnostics médicaux précis, et cela même en dehors d'un environnement hospitalier. L'efficacité énergétique et la fiabilité sont primordiales pour la conception des WBSNs, étant donné que ces derniers doivent pouvoir fonctionner sur une longue durée, tout en utilisant de petites batteries à faible encombrement, réduisant au minimum l'inconfort des patients. Ces contraintes de conception imposent à terme une élaboration soigneuse des architectures matérielles et logicielles, afin de supporter l'exécution d'applications biomédicales de plus en plus complexes.
Dans cette thèse, je développe et explore un ensemble de solutions au niveau de l'architecture et de la technologie des circuits afin d'améliorer l'efficacité énergétique et la fiabilité des WBSNs actuels.
Dans un premier temps, en suivant une approche descendante reposant sur les caractéristiques d'algorithmes biomédicaux, j'effectue l'exploration architecturale d'**une plateforme de traitement hétérogène et reconfigurable**, adaptée à l'analyse de signaux biologiques. En interfaçant un accélérateur matériel reconfigurable à gros grains (CGRA en anglais), cette plateforme peut atteindre de plus hautes performances de calculs et des économies d'énergie, au-delà de celles atteignables par un système multi-processeurs standard. Plus précisément, je propose trois versions différentes de l'architecture du CGRA, dont chacune contribue différemment à la maximisation du traitement parallèle de l'application. Les architectures de CGRA proposées sont dotées d'un chemin de données *Simple*, *Multiple* ou *Entrelacé*, permettant à la plateforme développée d'atteindre des économies d'énergie significatives jusqu'à 37 %, lors de

l'exécution d'applications biomédicales complexes, par rapport à une plateforme multi-cœurs traditionnelle.
Dans un deuxième temps, j'étudie comment la modélisation de problèmes de fiabilité au niveau des composants logiques et mémoires peut être exploitée, afin d'ajuster adéquatement la fréquence et la tension d'alimentation du circuit, dans le but d'optimiser ses performances et son efficacité énergétique. A cette fin, je propose **un nouveau flot de conception permettant d'analyser les effets de l'instabilité de la polarisation avec la température (BTI en anglais)** sur la qualité des résultats produits par les applications biomédicales. Ce flot de conception est capable de déterminer l'intervalle de fréquences optimisées et sûres pour le circuit, tout en évitant l'introduction de marges de sécurité de type « pire-cas ». Les expériences mettent en évidence la possibilité d'augmenter sans risque la fréquence jusqu'à 101 % au-dessus de la fréquence maximale obtenue habituellement avec la méthode d'analyse temporelle statique du circuit. Par la suite, à travers l'étude de plusieurs algorithmes biomédicaux, je propose **une nouvelle approche permettant des économies d'énergie en protégeant dynamiquement et non-uniformément une mémoire sous-alimentée**, comparativement aux stratégies de protection mémoire traditionnellement utilisées. Cette approche repose sur la *technique de compensation et de masquage d'erreurs dynamiques* (DREAM en anglais), qui réduit d'environ 21 % l'énergie totale habituellement consommée par des codes de correction d'erreurs.
En démontrant la possibilité de produire des résultats cliniquement acceptables, sous des conditions extrêmes de fonctionnement et de fiabilité, ce travail de thèse a contribué au développement de dispositifs médicaux portables et performants, dotés d'une consommation énergétique réduite.

Mots-clés : Appareils Biomédicaux Energétiquement Efficaces, Architectures Hétérogène et Reconfigurable pour le Traitement de Signaux Biologiques, Systèmes Multi-cœurs, Accélérateurs Matériels, Accélérateurs Reconfigurable à Gros Grains (CGRA), Exploration Technologique de la Fiabilité, Instabilité de la Polarisation avec la Température (BTI), Mise à l'Echelle de Tension et de Fréquence, Techniques de Protection contre les Erreurs.

Table of Contents

Acknowledgements i

Abstract iii

Résumé (Français) v

Table of Contents vii

List of Figures xi

List of Tables xv

List of Abbreviations xvii

1 Introduction 1
- 1.1 Fundamentals of Embedded Biomedical Devices . . . 3
- 1.2 Energy-Efficient Bio-Signal Processing Architectures Design . . . 5
  - 1.2.1 Optimization Techniques for Energy-Efficient Bio-Signal Processing Platforms . . . 5
  - 1.2.2 Challenges in the Design of Energy-Efficient Bio-Signal Processing Platforms . . . 7
    - 1.2.2.1 Algorithmic and Architectural Challenges . . . 8
    - 1.2.2.2 Technological Challenges . . . 8
- 1.3 Thesis Contributions . . . 10
  - 1.3.1 Heterogeneous and Reconfigurable Energy-Efficient Bio-Signal Processing Architectures . . . 11
  - 1.3.2 Technology-Level Reliability Exploration in Energy-Efficient Biomedical Systems . . . 12
    - 1.3.2.1 Logic-Level Reliability Study . . . 12
    - 1.3.2.2 Memory-Level Reliability Study . . . 13
- 1.4 Thesis Outline . . . 15

2 Heterogeneous and Reconfigurable Energy-Efficient Bio-signal Processing Architectures 17
- 2.1 Introduction and Motivations . . . 17

2.1.1 Parallel Processing and Computational Hotspots in Biomedical Applications 18
2.1.2 Contributions and Outline of the Chapter 20
2.2 State-of-the-Art and Selection of the Accelerator Architecture 23
2.3 Reconfigurable and Accelerated Multi-Core System 26
2.3.1 Multi-Core System 27
2.3.1.1 Domain-Specific Processors 28
2.3.1.2 Instruction and Data Memory Architecture 29
2.3.1.3 Hardware/Software Synchronization Mechanism 30
2.3.1.4 Acceleration Request Controller and Configuration Memory . . 32
2.3.1.5 Instruction Set Extensions for Synchronization and Hardware Acceleration Support 34
2.3.1.6 Acceleration Request Execution Flow 34
2.3.2 Domain-Specific Shared CGRA 37
2.3.2.1 CGRA Hardware Architecture 37
2.3.2.2 CGRA Execution Flow 40
2.3.2.3 Kernel Identification, Selection and Scheduling 40
2.4 Single-Datapath CGRA Accelerator Shared by a Multi-Core System 42
2.4.1 Overview 42
2.4.2 Experimental Evaluation 42
2.4.2.1 Experimental Setup 42
2.4.2.2 Experimental Results 48
2.5 Multi-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System 52
2.5.1 Overview 52
2.5.2 Experimental Evaluation 53
2.5.2.1 Experimental Setup 53
2.5.2.2 Experimental Results 55
2.6 Interleaved-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System 65
2.6.1 Overview 65
2.6.2 Experimental Evaluation 67
2.6.2.1 Experimental Setup 68
2.6.2.2 Experimental Results 69
2.7 Summary and Concluding Remarks 75

3 Technology-Level Reliability Exploration in Energy-Efficient Biomedical Systems 77
3.1 Introduction and Motivations 77
3.1.1 From Silicon-Level Reliability Issues to Application-Level Degradations 78
3.1.2 Contributions and Outline of the Chapter 80
3.2 BTI-Aware Logic Circuit Design 83
3.2.1 BTI Effects: A CMOS Reliability Issue 83
3.2.2 Proposed Framework for Workload-Dependent BTI Impact Analysis . . . 85
3.2.2.1 Background and Related Work 86
3.2.2.2 Description of the BTI Evaluation Framework 90

3.2.2.3 Experimental Evaluation 98
3.2.3 Insights on BTI-Induced Functional Error Detection and Mitigation . . . 118
3.2.3.1 State-of-the-Art 118
3.2.3.2 Hardware-Level Technique for BTI-Induced Functional Error Detection 122
3.2.3.3 Software-Level Technique for BTI-Induced Functional Error Detection 124
3.2.3.4 Hardware and Software Techniques for BTI-Induced Functional Error Mitigation 125
3.3 Reliability Analysis and Significance-Based Data Protection in Memories 126
3.3.1 Reliability Concerns and Challenges in Memories 126
3.3.2 Related Works 127
3.3.3 Analysis of the Inherent Resilience of Biomedical Applications 129
3.3.3.1 Representative Set of Biomedical Applications 130
3.3.3.2 Data Significance Analysis and Characterization of Biomedical Applications 131
3.3.4 Proposed Significance-Based Error Mitigation Technique for Memory Components 134
3.3.4.1 Presentation of the DREAM Memory Protection Technique . . . 134
3.3.4.2 Experimental Evaluation 138
3.4 Summary and Concluding Remarks 145

4 Conclusions and Future Work **147**
4.1 Summary and Contributions 147
4.2 Future Work 151

A Appendix: BTI Degradations Evaluation **155**

Bibliography **178**

Curriculum Vitae **179**

List of Figures

1.1 Proportion of global deaths under the age of 70 by cause of mortality 1
1.2 Deaths attributed to 19 leading risk factors, by country income level 2
1.3 Health monitoring scenario based on WBSNs 3
1.4 Average energy consumption breakdown of *Traditional* versus *Smart* WBSNs with a reasonable application and system architecture 4
1.5 The three main stages of typical Smart WBSNs 5
1.6 Structure of a typical bio-signal processing application 6
1.7 Research scenario: Energy-efficient WBSN platform, embedding a CGRA accelerator shared by multiple computing cores, and running under fine-tuned operating conditions to meet the performance, energy, and reliability constraints 14
1.8 Thesis structure 15

2.2 Examples of digital signal processing applications for ECG signals: a) DWT decomposition and b) compressed sensing 19
2.3 Bio-signal processing architecture coupling a multi-core system with a CGRA unit, supporting SIMD execution in both resources 21
2.4 The three design corners of five hardware accelerator and processor architectures 23
2.5 Design framework and challenges 25
2.6 Illustrative example of biomedical application workload decomposition and parallelization 26
2.7 High-level view of the accelerated multi-core platform 27
2.8 TamaRISC three-stage pipeline architecture 28
2.9 Synchronization for processors (μPs) in lock-step execution and in a producer/-consumer relationship 30
2.10 High level management of SIMD-kernels execution 32
2.11 Acceleration request execution flow diagram 35
2.12 Block scheme of the SIMD-CGRA 38
2.13 Architecture of the SIMD-CGRA RCs 39
2.14 Block scheme of the experimental framework, comprising RTL and cycle-accurate system views 43
2.15 Task graph of the 3L-MF and 3L-MMD benchmarks 46
2.16 ECG fiducial points of a normal heartbeat 46
2.17 Task graph of the RP-CLASS benchmark 47

2.18 Speed-ups of kernels running on the single-DP CGRA mesh with respect to their software execution . . . 49
2.19 Energy consumed by the different kernels employed in the considered benchmarks, when executed on the accelerator (CGRA 1 DP) and on the processing cores (SW) . . . 49
2.20 Multi-core utilization time with and without CGRA acceleration . . . 50
2.21 System energy consumption for the different applications, while executing on the multi-core platform with and without CGRA acceleration . . . 51
2.22 Task graph of the 6L-MF and 6L-MMD benchmarks . . . 54
2.23 Task graph of the 8L-CS benchmark . . . 54
2.24 Average speed-up with kernels running on the multi-core + CGRA platform w.r.t. kernels running only on the multi-core platform . . . 56
2.25 Repartition of kernel acceleration requests types, depending on employed data-path configuration . . . 57
2.26 Average waiting time spent for each acceleration request . . . 58
2.27 Energy consumption of the different kernels . . . 60
2.28 System energy consumption for processing the different benchmark applications 62
2.29 Area breakdown of the accelerated multi-core platform, embedding a CGRA with different SIMD widths . . . 63
2.30 Multi-core and CGRA utilization time in the considered platforms . . . 64
2.31 i-DP CGRA block scheme, interfaced with a multi-processor system . . . 66
2.32 Example of a simple kernel mapped in a single-DP and an i-DP CGRA . . . 67
2.33 Run-time of *Dbl Min Srch* kernel, varying the input data size . . . 69
2.34 Average kernels run-time executing on CGRAs with 1 DP and 2 i-DPs . . . 70
2.35 Multi-core and CGRA utilization time in the considered platforms . . . 71
2.36 Average sample processing time with the different benchmarks and CGRA architectures . . . 71
2.37 Energy consumption of the different kernels . . . 73
2.38 System energy consumption of the different benchmark applications . . . 73
2.39 Area breakdown of the i-DPs CGRA compared to the baseline single-DP CGRA . 74

3.1 Time-dependent BTI evolution . . . 84
3.2 High-level design flow, including the proposed BTI evaluation framework oriented to the analysis of processor pipelines . . . 85
3.3 Non-uniform CDW point decomposition . . . 90
3.4 Structural overview of the long-term BTI model . . . 91
3.5 ΔV_{th} evolution illustrating the initial ramp-up phase of BTI-induced degradations from an NMOS transistor . . . 92
3.6 Signal conversion hysteresis from the analog to digital domain . . . 94
3.7 Illustrative example of the transistion thresholds determination . . . 94
3.8 Overview of the proposed workload-aware BTI evaluation framework . . . 95
3.9 Time-dependent BTI degradation with deeply-scaled transistor technologies . 97

3.10 Slack time of the three TamaRISC pipeline stages at a frequency of 550 MHz . . 99
3.11 Simplified schematic of the ALU16 circuit 100
3.12 Task graph of the 3L-MMD benchmark 101
3.13 Example of delineated heartbeat with ECG fiducial points 102
3.14 Cumulative framework run-time averaged over all the experiments 105
3.15 Fitting extended to 1 year (*DC offset*), with final *AC* stress activity corresponding to one application workload of 2 seconds 106
3.16 Minimum slack time for different operating frequencies and benchmark circuits, measured after 2 seconds of operation without and with accumulated BTI aging of 0 seconds 108
3.17 Minimum slack time at different working frequencies for the ALU16 (32 bits output multiplier) 110
3.18 Heatmaps: BTI degradation of the ALU16 output signals (*Carry_Out*, *Result_Out*) with the operating frequency, measured after 2 seconds of circuit operation without and with accumulated BTI aging of 0 seconds, 1 year and 10 years . . . 111
3.19 3D Surfaces: BTI degradation of the ALU16 output signals (*Carry_Out*, *Result_Out*) with the operating frequency, measured after 2 seconds of circuit operation without and with accumulated BTI aging of 0 seconds, 1 year and 10 years 112
3.20 Evolution of the percentage of faulty operations for the ALU16 (32 bits output multiplier), at time = 0 seconds, 1 year and 10 years. 113
3.21 Example of signals impacted by BTI and overclocking functional errors relative to an error-free ECG after 10 years 115
3.22 Quality assessment of the output results generated by the MMD delineation application, impacted by BTI functional errors after 10 years 117
3.23 Devices failure rate evolution with mature versus emerging silicon technologies 118
3.24 ALU16 circuit with concurrent error detection scheme using Dong's code 122
3.25 Illustrative example of the accumulation of pseudo-permanent errors when the memory supply voltage is reduced 127
3.26 Block scheme of a typical WBSN 129
3.27 Data bits utilization percentages evolution 132
3.28 SNR vs. data bit positions of injected errors 133
3.29 DREAM technique - Simplified representation of the protection mask encoding 135
3.30 DREAM technique operating modes 136
3.31 Example of memory mapping with DREAM support for a 2 KiB data bank . . . 137
3.32 High-level view of the VirtualSoc platform 138
3.33 Memory cell failure probability and number of faulty memory cells vs. supply voltage 139
3.34 SNR evolution vs. supply voltage for each application 141
3.35 Memory energy consumption vs. supply voltage, for different error mitigation techniques and biomedical applications 142

List of Tables

2.1 Kernel utilization per benchmark application (3L-MF, 3L-MMD and RP-CLASS) 48
2.2 Kernel utilization per benchmark application (6L-MF, 6L-MMD, RP-CLASS and 8L-CS) 55
2.3 CGRA area exploration (in 65 nm UMC technology library) 61

3.1 Benchmark circuits from a 16-bit pipeline execution stage 100
3.2 Maximum operating frequency as determined with STA, for each benchmark circuit 101
3.3 Workload activity summary for each benchmark circuit for the considered 3L-MMD biomedical application 105
3.4 Maximum transistor V_{th} degradation measured after 2 seconds of circuit operation with accumulated BTI aging of 0 seconds, 1 year and 10 years, for the ALU16 107
3.5 DREAM error correction example for 16-bit data 135

A.1 Operation code ($Op\ Code$) from the 16 bits pipeline execution stage ($ALU16$) . 156
A.2 Quality assessment of the output results generated by the MMD ECG delineation application under BTI degradations (after 10 years) for different circuit operating frequencies 159

List of Abbreviations

ADC	Analog-to-Digital Converter
ALU	Arithmetic and Logic Unit
ASIC	Application-Specific Integrated Circuit
BER	Bit Error Rate
BTI	Bias Temperature Instability
CDW	Compact Digital Waveform
CGRA	Coarse-Grained Reconfigurable Array
CMOS	Complementary Metal-Oxide-Semiconductor
CPLD	Complex Programmable Logic Device
CR	Configuration Register
CS	Compressed Sensing
CVD	Cardiovascular Disease
DM	Data Memory
DMA	Direct Memory Access
DP	Datapath
DREAM	Dynamic eRror compEnsation And Masking technique
DSP	Digital Signal Processing
DTA	Dynamic Timing Analysis
DWT	Discrete Wavelet Transform
ECC	Error Correction Code
ECG	Electrocardiogram

EEG	Electroencephalogram
EMG	Electromyograms
EMT	Error Mitigation Technique
FA	Full Adder
FFT	Fast Fourier Transform
FIFO	First-In First-Out memory buffer
FPGA	Field-Programmable Gate Array
GPU	Graphics Processing Unit
HCI	Hot Carrier Injection
HDL	Hardware Description Language
HW	Hardware
IM	Instruction Memory
IoT	Internet-of-Things
ISA	Instruction Set Architecture
LSB	Least Significant Bit
MAC	Multiply-Accumulate operation
MF	Morphological Filtering
MIMD	Multiple-Instruction Multiple-Data
MMD	Multi-scale Morphological Delineation
MSB	Most Significant Bit
MSE	Mean Squared Error
MUX	Multiplexer
NCD	Non-Communicable Disease
NMOS	N-type Metal-Oxide-Semiconductor
PC	Program Counter
PMOS	P-type Metal-Oxide-Semiconductor
PPG	Photoplethysmogram

PRD	Percentage Root-mean-square Difference
RAM	Random-Access Memory
RC	Reconfigurable Cell
RF	Register File
RISC	Reduced Instruction Set Computer
RMS	Root-Mean-Square
ROM	Read-Only Memory
RP	Random Projection
RTL	Register Transfer Level
RTN	Random Telegraph Noise
SEC-DED	Single Error Correction / Double Error Detection
SEU	Single Event Upset
SIMD	Single-Instruction Multiple-Data
SNR	Signal-to-Noise Ratio
SRAM	Static Random-Access Memory
STA	Static Timing Analysis
SW	Software
TDDB	Time-Dependent Dielectric Breakdown
VCD	Value Change Dump file
VEC	Digital Vector File
Verilog-A	Verilog-Analog hardware description language
VHDL	Very high speed integrated circuit Hardware Description Language
WBSN	Wireless Body Sensor Node
X-Bar	Crossbar Interconnect

1 Introduction

The pervasiveness of smart devices in our daily lives has generated a growing demand for highly efficient systems incorporating a wide range of autonomous functionalities. Smart devices can be found in a multitude of areas, such as communication, entertainment, security, domotics, mobility, sports, as well as personalized medicine. The continuous development of these electronic devices will enable major innovations in the interactions between humans and machines. Among those breakthroughs, it will have an important impact on real-time healthcare provisions [1, 2].

Nowadays, this scenario is becoming more and more relevant, since the aging of the world population and the prevalence of unhealthy habits have positioned Non-Communicable Diseases (NCDs) in the first place of the leading causes of death worldwide, even ahead of infectious agents [5]. As illustrated in Figure 1.1, according to a study made by the World Health Organization, NCDs such as diabetes, respiratory diseases, cancers or cardiovascular diseases, represent more than half of the major causes of mortality worldwide for people

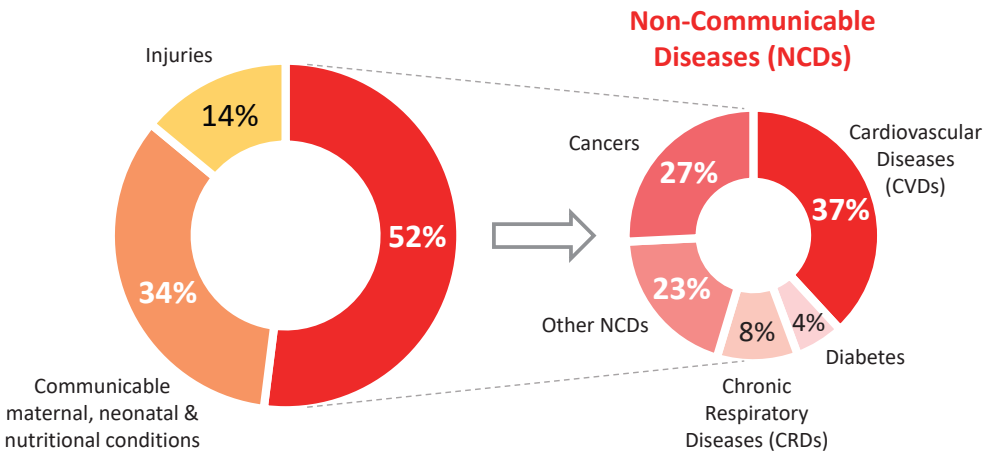


Figure 1.1 – Proportion of global deaths under the age of 70 by cause of mortality (Data extracted from [3]).

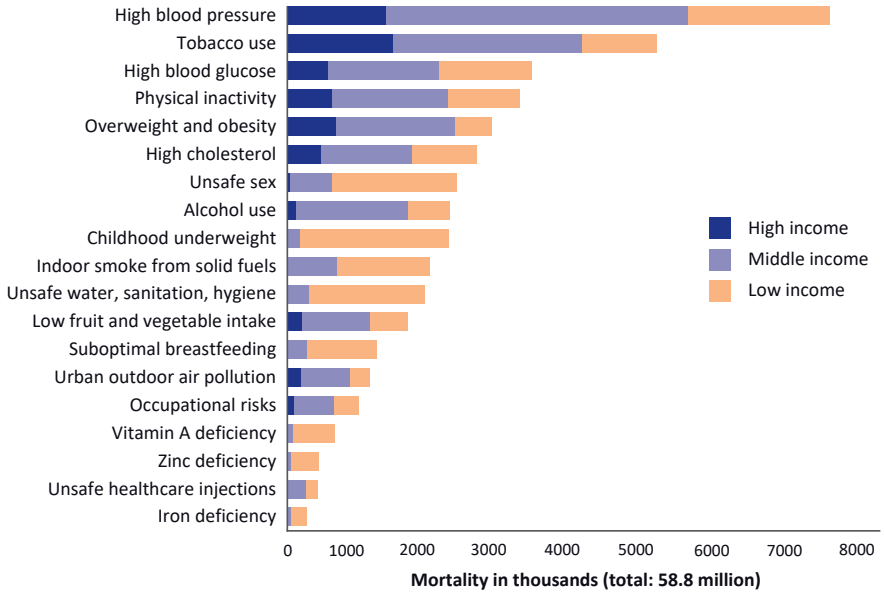


Figure 1.2 – Deaths attributed to 19 leading risk factors, by country income level (Chart extracted from [4]).

under the age of 70 [3]. Among these health issues, 37 % are related to Cardiovascular Diseases (CVDs), which represents more than one third of the total leading causes of death at a global scale. CVDs such as ischemic strokes and heart attacks are principally linked to the same risk factors, including high blood pressure, smoking, high cholesterol, and obesity. As shown in Figure 1.2, these risk factors are among the Top 6 leading to death every year, according to the World Health Organization [4]. Furthermore, heart-related diseases increase with age and affect both women and men, regardless of ethnicity and income levels.

Today, these diseases are treated on-demand in clinical environments, after being diagnosed by physicians. They require a continuous and long-term supervision of the affected patients, involving expensive medical resources, such as nurses, medical appliances, and hospital rooms, to name a few. As an example, in 2015 the burden of cardiovascular diseases was estimated to cost the European Union economy almost 210 billion euros per year, which corresponds to a cost of 218 euros per capita per annum [6]. These figures highlight the need for alternative solutions to deliver more affordable medical care and provide early recognition of the signs and symptoms of chronic diseases.

To address this challenge, wearable health monitoring devices, also known as Wireless Body Sensor Nodes (WBSNs), have emerged as a cost-effective unobtrusive solution to perform continuous and autonomous monitoring of clinically-relevant data, outside of a hospital environment and with little supervision from the medical staff [7]. WBSNs are able to autonomously acquire bio-signals with various modalities, such as blood oxygen saturation (SpO_2) with Photoplethysmograms (PPGs), muscle activity with Electromyograms (EMGs), brain activity with Electroencephalograms (EEGs) and so forth [8, 9, 10]. Among them, the

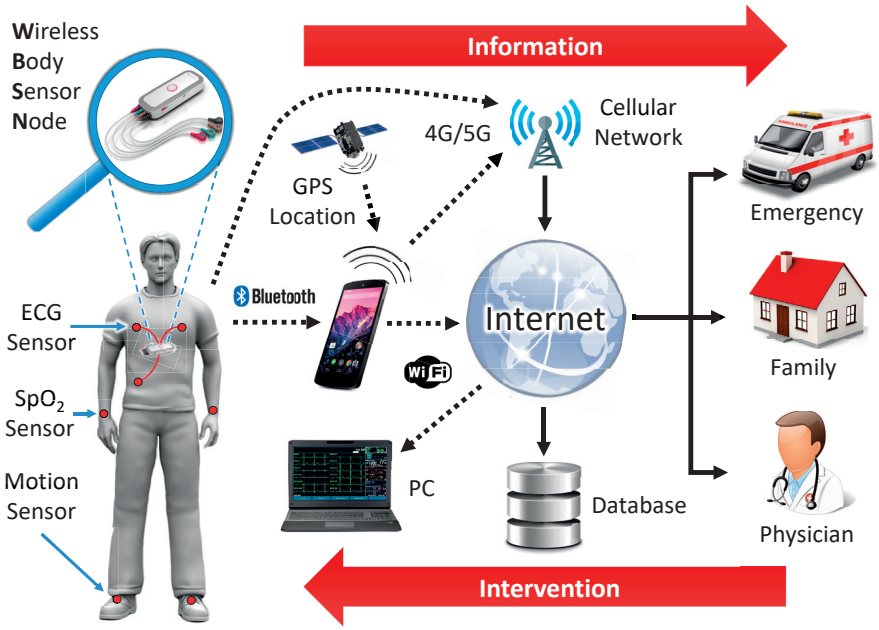


Figure 1.3 – Health monitoring scenario based on WBSNs.

most common one, on which I focus in this thesis, is the Electrocardiogram (ECG) [11], which measures the electrical activity of the heart. Additionally, cardiac data can be complemented with behavioral data from accelerometer acquisitions to discern body posture, movements or the activity level of the subject. Finally, other types of sensors can be integrated in WBSNs to measure, for instance, the electrical conductivity of the subject's skin, which assesses the amount of sweat-induced moisture, but also environmental parameters (e.g., air temperature, luminosity, noise, atmospheric pressure) which may have an effect on the current mood, stress or fatigue level of the subject.

In order to provide more insights on the context of this work and on the state-of-the-art, the remaining of this chapter is structured as follows. First, I present the main design requirements of WBSNs. Second, I introduce the existing techniques to optimize the energy efficiency of WBSNs, and more specifically, the on-board digital signal processing stage of these devices. Then, I detail the main contributions of this thesis, and finally, I conclude this chapter by providing a brief outline of this document.

1.1 Fundamentals of Embedded Biomedical Devices

The working principle of WBSNs is depicted in Figure 1.3. In the illustrated scenario, the WBSN device monitors the heart activity to detect possible abnormalities (e.g., arrhythmias, atrial fibrillation, bradycardia, tachycardia) or any warning signs of heart failure [12]. The acquired data is sent wirelessly to a smartphone or a remote computer for further processing. When a heart abnormality is detected, the information/alert can be transmitted through

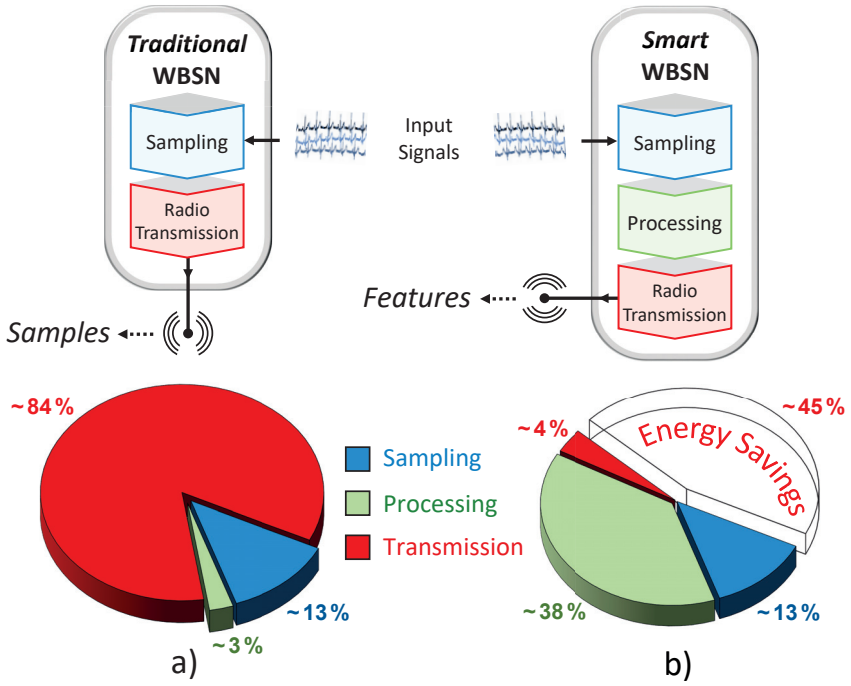


Figure 1.4 – Average energy consumption breakdown of *Traditional* versus *Smart* WBSNs with a reasonable application and system architecture. Data computed from [13] (Sampling), [14] (Processing) and [15] (Transmission).

Internet to the subject's family or to the medical staff, for requesting an emergency medical treatment. The log of abnormal events can also be stored locally into the device or in a remote database/cloud for further consultations by the physicians.

To be effective and not miss any of those critical random events, a key requirement of WBSNs concerns their autonomy. They must operate over extended periods of time, while relying on small batteries to reduce the weight of the device and subject discomfort. Thus, the design of WBSNs requires high energy efficiency, to also avoid frequent battery charging and disconnection of the electrodes from the body of the subject during recharge cycles.

In this context, the energy bottleneck of most WBSNs, which only perform acquisition and transmission of ECG samples (e.g., Holter monitors [16]), usually resides in the wireless communication stage, as shown in Figure 1.4.a. In fact, the slow dynamics of bio-signals allow their acquisition while employing little energy when performing their Analog-to-Digital Conversion (ADC) with reduced sampling frequencies (i.e., few hundreds of Hertz) [15]. To cut down the energy consumed by the radio transmission module, a striking alternative is proposed by *Smart WBSNs*, which perform advanced on-board Digital Signal Processing (DSP) to extract high-level relevant signal characteristics or *features* from acquisitions [17]. In this approach, only features (as opposed to samples) are transmitted through the energy-hungry wireless link, potentially resulting in large energy savings [18, 19]. Nevertheless, these benefits can only be leveraged by performing the DSP stage itself under tight energy constraints. In

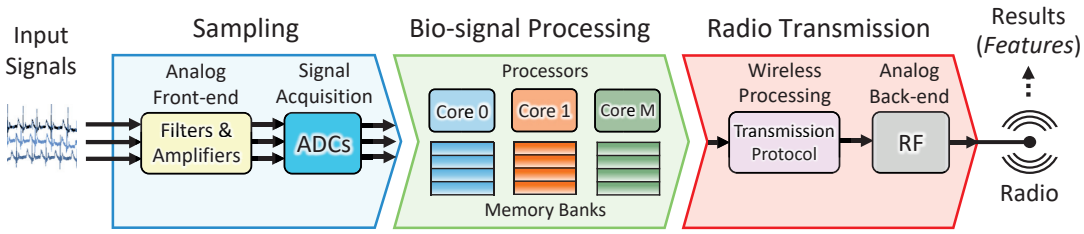


Figure 1.5 – The three main stages of typical Smart WBSNs (either integrated into a single chip or as separate components on printed circuit boards).

fact, thanks to progresses in the design of domain-specific analog-to-digital converters [20, 21] and wireless protocols [13], DSP tends to dominate the energy budget of Smart WBSNs [22], so that any increase in its efficiency has a tangible impact at the system level (see Figure 1.4.b).

Optimizing the DSP stage at the heart of Smart WBSNs (see Figure 1.5) represents a crucial challenge for embedded systems designers. Indeed, this processing stage requires a carefully tailored computing architecture for hosting the execution of more and more advanced biomedical applications, with a minimal energy budget. This in itself constitutes the main focus and technical challenge that I tackle in this thesis.

1.2 Energy-Efficient Bio-Signal Processing Architectures Design

As shown in the previous section, achieving optimizations at the DSP level is of paramount interest for the design of miniaturized and battery-powered WBSNs, since they must operate from several days to several weeks on a single charge [23]. The following sections provides an overview of the existing techniques and design opportunities to optimize these platforms at the hardware and software levels, as well as the resulting challenges arising from these optimizations.

1.2.1 Optimization Techniques for Energy-Efficient Bio-Signal Processing Platforms

Today, there is an increasing number of applications where energy saving and efficiency are at the top of embedded systems designers' priorities. For this reason, the scientific literature exposes a broad range of solutions to minimize the energy consumption of embedded systems such as WBSNs. I present hereafter a list of the main approaches which are also applicable in this thesis to complement the energy-efficient strategies proposed in the following chapters.

First, a common approach used in the ultra-low power domain is to reduce the bit width of the data manipulated by the processor [24, 25, 26], compared to standard 32 or 64 bits datapath architectures traditionally employed in mobile appliances [27, 28]. In fact, DSP applications running on wearable sensors typically process samples with a small range of variations. For

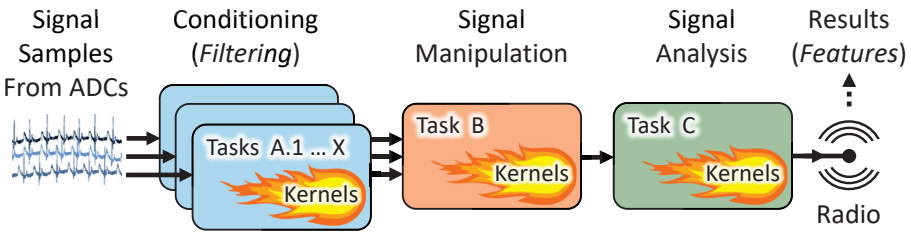


Figure 1.6 – Structure of a typical bio-signal processing application.

instance, the dynamic of an ECG signal is commonly represented on 16 bits integer data, which is a sufficient resolution to encode the information contained in the signal [29].

Second, following the same rationale, another approach consists of pruning/simplifying the hardware architecture by removing the extra operations or components that are seldom used, replaceable or superfluous when processing the DSP application [24, 30]. As an example, Floating-Point Units (FPUs) are usually not integrated into the execution stage of ultra-low power processor pipelines in order to reduce their energy consumption and silicon area [31, 32, 33]. Instead, several integer operations can be executed consecutively to perform the fixed- and floating-point arithmetic operations. This pruning approach is also frequently employed at the software level, to simplify and speed-up the execution of DSP applications [34]. For instance, in the context of ECG applications, the number of processed leads (i.e., channels) may be reduced from 12 to 6, 3, 2 or 1 lead to minimize the energy consumption of DSP platforms [35].

Third, similarly to the feature extraction approach (see Section 1.1), to diminish the energy spent during the transmission of the acquired/processed data, lightweight and lossy compression algorithms (e.g., Compressed Sensing) can be applied in real-time to reduce the bandwidth usage, together with the quantity of data transmitted through the wireless network [18, 36, 37].

Fourth, as illustrated in Figure 1.6, typical bio-signal processing applications can be represented as well-separated tasks executed sequentially or in parallel [38]. This structure provides the opportunity to be naturally mapped on low-power multi-core platforms, by distributing the workload over different computing processors [14, 39]. Moreover, a vast portion of the overall DSP run-time of each task is required to execute short and intensive computational hotspots (i.e., kernels), usually in the form of compact loops, which are data (as opposed to control) dominated. This application characteristic highlights the opportunity to achieve a better computing performance and energy savings, by employing dedicated hardware accelerators, such as Application-Specific Integrated Circuits (ASICs) [26], or Coarse-Grained Reconfigurable Array (CGRAs) [40], to efficiently support the computationally-intensive kernels from each task.

By leveraging the reduction of the application run-time, another effective technique consists of diminishing the operating parameters of the logic and memory components by means

of the Voltage and Frequency Scaling (VFS) technique. The decrease of the supply voltage and frequency translates into quadratic and linear energy savings respectively [41, 42, 43]. Additionally, when the processors are idle (i.e., when waiting for another core to finish its task or when waiting for another sample to process from the ADCs), power-gating and clock-gating techniques can be massively employed to decrease considerably the static and/or dynamic energy requirements of the DSP platform [14].

Then, because of task parallelism, the occurrence of concurrent accesses to the same memory address is frequent in bio-signal applications. In fact, the same signal processing routine is often performed on different input streams, and executed in parallel on several processors. Based on this observation, an alternative architectural strategy proposes to reduce the dynamic energy consumption of a multi-core platform, by reducing the number of read operations to the instruction and data memories. To do so, in [44], the authors describe a lightweight hardware/software synchronization mechanism, which provides the necessary support to enforce the lock-step (i.e., synchronized) execution of the processors running in parallel. In addition, specific crossbar (X-Bars) interconnects between the memories and the processors are used to the merge accesses from several cores to the same memory address, and subsequently, broadcast the retrieved data to all the requesting cores. By merging several memory access requests into a single one, substantial energy savings are achieved, especially in the case of large multi-core systems.

Finally, at the technology level, static and dynamic energy consumptions can be attenuated thanks to advanced silicon/wafer technologies (e.g., Fully Depleted Silicon On Insulator (FDSOI) substrates [45]) and transistor structures (e.g., Fin (FinFET), Nanowire (NWFET) or Carbon Nanotube (CNTFET) Field Effect Transistors [46, 47]). Moreover, energy and performance improvements can also be achieved by carefully selecting the appropriate logic and memory cell libraries matching with the requirements of the targeted system in terms of speed, energy, and area. For instance, in logic components, the static energy consumption is reduced by selecting large or medium technology nodes (e.g., from 90 nm to 28 nm) with a lower leakage current compared to smaller ones. Also, by choosing the proper threshold voltage option (i.e., High (HVT), Regular (RVT), Low (LVT) or Super Low (SLVT) Voltage Threshold), trade-offs between performance and static energy consumption can be achieved [48]. In the case of ultra-low power design, a high threshold voltage (HVT) is recommended to minimize static energy consumption at the cost of higher switching delays (i.e., lower maximum operating frequencies achievable).

1.2.2 Challenges in the Design of Energy-Efficient Bio-Signal Processing Platforms

All the strategies presented in Section 1.2.1 to optimize the energy consumption of bio-signal processing platforms raise several design and research challenges at the application, architecture, and technology levels. These challenges are highlighted in the following sections.

1.2.2.1 Algorithmic and Architectural Challenges

The design constraints imposed in the elaboration of wearable and energy-efficient health monitoring devices challenge the work of embedded systems designers.

Firstly, at the software level, when a biomedical application is ported towards these resource-contained platforms, a significant effort is required to adapt and optimize the program for the device's architecture [49]. For instance, when a DSP application has been initially developed with 32 and 64 bits variables, moving to a 16 bits processor requires substantial modifications of the algorithm, accompanied with a possible loss of accuracy of the application's results. This issue may also arise when the baseline arithmetic is changed or pruned to produce a lightweight version of the original application. As an example, the results accuracy may be degraded when moving from floating-point operations to integer operations, or when applying the hardware/software pruning techniques mentioned in Section 1.2.1.

Secondly, the biomedical application's memory footprint must be reduced in order to comply with the memory limitations/sizes of bio-signal processing platforms [50]. This can be achieved by resizing the memory buffers/arrays and by re-utilizing the memory space for several operations in a row, thanks to in-place processing techniques.

Thirdly, in the context of multi-core architectures, the decomposition of a single application task (i.e., thread) into several ones executed in parallel is not trivial. In particular, it requires the integration of different synchronization and communication mechanisms between the different processing units (e.g., synchronization barriers, shared memories, hardware/software First-In First-Out memory buffers (FIFOs), mailboxes) [14]. Furthermore, several profiling passes must be performed at the software level to identify and optimize the execution of the computationally-intensive kernels from each application task. Additionally, when hardware accelerators are interfaced with the single- or multi-core system to efficiently execute these kernels, extra hardware/software mechanisms must be introduced to configure and transmit the acceleration requests to the dedicated computing resources [51].

As a consequence, substantial energy and area overheads result from the different hardware mechanisms employed to increase the energy efficiency of bio-signal processing platforms. Trade-offs between the provided benefits and overheads of these techniques must be evaluated, which constitute serious research challenges when designing domain-specific DSP platforms [52].

1.2.2.2 Technological Challenges

As shown in the previous section, some of the modifications performed at the algorithmic level to save energy (e.g., pruning and limited bit widths data representation) compromise the accuracy of the results delivered by the biomedical application. Therefore, embedded systems designers are forced to re-evaluate and quantify the level of degradation in output of the application, compared to the original output produced by the non-optimized algorithm [19, 34, 35, 49].

In a similar way, when technology-level optimizations are performed, embedded systems designers may also encounter accuracy degradations, with possibly an impact on the performance and reliability of the logic and memory components.

As an example, when aggressive VFS is employed, the computing platforms' performance can be degraded beyond the real-time requirements of the applications. Performance degradations are even more dramatic in the context of emerging biomedical applications with heavy workloads, supporting for instance machine learning algorithms or the processing of many bio-signals in parallel. In addition, as the supply voltage approaches the threshold voltage of the transistors and as the circuit ages, functional errors induced by *time-zero* and *time-dependent* variabilities start to appear. More precisely, they can be the consequence of process and temperature variations [53], a linear increase of the memory cells sensitivity to radiation (e.g., particle strikes from outer space/cosmic rays)[54, 55], an increased sensitivity to voltage droops [56] or longer propagation delays/timing violations [57], when operating at near-threshold voltage or beyond the safe frequency limits of the circuit.

In this regard, measures must be taken to properly adjust the operating parameters (i.e., frequency, voltage) of the logic and memory components which compose the system. For instance, a minimal voltage level must be conserved for data retention in SRAM memories [52]. Additionally, error detection and correction mechanisms can be integrated at design-time, to counteract the reliability concerns occurring at run-time [58]. However, similarly to the architectural modifications employed to optimize the energy efficiency of the platform (see Section 1.2.2.1), significant energy and area overheads are resulting from the integration of error detection and correction mechanisms in the platforms [52, 59].

To address these challenges without costly hardware mechanisms, recent computation paradigms have been introduced, such as *approximate computing* and its extension *significance-based computing*. These paradigms exploit the fault tolerance inherent in many bio-signal processing applications [60]. In fact, even when recorded in a controlled environment, raw bio-signals contain various types of noise [61]. As a consequence, most bio-signal processing algorithms embed noise filtering capabilities, reinforcing their robustness against possible sample or data corruptions. Moreover, the biomedical applications processing these signals usually produce statistical or qualitative results [62]. The quality assessment of these results may depend on human perception, and thus does not rely on strict or precise quality criteria. Therefore, the aforementioned paradigms leverage the opportunity of saving energy by relaxing the traditional reliability requirements for 100 % computational precision. As a result, these paradigms are applicable to specific biomedical applications producing correct or acceptable results even if the DSP platform reliability (on which these applications are running) is compromised to a certain degree [63, 64, 65].

Finally, even though the DSP platform is operating in a safe region with the appropriate parameters (i.e., frequency, voltage, and temperature) to avoid reliability concerns, after a prolonged period of time the system will suffer from various aging effects, impacting the

reliability of its components. On one hand, at the transistor level, Bias Temperature Instability (BTI) effects are the most important ones and are complemented by relevant but less dominant aging-related reliability issues, such as Random Telegraph Noise (RTN), Hot Carrier Injection (HCI), and high-κ Time-Dependent Dielectric Breakdown (TDDB) [66]. On the other hand, at the interconnect level, Electromigration (EM) is the most critical wear-out effect occurring in metal layers (e.g., wires supporting unidirectional currents, power grids), in addition to low-κ TDDB effects in the surrounding copper barrier dielectric of latest technology nodes [67].

All those physical issues must be analyzed and quantified to guarantee the clinical accuracy of the output results delivered by the applications, but also to ensure that the biomedical devices work safely when used by doctors on the patients. However, the modeling of these time-dependent aging effects raises several complex research challenges [66, 68]. In particular, the most crucial one is the workload-dependent nature of the involved aging mechanisms. So far, to anticipate the effect of aging issues on the electrical properties of the circuit, a traditional design approach based on worst-case margins has been used. Nevertheless, with more advanced and strongly scaled technology nodes, these margins are becoming too extreme to guarantee safe system operations [48, 69]. Hence, alternative solutions must be developed to ensure system dependability, such as methodologies for accurate prediction/estimation of time-dependent aging degradations. Based on these precise estimations, proper hardware and software mechanisms can be integrated at design time, to enable the future detection and mitigation of aging-related issues at the potential cost of performance penalties and energy/area overheads.

1.3 Thesis Contributions

The main purpose of this thesis is to propose a set of architectural and technological solutions to further enhance the energy efficiency and reliability of current wearable medical devices. Bearing in mind this objective and based on the challenges mentioned in Section 1.2.2, I develop this thesis on two complementary research lines.

- Firstly, I follow a top-down approach, driven by the software characteristics of bio-signal processing algorithms to derive domain-specific and energy-efficient hardware architectures.
- Secondly, I investigate how the modeling of technology reliability issues in logic and memory components can be exploited to adequately adjust the frequency and voltage parameters of a circuit.

In both cases, I explore the beneficial impact of the proposed techniques at the system level, in terms of performance, energy savings and reliability, to ensure full compliance with the tight real-time, energy, and clinical accuracy requirements of biomedical devices.

In order to provide a global overview of this research work, its main contributions have been summarized as follows.

1.3.1 Heterogeneous and Reconfigurable Energy-Efficient Bio-Signal Processing Architectures

In the first part of my research and within the context of WBSNs, I explore how intensive segments (i.e., computational kernels) of biomedical applications can be efficiently executed on a reconfigurable hardware accelerator, in order to speed up the system and minimize its energy consumption.

To achieve this goal, I develop and evaluate a novel heterogeneous and reconfigurable computing platform devoted to bio-signal analysis and interfaced to three different CGRA accelerators. Each proposed CGRA architecture offer large raw computation capabilities with a low-cost implementation consuming a reduced amount of energy. In contrast to state-of-the-art solutions [70], the access to each of these customized reconfigurable meshes (i.e., CGRAs) is shared concurrently among several computing processors. The resulting CGRA-based multi-core system seamlessly integrates these heterogeneous computing blocks by unifying the concepts of synchronization among processors and shared reconfiguration of the coarse-grained accelerator. To this end, I develop an innovative hardware/software synchronization and reconfiguration methodology for Smart WBSNs, based on instruction set extensions supported by dedicated hardware elements. The proposed framework allows also to devise novel strategies for system-wide energy management based on the clock and power gating of the heterogeneous computing resources.

Moreover, the computing capabilities of each studied CGRA architecture contribute differently to the maximization of the application parallelization. Thus, I bring further contributions by proposing the following CGRA architectures:

Single-Datapath CGRA Accelerator Shared by a Multi-Core System

- I describe and evaluate a single-datapath CGRA architecture, supporting the concurrent execution of different acceleration requests issued by multiple processors running in parallel. This single-datapath CGRA architecture represents the simplest CGRA version employed in this work. It allows the developed platform to achieve substantial energy savings of up to 18.6 %, with respect to an equivalent multi-core solution without CGRA accelerator.

Multi-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

- I present and study a multi-datapath CGRA architecture optimized for Single Instruction / Multiple Data (SIMD) operations. In comparison with the previous architecture, which only considers SIMD operations at the multi-processor level, I now explore the benefits of also supporting SIMD kernel executions on a CGRA accelerator. In this context, I investigate how multiple requests of accelerated functions from different processors can be merged into a single configuration, transparently from the application perspective. Thanks to the optimized SIMD execution of the kernels, energy savings of up to 37.2 % are achievable by the platform featuring a multi-datapath CGRA, compared to a multi-core system without hardware acceleration capabilities.

Interleaved-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

- I introduce and analyze a novel CGRA architecture which, by employing complex computing cells with multiple interleaved-datapaths, leverages further the execution parallelism of individual kernels to speed-up their computation, reduce the required memory bandwidth and decrease the area/energy overheads of the control logic. In particular, this interleaved-datapath CGRA architecture enables a reduction of up to 69.4 % of the energy consumed by the accelerated kernels, compared to their execution on the processors, without hardware accelerator. Furthermore, similarly to the single-datapath CGRA, this new architecture is also profitable for a single-core system.

In addition, throughout this study, for each CGRA architecture I showcase the performance and energy gains of the developed heterogeneous and reconfigurable computing systems, while executing complex bio-signal processing applications, based on filtering, classification, compression and feature extraction algorithms for multi-lead ECG acquisitions.

1.3.2 Technology-Level Reliability Exploration in Energy-Efficient Biomedical Systems

In the second part of my research, I explore how the fine-tuning of the operating parameters (i.e., frequency and voltage) can be employed in accordance with the physical limitations of the technology, to reach higher energy savings and computing performance. In contrast with the architectural exploration presented in the previous section, I propose herein to study technology-level reliability-aware solutions in logic and memory components, which do not impose any modifications of the computing architecture. In fact, the proposed solutions are generic and flexible enough to be directly applicable to other embedded DSP platforms. To this end, I exploit the inherent characteristics and fault tolerance of several biomedical applications, allowing to push the limits of the circuit operating parameters. The main contributions stemming from this research line are presented as follows.

1.3.2.1 Logic-Level Reliability Study

The trend to smaller technology nodes and the extended operation of devices increase the impact of various aging effects on the timing properties of the circuits (see Section 1.2.2.2). In particular, Bias Temperature Instability (BTI) affects the timing characteristics of individual transistors by altering their threshold voltages [66]. These timing variations may induce functional errors in the behavior of the complete system. The traditional method for coping with those issues, which consists simply of guaranteeing large enough margins, leads to inefficient worst-case based designs, with non-optimized performances [71].

By analyzing the impact of BTI effects at the different abstraction levels of the system, I propose to maximize the computing performance of logic components, by increasing the frequency beyond the traditional safety margins. This approach aims at determining the range of safe operating frequencies under BTI-induced timing degradations. To this end, a

workload-dependent analysis is proposed to avoid an overly pessimistic estimation of the timing degradations usually obtained with application-independent analyses.

Besides, the proposed approach is beneficial for embedded systems with performance, energy and reliability constraints, but also for computing platforms with extremely different needs. On one hand it can be employed for High Performance Computing (HPC) systems, for which the highest possible and safe frequency is required to perform the maximum amount of correct operations in a minimum amount of time and at a nominal supply voltage. While on the other hand, in the context of this thesis, WBSN systems may benefit from this approach by looking at the minimum supply voltage and frequency, satisfying the tight real-time, processing quality, and energy constraints of these devices. In more detail, the main contributions are listed below:

- I introduce an innovative framework able to perform a workload-dependent BTI-aware analysis on the logic of a synchronous processor. The considered workload, generated by a complex ECG processing application, presents several interesting characteristics perfectly suited to highlight the physical phenomena involved in the transistor-level BTI variability.
- I describe the different mechanisms used by the proposed framework to analyze the impact of BTI-induced timing degradations on functional error rates. This framework can either be employed to improve system performance and correctness (i.e., avoiding unsafe working points) or to achieve a graceful degradation of system characteristics according to specific application requirements. Moreover, the structure of this framework is generic enough to be reused for the evaluation of other device-level aging effects (e.g., RTN, HCI, or high-κ TDDB), by simply replacing the employed technology-dependent aging model.
- For several operating points, I explore the short- and long-term effects of BTI-induced timing violations on the functionality delivered by the circuit.
- At the application level, I characterize the accuracy degradation of the results produced by the considered biomedical DSP algorithm when executed on a processor prone to BTI reliability issues. Thanks to the proposed framework and based on the considered experimental setup, the obtained results showcase the possibility to safely double the operating frequency of the system, with respect to the maximum one determined with the classical Static Timing Analysis (STA).
- Lastly, to cope with BTI reliability concerns at run-time, I provide insights on the mitigation of BTI-induced functional errors at the hardware and software levels.

1.3.2.2 Memory-Level Reliability Study

As presented in Section 1.2.1, state-of-the-art embedded DSP platforms such as WBSNs apply voltage scaling to lower as much as possible their energy consumption and achieve longer battery lifetimes. The lower bound is given by the fact that reliability issues appear

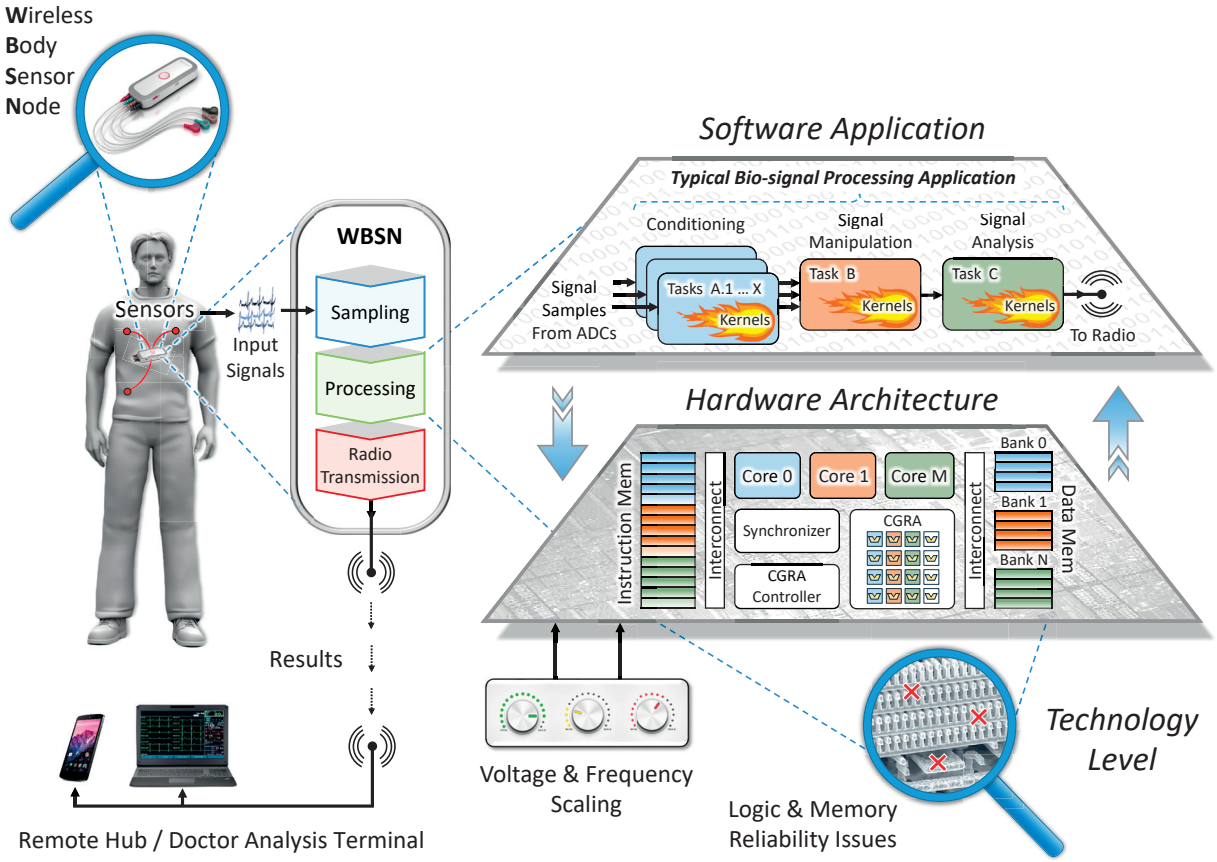


Figure 1.7 – Research scenario: Energy-efficient WBSN platform, embedding a CGRA accelerator shared by multiple computing cores, and running under fine-tuned operating conditions to meet the performance, energy, and reliability constraints.

as we approach the transistors' threshold voltage. While embedded memories often rely on Error Correction Codes (ECCs) for error protection, in the last part of my thesis I explore how the characteristics of biomedical applications can be exploited to develop new error mitigation techniques with lower energy overheads. As a result, I introduce a novel approach to minimize the energy consumption of memories, by reducing aggressively the supply voltage of these components, while taking full advantage of the inherent resilience of several bio-signal processing algorithms. In particular, the main contributions of this research path are listed as follows:

- I study the significance of the data bits processed by different widely used biomedical applications. Following this analysis, I show the output quality degradation as a function of the pseudo-permanent errors injected inside memory words, by using a stuck-at fault model. This information is used to order by significance of criticality the data which must be protected by any error mitigation technique.
- I introduce the *Dynamic eRror compEnsation And Masking protection* (DREAM), a new asymmetric error mitigation technique that consumes 21 % less energy than traditional Error Correction Code (ECC) with Single Error Correction / Double Error Detection

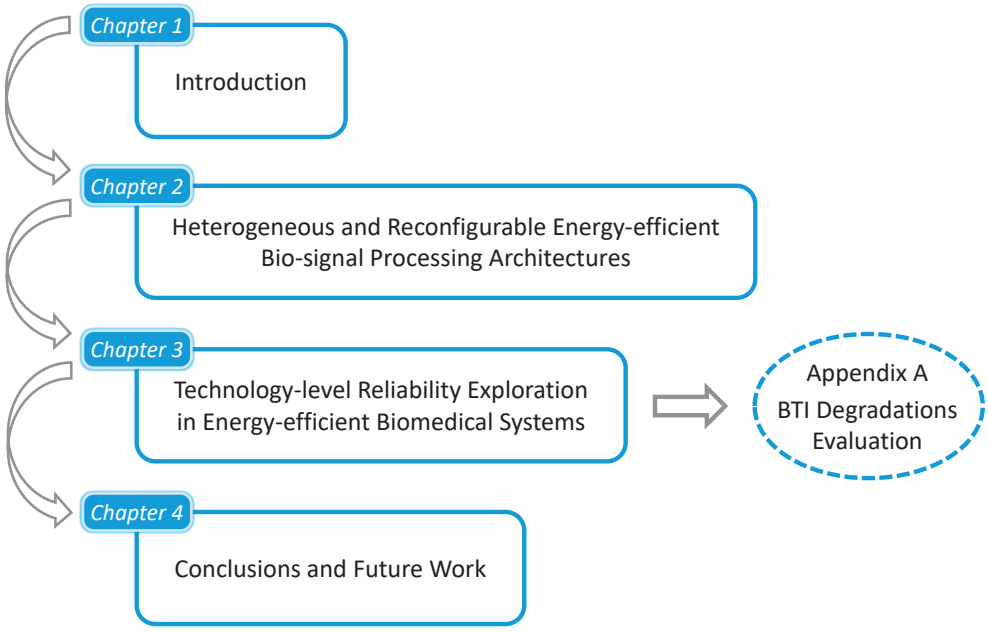


Figure 1.8 – Thesis structure.

(SEC-DED). The correction ability, energy consumption and area overhead of DREAM are compared to the SEC-DED ECC memory protection.

- I analyze the efficiency of different error mitigation techniques according to the supply voltage of the memory, to get the best trade-off between energy consumed and output quality in energy-efficient WBSNs. As an example, by tolerating a -1 dB degradation of the output data produced by the applications, DREAM allows to save up to 31 % of the energy consumed by the memories, compared to a system running at the nominal supply voltage with no protection.

Finally, to illustrate and summarize all the aforementioned contributions of this thesis, Figure 1.7 provides an overall view of the research scenario. More precisely, the work presented in this thesis focuses on the processing stage of Smart WBSNs to improve their energy efficiency. It mainly covers three design levels, such as the software application, hardware architecture and technology levels. First, an energy-efficient bio-signal processing platform featuring a CGRA accelerator (with different architectures) is proposed and studied. Then, an analysis of the optimal operating conditions of logic and memory components is performed, to meet the tight performance, energy, and reliability constraints of Smart WBSNs.

1.4 Thesis Outline

As shown in Figure 1.8, the rest of this thesis follows the structure described in Section 1.3. Each chapter provides the necessary background information related to each research path, supported by a review of the current state-of-the-art in the field. In particular, the structure of the thesis is organized as follows:

- **Chapter 2** deals with the design of energy-efficient heterogeneous and reconfigurable systems devoted to bio-signals analysis. The first part of this chapter presents the context and highlights the promising benefits of accelerated DSP systems, supporting optimized parallel execution of computationally-intensive kernels. The second part details the pros and cons of various hardware accelerator architectures, providing guidelines to the selection of the optimal design. The third part introduces three different architectures of CGRA accelerators, before analyzing individually their performance and energy benefits, when integrated within a multi-core system. The last part discusses the results obtained from this architecture exploration and concludes the chapter.

- **Chapter 3** details the reliability exploration performed in logic and memory components of WBSNs, to achieve higher performance and energy savings. The first half of this chapter deals with the BTI reliability study in logic components. The origins and effects of BTI-induced transistor degradations are first presented, before introducing and evaluating the proposed framework for short-term and long-term workload-dependent BTI effects analysis in logic circuits. To conclude the first part of the chapter, several insights are provided on the appropriate error mitigation techniques to tackle BTI reliability issues at the functional level. Then, the second part of the chapter presents the reliability study in memory components operating at low supply voltages. First, the different reliability issues in current memory technologies are described, followed by a presentation of the common error mitigation and correction strategies to address them. Second, a characterization of the inherent error resilience of biomedical applications is performed, before describing and analyzing the proposed energy-efficient DREAM technique. Finally, the last part of the chapter concludes by highlighting the existing trade-offs for the selection of the optimal error mitigation technique in memories for a given biomedical application and supply voltage.

- **Chapter 4** concludes this thesis by summarizing its main contributions and by providing directions for future work linked with its research paths.

2 Heterogeneous and Reconfigurable Energy-Efficient Bio-signal Processing Architectures

2.1 Introduction and Motivations

THE aging of the world population and the growing trend of unhealthy lifestyles are causing a major shift in the healthcare landscape [72, 73]. Nowadays, the number of cases of ailments caused by infectious agents have been displaced by chronic cardiac diseasecardiac diseases as the first cause of death worldwide [5]. These chronic conditions require the long-term and continuous monitoring of affected patients, which has a major impact on their quality of life, while also presenting a high financial burden for healthcare providers [6].

Today, Wireless Body Sensor Nodes (WBSNs) are emerging as an important technological aid to cope with this important societal challenge, as they allow the uninterrupted acquisition of clinically-relevant bio-signals outside of a medical environment and with limited supervision from the medical staff. WBSNs are wearable devices, able to monitor bio-signals generated by several organs and with different characteristics. For instance, these devices enable the recording of the electrical activity from the heart with Electrocardiograms (ECGs), the blood oxygen saturation (SpO_2) with Photoplethysmograms (PPGs), the muscles activity with Electromyograms (EMGs) and the brain activity with Electroencephalograms (EEGs) [8, 9, 10]. Also, cardiac data can be complemented by accelerometer acquisitions for discerning posture and/or level of activity and by measures of the electrical conductivity of the patient skin, which assess the amount of sweat-induced moisture.

Furthermore, one of the main features of WBSNs concerns their high energy efficiency, since they must operate over a prolonged period of time while relying on compact batteries, to reduce the discomfort of the patients wearing these devices. In this context, the energy bottleneck of most WBSNs, which only perform acquisition and transmission, usually resides in the wireless communication stage, because the slow dynamics of bio-signals allow their acquisition while employing little energy [15].

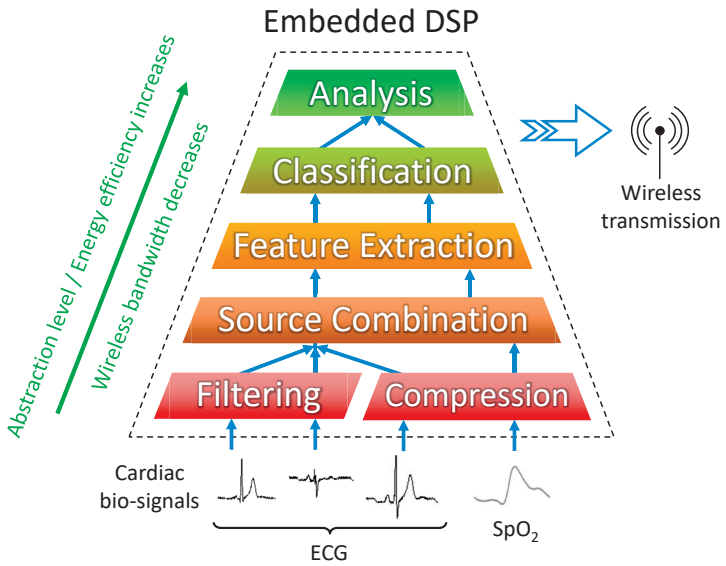


Figure 2.1 – On-board DSP allows for energy savings in the wireless transmission link.

A striking alternative is embodied by *Smart WBSNs*, which perform advanced on-board Digital Signal Processing (DSP) to extract relevant high-level signal characteristics or features from acquisitions [17]. Thanks to this approach, only signal features (as opposed to samples) are transmitted through the energy-hungry wireless link, potentially resulting in significant energy savings [18]. However, these benefits can only be leveraged by providing substantial processing capabilities within the tight budget constraints of wearable and battery-supplied devices. In fact, their design demands a carefully tailored computing architecture for hosting the execution of on-chip applications.

2.1.1 Parallel Processing and Computational Hotspots in Biomedical Applications

The development of DSP applications is dictated by both the clinical requirements and the allowed transmission bandwidth, which is controlled by the abstraction level of the output results, as depicted in Figure 2.1. The following paragraph provides several examples of biomedical applications producing results at the different abstraction levels illustrated by this figure.

In order to reduce the amount of transmitted data, an approach based on signal compression allows to recover on the receiver side, the whole sampled signals with possibly a degree of degradation. For this purpose, several lightweight compression strategies are presented in the literature, based on Compressed Sensing [18, 36, 37], Discrete Wavelet Transform (DWT) compression [74] and Adaptive Differential Pulse Code Modulation (ADPCM) [75]. Alternative solutions propose the computation and extraction of features of interest from combined input signals, with the aim of identifying their time or frequency domain characteristics [76, 77]. Ultimately, at a higher level of abstraction, the automated identification of different physical or pathological conditions can be achieved. For instance in [78] a Principal and Independent

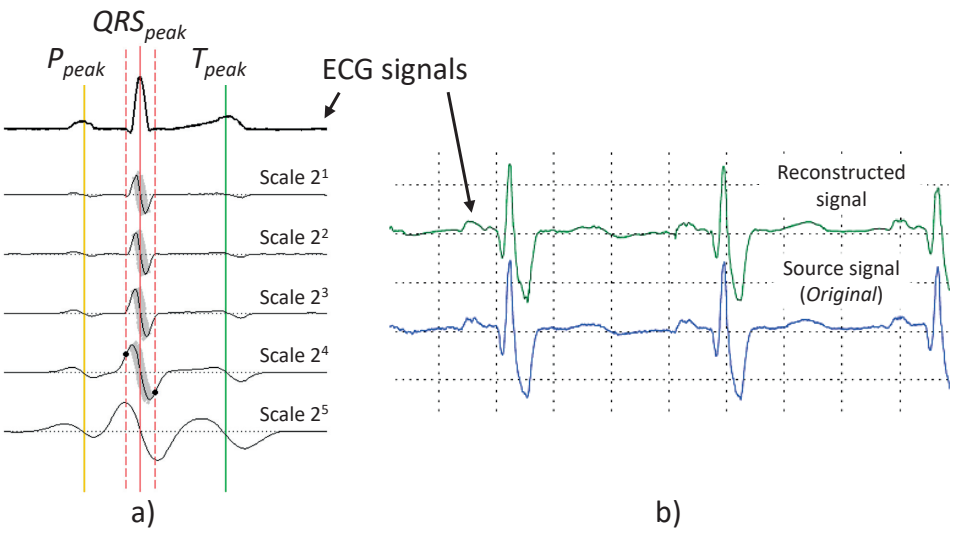


Figure 2.2 – Examples of digital signal processing applications for ECG signals: a) DWT decomposition [35] and b) compressed sensing [19].

Component Analysis (PCA) is used jointly with neural networks to classify cardiac arrhythmias, while in [79] pathological heartbeat classes are detected using Independent Component Analysis (ICA).

Regrettably, these examples of biomedical applications require a significant amount of computations, which must be executed on a wearable device with limited resources. More importantly, the energy required in the DSP stage of a WBSN should not offset the benefits obtained from the reduction in transmission bandwidth. For all these reasons, to sustain complex signal processing workloads at ultra-low power levels, a number of single-core processors have been proposed, specifically tailored for embedded bio-signal processing. These domain-specific processor architectures usually feature a small area footprint in conjunction with advanced power management units to adjust at run-time the operating parameters of the system (i.e., frequency and voltage) to different workload conditions [41, 42, 43].

Today, to address the increasing demand in computing power and at the same time reach higher energy savings, the design of domain-specific processors is evolving from single-core to multi-core architectures [14, 44, 80, 81]. Often, this new branch of processor architectures supports Single Instruction / Multiple Data (SIMD) execution modes [25], exploiting the parallelism intrinsic in bio-signal processing algorithms when multiple signals are concurrently acquired and processed [82, 83, 84]. For instance, in the case of ECG signal processing, 3-lead (inputs), 6-lead, and 12-lead configurations are commonly employed. Additionally, since bio-signal processing applications must be executed within real-time constraints, the parallelization over multiple cores allows to decrease the global clock frequency and conserve the same computing performance. That in turn enables the use of lower supply voltages, resulting in substantial savings in both static and dynamic power consumption due to voltage/frequency scaling.

Furthermore, another characteristic of bio-signal processing applications concerns their non-uniform distribution of the computational effort throughout their execution. In fact, a significant part of the computing power is required to execute short and intensive computational hotspots (i.e., kernels), usually in the form of compact loops which are data (as opposed to control) dominated. This observation underlines the opportunity for single-core and multi-core architectures to achieve a better computing performance, thanks to dedicated hardware (HW) blocks (e.g., custom instructions [85] or dedicated accelerators [26, 40]), to efficiently support computationally-intensive kernels of applications. Notable examples are the computation of matrix multiplication, Fast Fourier Transform (FFT), DWT scales decomposition when performing wavelet-based ECG delineation (see Figure 2.2.a), or extracting feature vectors when executing compressed sensing (see Figure 2.2.b). This strategy based on hardware accelerators can also result in orders-of-magnitude energy reductions by allowing a fine-grained and optimized software (SW) application execution [86, 87, 88].

Similarly to multi-core systems which require software and/or hardware support (e.g., synchronization mechanism) to orchestrate the execution of the parallel tasks between the different cores [44], hardware accelerators such as Coarse-Grained Reconfigurable Arrays (CGRAs), require also support at the software and hardware levels to perform the sub-tasks or kernels allocation and distribution among their shared computing resources [89]. However, in addition to the integration of this accelerator support, an initial and careful execution of the following steps is necessary at design time:

1. First, a characterization of the different software application tasks must be performed to identify the ones which can be serialized or parallelized on the processing system.
2. Then, inside each of these processing tasks, a computationally-intensive sub-tasks or kernels selection takes place to identify the most frequently executed at run-time.
3. Finally, among the selected kernels, an analysis is carried out to highlight the ones which can efficiently benefit from the accelerator when executed within a single- or multi-core system. During this last step, the employed profiling and design tools check if each kernel can fit on the shared computing resources of the hardware accelerator, and if its execution on the accelerator can bring performance and/or energy improvements to the complete system.

All these challenging steps constitute an important part of this work and are presented in detail in Section 2.3 of this chapter.

2.1.2 Contributions and Outline of the Chapter

This chapter deals with an architectural exploration of a heterogeneous and reconfigurable system devoted to bio-signal analysis. More particularly, in this work I propose several architectural solutions to exploit and maximize the parallel computing of well-known biomedical applications. The intended goal is to achieve higher performance and energy savings, beyond the capabilities offered by a baseline multi-core system.

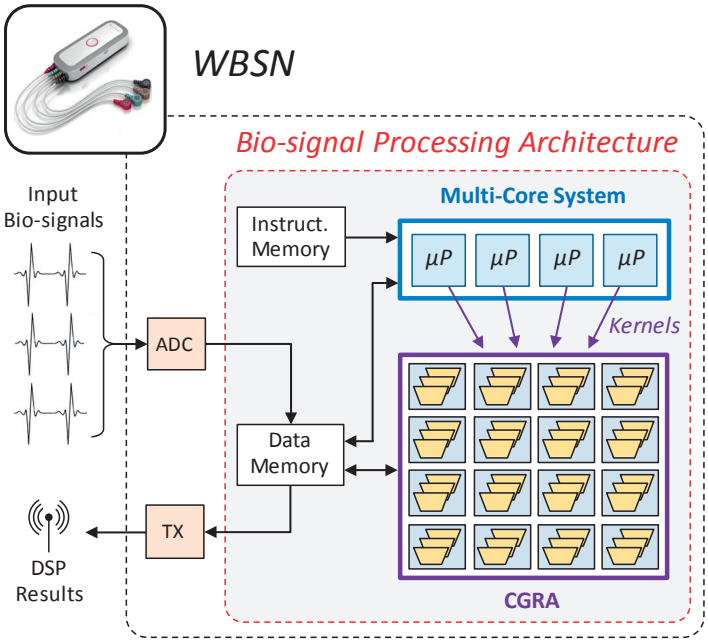


Figure 2.3 – Bio-signal processing architecture coupling a multi-core system with a CGRA unit, supporting SIMD execution in both resources.

As depicted in Figure 2.3, thanks to a domain-specific CGRA accelerator shared by a multi-core system, I propose an efficient approach to accelerate the execution of the computationally-intensive kernels, previously identified within biomedical applications.

In the following sections of this chapter, three different architectures of a CGRA accelerator are presented and studied. An analysis, both at the system and component levels, is carried out to evaluate the contributions of the different CGRA architectures in terms of performance and energy savings, with respect to the introduced overheads. In addition, all along this study the benefits provided by the different CGRA architectures are systematically compared to a state-of-the-art multi-core processing system without kernel accelerations.

In more detail, the most relevant contributions of this chapter associated to the proposed CGRA architectures, are listed bellow:

Single-Datapath CGRA Accelerator Shared by a Multi-Core System

- I introduce and evaluate a reconfigurable and heterogeneous system devoted to bio-signal processing, which integrates multiple processors and a shared *Single-Datapath* CGRA accelerator.
- I propose a unified hardware/software mechanism to jointly support synchronization among cores, acceleration of kernels, and run-time energy management policies at the system level with very low overheads.

- I showcase the efficiency of the developed system while executing complex bio-signal DSP on ECG acquisitions, such as applications for filtering, classification and feature extraction.

Multi-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

- I introduce and explore the performance, from an energy-efficiency perspective, of a novel CGRA design, optimized for SIMD operations. In comparison with the previous architecture, which only considers SIMD executions at the multi-processor level, I now investigate the benefits of also supporting SIMD-kernels, by adopting a *Multi-Datapath* CGRA. This new CGRA architecture is able to execute several identical kernels in parallel on the same Reconfigurable Cells or RCs (i.e., CGRA computing units).
- I detail how the developed SIMD-CGRA can be integrated in an energy-efficient multi-core system devoted to bio-signal analysis.
- I propose a set of synchronization and reconfiguration methodologies for a shared multi-datapath CGRA in a multi-core system.
- I perform an experimental evaluation to showcase the energy benefit of the multi-core heterogeneous system when performing complex signal analysis on multi-lead ECG acquisitions.

Interleaved-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

- I describe a novel CGRA architecture which, by employing complex RCs with multiple *Interleaved-Datapaths* (i-DPs), leverages the execution parallelism inside each kernel to speed-up its computation and decrease the area/energy burden of the control logic.
- I propose an interleaving mechanism between the datapaths of the CGRA cells, which minimizes the required memory bandwidth and avoids extra area/energy overheads induced by wider memory ports.
- I perform a systematic investigation of the benefits derived from the introduced architectural choices, considering various kernels with different characteristics and two real-world bio-signal analysis applications. Similarly to the single-datapath CGRA, this new architecture is also applicable and beneficial for a single-core system.

The rest of this chapter is structured as follows. First, Section 2.2 summarizes the current state-of-the-art in the field and in particular, the different accelerator architectures that can be employed to achieve the envisioned goal. Secondly, Section 2.3 describes the proposed CGRA-based multi-core system and how the selected CGRA mesh can be efficiently integrated and shared by multiple processors at the same time. Then, Sections 2.4, 2.5, and 2.6 present and evaluate the three different CGRA architectures developed in this research, and how they can be beneficial for the overall system. Finally, Section 2.7 summarizes the main achievements of this chapter.

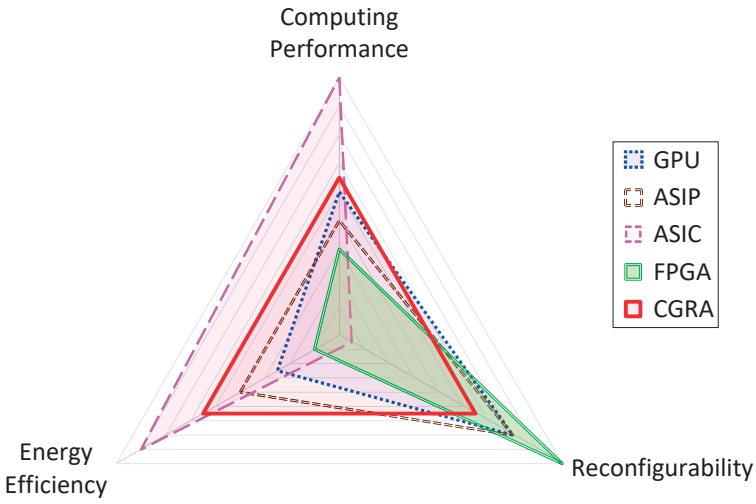


Figure 2.4 – The three design corners of five hardware accelerator and processor architectures, employed with the same operating parameters.

2.2 State-of-the-Art and Selection of the Accelerator Architecture

Nowadays, thanks to the evolution of the technologies, a broad set of computing architectures can be employed as hardware accelerators. The following paragraphs highlight the strengths and weaknesses of commonly employed accelerator architectures, before presenting the one selected for this study.

GPU: In the context of embedded computing, general-purpose processor architectures are commonly employed to execute the wide spectrum of mobile applications, solicited by the evolution of the consumer electronics market [90]. In fact, the increasing demand in computing performance with the latest multimedia applications (e.g., video games, 3D face recognition, machine learning algorithms) has pushed toward the integration of Graphics Processor Units (GPUs) coupled with embedded general-purpose processors inside a single chip [91, 92]. This design strategy improves considerably the computing performance of the complete system (e.g., smartphone, tablets). However, it has a significant area and energy cost, which is not suitable for battery-powered embedded systems with an expected operating life ranging from several days to several weeks on a single charge [23].

ASIP: For all these reasons an alternative design strategy has been chosen for energy efficient WBSNs. As shown in Section 2.1.1, current WBSNs rely on domain-specific or Application Specific Instruction-set Processors (ASIPs) with a customized single- or multi-core architecture. Nowadays, similarly to multimedia embedded systems, advanced biomedical applications require more and more computing power, while retaining the energy consumption as low as possible (see Section 2.1.1). To address these challenges, hardware accelerators used in conjunction with domain-specific processors offer interesting opportunities to perform an optimized execution of application workloads, and in particular a fined-grained execution of the computationally-intensive kernels.

ASIC: Several accelerator architectures are presented in the literature and propose different trade-offs between performance, reconfigurability and energy efficiency, as illustrated in Figure 2.4. For instance, employing dedicated hardware blocks such as Application-Specific Integrated Circuit (ASIC) accelerators on the computationally-intensive kernels, can help to simultaneously meet tight performance/energy requirements [85, 93]. However, this strategy is also inflexible, as each block can perform only a single operation. These characteristics are even more preeminent when accelerators are integrated in a multi-core environment [87, 94], because multiple requests for accelerated functions issued by each core must be arbitrated when trying to access the same computing resource. As a consequence, the increasing convergence of different functionalities combined with high non-recurring costs involved in designing ASICs have oriented designers towards more flexible solutions that are post-programmable [86].

FPGA: Reconfigurable array components, such as Field-Programmable Gate Arrays (FPGAs) or Complex Programmable Logic Devices (CPLDs), are composed by islands of flip-flops and Look-Up Tables (LUTs), which do not impose any restriction on the boolean function that can be mapped on the reconfigurable fabric. On the other hand, this high degree of flexibility offered by the bit-level reconfigurability incurs in huge area and energy overheads. As an example, an increase of 35-times in area, 4-times in critical path delay and 14-times in power consumption compared to equivalent ASIC implementations is reported in [93]. Nonetheless, despite the drawbacks of older FPGA generations, emerging FPGA-based accelerators are attracting more and more attention in the context of energy-efficient embedded systems similar to WBSN platforms [95]. In fact, for small-volume productions the development of costly ASICs may not be worthwhile in comparison to the fast development rounds of FPGAs. Therefore, the utilization of programmable System-On-Chips (SoCs) composed of multiple cores interfaced with an FPGA accelerator represents a cheaper alternative. However, the silicon area required to manufacture these new FPGA designs remains higher with respect to classical ASIC implementations.

CGRA: In contrast to FPGAs, CGRAs dramatically reduce the hardware overhead induced by reconfiguration, thanks to the utilization of Arithmetic and Logic Units (ALUs) as cell elements. This architectural choice results in a specialized reconfigurable mesh, on which arithmetic operations composing kernels can be efficiently mapped [89, 96]. Moreover, this choice leads to orders-of-magnitude improvements in reconfiguration times, as only the connection between computing elements and the operation to be performed by each cell must be configured at run-time. Finally, CGRA meshes allow for multiple configurations to be activated on a cycle-by-cycle basis. The resulting combination of spatial and temporal scheduling maximizes the utilization of available hardware resources by pipelining the execution of loop iterations [97], at the expense of the area overhead required to store configurations and control their activation.

This thesis work aims at aggressively exploiting the parallel nature of coarse-grained reconfiguration by interconnecting a CGRA instance as a shared accelerator in a multi-core system.

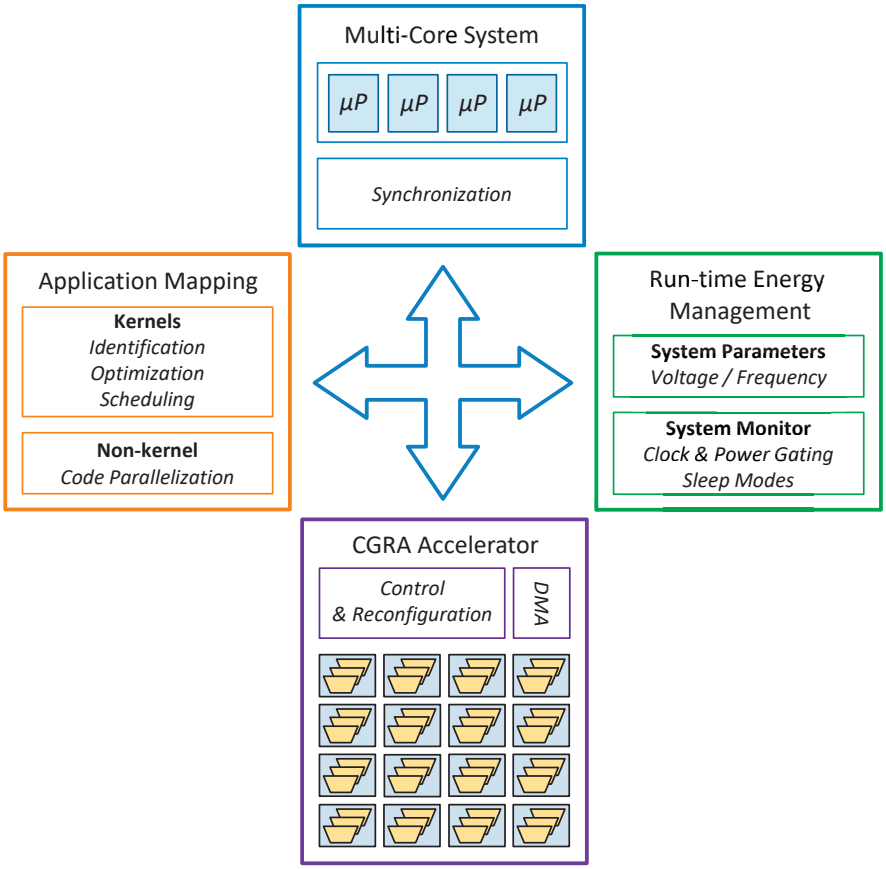


Figure 2.5 – Design framework and challenges.

From a higher-level perspective, this CGRA-based architecture behaves as a traditional multi-processor architecture coupled with GPUs. However, compared to GPUs (i.e., tiles of general purpose processors interfaced with large memory buffers), CGRAs provide better performance while consuming less energy due to their reduced logic/memory size and optimized capabilities for the targeted application domain (i.e., WBSNs) [98]. To confirm these observations, a comparison between the benefits provided by the proposed CGRA architectures and state-of-the-art hardware accelerators (e.g., GPU, ASIC, FPGA) can be also carried out. For reasons of time, this evaluation is left for future works.

Finally, my approach has some similarities with the one adopted in [70]. Nonetheless, the authors of this work adopt the limiting assumption that the reconfigurable fabric can be accessed only by one core at a time. Conversely, the proposed CGRA-based platform supports and orchestrates concurrent acceleration requests coming from multiple cores, either in a Multiple Instruction / Multiple Data (MIMD) or SIMD fashion [94]. This feature represents by itself a notable improvement over the state-of-the-art.

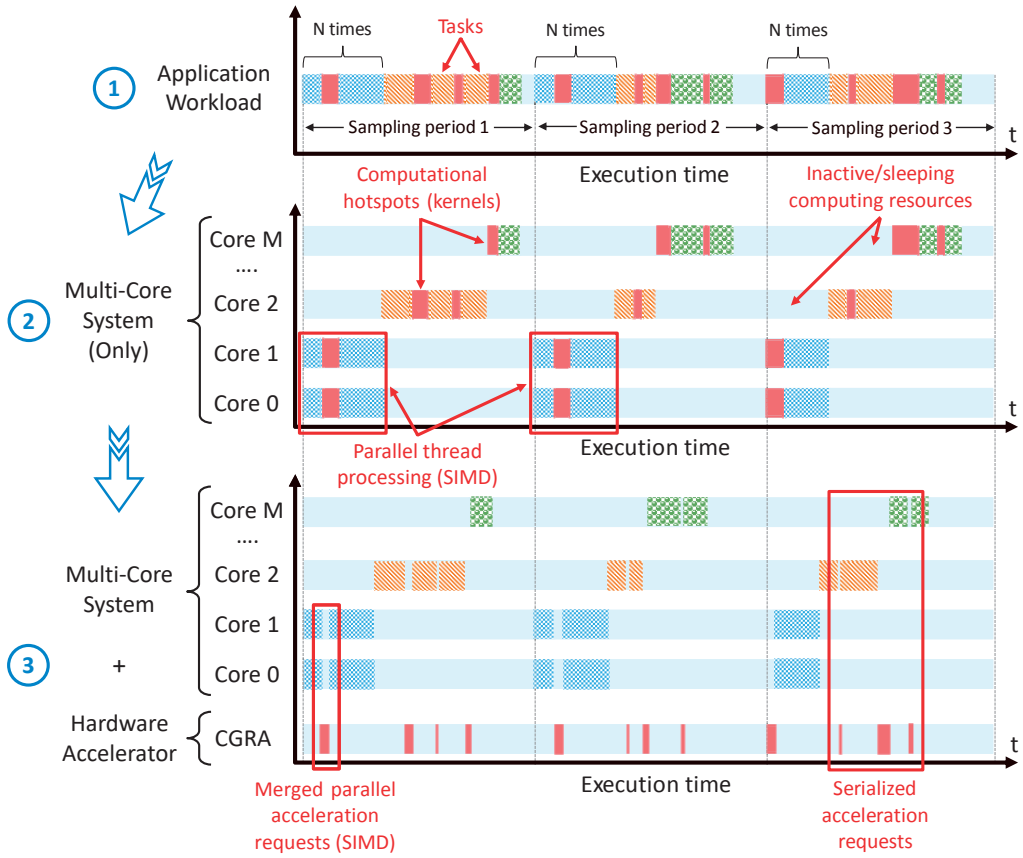


Figure 2.6 – Illustrative example of biomedical application workload decomposition and parallelization.

2.3 Reconfigurable and Accelerated Multi-Core System

As mentioned in the previous sections, by combining the benefits of multi-core systems and coarse-grained computation of application kernels, this work aims to achieve a disruptive improvement in the energy efficiency of WBSNs.

The study of this design space is performed systematically, as illustrated in Figure 2.5. It relies on four fundamental blocks to elaborate an energy-efficient heterogeneous and reconfigurable platform. My research addresses the challenges involved in the design of each of these blocks. In particular, I carry out an architectural exploration of domain-specific CGRA accelerators and their integration in the target heterogeneous computing system devoted to bio-signal analyses. Moreover, I devise novel methodologies to perform the partitioning and mapping of software applications on the envisioned hardware. Finally, I evaluate the benefits of run-time management policies in maximizing the energy efficiency of the overall system. The detailed description of these design challenges associated to the proposed hardware/software solutions is provided in the following sections. The resulting platform and CGRA accelerators were de-

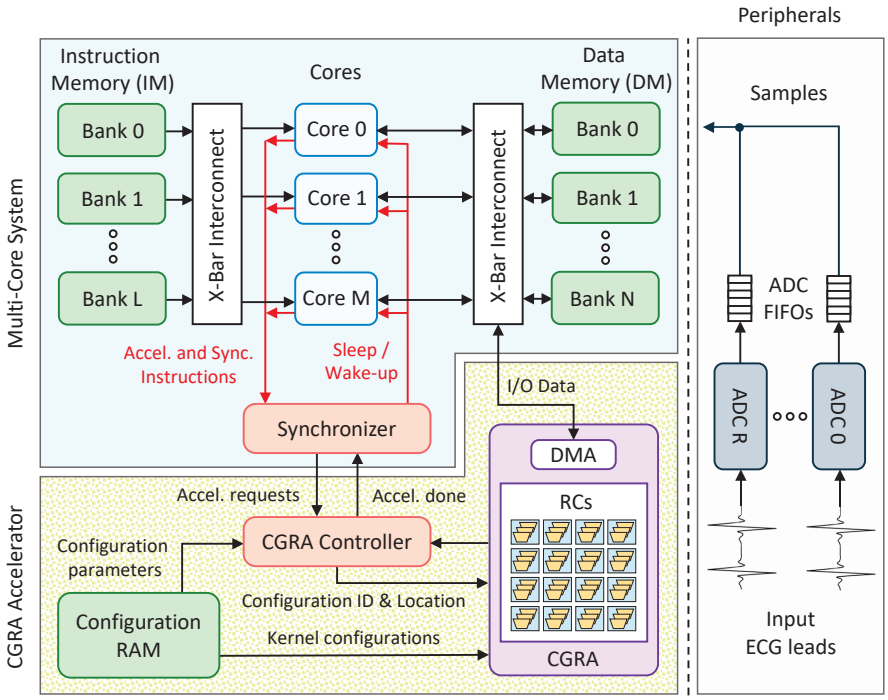


Figure 2.7 – High-level view of the accelerated multi-core platform.

signed and implemented with the help of *Soumya Basu*, who established the scheduling of the kernels, and performed the power/area characterization of the different CGRA architectures, based on an HDL implementation.

2.3.1 Multi-Core System

The proposed platform interfaces the envisioned CGRA meshes with a multi-core system, whose structure and run-time paradigm are described in this section. As represented in Figure 2.6, the system leverages the potential of parallel processing at two different levels to maximize its energy efficiency. First, at the thread level, it supports a flexible SIMD execution strategy over multiple cores. Second, at the loop level, it allows for spatial execution of computationally-intensive kernels (possibly in SIMD) on the coarse-grained reconfigurable mesh.

The platform allows for a time- and space-shared utilization of the CGRA module to execute the kernels to be accelerated. Cores are clock-gated when requesting an acceleration, waiting for synchronization after branches (see Section 2.3.1.3), and when no input data are available. The CGRA columns are composed of Reconfigurable Cells (RCs) dynamically programmed by several Configuration Registers (CRs) (see Section 2.3.2.1). The processing datapaths (passing through the RCs) are power-gated when not in use, as no data has to be retained across subsequent executions of accelerated functions. All these features are provided by a unified mechanism supporting fine-grained code synchronization, power management and

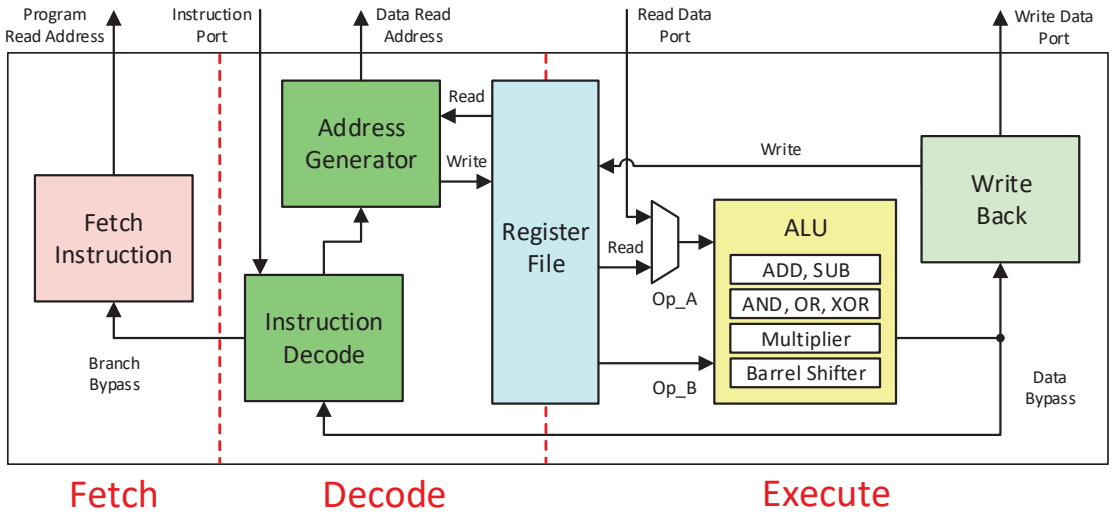


Figure 2.8 – TamaRISC three-stage pipeline architecture [32].

acceleration of computational hotspots. Its design is detailed in the following parts of this section.

The illustrative block diagram of the proposed platform is depicted in Figure 2.7. The upper part represents the hardware components of the multi-core system, while the bottom part illustrates the components of the CGRA accelerator. The lateral part on the right represents the sensing part of the platform, with the Analog-to-Digital Converters (ADCs) connected to the ECG electrodes. These peripherals are not covered in this work, since they are not part of the DSP stage from a WBSN.

2.3.1.1 Domain-Specific Processors

As a first step, the design of an energy-efficient DSP platform relies on a careful selection of its computing core(s). The state-of-the-art proposes a wide range of domain-specific processors, featuring tailored architectures devoted to embedded sensor data or signal processing [25, 26, 41, 42, 43, 81, 99].

In the context of this thesis, the choice of the processor has been oriented towards the TamaRISC architecture [32, 100], which integrates specific bio-signal processing capabilities widely exploited in the literature [38, 44, 85, 101]. In fact, the main characteristic of TamaRISC concerns its extremely small instruction set, which only supports 23 instructions, compared to standard Reduced Instruction Set Computers (RISCs) with hundreds of instructions [102]. By featuring a lightweight RISC architecture, able to execute all the typical operations involved in bio-signal processing applications, the hardware design of TamaRISC is substantially simplified, resulting in limited area and energy footprints for the complete processing platform. For instance, TamaRISC showcases an outstanding energy efficiency with an average consumption of 17.1 pJ per operation, at a nominal supply voltage of 1.2 V, and when implemented with a

90 nm transistor technology [32]. This processor also supports voltage and frequency scaling to achieve further energy savings [24].

As illustrated in Figure 2.8, TamaRISC adopts a Harvard architecture, based on a three-stage pipeline featuring the elementary *Fetch*, *Decode*, and *Execute* stages. The processor integrates a 24-bit read port to the instruction memory (IM), one 16-bit read and one 16-bit write ports to the data memory (DM). The processing datapath of the pipeline operates on a 16-bit data word length and is interfaced to a register file of 16×16-bit registers. The instruction set architecture (ISA) of TamaRISC comprises eight arithmetic and logic integer operations, one general data move operation and two program flow instructions. The ALU supports addition (with or without carry), subtraction (with or without carry/borrow), logic operations (AND, OR, XOR), right and left shift (arithmetic or logic) with a barrel shifter. Full signed/unsigned 16-bit×16-bit multiplications are also supported, and generate a 32-bit result splitted into 2×16-bit data words. Three addressing modes are allowed: register direct, register indirect with offset and register indirect with pre- and post-increment/decrement. To control the execution flow of the program, branching instructions are available in direct and indirect mode. They also support an optional address offset relative to the program counter (PC), with 15 condition modes based on the Carry (C), Zero (Z), Negative (N) and Overflow (O) processor status flags [32]. The instruction set of TamaRISC can also be extended or customized [85], which is an interesting feature for the proposed heterogeneous and reconfigurable platform (see Section 2.3.1.5).

Additionally, as bio-signal processing applications are usually divided into well-defined phases, similarly to [14], several instances of the TamaRISC core are employed in parallel to build this platform. In this way, the execution of the control-dominated parts of applications are spread over several homogeneous processors, while the data-dominated (i.e., computationally-intensive) parts are processed by the CGRA. Also, each TamaRISC core can be individually clock-gated when idle (i.e., when waiting for another core to finish its task or when waiting for another sample to process from the ADCs), thus reducing considerably the dynamic energy requirements of the platform.

2.3.1.2 Instruction and Data Memory Architecture

The TamaRISC processors instantiated within the multi-core system are interconnected to separate multi-banked instruction and data memories through combinational crossbars (X-Bars). The data memory space is partitioned in a shared section, devoted to inter-processor data communications, and in private sections for each core. Similarly to [44], the crossbars arbitrate and merge memory requests when the same data or instruction is read by multiple cores in SIMD mode, thus providing energy savings by reducing the amount of memory accesses.

Nonetheless, data-dependent branches can cause the execution flow on cores acting in SIMD to diverge. To recover at the end of those branches from run-time divergences, the hardware/-software solution presented in the next section has been adopted.

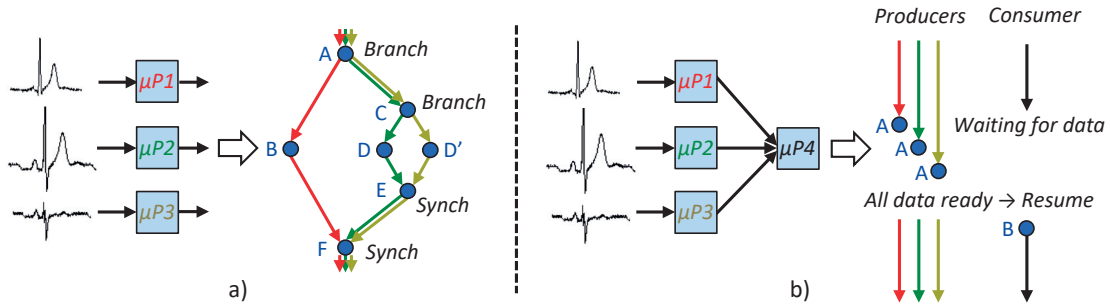


Figure 2.9 – Synchronization for processors (μPs) in lock-step execution (a) and in a producer/consumer relationship (b).

2.3.1.3 Hardware/Software Synchronization Mechanism

A) Lock-Step Execution Mode

In many bio-signal analysis applications [14, 82, 101, 103], the processing is performed on each signal independently, as represented in Figure 2.9.a. Three common scenarios are the noise filtering/removal from acquisitions, data compression, and features extraction from several input sources. These computations are inherently parallel, allowing the distribution of their

Algorithm 1: Example of data-dependent code branches executed in lock-step

Code snippet illustrated in Figure 2.9.a

```
1  ...
2  // Enter 1st lock-step region
3  switch Branch_Condition_A do
4      case Left_Branch_A do
5          // μP1 executes Step B → Step F
6      end
7      case Right_Branch_A do
8          // μP2 & μP3 execute Step C & Enter 2nd lock-step region
9          switch Branch_Condition_C do
10             case Left_Branch_C do
11                 // μP2 executes Step D → Step E
12             end
13             case Right_Branch_C do
14                 // μP3 executes Step D' → Step E
15             end
16         end
17         Wait_Sync() // Step E - Wait for μP2 & μP3 (Synchronization barrier)
18         // Exit 2nd lock-step region (μP2 & μP3 synchronized)
19     end
20 end
21 Wait_Sync() // Step F - Wait for all processors (Synchronization barrier)
22 // Exit 1st lock-step region (All processors synchronized)
23 ...
```

Algorithm 2: Example of producer/consumer relationship execution

Code snippet illustrated in Figure 2.9.b

```
1  ...
2  if Processor_ID < 4 then
3  |  // µP1, µP2 & µP3 execute Step A (Produce data)
4  end
5
6  Wait_Sync() // Wait for all processors (Synchronization barrier)
7  // Data ready (All processors synchronized)
8
9  if Processor_ID = 4 then
10 |  // µP4 executes Step B (Consume data)
11 end
12 ...
```

execution on different computing cores. When the same processing routine is concurrently executed on several computing units, the processors can be in lock-step (i.e., SIMD mode). In other words, they fetch the same instruction from the program memory all along the routine execution (see Section 2.3.1.2). Multiple memory requests can then be merged to a single memory access, resulting in substantial energy savings. Nonetheless, as shown in Algorithm 1, lock-step execution can be impeded by data-dependent branches, as conditional execution paths can diverge due to different input data. A synchronization mechanism must then be employed to recover lock-step at the end of each conditional statement or branch operation [44].

B) Producer/Consumer Execution Mode

In another execution scenario, illustrated in Figure 2.9.b, the processing of input streams cannot be disjoint. This is the case for many applications with multiple input channels combined to extract common features. In addition, some applications can also activate a more detailed processing of all the input sources when the presence of abnormal conditions is detected on a subset of the input data. To support these scenarios while still taking advantage of the parallel execution paradigm, synchronization should include, in addition to synchronization across branches, the support of producer/consumer relationships among processors as described in Algorithm 2.

C) SIMD-Kernel Execution Mode

In addition to lock-step and producer/consumer execution modes, a multi-core synchronization mechanism is also a major requirement in the case of CGRA meshes supporting SIMD-kernel executions (see Section 2.5). In this context, when identical acceleration requests are issued at the same time by cores running in lock-step, they will be mapped on the same set of reconfigurable cell columns (i.e., CGRA computing resources) and share the same configuration, fetched from the memory (i.e., CGRA Configuration RAM). However, in absence of synchronization, the cores will not be running in lock-step. Hence, the acceleration requests

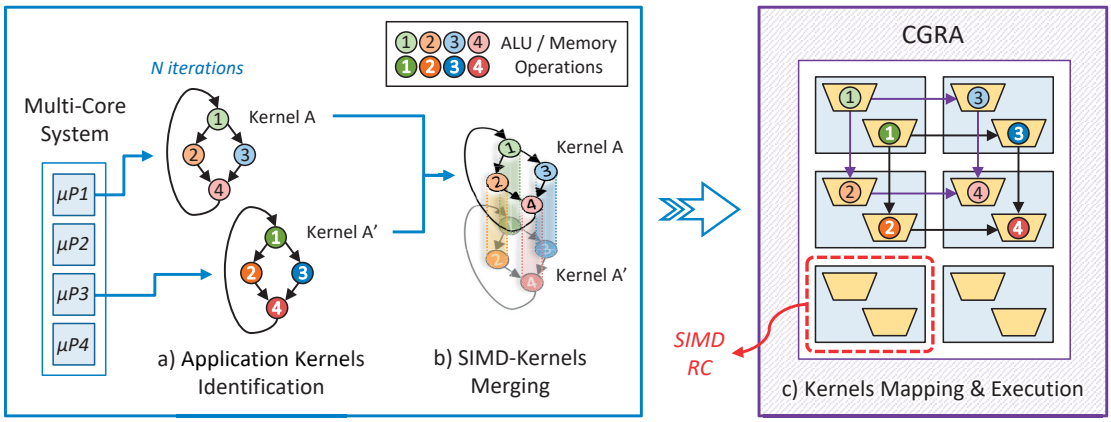


Figure 2.10 – High level management of SIMD-kernels execution. In this example, the elementary operations of two identical kernels from two cores operating in SIMD mode are merged, mapped and executed on the two datapaths of each CGRA RC columns. If another kernel acceleration request is issued by a third core, its execution on the CGRA will be delayed until the release of computing resources (i.e., when the first and second kernels have finished their executions on the two RC columns).

will be mapped at a different moment in time on different RC columns. This scenario leads to an overuse of the CGRA computing resources, followed by possible resource contentions, and unnecessary energy overheads when additional RC columns must be configured with the same configuration.

For all these reasons, the proposed heterogeneous platform integrates the lightweight hardware/software synchronization mechanism first described in [14]. This synchronization mechanism or *Synchronizer* supports lock-step/SIMD execution modes and the management of producer-consumer relationships between individual application threads. It relies on dedicated instructions (see Section 2.3.1.5) used to synchronize the code execution using reserved locations (i.e., synchronization points) in the shared data memory, recording information about the execution flow.

Finally, beyond the management of the computations between multiple cores, with the proposed platform the Synchronizer is also employed to co-orchestrate the acceleration requests and their execution on the reconfigurable CGRA architectures.

2.3.1.4 Acceleration Request Controller and Configuration Memory

In order to manage the execution of the acceleration requests on the CGRA, the aforementioned Synchronizer must be interfaced to a *CGRA Controller*.

The role of this controller is to coordinate the acceleration requests from the cores to the CGRA through a request queue, implemented with a First-In First-Out memory buffer (FIFO). The controller checks that sufficient resources are available at run-time to map an accelerated

Algorithm 3: Example of kernel implementation

Code snippet of the kernels illustrated in Figure 2.10

```
1  ...
2  // For each input data to process
3  for n ← 1 to N do
4      Operation_1() // Memory read access (Read input data)
5      switch Branch_Condition_1 do
6          case Left_Branch_1 do
7              Operation_2() // Arithmetic or logic operation
8          end
9          case Right_Branch_1 do
10             Operation_3() // Arithmetic or logic operation
11         end
12     end
13     Operation_4() // Memory write access (Write back result)
14 end
15 ...
```

function. It also arbitrates the accesses to the reconfigurable accelerator when multiple requests for *different* kernels are received in the same clock cycle.

Furthermore, when the CGRA architecture supports multiple processing datapaths (Multi-DPs) per RC, the controller is in charge of merging requests of the *same* kernels from cores executing in SIMD. As shown in Figure 2.10 and Algorithm 3, the kernel structure can be decomposed into elementary operations (e.g., addition, subtraction, multiplication, memory accesses) executed iteratively within a loop. Each operation which composes the kernel is individually executable on the reconfigurable cells of the CGRA mesh (i.e., coarse-grained reconfiguration and execution at the operation level). In the example illustrated in Figure 2.10, all the operations from two identical kernels are merged, mapped and concurrently executed on two DPs inside the same reconfigurable cells of the SIMD-CGRA. In case more SIMD-kernel accelerations are requested than the SIMD width (i.e., RC datapath size) of the CGRA, the ones corresponding to the cores with the lower processor identifier (PID), are executed first.

In addition, the storage for the configuration words (or bit-streams) used to program kernels on the CGRA fabric is provided by a *Configuration RAM*. A subset of these configuration words, called *Preambles*, are read by the controller to determine for instance, the number of RC columns required for the execution of a kernel. These preambles are also used to define the location (i.e., column identifiers) where the kernel will be mapped on the CGRA.

The remaining configuration words of a kernel are directly read by the CGRA. They dictate the functionality (i.e., arithmetic or logic operation) of a reconfigurable cell at a given clock cycle during the execution of a kernel iteration. These configuration words also provide the source of the operands (either from the neighboring cells or from the local register file) and the output destination of the result computed with the selected operation. Only few bits are necessary to encode these information, enabling the support of multiple configuration words

per RC with little memory overhead. Further details concerning the internal structure of RCs and the execution of kernels on the CGRA are provided in Section 2.3.2.

2.3.1.5 Instruction Set Extensions for Synchronization and Hardware Acceleration Support

In order to support the hardware synchronization strategy presented in Section 2.3.1.3, the instruction set of the TamaRISC processors is extended with four custom instructions, initially introduced by the authors of [14]. On one hand, the *SINC #literal* (i.e., *Synchronization Increment*) and *SDEC #literal* (i.e., *Synchronization Decrement*) instructions are employed to perform respectively the *check-in* and *check-out* processes at the different synchronization points (i.e., barriers), identified with a unique flag or integer number *#literal*. On the other hand, the *SNOP #literal* (i.e., *Synchronization No Operation*) instructions are used to register a core at specific synchronization point, without modifying its register or counter value.

These dedicated instructions are employed to enforce the producer-consumer relationships among the identified threads: *SNOP* instructions are inserted in the program of the consumer cores, while *SINC* and *SDEC* in the program of the producer cores. This last pair of instructions is also integrated before and after data-dependent code branches, to build synchronization barriers for cores assigned to parallel computation streams. These branches may also include the call to an acceleration request, which needs to be synchronized with the ones issued by other cores, when multi-datapath SIMD-CGRA architectures are employed (see Section 2.5).

Moreover, in order to reduce the dynamic energy consumption when a core is waiting at specific synchronization barriers or when an acceleration request has been sent to the CGRA, the *SLEEP* instruction is employed to clock-gate the core and pause its execution [14].

Furthermore, CPUs instantiated to hardware accelerators (e.g., GPUs) traditionally require an extra software support to launch acceleration requests from the software application to the selected hardware accelerator. As an example, CUDA (Compute Unified Device Architecture) is a complete Application Programing Interface (API), allowing software developers to use CUDA-enabled GPUs for general purpose processing [104]. Similarly to this approach and in the context of energy-efficient WBSNs, a lightweight solution is proposed in this research to support acceleration requests from the processors to the CGRA. To do so, an additional instruction is added to the TamaRISC ISA. It is defined as *ACCEL #literal*, where the literal specifies the unique kernel ID that the core needs to execute on the CGRA. The interaction between the hardware CGRA controller and the custom *ACCEL* instruction is described in the next section, which details how the proposed platform handles kernel acceleration requests.

2.3.1.6 Acceleration Request Execution Flow

The acceleration requests launched by the processors are processed according to the flow-chart presented in Figure 2.11. The different steps are detailed in the following paragraphs.

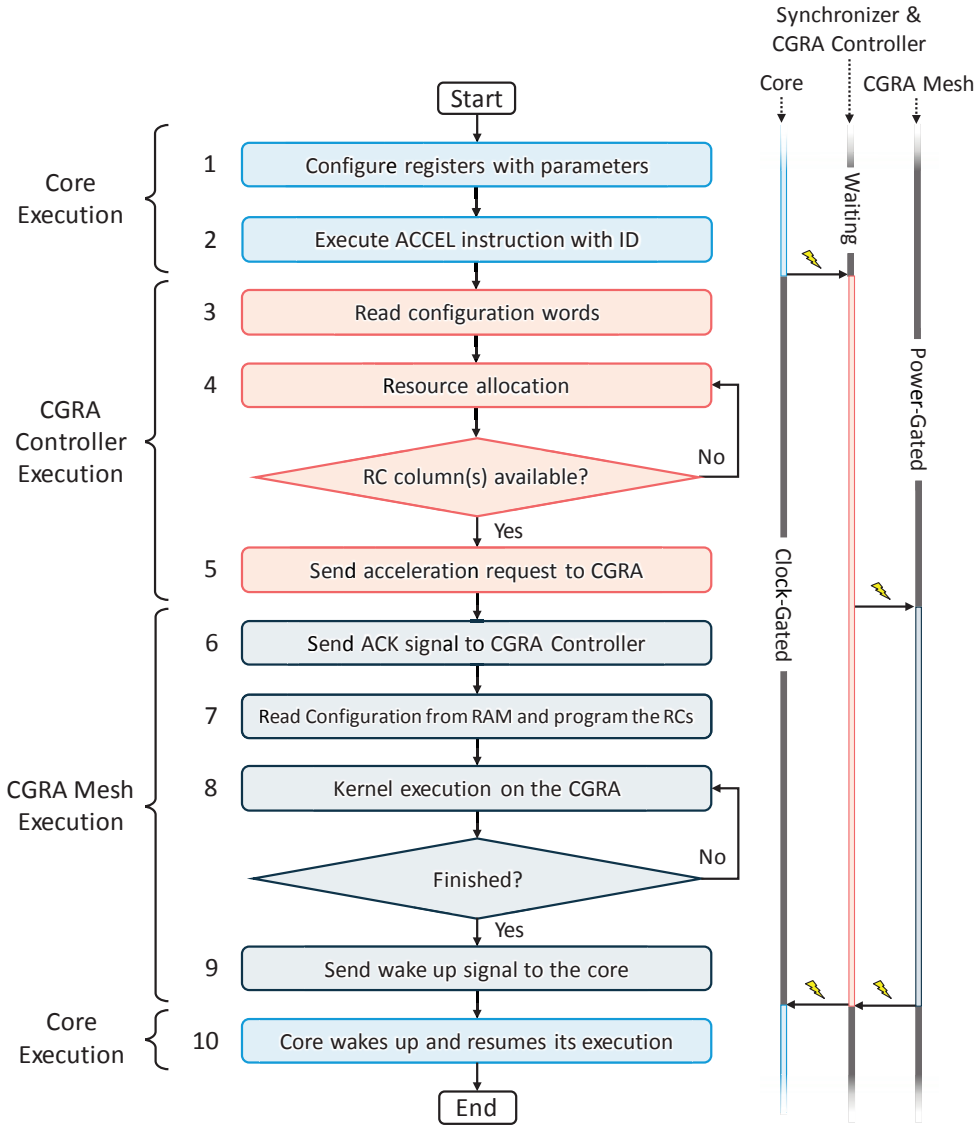


Figure 2.11 – Acceleration request execution flow diagram.

Step 1: Before requesting the execution of a kernel on the CGRA with the *ACCEL* instruction, each core has to configure a set of private memory-mapped registers, only accessible by the corresponding core and the CGRA. These registers are used to define run-time parameters, which can change from one kernel invocation to another. They specify:

- The address and length of the input data to be processed by the kernel running on the CGRA.
- The address and length of the output data buffer where results have to be delivered by the kernel executed on the CGRA.
- The number of required loop iterations.
- Possible kernel invariants.

Step 2: After setting the value of these registers, the *ACCEL* instruction with the correct configuration ID is called by the core. Consequently, an acceleration request signal is raised from the corresponding core and forwarded with the configuration ID to the CGRA Controller through the Synchronizer (see Figure 2.7). In the meantime, the core is clock-gated and the acceleration request received by the CGRA Controller is stored into a request queue implementing a FIFO priority (see Section 2.3.1.4).

Step 3: For each request popped from the queue, the CGRA Controller reads two configuration words (Preamble #0 and #1) from the Configuration RAM. The first word (Preamble #0) is used to address and specify the length of the kernel configuration. The second one (Preamble #1) specifies the resources (RCs and CGRA columns) required to accelerate the desired kernel. These two words are temporarily stored in local registers until **Step 5**.

Step 4: Before forwarding the request signal to the CGRA mesh, the CGRA Controller searches for a number of free RCs column(s) equal to the ones required by the desired acceleration. Based on a local representation of the current CGRA execution state, this work is achieved in three sub-steps:

a) The CGRA Controller checks if this request was already mapped on the CGRA. To do so, the controller considers the required amount of RC columns forming a kernel (which is provided by the configuration word #1), and tries to find contiguous available RC columns already configured with the requested configuration ID.

b) If this search step fails, the CGRA Controller tries to find the first set of contiguous RC columns never used (i.e., never allocated/configured since platform startup).

c) If all the RC columns have been used at least once, the CGRA Controller tries to find the first set of contiguous RC columns not in use (i.e., not currently used by a kernel on the CGRA). If insufficient resources are available, the execution is stalled and waits until the release of some RC columns by other kernels.

When available RC columns have been identified, a column index (i.e., configuration location) is determined, specifying the location where the kernel or acceleration has to be mapped on the CGRA. This index corresponds to the first RC column of the kernel mapping on the CGRA. The other column indexes are determined by knowing the total amount of columns required by an acceleration (as specified by configuration word #1).

Step 5: When the CGRA Controller finds suitable RC columns to execute the kernel, it sends an acceleration request signal to the CGRA, along with the acceleration ID, the retrieved location and the previously read configuration words.

Step 6: Then, the CGRA sends an acknowledgment (ACK) signal to inform the CGRA Controller that the acceleration requests has been received and taken into account.

Step 7: The CGRA fetches (if necessary) the remaining configuration words of the accepted acceleration from the Configuration RAM, to program the RCs of the used columns.

Step 8: When the kernel is ready, the execution starts. The input and output data processed by the kernel are read or written from/to the data memory through the Direct Memory Access component (DMA), so that they can be used by the requesting core(s) after the kernel completion.

Steps 9 & 10: Finally, resources in the local state representation of the CGRA are released, and a wake up signal is sent through the CGRA Controller and the Synchronizer, to the cores which have previously requested the acceleration, thus allowing them to continue their execution.

As shown in this presentation of the acceleration request execution flow, the CGRA accelerator is located at the heart of the proposed platform. In the following sections, further details are provided regarding its optimized architecture and the methodology employed for the selection of the accelerated kernels.

2.3.2 Domain-Specific Shared CGRA

2.3.2.1 CGRA Hardware Architecture

Similarly to FPGAs, CGRA architectures are structured as a two-dimensional mesh of Reconfigurable Cells (RCs), tightly interconnected with each other. The structure of the RCs differentiates CGRAs from FPGAs: while the latter can perform any boolean function of the input data, thus providing bit-level flexibility, the functionality of RCs is defined at the operation level, by embedding a dedicated Arithmetic and Logic Unit (ALU) coupled with a small local register file. This arrangement allows CGRAs to efficiently execute Data Flow Graphs (DFGs), extracted from loop-intensive code segments (i.e., kernels), in a spatial way [97].

As opposed to fixed-function ASICs, CGRAs can be easily re-programmed at run-time. Their configuration overhead, as well as the area devoted to the configuration logic, is orders-of-magnitude smaller than that of fine-grained FPGAs, as only the desired ALU operations and the routing of operands must be specified for each cell (see Section 2.2). Hence, only a short configuration interval is sufficient to provide this information [105]. Multiple operations can be cyclically performed by each cell on the mesh [106], by providing a set of configuration words and activating the proper one at each clock cycle, during execution.

CGRAs are particularly effective in the acceleration of loops. Their structure allows to partially overlap the execution of different iterations, a technique termed modulo-scheduling and initially developed for Very Long Instruction Word (VLIW) processors [107]. The speed-up obtained by modulo-scheduling loops is determined by the achieved initiation interval, which measures the difference, in clock cycles, between the start of two subsequent loop iterations. The achievable initiation interval for a loop is limited by the amount of RC resources, as well as by the presence of loop-carried dependencies [97].

Similarly to [108], the employed CGRA architectures are composed of 16 RCs, organized in a 4×4 mesh and connected by nearest-neighbor links in a torus configuration. The mesh size

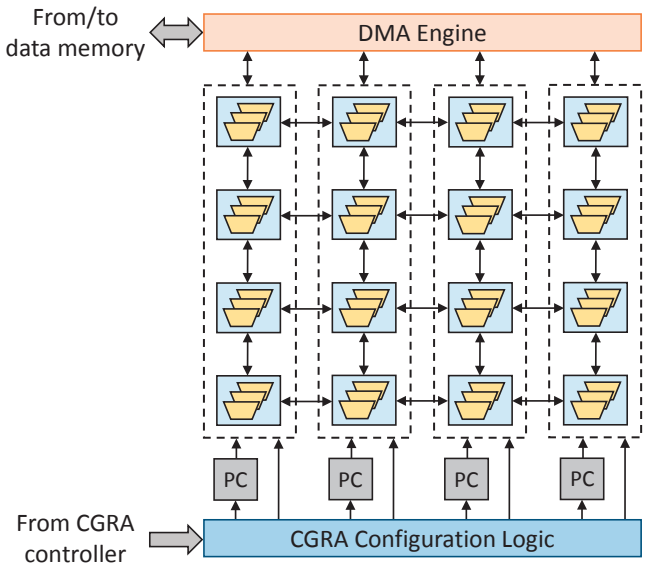


Figure 2.12 – Block scheme of the SIMD-CGRA.

represents a tunable design parameter which can be adjusted at design time, depending on the amount of computing resources required. Figure 2.12 provides a high-level view of the mesh, while Figure 2.13 details the structure of the RCs. A distinguishing feature of the reconfigurable arrays studied in Sections 2.5 and 2.6 is that, as opposed to state-of-the-art CGRAs, two of the three CGRA architectures proposed in thesis, feature multiple datapaths in each cell. In this way, single-instructions/multiple-data execution can be efficiently supported at the RC level. Datapaths (DPs) are composed of an ALU, a local register file, and the multiplexers required to select the input operands (from the register file, or from the output of its own ALU, or from those of neighboring RCs).

The ALU can execute arithmetic operations (addition, subtraction, multiplication), arithmetic and logic shifts, and bit-wise operations (AND, OR, XOR). In addition, execution branches are supported by if-conversion, i.e., both branches of an “if" statement are executed, and the correct result is then selected based on the outcome of the test condition. To do so, 1-bit flags (negative, zero, and overflow) are generated by arithmetic and shift operations. These flags are routed along with data, and used by a dedicated multiplexer (MUX) operation. Finally, one ALU per column can perform square root operations. These operations are required by biomedical applications performing the Root-Mean-Square (RMS) combination of multiple signals into a single one (see Section 2.4.2.1-C).

As illustrated in Figure 2.13, in the case of SIMD-CGRA architectures (see Sections 2.5 and 2.6), the DPs belonging to the RCs share the same set of configuration words. Therefore, each of these instances of the same kernel can operate in parallel on its own dataset. Instances of the same kernel called by different cores are mapped onto different DPs inside the RCs. As kernel instances do not exchange data, no links are present between DPs in a cell, but only among corresponding DPs in neighboring RCs.

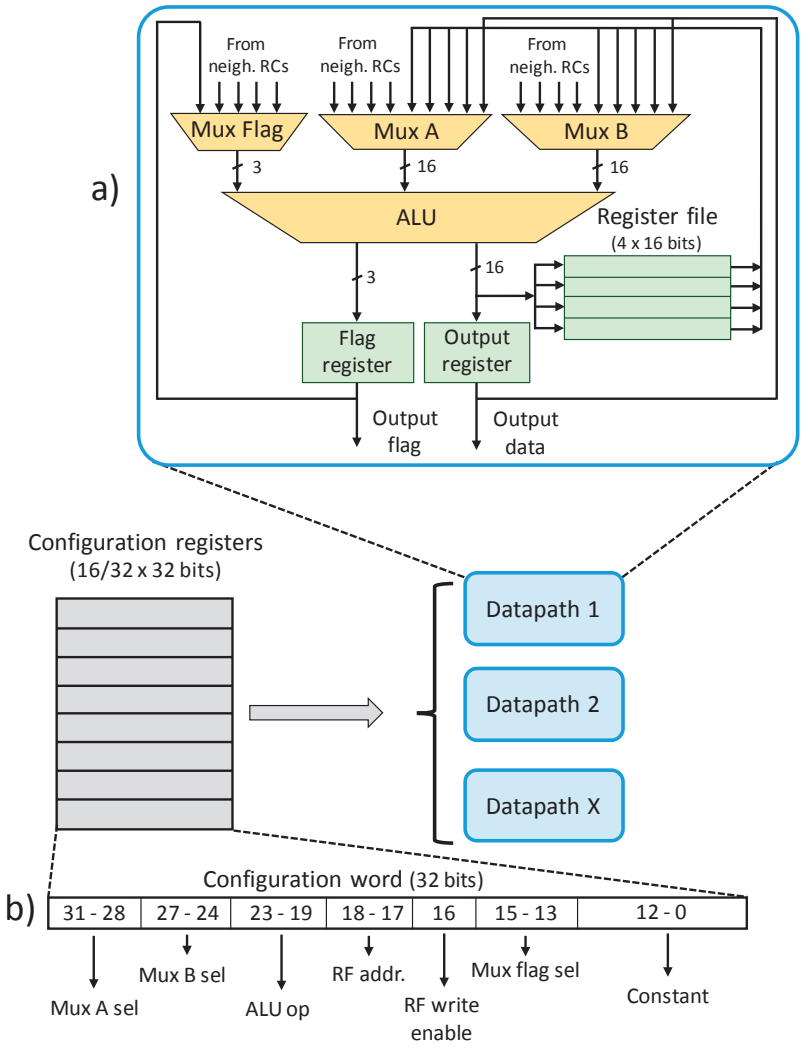


Figure 2.13 – Architecture of the SIMD-CGRA RCs, highlighting a) the datapath structure, and b) the format of the configuration words. All datapaths in the RC receive the same configuration word at each clock cycle.

At run-time, the functionality of the DPs of an RC is dictated by its active configuration word, selected among the ones stored in its local configuration registers. Column-wise Program Counters (named PCs in Figure 2.12) are in charge of this selection at each clock cycle, so that different kernels can be concurrently mapped on each column of the SIMD-CGRA architectures.

At the periphery of the reconfigurable mesh, a multi-channel DMA block is in charge of providing the interface toward the system data memory. Such transfers operate using the data memory ports of the processors that requested the execution of the kernel, therefore they do not require extra read and write connections toward the memory subsystem.

2.3.2.2 CGRA Execution Flow

When an acceleration request is issued to the CGRA, each kernel mapped on its mesh requires a configuration and an execution phase. During configuration, the parameters of the kernel invocation (such as the addresses of input and output data in memory and the total number of iterations) are retrieved from the issuing cores and are employed to configure the program counters of the dedicated RC columns and the required DMA ports. The corresponding configuration words are then transferred from the appropriate CGRA Configuration RAM location to the Configuration Registers (CR) of each cell. Kernel mappings may have different lengths, so they may be expanded on multiple RC columns, up to the entire CGRA mesh (i.e., 4 columns in this case). They can also be invoked with a varying SIMD width ranging from one (i.e., no SIMD) to the number of DPs available in each cell. If the SIMD width is bigger than the DP width, the kernel invocation is split into multiple instances, mapped on different CGRA regions. If instead the SIMD width of the kernels is less than the number of DPs per cell, unused DPs are power-gated to save static and dynamic energy.

Once mapped, the modulo-scheduled [107] kernel is activated and the RCs compute the desired output data, which are stored by the DMA engine in the system data memory. To avoid a multi-ported implementation of the CGRA configuration RAM, and a replication of the configuration logic, a single (possibly SIMD) kernel is configured at a time. Execution of different kernels can instead proceed concurrently on separate CGRA columns, effectively employing the available RCs. Finally, the termination of a kernel execution is notified to the requesting core, which can then resume its execution after the call to the acceleration request.

So far, this chapter has presented the different architectural blocks that compose the proposed heterogeneous and reconfigurable system. Apart from the hardware design of the accelerated computing platform, embedded systems designers need also to identify the computationally-intensive code segments which contribute considerably to the overall execution time of the application. To this end, the following section presents the methodology adopted for selecting candidate kernels to accelerate on the different CGRA architectures developed in this work.

2.3.2.3 Kernel Identification, Selection and Scheduling

In this research work, the identification of the computationally-intensive kernels from biomedical applications is performed by means of profiling passes [109] developed on top of the LLVM infrastructure [110]. Kernels are selected among the most frequently executed basic blocks of the application. Candidate kernels with specific control or data flow structures that cannot be executed on the CGRA are discarded during the analysis.

In fact, hardware- and software-level rules have been established to specify the basic requirements for selection of eligible kernels to accelerate. These rules are summarized below.

Kernel Selection Rules

a) The search for candidate kernels to accelerate starts from the most frequently used and then progressively goes down through the list of candidate kernels, toward the ones executed less often at run-time. A maximum number of kernels to select must be defined to limit memory footprint and the size of the resulting CGRA Configuration RAM.

b) A fixed number of private memory-mapped registers can be read or written by each core from the software application, to configure the dynamic parameters of each acceleration request (see *Step 1* in Section 2.3.1.6). If the number of registers is not sufficient for a specific kernel, designers have two options: either discard the candidate kernel or increase the number of memory-mapped registers at the cost of higher energy, area, and software execution time overheads. To minimize these overheads, a reasonable number of registers must be chosen at design-time for being able to configure most of the kernels to accelerate.

c) The proposed CGRA meshes are composed of a fixed amount of RCs (specified at design time) with a torus configuration, allowing only data/flags propagation between contiguous RCs. The mapping/scheduling of the candidate kernels must be compliant with this arrangement for being selected.

d) The inner-most loop of a processing block can be considered as a candidate kernel to accelerate. The outer loop can also be selected as a kernel, only if it is possible to unroll the inner loops into a reasonable amount of instructions supported by the limited CGRA resources (e.g., RC columns, configuration registers, and register files).

e) Each proposed CGRA architecture has only one data port per RC column. Hence, two RCs of the same column cannot read or write data at the same clock cycle. The mapping/scheduling of the candidate kernels must support this restriction.

f) Data memory allocation, dynamic address calculations and accesses from inside the kernel are not allowed. In fact, the processing parameters of the input data vector (e.g., start, length, and frequency of access per element) have to be pre-determined before the execution of the kernel.

g) The number of allowed kernel loop iterations is limited by the design of the RCs. With the employed RCs, a loop counter of 20 bits is used, which limits the number of iterations to $2^{20} - 1 = 1048575$. However, this is more than sufficient for the employed kernels of this study.

Kernel scheduling on CGRA meshes is not straightforward, as operations must be assigned to spatially distributed components (i.e., RCs), as well as to a temporally defined execution cycle. For the experimental evaluations presented in Sections 2.4.2, 2.5.2, and 2.6.2, this task was performed manually, mimicking the automated modulo strategy described in [97]. Other CGRA scheduling algorithms include [108, 111, 112, 113].

A possible framework extension is the full automation of this step, by including the aforementioned rules and by adapting existing instruction set extension identification techniques, such as the ones proposed by [114] and [115]. A pass built on LLVM is also used to derive the Data

Flow Graphs (DFGs) of the kernels, which are then modulo-scheduled on the CGRA. In this way, it becomes possible to derive the number of columns and RCs required by the kernels, the operations performed at each clock cycle during a kernel execution, the number of cycles per iteration, and ultimately the content of each configuration word.

Once the identification, selection, and scheduling of the candidate kernels have been performed, it becomes possible to evaluate at the system level, their contributions in terms of execution speed improvements and energy savings.

In order to evaluate the proposed platform, the following sections introduce the different CGRA datapath architectures. As shown during their experimental evaluations, they showcase different performance, depending on the size of the kernels to accelerate and on the level of contention between the different cores trying to access the shared CGRA. In addition, the benefits and drawbacks of each CGRA datapath architecture are presented and compared to a non-accelerated computing system.

2.4 Single-Datapath CGRA Accelerator Shared by a Multi-Core System

2.4.1 Overview

The CGRA architecture introduced in this section integrates all the hardware and software components presented in Section 2.3. However, it only features a Single processing DataPath *(Single-DP)* per RC. In other words, a single kernel can only be processed in one or several RC columns of the CGRA mesh at a time. Nonetheless, the remaining free columns of the CGRA can be used for the execution of other kernels in parallel, allowing a shared utilization of the mesh at run-time.

This single-DP architecture has the main advantage of being the simplest one, with the minimum area and energy overheads. In this thesis, it is considered as the *baseline CGRA architecture*, systematically compared to the more advanced CGRA designs proposed in this chapter. The resulting execution speed-up and energy savings provided by this CGRA architecture at the system level are evaluated with a moderate application workload and in comparison with the *baseline multi-core system*. The experimental setup and the obtained results are presented in the following experimental evaluation.

2.4.2 Experimental Evaluation

2.4.2.1 Experimental Setup

This section first presents the targeted and baseline systems employed in this experimental evaluation. Then, the architectural parameters of the single-DP CGRA-based platform

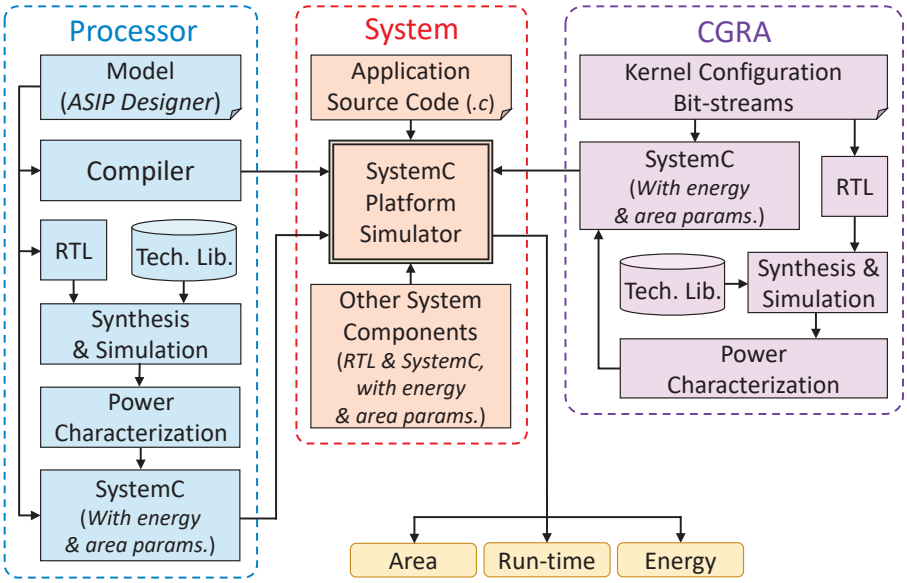


Figure 2.14 – Block scheme of the experimental framework, comprising RTL and cycle-accurate system views.

are introduced, followed by a presentation of the framework developed to investigate its performance. Finally, the employed benchmarks and the accelerated kernels are described.

A) Target and Baseline Systems

As seen in Section 2.1.1, previous studies have showcased that multi-core architectures achieve higher energy efficiency than single-core ones when performing bio-signal applications. Hence, inhere a homogeneous multi-core system is considered as a baseline solution. The energy consumption and performance of this system are evaluated in comparison with the ones of the single-DP CGRA-based platform (i.e., multi-core system interfaced with a single-DP CGRA accelerator). In more detail, this section focuses on two different architectural configurations, described below.

- **Multi-core architecture only:** No CGRA accelerator is employed, and the software applications are entirely executed by the processors (*SW Only*). This system is similar to the one described in [14]. It supports SIMD execution to reduce accesses to the instruction and the data memories. A comparative evaluation with respect to this baseline therefore explores the efficiency of a shared reconfigurable acceleration resource in the WBSN digital signal processing domain.
- **Multi-core architecture with single-DP CGRA:** Processors are interfaced with a CGRA having only one DP per RC, as presented in Section 2.4.1.

B) Hardware Architecture and Simulation Parameters

To evaluate the energy and performance benefits of the single-DP CGRA-based platform, a hybrid framework was developed. It comprises both a post-synthesis and a (higher-level)

cycle-accurate view of the system. Its block scheme is presented in Figure 2.14.

The RTL implementation, cycle-accurate model and compiler of the TamaRISC processors (see Section 2.3.1.1) were defined using Synopsys® ASIP Designer [116]. The gate count (12 kilogates) and the energy-per-cycle of the processors is comparable to that of the low-power ARM Cortex M0 architecture [28].

The CGRA accelerator was instead designed from the ground up as an RTL template, allowing the definition of architectural parameters through configuration directives. As explained in Section 2.3.2, the mesh comprises 4 rows and 4 columns of RCs. In all the considered experiments of this chapter, this mesh size suffices to map the biggest kernels using 4 columns. The run-time behavior of the CGRA was also modeled in the SystemC simulator. A similar strategy was followed for the other system blocks, such as memories and interconnects.

The complete system includes 8 processors, a data memory of 64 kiloBytes (32 kilowords of 16 bits) and an instruction memory of 96 kiloBytes (32 kilowords of 24 bits). The Configuration RAM has a size of 6 kiloBytes (1.5 kilowords of 32 bits), which is sufficient to store all the configurations of the considered kernels. Each RC has 16 configuration registers, while each DP has a 4-word local register file. From the software side, 5 memory-mapped 16-bit registers are reserved for each core, to configure the dynamic parameters of each acceleration request.

Furthermore, the design of embedded biomedical devices traditionally relies on large and cheap transistor technologies (e.g., 90 nm [117] or 130 nm [118]) characterized by a high energy consumption and area footprint. However, as mentioned in introduction, the primary goal of the proposed architectural exploration is to improve the energy-efficiency of the envisioned platform beyond state-of-the-art multi-core processing systems. To this end, a more advanced technology (i.e., 65 nm UMC low-leakage [119], Slow-Slow (SS) process corner) has been considered to further reduce the power consumption of the design, compared to systems implemented with older semiconductor technologies. In fact, even if the leakage power is higher with more advanced technologies, the absolute energy consumption of the circuit is lower, since the dynamic energy consumption (resulting mainly from memory accesses) represents the biggest portion of the total system energy consumption.

Additionally, thanks to the higher computing performance achievable by newer technologies, strategies such as the one presented in [38] can be explored to execute at high frequency the processing of each bio-signal sample during a short period of time, and then power-gate the complete device before the arrival of the next processing period. Despite the energy benefits provided by this solution, it has not been selected in this study, since it requires Non-Volatile Memories (NVMs) to be able to power-gate the memory banks without losing the data from one processing period to another. Instead, a reduced operating frequency of 1 MHz has been considered to perform the processing of the benchmarks in real-time with low energy expenditures.

Moreover, the nominal voltage (i.e., 0.9 V) of the employed design kit has been used to supply both the logic and memory components of the platform. A strategy to save energy consists of lowering this supply voltage to minimize the energy consumption of the whole platform. However, this solution requires a comprehensive system reliability exploration to evaluate the impact of run-time errors at low voltage levels, which is not the scope of this chapter. Nonetheless, this solution has been explored in joint research works [56, 65], to complete the architectural exploration performed in this work.

As concerns the silicon area occupied by the platform, numbers were directly derived from the synthesized netlist of the system, integrating the single-DP CGRA mesh. To accurately measure the energy consumption of the resulting platform, long logic simulations would be required to collect the switching-activities corresponding to the processing of entire ECG signal windows. Such approach would lead, in turn, to time-consuming simulations at the post-synthesis level which would make the space exploration unfeasible. Instead, post-synthesis simulations were used on a smaller scale, only to perform the energy characterization of the different system blocks. To this end, similarly to the approach in [14], the performance of the multi-processor architecture was evaluated when executing small synthetic benchmarks, both with and without SIMD execution (using Synopsys® Design Compiler [120] and Mentor Graphics® ModelSim [121]). In addition, the execution of all kernels on the single-DP CGRA was simulated at the post-synthesis level.

Through this analysis, detailed static and dynamic energy profiles were derived for each of the system computational components (processors, CGRA RCs) when in active and sleep mode. The obtained reports also include the power required for all the memory accesses. The behavior of the system components while in sleep mode depends on whether their state must be retained or not across idle periods. DPs can be safely power-gated when not in use, because values residing in the CGRA local register files are not used after a kernel execution. On the other hand, CGRA Configuration Registers (CRs) are clock- (but not power-) gated, so that their content can be reused by a further invocation of the same kernel. Processors are also clock-gated when idle, since the content of their internal registers must be preserved at run-time.

The energy profiles, along with the run-time required for the execution of the CGRA kernels, were imported in the cycle accurate (SystemC) simulator, which allows much faster experimental evaluations. Using the simulator, different metrics were gathered such as the number of active or inactive clock cycles for each computing units (i.e., processors and CGRA RCs), and data/instruction memory accesses. By combining the information provided at each abstraction level, the performance metrics of the system were obtained over long simulated periods of time. At the cycle-accurate level, each benchmark was executed for a total of 10 seconds, corresponding to processing 5000 ECG samples extracted from the T-Wave Alternans Challenge database [29].

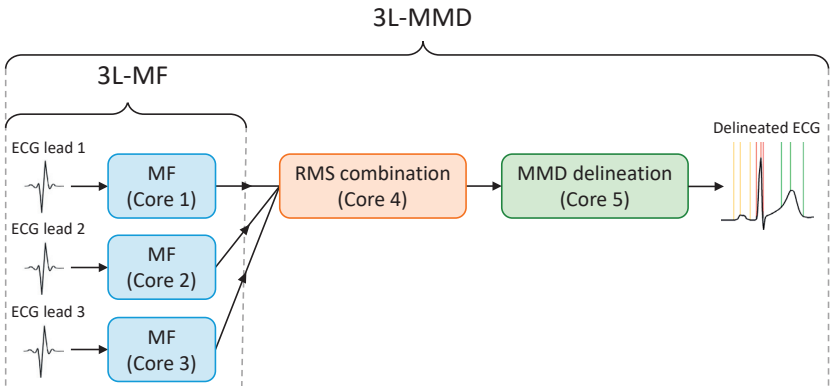


Figure 2.15 – Task graph of the 3L-MF and 3L-MMD benchmarks.

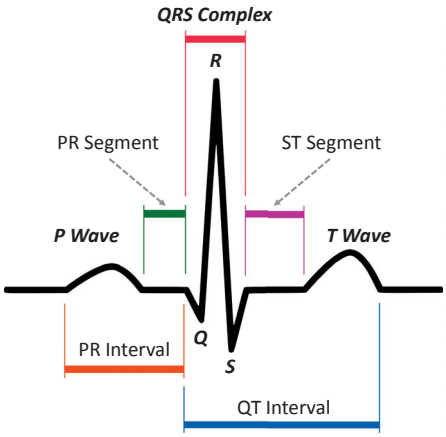


Figure 2.16 – ECG fiducial points of a normal heartbeat.

C) Biomedical Benchmarks

Experimental evidence was gathered for the considered platforms when processing ECG records acquired with a sampling rate of 500 Hz per lead, common in high-quality ambulatory recordings [29]. Three complex cardiac processing routines are evaluated as benchmark applications.

- **3L-MF:** Three-lead Morphological Filtering (MF). Removes artifacts (due to muscles activity, system AC supply interferences and base drift caused by breathing) from an ECG acquisition, using structuring elements to remove low- or high-frequency noise components, according to the algorithm described in [77]. This benchmark operates in parallel on three different input streams and is therefore mapped on three cores (see Figure 2.15, first blocks).
- **3L-MMD:** Three-lead delineation using Multi-scale Morphological Derivatives (MMD) [122]. It detects the characteristic fiducial points (P, Q, R, S, and T, as depicted in Figure 2.16) of each heartbeat. This application relies on 3L-MF to properly filter the

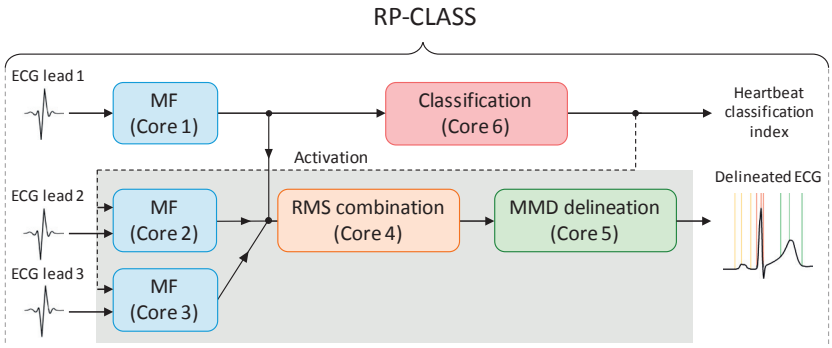


Figure 2.17 – Task graph of the RP-CLASS benchmark. The shaded part is only activated when an abnormal heartbeat is detected by the classifier.

acquired signals, then performs a RMS combination of the filtered streams, and finally the delineation of the fiducial points. It is divided into three different tasks mapped on five processing cores, as shown in Figure 2.15.

- **RP-CLASS:** Uses a heartbeat neuro-fuzzy classifier, operating on a single-lead, to identify pathological heartbeats, by applying a random projection over the heartbeat samples [123]. When a heartbeat abnormality is detected, a three-lead delineation is activated for a short window of signal. RP-CLASS is mapped onto six cores (see Figure 2.17), among which the four cores in the delineation chain are seldom activated.

D) Accelerated Kernels

The computational kernels of the considered applications were identified with the approach described in Section 2.3.2.3. The profiler highlighted seven kernels out of fifteen, as the most promising candidates to be accelerated on the CGRA:

- ***Dbl Min Srch*** performs a search of the first and second minimum values inside a window of samples. This kernel is used by the cores running the MF algorithm.
- ***Dbl Max Srch***, similarly to *Dbl Min Srch*, performs a search of the first and second maximum values inside a window of samples. This kernel is used by the cores running the MF algorithm.
- ***Min Max Srch*** determines the minimum and maximum limit values used during the *erosion* and *dilation* steps [77], executed by the cores running the MF algorithm.
- ***Sqrt 32*** executes a 32-bit square root algorithm. This kernel is used by the core performing the RMS combination of the filtered ECG signals.
- ***Lin Srch*** performs a linear search of the two minimum values and two maximum values inside an array of samples. Its execution is more compact than the one of the kernels *Dbl Min Srch* and *Dbl Max Srch* executed sequentially. This kernel is used by the core executing the RMS combination of the filtered ECG signals.

- ***Apply RP*** calculates the random projection [123] on a signal window, by performing a matrix-vector multiplication. This kernel is used by the core executing the classifier.
- ***Lin Min Max*** performs a linear search of a single minimum value and single maximum value inside an array of samples. This kernel is used by the core performing the classification task.

In order to underline the sources of possible contentions when trying to access the computing resources from the CGRA mesh, Table 2.1 shows for each application, which kernels are accelerated, and how many cores request them (possibly at the same time). The effect of these resource conflicts are studied in the following section.

Table 2.1 – Kernel utilization per benchmark application. Each cell indicates the number of processing cores requesting each acceleration.

Kernels \ Apps	3L-MF	3L-MMD	RP-CLASS
Dbl Min Srch	3 cores	3 cores	3 cores
Dbl Max Srch	3 cores	3 cores	3 cores
Min Max Srch	3 cores	3 cores	3 cores
Sqrt 32	-	1 core	1 core
Lin Srch	-	1 core	1 core
Apply RP	-	-	1 core
Lin Min Max	-	-	1 core

2.4.2.2 Experimental Results

To investigate the benefits deriving from the single-DP CGRA accelerator shared by a multi-core system, this section starts by analyzing the obtained performance at the kernel level and the resulting savings from both run-time and energy perspectives.

As shown in Figure 2.18, execution on the CGRA mesh achieves speed-ups ranging from 1.6x to 11.0x while executing the selected kernels, compared to a software-only (i.e., multi-core only) alternative. The reported results account for resource conflicts, which arise when several concurrent requests cannot be allocated at the same time on the limited single-DP CGRA resources. When a resource conflict occurs (i.e., not enough RC columns to execute the kernel on the CGRA), the acceleration request stays into the request/FIFO queue. Its execution is delayed until the release of CGRA computing resources (i.e., RC columns). Throughout this process, the core that has issued the request will remain clock-gated (see Section 2.3.1.6). Therefore, no dynamic energy is lost during resource conflicts. However, performance degradations are observable for each kernel and for the whole application, since kernel accelerations will take more time to be executed (see Section 2.5.2.2). With the chosen benchmark applications and CGRA size, the amount of resource conflicts remains relatively low, allowing outstanding execution speed-ups for all the kernels.

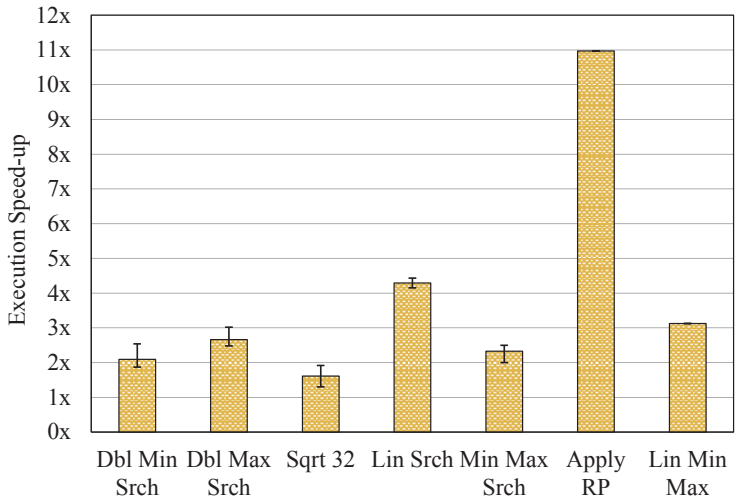


Figure 2.18 – Speed-ups of kernels running on the single-DP CGRA mesh with respect to their software execution. These results have been averaged with all the kernel invocations performed by the different benchmark applications (see Table 2.1). Hence, an error bar is depicted for the kernels having a non-negligible speed-up variation.

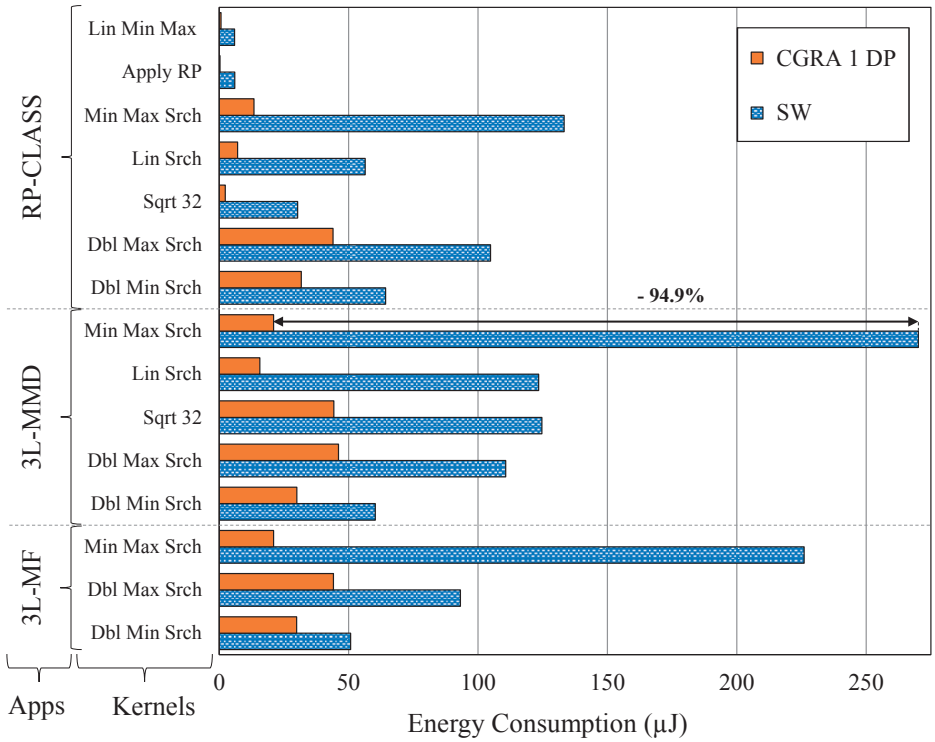


Figure 2.19 – Energy consumed by the different kernels employed in the considered benchmarks, when executed on the accelerator (CGRA 1 DP) and on the processing cores (SW).

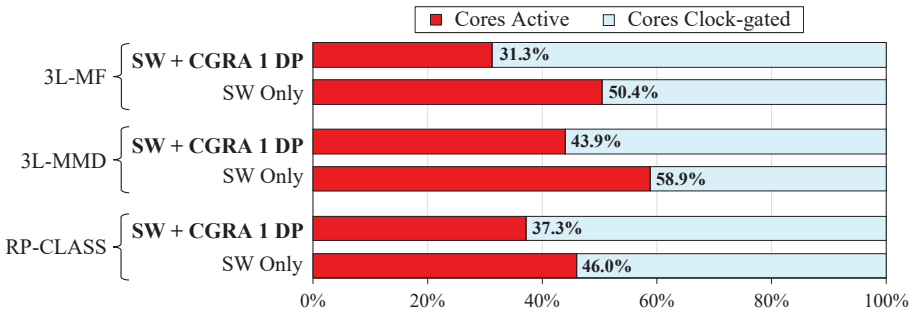


Figure 2.20 – Multi-core utilization time with and without CGRA acceleration (% of total run-time).

The considerable time reductions achieved are coupled with a superior energy efficiency of the CGRA unit, which is represented in Figure 2.19. This figure compares the total energy (i.e., dynamic + static energy) consumed by executing the selected kernels on the multi-core system and on the CGRA. It shows that by accelerating the kernels on the CGRA it is possible to achieve energy savings of up to 94.9 % when compared to a software-only execution (i.e., kernel execution on the processor without CGRA support). Moreover, an average energy reduction of 73.3 % is achievable for all kernels of the different applications.

These energy savings are inherent to the internal CGRA architecture. In fact, the decomposition and execution of the kernel's operations over multiple RCs in parallel, is more energy-efficient than the sequential execution of the kernel's instructions by the different pipeline stages of the processor. As mentioned in Section 2.3.2, the CGRA is tailored to the processing of loop-intensive code segments, composed of a set of operations repeated several times (i.e., for each loop iteration). Once configured for a specific kernel, the RCs operate independently of the Configuration RAM, throughout the kernel execution on the CGRA. Hence, only data memory accesses are performed between the CGRA and Data Memory (DM), minimizing the dynamic energy consumption during kernel executions. Conversely, when a processor executes a kernel, an instruction must be fetched from the Instruction Memory (IM), decoded and executed for each operation. This leads to a significant energy consumption, compared to a kernel execution on the CGRA mesh.

Secondly, at the system level, the CGRA acceleration of just few kernels per application results in a sizeable reduction of the active times of processors (as highlighted in Figure 2.20), leading to an increase in overall energy efficiency of the platform. As mentioned previously, by executing kernels on the CGRA, not only the dynamic energy of the cores is decreased (i.e., less instructions to execute), but also fewer accesses to the Instruction Memories (IMs) are required by the processors. For instance, in the case of the 3L-MF benchmark, this effect is particularly noticeable, as the active time of cores is reduced from 50.4 % to 31.3 %.

The resulting energy savings are detailed in Figure 2.21, which provides the consumption breakdown of the multi-core system with and without the single-DP CGRA accelerator for

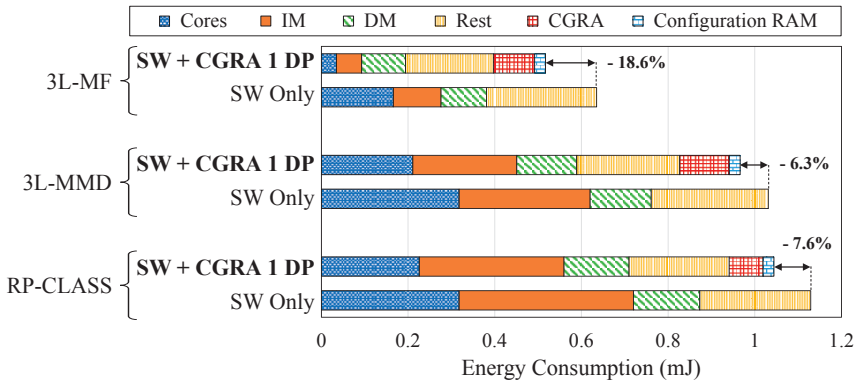


Figure 2.21 – System energy consumption for the different applications, while executing on the multi-core platform with and without CGRA acceleration. NB: In the legend, the *Rest* category encompasses the energy consumption of the crossbar interconnects, clock-tree and synchronizer.

the three investigated benchmarks. The comparison showcases that the energy consumed by the cores and the instruction memory is reduced by a large margin when the CGRA is employed, as a considerable part of the applications' workloads are outsourced to the CGRA and a much smaller amount of instructions are fetched at run-time. For all applications, significant reduction in energy consumption can be also seen in the rest of the system components, which greatly compensates the energy overhead associated to the inclusion of CGRA accelerator, resulting in a decrease of the overall energy budget, including both dynamic and static consumption, of up to 18.6 % (for the 3L-MF case).

Furthermore, the energy consumed by the instruction and data memories represents a significant part of the overall platform's consumption. This is even more noticeable in the case of the 3L-MMD and RP-CLASS benchmarks, as depicted in Figure 2.21. By employing complementary memory optimization techniques [38, 124], these energy expenditures can be further reduced, which in turn will increase the relative energy gains provided by the CGRA, compared to the software-only execution.

Moreover, with the current experimental setup and thanks to the selected low-leakage cell library (see Section 2.4.2.1-B), the dynamic energy consumption measured in this study is always dominant compared to the static one. Therefore, the CGRA-based computing system proposed in this section and in the following ones (see Sections 2.5 and 2.6), aim at minimizing the largest energy portion consumed by the platform. Nonetheless, the static energy consumption of this platform is also reduced by power-gating the unused datapaths of the RC columns (see Section 2.4.2.1-B).

Regarding the introduced area overhead of the proposed accelerator, the integration of a single-DP CGRA (with all its additional components) represents 29.5 % of the total platform area. Further details about area breakdown of the accelerated platform are provided in Section 2.5.2.2.

Two improvements to the single-DP CGRA architecture are described in the next section of this chapter. This enhanced CGRA architecture is able to concurrently minimize the area/energy envelope of the mesh and increase its performance, by featuring multiple DPs per RC, governed by the same control logic.

2.5 Multi-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

2.5.1 Overview

The main drawback of the single-DP CGRA architecture presented in Section 2.4 concerns its limitation to handle efficiently multiple acceleration requests to the same kernel in parallel. As a consequence, the access contention to the shared CGRA increases steeply when additional bio-signal channels are processed concurrently by the application (see Section 2.5.2.2-A.2). Moreover, the RCs that compose the single-DP CGRA must be reconfigured for each additional kernel executed on the CGRA. In fact, the RCs must be individually reprogrammed since the configuration words are not shared between two identical kernel configurations and executions on the mesh. Hence, duplicated accesses to the Configuration RAM are performed and accompanied with extra energy overheads.

To face these issues, this section investigates the benefits offered by a Multi-DataPath *(Multi-DP)* CGRA, supporting SIMD-kernels execution. By adopting this accelerator architecture, the proposed DSP platform supports SIMD execution modes, both during software execution and hardware accelerations. In this way, the platform leverages SIMD in order to (1) coalesce accesses performed by different processors to the memory subsystem, (2) minimize the number of reconfigurations and energy consumption required to map accelerated loops, and (3) streamline the control logic of the CGRA fabric, whose cells embed multiple datapaths. The resulting SIMD-based CGRA therefore supports the execution of simultaneous acceleration requests (on the same set of RC columns), issued by processors executing in SIMD.

Similarly to a single-DP CGRA, with a multi-DP architecture the same amount of configuration registers is required for each core to dynamically configure a kernel executed on the CGRA (i.e., five configuration registers in this study). To access these memory-mapped registers, the data memory accesses are multiplexed between the different RC columns and for each datapath inside each column. These connections are managed by the DMA component (see Section 2.3.2), while the memory access conflicts are automatically arbitrated by the crossbar interconnect interfaced to the multi-banked data memory (see Section 2.3.1.2). Furthermore, the same memory ports are used by the cores and CGRA to access the data memory. A multiplexer is in a charge of switching the data memory connection from one core to a specific CGRA datapath, when a kernel execution is requested. The connection between the core and data memory can be interrupted because during a kernel execution on the CGRA, the core is

in sleep mode and does not perform any accesses to the data memory. Regarding the kernel assignation to a datapath, it is based on the processor identifier (PID) requesting the kernel execution. An acceleration request with a lower PID is executed on lower Datapath ID (DID) inside the available RC columns.

To evaluate the proposed multi-DP CGRA architecture, three different scenarios are considered, with one, two and three DPs per RC, respectively. The performance and energy efficiency of the three CGRA scenarios interfaced with the multi-core system, are analyzed in the following sections.

2.5.2 Experimental Evaluation

This section proceeds as follows. It starts by introducing the experimental setup employed in the evaluation of the proposed multi-DP CGRA-based platform. Then, the obtained results are discussed by comparing the proposed platform with the baseline multi-core only and multi-core + single-DP CGRA systems.

2.5.2.1 Experimental Setup

A) Target and Baseline Systems

To evaluate the proposed approach, I focused on three different architectural configurations, described as follows.

- **Multi-core architecture only:** Baseline multi-core system without CGRA accelerations (*SW Only*), and supporting SIMD execution modes at the level of the processors. The architecture of this system is based on the work of the authors of [14].
- **Multi-core architecture with single-DP CGRA:** Processors are interfaced with a CGRA having only one DP per RC. Hence, the execution of SIMD kernels is not supported, and SIMD acceleration requests are mapped on different CGRA regions. This setup corresponds to the platform evaluated in Section 2.4 and represents the baseline CGRA architecture.
- **Multi-core architecture with multi-DP CGRA:** Target architecture, as described in Section 2.5.1. It presents multiple DPs in each RC, allowing the efficient acceleration of SIMD-kernels on the same set of RC columns. Two CGRA instances are considered here with two or three DPs per RC.

B) Simulation Framework and Biomedical Benchmarks

In order to be consistent with the experimental evaluation of the single-DP CGRA (see Section 2.4.2), the same experimental framework is employed in this section to evaluate the above-mentioned platforms. However, the application workload of some benchmarks has been increased to stimulate further the utilization of the CGRA mesh shared by multiple cores running in lock-step/SIMD mode. By employing higher application workloads, the experimental

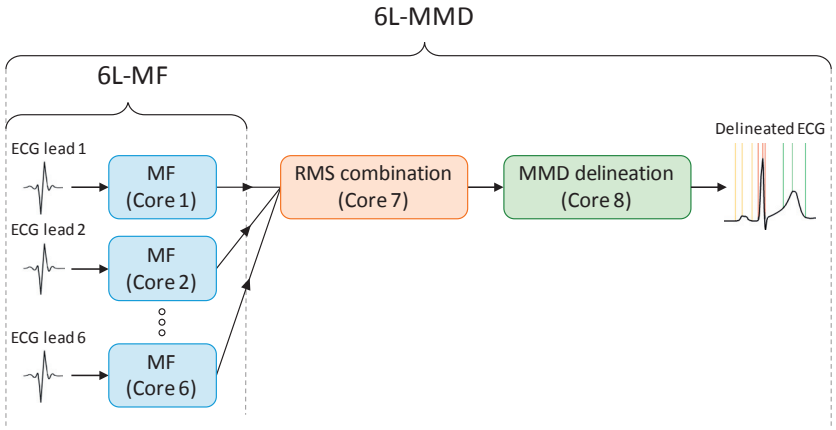


Figure 2.22 – Task graph of the 6L-MF and 6L-MMD benchmarks.

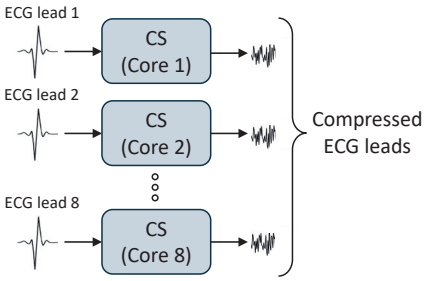


Figure 2.23 – Task graph of the 8L-CS benchmark.

conditions emphasize the need for SIMD-kernels execution capabilities, which are naturally enabled by a multi-DP SIMD-CGRA.

To do so, as depicted in Figure 2.22, the 3L-MF and 3L-MMD benchmarks, introduced in Section 2.4.2.1, have been extended from three to six ECG leads processed in parallel. The tasks mapping on the cores has been modified accordingly, to support respectively six and eight cores for the 6L-MF and 6L-MMD benchmarks.

In addition, a forth benchmark from the bio-signal processing domain is employed in this study: **8L-CS** is an 8-lead Compressed Sensing (CS) algorithm [37], mapped onto eight cores (see Figure 2.23). It applies a 50 % lossy compression on input ECG signal windows of 1024 samples.

All the selected benchmarks present different workloads and computational characteristics: On one hand, the execution of 8L-CS is dominated by a single and intense kernel, which is always executed in SIMD. On the other hand, RP-CLASS (see Section 2.4.2.1) has a more complex run-time behavior and fewer opportunities for SIMD execution. Alternatively, intermediate workloads in-between those two extremes are achieved with the 6L-MF and 6L-MMD benchmarks.

Finally, as concerns the operating parameters of the platform, similarly to Section 2.4.2.1, the operating frequency is set at 1 MHz for all the benchmarks, except for the computationally-intensive 8L-CS benchmark, which operates at 2 MHz, in order to respect the real-time constraint of the system.

C) Accelerated Kernels

The benefits provided by the multi-DP CGRA architecture are analyzed with all the kernels presented in Section 2.4.2.1. Moreover, an eighth kernel named *CS*, from the 8L-CS benchmark, is used to compute the sequence of random indexes necessary to perform the Compressed Sensing. This kernel is used by the eight cores running the CS algorithm, thus generating a significant amount of contention when trying to access the four CGRA columns (one RC column/datapath is required per kernel acceleration).

Table 2.2 – Kernel utilization per benchmark application. Each cell indicates the number of processing cores requesting each acceleration.

Apps / Kernels	6L-MF	6L-MMD	RP-CLASS	8L-CS
Dbl Min Srch	6 cores	6 cores	3 cores	-
Dbl Max Srch	6 cores	6 cores	3 cores	-
Min Max Srch	6 cores	6 cores	3 cores	-
Sqrt 32	-	1 core	1 core	-
Lin Srch	-	1 core	1 core	-
Apply RP	-	-	1 core	-
Lin Min Max	-	-	1 core	-
CS	-	-	-	8 cores

To summarize the possible sources of contention, Table 2.2 presents the number of cores requesting in parallel (or not), each kernel from the employed benchmarks. As shown by this table, more SIMD execution opportunities at the multi-core and CGRA levels, are present by employing these benchmarks, compared to the previous experimental evaluation in Section 2.4.2.

2.5.2.2 Experimental Results

This section showcases the performance of the heterogeneous and reconfigurable system in a bottom-up fashion. First, the performance, area, and energy-efficiency of the multi-DP CGRA is evaluated when executing each kernel individually. Then, a system-level assessment is performed on the overall platform and over all benchmark applications, and is concluded by providing insights on the trade-off implied by different choices for the CGRA SIMD width.

A) CGRA Performance Evaluation

A.1) Kernel Execution Speed-Up: Figure 2.24 shows the speed-ups of the considered kernels when executed on the CGRA, with respect to their execution time on the processors. They range from 0.8x to 11.3x, depending on the kernel structure (e.g., number of iterations, initiation

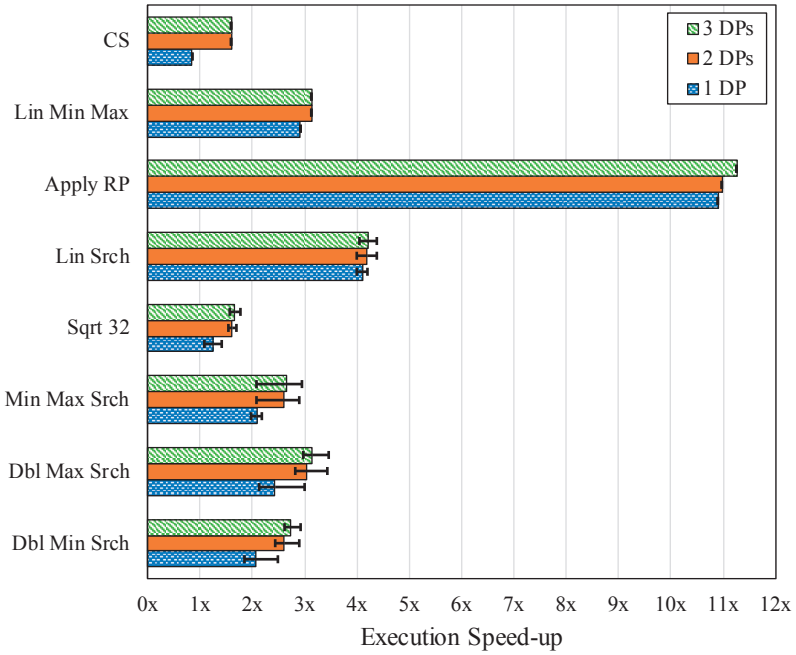


Figure 2.24 – Average speed-up with kernels running on the multi-core + CGRA platform w.r.t. kernels running only on the multi-core platform. These results have been averaged with all the kernel invocations performed by the different benchmark applications (see Table 2.2). Hence, an error bar is depicted for the kernels having a non-negligible speed-up variation.

interval) as well as the number of DPs embedded in the cells of the CGRA mesh (i.e., the CGRA SIMD width). For each kernel, the presented result considers both configuration and execution time, as well as the overhead due to the management of acceleration requests. Since kernels compete for the limited CGRA resources, their speed-up may vary across invocations. Hence, Figure 2.24 provides average, maximum, and minimum speed-up values for the cases having a non-negligible variance. This speed-up variation may originate from the level of contention, but also from size of the data vector to be processed by the kernel from one invocation to another.

In all cases, kernels require a smaller execution time on the CGRA than in the processors. The *CS* kernel on a 1 DP (i.e., single-DP) CGRA is an outlier, as it has a speed-up of 0.8x (a slowdown), due to the high amount of contention during its execution. Such a contention level is substantially reduced by increasing the number of DPs, as SIMD accelerations can be configured and mapped in parallel on the CGRA, ultimately improving run-time performance. In fact, the average execution time decreases for all kernels when a multi-DP CGRA is employed, allowing for instance the *CS* kernel to reach 1.6x speed-up when multiple datapaths are used.

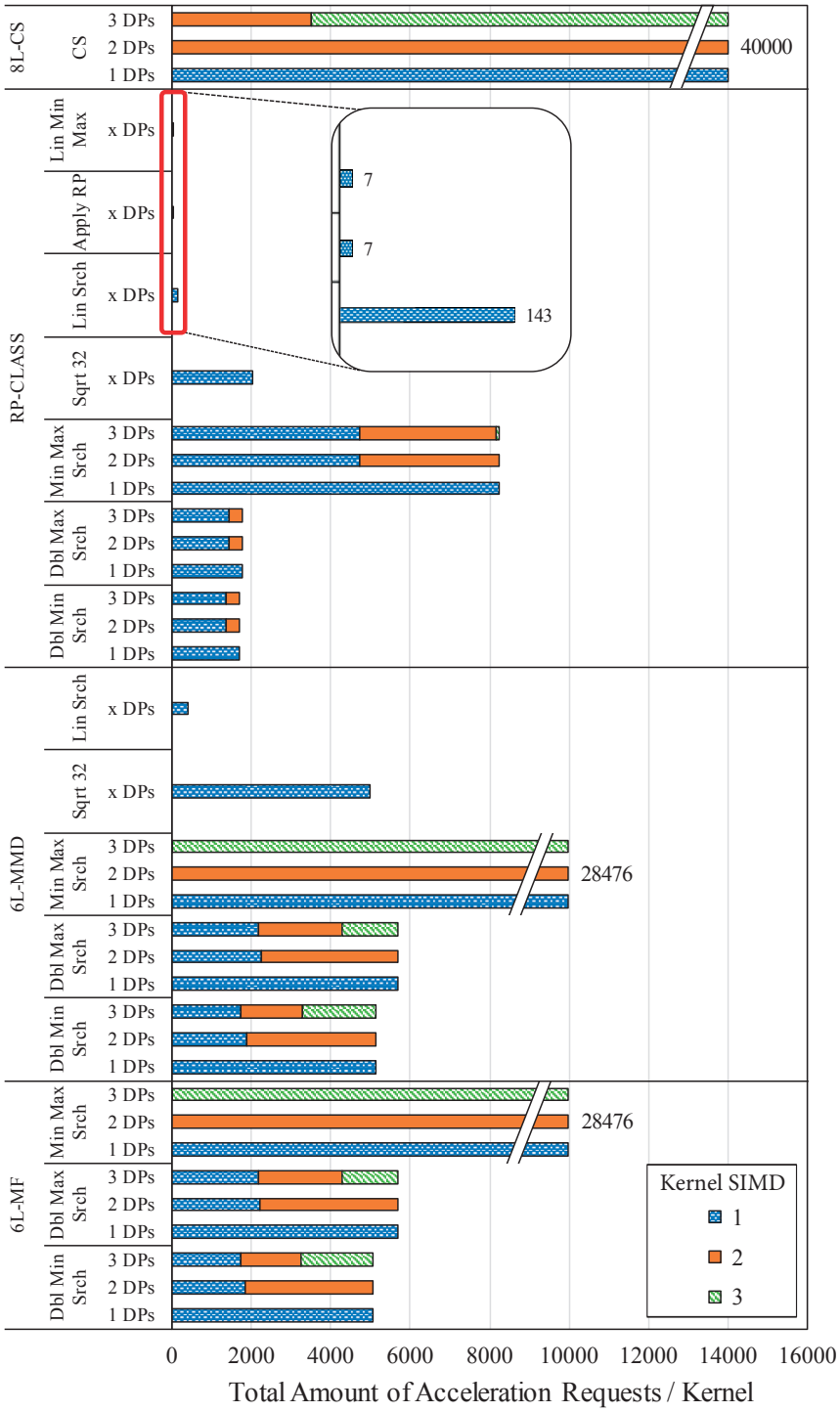


Figure 2.25 – Repartition of kernel acceleration requests types, depending on employed datapath configuration.

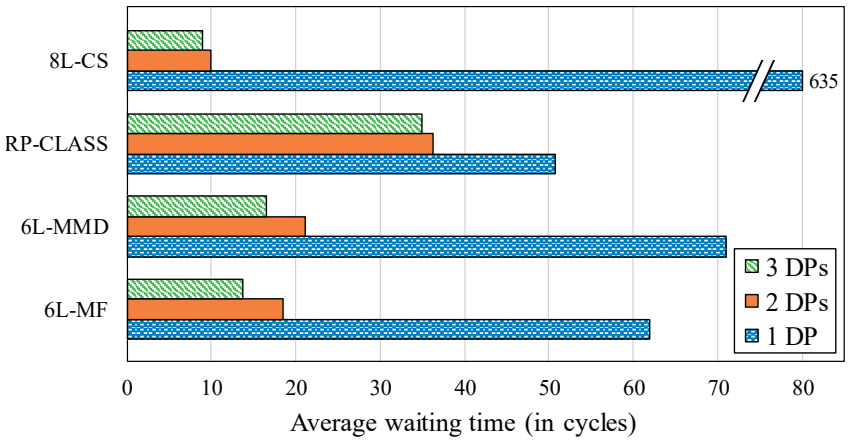


Figure 2.26 – Average waiting time (in cycles) spent for each acceleration request (called by the cores executing the different benchmarks), before being configured and executed on the CGRA.

A.2) Resource Utilization Analysis: By employing multi-DP RCs, concurrent SIMD accelerations can be mapped and executed at the same time on the same set of RC columns. To investigate how much the different kernels benefit from the multi-DP SIMD-CGRA, Figure 2.25 depicts the repartition of the different types of acceleration calls over the execution of the considered benchmarks for single- and multi-DP configurations. In this figure, kernels *Sqrt 32*, *Lin Srch*, *Apply RP* and *Lin Min Max* run on a single core and thus can not take advantage of the SIMD execution mode (see Table 2.2 in Section 2.5.2.1). For these cases, Figure 2.25 only shows one bar that remains identical for the different configurations. On the other hand, kernels *Min Max Srch* and *CS* benefit the most from the SIMD mode, with 100 % of the acceleration requests executed in this mode. This is possible because all the invocations to these kernels are received and mapped at the same clock cycle on the multi-DP CGRA. It showcases a *perfect* lock-step execution of the cores requesting these kernel accelerations.

However, due data-dependent branches in other parts of the program (see Section 2.3.1.3), other kernels such as *Dbl Min Srch* and *Dbl Max Srch* may be called by one or several cores that are not always running in lock-step. When concurrent acceleration requests are not exactly received at the same clock cycle by the CGRA Controller, they will be mapped on different RC columns. Thus, they will not benefit from the SIMD-CGRA architecture. As an example, the *Dbl Min Srch* kernel called by the 6L-MF benchmark, is executed 64 % of the time in parallel over 2 datapaths (of the same RCs) when a 2 DPs CGRA is employed. The rest of the kernel invocations (i.e., 36 %) are executed alone and sequentially on a single datapath inside the allocated RC columns. Nevertheless, in the case of *Dbl Min Srch* and *Dbl Max Srch*, the multi-DP architecture is still beneficial, since most of the acceleration requests are executed in SIMD mode on the CGRA featuring 2 or 3 DPs per RC.

In addition to improving the execution speed-up of the kernels and the parallelism on the CGRA mesh, the SIMD execution mode allows also a faster access time to the accelerator (i.e., it reduces the CGRA access contention among the cores). When the cores request a kernel

acceleration, a variable waiting time occurs between the moment in which the request is received by the Synchronizer and the moment in which it is transmitted to the accelerator by the CGRA Controller. This waiting time is depicted in Figure 2.26 for the different DP configurations. The minimum waiting time achievable with the proposed implementation and without resource contention is 8 cycles. It corresponds to the time required to find the proper RC columns used to execute a kernel.

As shown by Figure 2.26, there is a sharp drop between the 1 DP case and the other datapath configurations. The 1 DP configuration induces a large amount of contention on the CGRA, i.e., 17.9x more waiting time in average for all the considered benchmarks with the 1 DP configuration compared to the 2 DP one. In fact, the majority of the kernels are executed by three to eight cores operating most of the time in lock-step/SIMD mode. Therefore, multiple DPs are required to map and accelerate several kernels in parallel, without waiting too long for available RC columns.

A.3) Kernel Energy Consumption: Figure 2.27 compares the total energy consumed when executing the selected kernels in each application, on the multi-core architecture only and on the CGRA with different SIMD widths. The data is aggregated across all invocations of a kernel in the indicated benchmark.

Similarly to the observations made in Section 2.4.2.2, Figure 2.27 highlights that execution on the CGRA accelerator is more energy-efficient by a large margin than on the multi-core system (i.e., *SW Only* execution without HW accelerations). Moreover, by supporting SIMD in the CGRA mesh, further energy savings can be achieved, especially for kernels which present a high ratio of concurrent calls by multiple cores (*Dbl Min Srch, Dbl Max Srch, Min Max Srch,* and *CS*) (see Figure 2.25). The average savings relative to these kernels for a CGRA with a SIMD width of 2, averaged over all the applications where the kernels are used, range from 53 % (*Dbl Max Srch*) to more than 90.8 % (*Min Max Srch*).

Despite the energy benefits obtained for *Dbl Min Srch, Dbl Max Srch, Min Max Srch,* and *CS*, other kernels do not take advantage of additional SIMD datapaths. This is particularly the case for kernels only called by one core and always executed on a single DP, such as *Lin Min Max, Apply RP* and *Lin Srch* in Figure 2.27. Therefore, as shown later in Section 2.5.2.2, a complete study at the system level is required to assess the global energy savings provided by the multi-DP CGRA for each benchmark application. This comprehensive study allows also to make a final choice between a 1, 2 or 3 DPs CGRA configuration.

A.4) CGRA Area Footprint Exploration: Table 2.3 details the breakdown of the area footprint of the different CGRA components. An important part of the mesh area is used by the Configuration RAM, which occupies more than two-third of the CGRA real estate in the case of 1 DP instance (which requires roughly 0.36 mm^2). The size of the Configuration RAM is constant for all the CGRA architectures with 1, 2 or 3 DPs. In addition, the Configuration RAM size could be reduced by supporting less kernels per application (or smaller ones), or

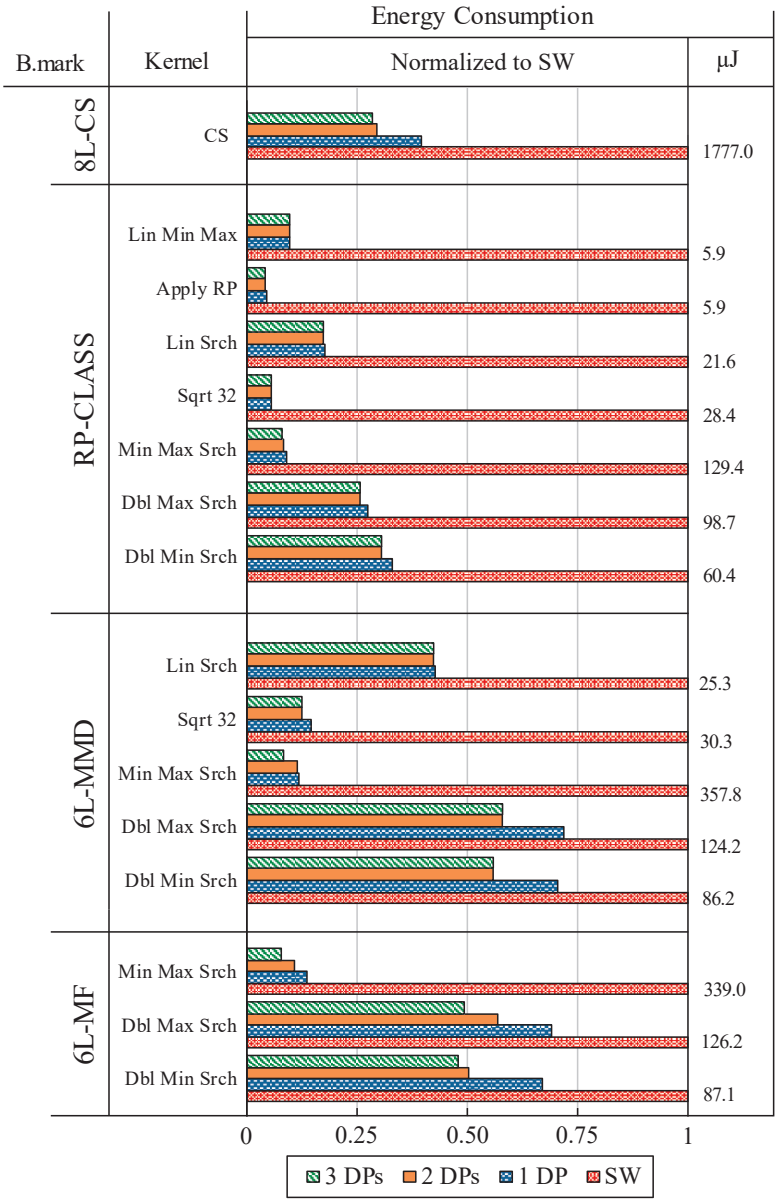


Figure 2.27 – Energy consumption of the different kernels. For each kernel, the bars are normalized to the *SW Only* consumption, which is indicated on the right side of the graph.

by storing configurations in a compressed form [125]. The mesh itself is extremely compact, as only few operations have to be supported by the ALUs and only a simple control logic is required. While ALUs supporting the square root operation are bigger than the ones that do not, such overhead has a small overall impact, as this feature is present only in one cell per column.

In fact, each DP inside an RC (ALUs + local register file + multiplexers) is smaller than the configuration memory of the RC itself. Since the configuration storage and the control logic are

Table 2.3 – CGRA area exploration (in 65 nm UMC technology library).

Components	Area (μm^2)
Configuration register (per RC)	6721.5
CGRA control	5227.9
Cell control (per RC)	881.7
Register file (per DP)	1374.2
Multiplexers (per DP)	822.9
ALU Standard (per DP)	3962.8
ALU Square Root (per DP)	7859.1
Configuration RAM	278978.2
Total CGRA area 1 DP	**361591.2**
Total CGRA area 2 DPs (+34.1 % w.r.t 1 DP)	**484900.7**
Total CGRA area 3 DPs (+22.4 % w.r.t 2 DPs)	**593348.6**

shared among the different DPs, the overhead of doubling the SIMD width from 1 to 2 DPs is therefore only 34 %. Moreover, adding one extra datapath to the 2 DPs CGRA increases the total area by only 22 %. Therefore, the parallel processing capabilities of the CGRA improves faster than the required area to implement the CGRA accelerator. As will be shown in Section 2.5.2.2, this area overhead is even smaller when considering the complete system.

B) System-Level Assessment

In this section, the energy benefits, area overhead, and run-time performance of the multi-DP CGRA-based platform are investigated.

B.1) Energy Consumption per Application: Figure 2.28 analyzes the energy consumed by the multi-core platform with and without the multi-DP CGRA, for the processing of the targeted applications. This figure highlights that using the multi-DP CGRA improves energy efficiency, as the accelerator takes advantage of SIMD processing. In particular, the processing system including the 2 DPs SIMD-CGRA accelerator results in significant energy savings for all the considered benchmarks, ranging from 9.2 % to 37.2 %.

These energy savings are dependent of the nature of the employed algorithms. Substantial energy gains are noted in the case of 6L-MF and 6L-MMD applications, which similarly spend a large amount of their processing time in the accelerated kernels. Even in the case of the RP-CLASS application, where there are far fewer acceleration calls (see Figure 2.25), the obtained energy savings are still significant (9.2 %).

The maximum reduction of energy consumption is observed for the 8L-CS application, because its workload is mostly concentrated in the accelerated *CS* kernels. In fact, for each ECG sample processed, eight *CS* kernel invocations are performed by eight cores fully operating in lock-step. Hence, with a 2 DPs SIMD-CGRA all the acceleration requests (using one RC column) can be mapped and executed at the same time on the four columns of the mesh. More

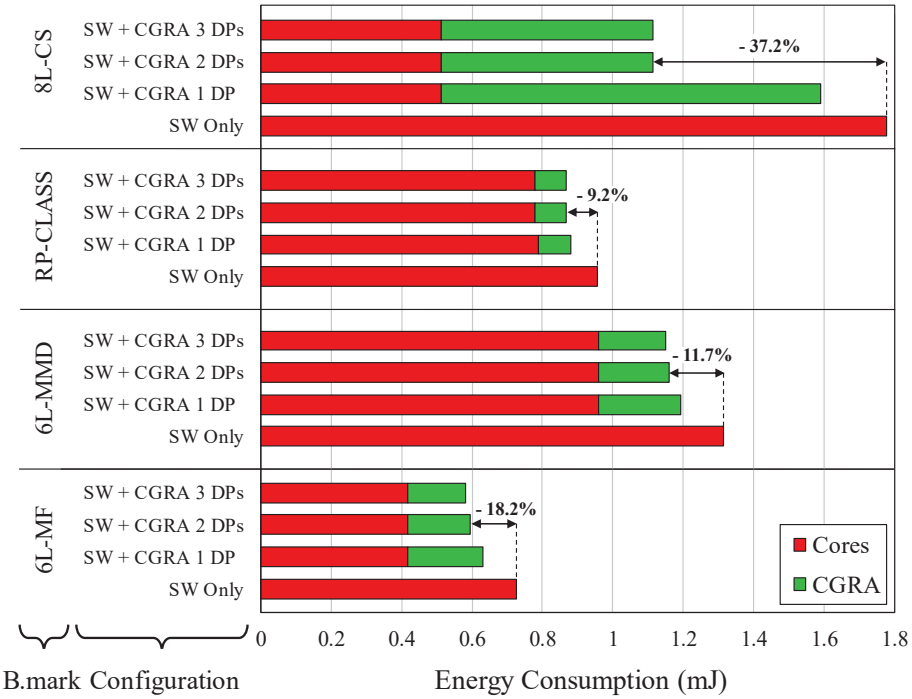


Figure 2.28 – System energy consumption for processing the different benchmark applications.

precisely, with this architecture two *CS* kernels are executed on the 2 DPs of each RC column (i.e., 2×4 = 8 *CS* kernels). In this way, significant energy savings are obtained (i.e., 37.2 % w.r.t *SW Only*) by configuring and controlling at the same time all the datapaths executing the same kernel. However, if an additional datapath is added to the 2 DPs SIMD-CGRA, no extra energy savings are obtained, as shown at the top of Figure 2.28 (see *8L-CS* / *SW + CGRA 3 DPs*). With a 3 DPs SIMD-CGRA, three kernels are concurrently executed on the two first RC columns (i.e., 2×3 = 6 *CS* kernels), while the two other kernels are processed in parallel on the two datapaths of the third RC column (the fourth column is left unused and the remaining datapaths are power-gated). This SIMD-kernel repartition has also a visible effect on the first bar at the top of Figure 2.25. As a consequence, only three-fourth of the 3 DPs SIMD-CGRA is used to execute the eight *CS* kernel invocations. In other words, the processing capabilities of the CGRA are not fully exploited with 3 DPs, compared to the 2 DPs case. This observation is also valid for the other benchmarks. Therefore, as shown later in Section 2.5.2.2-B.3, to avoid unnecessary area overheads without substantial energy benefits, the choice of the CGRA architecture must be tailored to the access contention level and processing requirements of the application.

In particular, with the employed benchmarks the 3 DPs SIMD-CGRA does not improve drastically the energy consumption of the platform with respect to a 2 DPs CGRA accelerator. In fact, the 3 DPs architecture would be more appropriate in the case of heavily parallel DSP benchmarks, such as applications with a higher number of ECG leads (e.g., 12 leads) or applications from the EEG processing domain. By processing a bigger number of signals in parallel over multiple cores operating in lock-step, additional kernel acceleration requests are issued at the

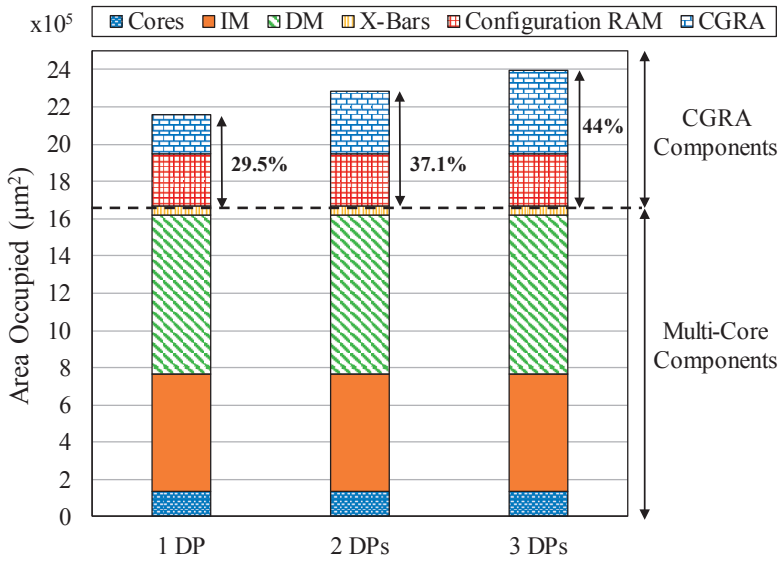


Figure 2.29 – Area breakdown of the accelerated multi-core platform, embedding a CGRA with different SIMD widths.

same time, thus increasing the workload and access contention on the CGRA. In this context, many more opportunities are present to merge up to 3 acceleration requests at the same clock cycle and on the same 3 DPs of each RC column. Hence, a marked improvement of the energy consumption can be obtained in this scenario with a 3 DPs SIMD-CGRA compared to a 2 DPs architecture.

B.2) Platform Area Breakdown: This section investigates the effects of using the proposed multi-DP CGRA on the area footprint of the whole system. Figure 2.29 displays the area breakdown of the different components of the envisaged heterogeneous and reconfigurable platform. It showcases that the addition of the CGRA with a single-DP results in the increase of the area occupied by only 29.5 %, with respect to a multi-core system alone. Moreover, the area cost of extending the CGRA to support SIMD kernels is marginal: less than 8 % of the total area for each additional DP. Thus, the area penalty deriving from the adoption of complex multi-DP cells, which achieve a better energy efficiency when the execution of SIMD kernels dominate at run-time (as seen in Section 2.5.2.2-B.1), is affordable. For example, in the case of the 8L-CS benchmark, the system energy consumption is 29.9 % lower when a CGRA with 2 DPs per RC is used instead of a single-DP CGRA (see Figure 2.28). In that case, the system area penalty incurred due to the additional DP is only 7.6 %. In other words, the area penalty resulting from the doubling of the number of CGRA datapaths is less than the double of the single-DP CGRA area, since only the datapath itself is replicated without the configuration registers associated to each RC.

B.3) CGRA SIMD Width: From a design perspective, the choice of which SIMD width to support is a trade-off between complexity and flexibility. A higher SIMD degree can potentially decrease energy consumption by harnessing more parallelism from applications. On the

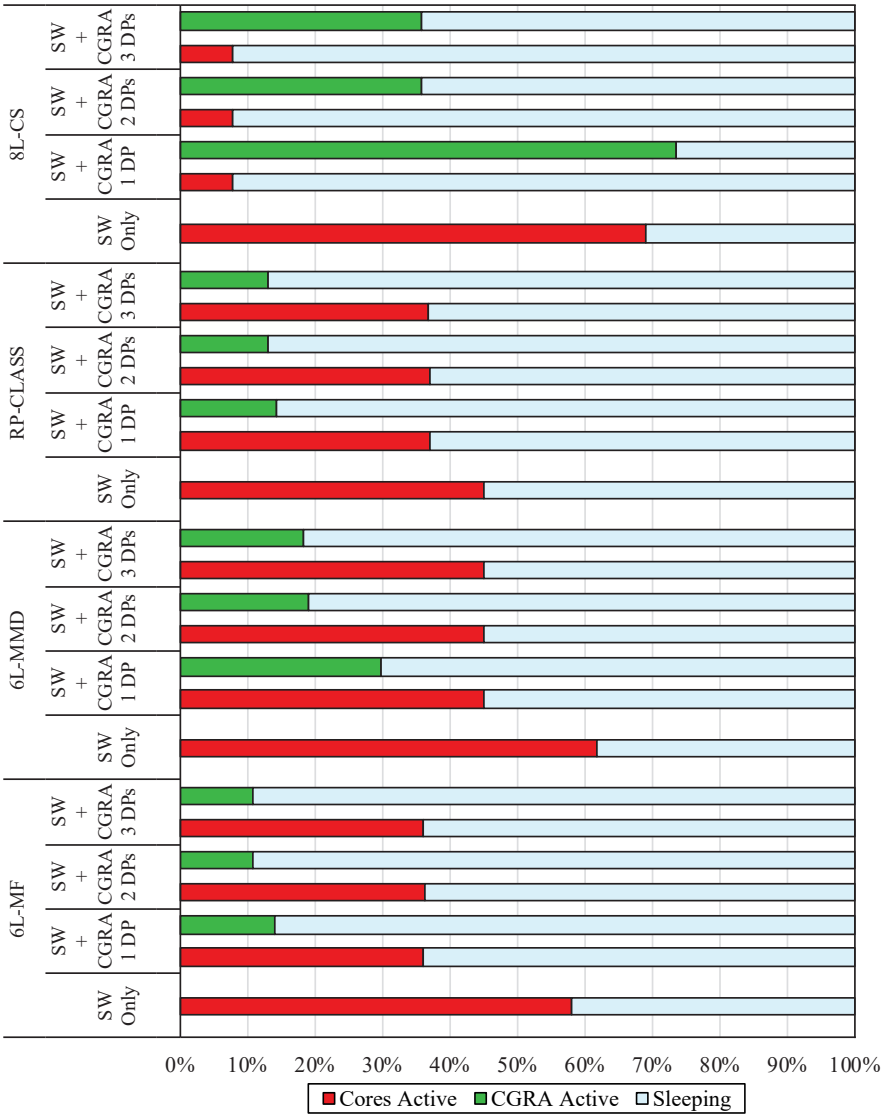


Figure 2.30 – Multi-core and CGRA utilization time in the considered platforms (% of the total run-time).

other hand, the design of more complex RCs requires more silicon area, and a more elaborate controller (i.e., with additional multiplexers). Ultimately, the parallelism supported by the accelerator should closely match the one present in the kernels of application.

A configuration with a single-DP per cell is clearly sub-optimal. As discussed in Section 2.5.2.2-A.1, in the case of the *CS* kernel, the adoption of a single-DP mesh even results in a slowdown (instead of a speed-up), due to the presence of a high level of contention, which leads to significant waiting times (reported in Section 2.5.2.2-A.2). Ultimately, as detailed in Section 2.5.2.2-A.3, the energy efficiency of a single-datapath CGRA trails that of an alternative supporting SIMD. In addition, Figure 2.28 shows that the energy savings when using a CGRA with 2 DPs per RC are very similar to the case when 3 DPs are employed, while requiring a bigger

area (see Section 2.5.2.2-A.4). Therefore, a choice of 2 DPs per RC strikes a good trade-off between performance, energy efficiency, and silicon area for the considered benchmarks.

Furthermore, the good performance of this configuration is highlighted by analyzing in Figure 2.30 the utilization of its computing components. Results show that, when the system is interfaced with a 2 DPs SIMD-CGRA, the active time of the cores is decreased on average from 58.7 % to 31.6 %, compared to the software-only version. This reduction is instrumental in improving the overall power consumption of the platform, because of the superior energy efficiency of the CGRA module when processing computationally-intensive kernels.

Moreover, for all the studied benchmarks, a multi-DP CGRA with 3 DPs only provides marginal benefits with respect to the 2 DPs configuration, justifying the choice of the multi-DP CGRA employing 2 DPs per RC. Finally, Figure 2.30 also shows that in the case of the computationally-intensive 8L-CS benchmark, the overall utilization of the multi-core system and the CGRA is below 50 % when employing a multi-DP configuration, allowing a potential reduction of the operating frequency from 2 MHz to 1 MHz. As a consequence, by dividing the operating frequency by two, a linear decrease of the dynamic energy consumed by the platform can be obtained (see Equation 3.1 in Chapter 3). Since the dynamic energy consumption is dominant (~95 %) with the proposed gate technology (see Section 2.4.2.1-B), the energy consumed by the processing platform can be roughly halved by employing a 2 DPs SIMD-CGRA with the 8L-CS benchmark.

As shown in this experimental evaluation, the multi-DP CGRA showcases significant performance improvements both in terms of system energy reductions and kernel execution speed-ups. However, the usefulness of such CGRA architecture is only justified in the context of a multi-core system running in SIMD mode and requesting identical kernel accelerations in parallel. In contrast, the following section introduces a novel CGRA architecture able to further accelerate the execution of individual kernels, requested by a single core.

2.6 Interleaved-Datapath SIMD-CGRA Accelerator Shared by a Multi-Core System

2.6.1 Overview

The multi-DP CGRA mesh presented in Section 2.5, allows multiple acceleration requests to be merged, mapped, and executed together on the same CGRA resources (i.e., RC columns). By sharing the same computing resources and the cost of reconfiguration among identical acceleration requests, substantial speed-ups and energy savings are obtained at the system level. However, this approach is only beneficial when the same kernel acceleration is requested by different processors working in SIMD mode. Moreover, this architecture requires high communication bandwidth between the CGRA and the data memory to support the parallel execution of the SIMD-kernels on the mesh.

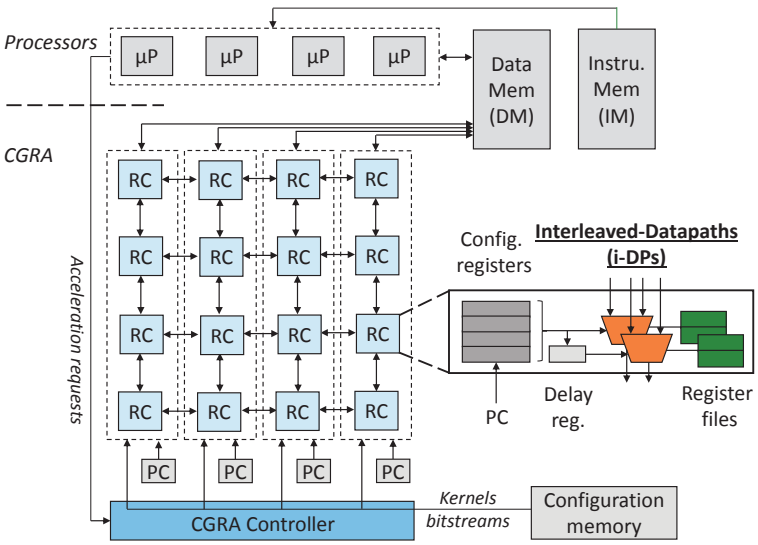


Figure 2.31 – i-DP CGRA block scheme, interfaced with a multi-processor system.

In comparison with the previous architecture, I showcase in this section an optimized CGRA design that, by employing complex cells with multiple Interleaved-DataPaths *(i-DPs)*, leverages the execution parallelism of individual kernels to speed-up their computation and decrease the area and energy budget of the associated control logic. In addition, thanks to an interleaving mechanism, the proposed CGRA architecture minimizes the required bandwidth (i.e., number of memory ports) between the CGRA and data memory, thus avoiding extra area and energy overheads induced by wider memory interfaces.

As opposed to the multi-DP CGRA presented in Section 2.5, this improved CGRA architecture results in high efficiencies regardless of the structure and operating states of other system components (e.g., processors) at run-time. In particular, the i-DP CGRA can efficiently parallelize the execution of individual kernels called by a single processor, without the need for cores operating in lock-step/SIMD mode and requesting the same kernel acceleration.

Furthermore, the proposed i-DP CGRA can access the data memory by multiplexing the port of the processor that issued the acceleration request between several DPs. This strategy does not require dedicated memory ports, while allowing the transfer, at each clock cycle, of one data word per active kernel. In order to avoid data memory access conflicts, the i-DP strategy skews (using a delay register) by one clock cycle the active configuration word between DPs (see Figure 2.31). As a consequence, the memory accesses from each DP are temporally spaced by a number of clock cycles at least equal to the number of DPs. Thus, kernels can be effectively split into slices, with each DP processing only a kernel slice (i.e., a sub-window or vector of data).

Figure 2.32 shows an example of a simple kernel being mapped on a single-DP and on an i-DP CGRA with two datapaths. The two architectures have the same configuration words, but in the i-DP case the same set of configuration registers drives two different DPs. Separate kernel

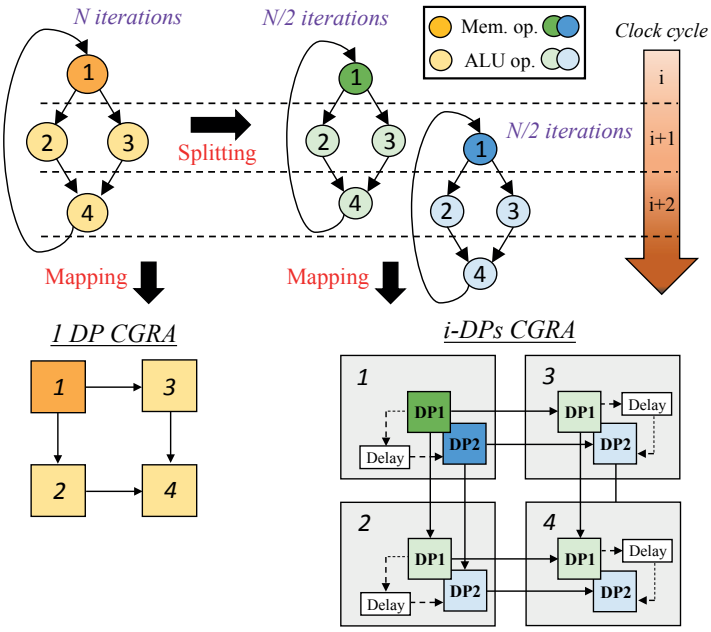


Figure 2.32 – Example of a simple kernel mapped in a single-DP and an i-DP CGRA. In the latter case, the kernel is split into two slices and mapped on the interleaved DPs of the RCs.

slices can then be executed concurrently on the CGRA (skewed by one clock cycle), with the ensuing gain in parallelism being only limited by the amount of load/store operations, and the available bandwidth between the data memory and the CGRA.

Little hardware and run-time overhead is required to support interleaved-datapaths. On the hardware side, delay registers in each RC store one configuration word of 32 bits per DP. Furthermore, a multiplexer is required to select, at each clock cycle, which DP can access the data memory. As for run-time, scalar constants have to be written for each slice in the local register file of the DPs, and scalar results transferred back to data memory. Section 2.6.2.2 demonstrates that these overheads are dwarfed by the gains attained in execution time.

The adopted strategy can effectively map two common kernel structures. First, kernels that do not present loop carries can perform a slice of all iterations in each DP, without further modifications. Second, reduction kernels, which compute one (or few) scalar values from an input array, can be divided into multiple parts, but require a wrap-up phase in software to aggregate the obtained results. Even in this last case, notable speed-ups can be obtained with i-DP with respect to a single-DP arrangement, when the input set is sufficiently large.

2.6.2 Experimental Evaluation

To characterize the benefits delivered by the i-DP CGRA accelerator at the system-level, this experimental evaluation is structured as follows. First, the main features of the experimental setup are presented, including the target and baseline systems taken under consideration in

this study, followed by the employed benchmarks. Finally, three analyses are carried out to assess the i-DP CGRA performance from a run-time, energy and area perspective, with respect to the considered baseline systems.

2.6.2.1 Experimental Setup

A) Target and Baseline Systems

To analyze and compare the proposed i-DP CGRA mesh with the baseline computing platforms, three different architectural configurations are considered. They are composed as follows.

- **Multi-core architecture only:** Baseline multi-core system without CGRA accelerations (*SW Only*). This system architecture relies on the work of the authors of [14].
- **Multi-core architecture with single-DP CGRA:** Processors are interfaced with a CGRA having only one DP per RC. This architecture configuration corresponds also to the platform evaluated in Sections 2.4.2 and 2.5.2. In this thesis, it is considered as the baseline CGRA architecture.
- **Multi-core architecture with i-DP CGRA:** Target architecture, as described in Section 2.6.1. It integrates multiple i-DPs in each RC, allowing the efficient parallelization of an individual kernel, by uniformly distributing its execution over multiple interleaved-datapaths. Several datapath configurations are evaluated in Section 2.6.2.2, including two, four, and eight i-DPs per RC.

B) Simulation Framework and Biomedical Benchmarks

The experimental framework employed in Sections 2.4.2 and 2.5.2, is reused hereafter for the experimental evaluation of the i-DP CGRA architecture.

First, to assess the performance of the proposed accelerated platform, two bio-signal processing applications were selected, namely 6L-MF and 8L-CS (previously described in Section 2.5.2.1-B). These two benchmarks execute kernels that can be easily split over multiple datapaths in parallel (which is not the case, for instance, for a square root kernel such as *Sqrt 32*, presented in Section 2.4.2.1-D). During the execution of the 8L-CS benchmark, the *CS* kernel computes the random indexes of the compression, by using a linear feedback shift register. This kernel does not present loop-carried dependencies, and can therefore be straightforwardly mapped on the accelerator by distributing its iterations uniformly among the available DPs.

Secondly, the 6L-MF benchmark cancels the baseline wandering of an ECG acquisition by employing sliding windows of 100 and 150 elements [61]. The considered low-pass filtering part of the 6L-MF benchmark involves the computational kernels *Dbl Min Srch* and *Dbl Max Srch*, which are employed to compute the first and second minimum and maximum along the

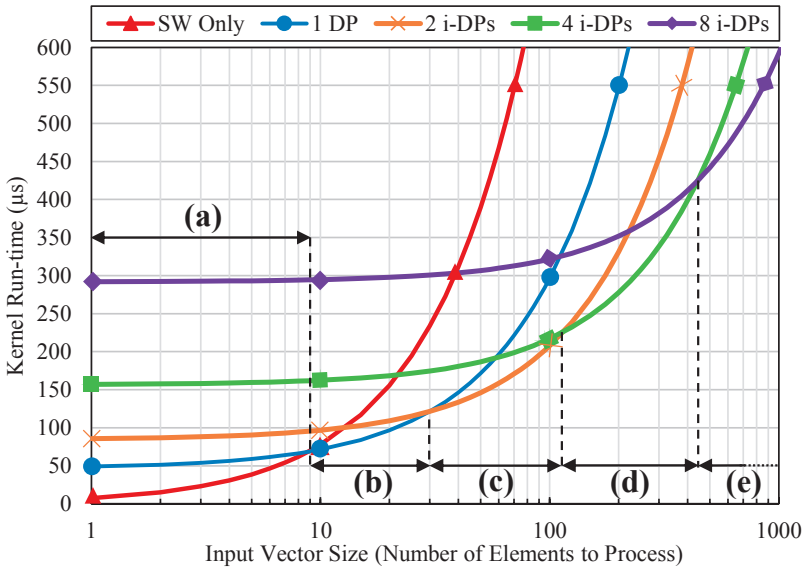


Figure 2.33 – Run-time of *Dbl Min Srch* kernel, which computes the first and second minimum element of an array, on a single-DP CGRA and on i-DPs with different widths, varying the input data size.

windows of elements. The computations of these kernels can therefore be divided in slices (e.g., two slices in two i-DPs case), each of them returning the two lower or higher values extracted from the processed sub-window. Then, a small software wrap-up routine is in charge of determining the two final outputs among the four values computed by the i-DP CGRA.

Additionally, to support acceleration requests to the i-DP CGRA, the program running on each processor must configure a set of private memory-mapped registers, providing the dynamic parameters of the kernel to the CGRA (see *Step 1* in Section 2.3.1.6). In the case of the i-DP CGRA, each DP employed during the hardware acceleration requires the configuration of an extra set of 5 registers, before launching the request to the CGRA. For instance, when an i-DP CGRA with 2 DPs per RC is used, 10 (2×5) registers must be written by the processor and read by the CGRA for each acceleration request. With this setup, the instruction overhead introduced by the configuration of extra registers and the software wrap-up routine represent less than 2 % of the total binary code of each benchmark.

Finally, at the hardware level, the scheduling of the considered kernels has been reevaluated to support the i-DP CGRA architecture and the skewing mechanism previously presented.

2.6.2.2 Experimental Results

A) Performance Analysis

While kernels can be conceivably divided in a high number of slices, the attainable gains may offer diminishing returns in terms of execution time, due to the bandwidth bottleneck

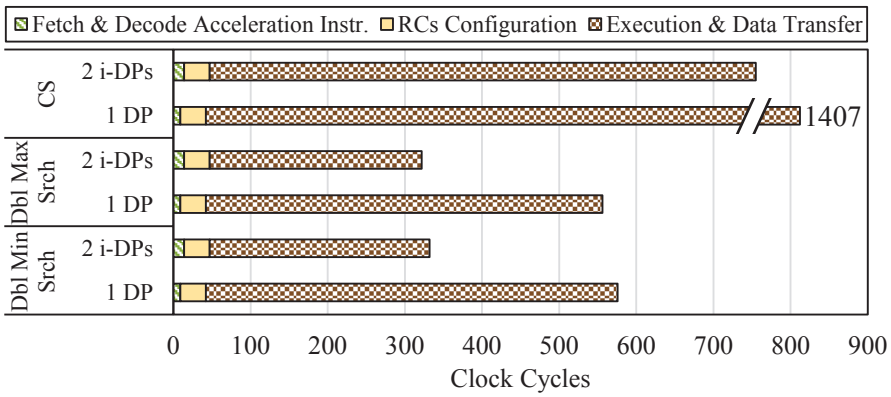


Figure 2.34 – Average kernels run-time (in clock cycles) executing on CGRAs with 1 DP and 2 i-DPs.

between the data memory and the reconfigurable fabric. Moreover, a large number of DPs may incur a significant timing overhead for the transfer of initialization values (through the memory-mapped registers) and scalar outputs to/from the CGRA for each slice. In addition, for reduction kernels, the time required for the wrap-up phase (see Section 2.6.2.1) increases proportionally with the number of slices. Therefore, the selection of a proper DP width for a kernel depends on the amount of its memory accesses and on its number of iterations.

This last aspect is investigated in Figure 2.33, which showcases the trade-offs between the number of elements to process and the number of interleaved DPs, depicting the global execution time of the *Dbl Min Srch* kernel, a computationally-intensive hotspot of the 6L-MF benchmark. In region (a) (i.e., from 1 to 9 data elements), the execution of the kernel directly on the core without CGRA (*SW Only*) is faster compared to an execution with a single or i-DPs CGRA. Indeed, for such a small dataset, it is not worthwhile to configure and invoke the accelerator, as the entailed overhead exceeds the benefits of hardware acceleration. Then, in regions (b) and (c), the 1 and 2 i-DPs CGRA accelerators present better run-time performance, for input sizes respectively from 10 to 30 and from 31 to 120 elements. In region (d) (i.e., from 121 to 445 data elements), the 4 i-DPs yield the best performance, while 8 i-DPs have the lowest run-time if the input size exceeds 446 elements (see region (e)).

Since, in the considered benchmarks, the kernel input data vectors have an average size of 100 elements (which is a typical scenario for bio-signal analysis applications), in the rest of this section only a 2 i-DPs CGRA configuration is considered.

Similar results were obtained for the other investigated kernels. Figure 2.34 illustrates their execution time on a single-DP and on an i-DPs CGRA with 2 DPs, without considering the software overhead required by the processors to configure, launch, and recover the results from an acceleration. The graph shows that, by adopting i-DPs, a large reduction is obtained in the time required for computing the kernel outputs (*Execution & Data Transfer* phase), which is almost halved. An increase in the *Fetch & Decode* phase is also observed, due to the

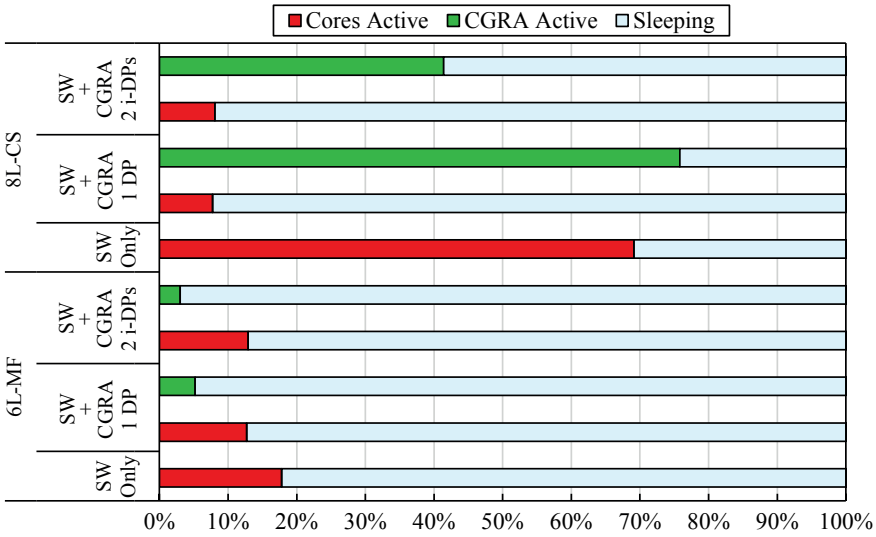


Figure 2.35 – Multi-core and CGRA utilization time in the considered platforms (% of the total run-time at a 2 MHz clock frequency).

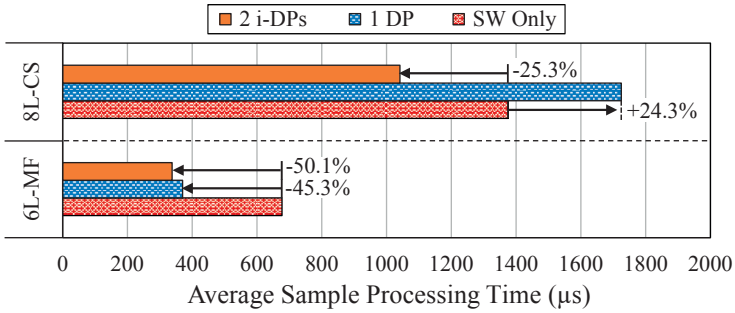


Figure 2.36 – Average sample processing time with the different benchmarks and CGRA architectures. With the current experimental setup, 2000 µs represents the sampling period of the ADCs, performing the acquisition of the ECG signals.

extra register settings required for the initialization of multiple kernel slices, but its impact is negligible (less than 1.6 % in all cases).

The above-mentioned gains are reflected at the system level. To assess them, a multi-core system interfacing the CGRA is considered (see Section 2.6.2.1). Figure 2.35 reports the active and sleeping time of the multi-core system and the CGRA, as a percentage of the total run-time of the application. A first consideration that can be drawn from this data is that both applications are rather kernel-intensive, with 28 % and 88 % of the active time spent in the kernel functions for 6L-MF and 8L-CS, respectively. Furthermore, Figure 2.35 shows that the i-DPs CGRA allows a marked decrease of CGRA active time, compared to its single-DP version. In fact, the CGRA activity is decreased by almost half in the case of the 8L-CS, and by one third for the 6L-MF application.

In addition, as shown in Figure 2.35 and similarly to the performance evaluation in Section 2.5.2.2-A, the 8L-CS benchmark is again an outlier when executed on a multi-core system interfaced with a single-DP CGRA (*SW + CGRA 1 DP*). The CGRA active time of this platform is 7 % higher than the active time of the multi-core system without hardware accelerator (*SW Only*). In fact, the single-DP CGRA is active over a longer period of time due to the high amount of contention during the execution of the 8L-CS benchmark. In this context, the eight SIMD cores of 8L-CS are trying to access the single-DP CGRA, which can only execute four kernels on its four RC columns at the same time. Therefore, the execution of the four remaining acceleration requests must be delayed and processed after the completion of the first set of four kernels. By processing two sets of four kernels sequentially, the active time of the single-DP CGRA is increased. In this case, the execution of the kernels at the software level (*SW Only*) is faster, since the cores are operating independently without waiting for available computing resources.

This effect of resource contention is also visible in Figure 2.36, which depicts the average sample processing time with the different benchmarks and CGRA architectures. The resulting waiting time to access the single-DP CGRA increases by 24 % the time necessary to process an ECG sample. However, when the i-DP CGRA is employed, the 6L-MF and 8L-CS benchmarks spend in average less time to process a sample, with -50 % and -25 % respectively, compared to a multi-core platform without CGRA accelerator.

As a conclusion of this CGRA performance evaluation, and based on the results obtained from the other CGRA architectures (see Sections 2.4.2.2 and 2.5.2.2), it can be noted that the resulting execution speed from an accelerated kernel is mainly dependent of four factors. They are listed below in chronological order, following the execution steps of an acceleration request:

- **The software initialization phase of the kernel** before calling the acceleration request (e.g., the time spent for saving the dynamic parameters of the kernel into the private memory-mapped registers of the core requesting the acceleration).
- **The resource allocation time**, depending on: the size of the CGRA mesh, the resource access contention with the other cores using the CGRA, the number of RC columns required to map the kernel and the number of DPs per RC.
- **The intrinsic execution speed of the kernel** based on its scheduling/mapping on the CGRA mesh and on the number of iterations/elements to process from the input vector.
- **The duration of the software wrap-up routine** after the execution of an acceleration request (only when an i-DP CGRA is employed).

B) Energy Analysis

From an energy viewpoint, the savings obtained by the i-DPs CGRA stem from two sources. First, increased idle times are exploited to aggressively clock-gate idle components (cores),

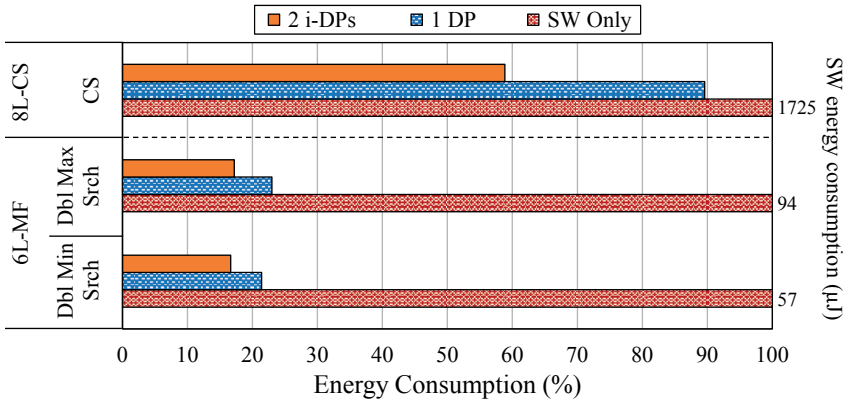


Figure 2.37 – Energy consumption of the different kernels. For each kernel, the bars are normalized to the SW-only energy.

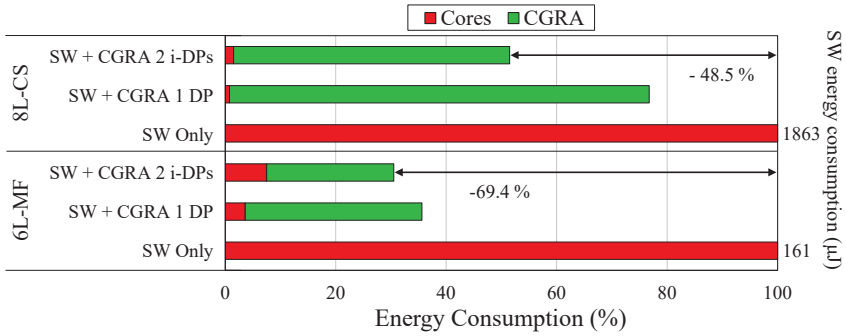


Figure 2.38 – System energy consumption of the different benchmark applications, normalized to the *SW Only* consumption for the part of the application transfered to the CGRA.

resulting in a decrease in dynamic energy. Second, the i-DPs scheme results in a high ratio between the CGRA logic devoted to computing (RCs) and that used to control the execution flow (i.e., configuration registers), which is exploited to increase efficiency. Figure 2.37 highlights that important savings are attainable for each kernel by employing interleaved datapaths. In the case of the *CS* kernel, the energy budget (with respect to an equivalent single-DP architecture) is reduced by 34 %. For the two kernels of 6L-MF benchmark (*Dbl Min Srch* and *Dbl Max Srch*), the reductions are 22 % and 25 %, respectively.

Figure 2.38 compares the energy consumption of the part of the workload that is accelerated for the two considered benchmarks. The three architectural choices presented in Section 2.6.2.1 are employed to produce this figure: (1) a multi-core platform, that does not embed a reconfigurable accelerator, (2) a platform that couples a multi-core system with a single-DP CGRA and (3) the proposed platform, featuring an i-DPs accelerator. It can be noted that, even in the two latter cases, certain software overhead is required for setting up, launching, and retrieving the outputs of an acceleration request. This component of the energy budget of kernels is even more pronounced for the i-DPs case, since, as discussed before, i-DPs requires a more complex initialization phase. The increase is particularly noticeable for the

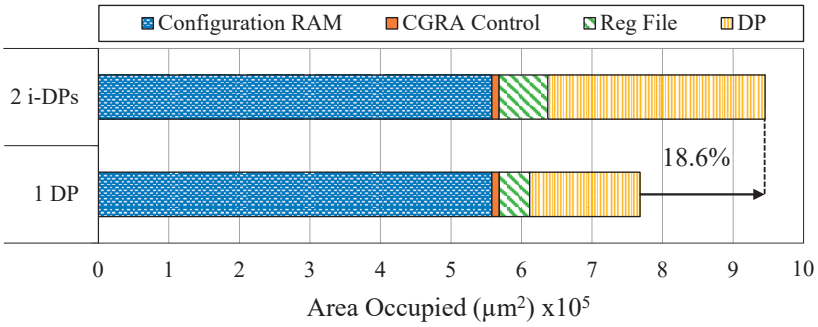


Figure 2.39 – Area breakdown of the i-DPs CGRA compared to the baseline single-DP CGRA.

6L-MF benchmark which, being a reduction algorithm, also requires wrap-up computations, performed in software. The energy efficiency derived from the use of i-DPs is nonetheless substantial: 69.4 % and 48.5 % for 6L-MF and 8L-CS, respectively, when compared with the execution of the kernels on the multi-core system without CGRA support. Furthermore, the resulting energy envelopes are always smaller by a large margin with respect to the single-DP CGRA alternative (see *SW + CGRA 1 DP* in Figure 2.38).

Finally, as shown during the performance analysis, the 8L-CS benchmark executed on the single-DP CGRA leads to a higher active time of the overall platform (see *SW + CGRA 1 DP* in Figure 2.35). In addition, the average sample processing time is also increased as shown in Figure 2.36. Despite the performance degradation when the single-DP CGRA is employed, the real-time constraints of the platform are still met (i.e., the ECG samples are processed in less than 2000 µs, which corresponds to the sampling period of the ADCs). Moreover, as depicted in Figure 2.38, even if the platform activity is increased with the single-DP CGRA accelerator, 23 % of energy savings are obtained due to the higher energy efficiency of the CGRA when processing the *CS* kernels.

C) Area Analysis

Figure 2.39 presents a breakdown view of the silicon area required for two CGRA configurations with a single and two interleaved DPs (including the delay registers). In both cases, a 4×4 mesh has been considered with RCs supporting 32 configuration registers of 32 bits each. Similarly to all the computing architectures explored in this chapter, the considered data bitwidth is 16 bits. Register files, local to each DP in the RCs, can store 4×16 bits words (see Figure 2.13 in Section 2.3.2.1). Since increasing the i-DPs width only impacts the logic required for the datapaths themselves and the local register files, doubling it from 1 to 2 only entails an area overhead of less than one fifth of the total CGRA area. In fact, a sizable portion of the CGRA area is employed by the Configuration RAM, whose area does not depend on the number of datapaths, but rather on the number of configuration words required to configure the kernels on the RCs.

2.7 Summary and Concluding Remarks

Energy efficiency is a major concern in the design of digital systems across the computing landscape. It is especially important in the context of wearable health monitoring devices, such as Wireless Body Sensor Nodes (WBSNs), where the power budget is drastically limited. To achieve the required ultra-low power operating levels, a careful optimization of the architectural components is necessary. For such optimization to be effective, it must be *domain-specific*. In other words, it has to take into account and exploit the characteristics of the targeted workloads. Herein, I addressed this complex endeavor by proposing a heterogeneous and reconfigurable architecture devoted to bio-signal processing applications.

DSP applications from this domain rely on run-time execution profiles which are often divided between control-dominated phases and computationally-intensive ones in the form of compact loops (i.e., kernels). The illustrated DSP platform can efficiently support both: the former on multiple ultra-low power processing cores and the latter by employing a coarse-grained reconfigurable array (CGRA) interfaced to the cores as a shared acceleration resource.

In this chapter, three different CGRA architectures are proposed and characterized from a performance, energy and silicon area perspective.

Firstly, I have introduced a *Single-Datapath CGRA* architecture that is shared by multiple cores and able to process several acceleration requests in parallel on the different computing resources (i.e., reconfigurable cells) of the CGRA mesh. It is the simplest CGRA design proposed in this work, with the minimum area and energy overheads. It allows the developed platform to achieve tangible overall energy savings of up to 18.6 %, when executing complex ECG processing applications, in comparison to an equivalent multi-core solution without CGRA acceleration of kernels.

Secondly, I have proposed a *Multi-Datapath CGRA* design, allowing the platform to support SIMD execution modes, both at the processor and CGRA levels. In this way, the platform leverages SIMD in order to (1) merge the memory accesses from different processors running in lock-step, (2) minimize the number of reconfigurations and the energy consumption required to execute accelerated kernels on the CGRA, and (3) reduce the access contention to the CGRA resource by merging multiple acceleration requests into the same reconfigurable cells. Thus, the proposed platform is perfectly tailored to the acceleration of heavily-parallel biomedical applications. Thanks to its optimized architecture, energy savings of up to 37.2 % are achievable when executing real-world bio-signal processing applications, compared to a multi-core system without hardware acceleration capabilities.

Thirdly, I have showcased the efficiency of an *Interleaved-Datapath CGRA*, which is able to parallelize the execution of an acceleration request from a single core over multiple processing datapaths and inside the same computing resources. Its design is particularly suitable for the acceleration of kernels with a large number of iterations and/or called by a single core in the whole application. The interleaved run-time scheme of this CGRA architecture maximizes the

utilization of resources and available bandwidth between the CGRA and the data memory. This leads to notable run-time and energy efficiency gains. In particular, this architecture enables a reduction of up to 69.4 % of the energy consumed by the accelerated kernels, compared to their execution on the processors, without hardware accelerator.

To conclude this chapter, architectural solutions have been proposed to achieve higher energy savings and/or computing performance. In the next chapter, I investigate how the fine-tuning of the system operating parameters (i.e., frequency and voltage) in accordance with the physical limitations of the technology, can be applied to achieve the same end.

3 Technology-Level Reliability Exploration in Energy-Efficient Biomedical Systems

3.1 Introduction and Motivations

NOWADAYS, there is an increasing number of applications where *energy efficiency*, *reliability* and *performance* are at the top of embedded systems designers' priorities. In particular, in the domain of personalized healthcare delivery, wearable biomedical appliances, termed Wireless Body Sensor Nodes (WBSNs), are relying on these three fundamental aspects [23, 126].

WBSNs are fostering a revolution in health monitoring for patients affected by chronic ailments. This breakthrough is especially relevant in a world where the aging of the population and unhealthy lifestyles have positioned cardiovascular diseases in the first place of the leading causes of death worldwide [5]. The emergence of WBSNs has enabled cheap, unobtrusive and long-term monitoring of clinically-relevant bio-signals, with little supervision from the medical staff, even outside of a hospital environment. State-of-the-art WBSNs embed complex on-node Digital Signal Processing (DSP) routines to process and extract autonomously high-level features from bio-signal acquisitions, such as Electrocardiograms (ECGs) [35]. By only transmitting the main bio-signal features (as opposed to acquired samples) through the energy-hungry wireless link, these *Smart WBSNs* result in large efficiency gains, thus enabling longer acquisitions with miniaturized and lightweight devices, embedding smaller batteries. Nonetheless, these benefits can only be leveraged by performing the DSP stage within a tiny energy envelope.

To address this challenge, the literature provides a plethora of architectural solutions ranging from domain-specific single-core platforms [26, 80], to accelerator-based (e.g., CGRA-based) multi-core platforms [51, 52], such as the one presented in Chapter 2. The parallel nature of bio-signal processing applications enables the efficient decomposition and real-time execution of the different tasks on each computing resource (i.e., processors or accelerators) concurrently [38]. However, a substantial area increase is resulting from the integration of additional

computing resources, and hardware mechanisms to orchestrate and synchronize the execution of the processing tasks among these resources [51]. Moreover, the introduction of extra computing units will not further increase the energy efficiency of an application, if its execution cannot take advantage of the additional hardware parallelism. An effective strategy to achieve higher energy savings independently of the application parallelization is then to tune the operating parameters of the platform in accordance with the performance and reliability requirements of the application. This energy-efficient solution is detailed in the following section of this chapter.

3.1.1 From Silicon-Level Reliability Issues to Application-Level Degradations

A common strategy to increase the energy efficiency of a computing platform is to employ Voltage and Frequency Scaling (VFS) on the most energy-hungry components of the circuit [127]. This strategy is applicable to both logic and memory components, which represent the main building blocks of the processing stage from WBSNs (see Chapter 1). In addition, VFS provides knobs to adapt their energy consumption and performance to the different workload conditions, either statically at design-time or dynamically at run-time.

In order to give a better understanding of how VFS operates, Equation 3.1 represents the total power dissipation P_{total} of a CMOS circuit:

$$\begin{aligned} P_{total} &= P_{dynamic} + P_{static} \\ P_{dynamic} &= \alpha \times C_L \times V_{DD}^2 \times f \\ P_{static} &= P_{leakage} + P_{short\_circuit} \end{aligned} \tag{3.1}$$

Where:
- P_{total} = Total power dissipation [W]
- $P_{dynamic}$ = Dynamic switching power dissipation [W]
- P_{static} = Static power dissipation [W]
- $P_{leakage}$ = Transistor leakage current power dissipation [W]
- $P_{short\_circuit}$ = Short circuit power dissipation (i.e., when a CMOS gate switches) [W]
- α = Switching activity factor from 0 to 1.
- C_L = Capacitance load of the gates [F]
- V_{DD} = Supply voltage [V]
- f = Operating frequency [Hz]

With the current manufacturing technologies relying on Fin Field-Effect Transistors (FinFETs) and high-κ dielectric materials to drastically reduce leakage currents [128], the dynamic power consumption $P_{dynamic}$ is now the main portion of the total power dissipation P_{total}. As shown by the Equation 3.1, $P_{dynamic}$ is linearly increased conjointly with the activity and performance of the device. In particular, the switching activity factor α is dependent of the application workload, while the operating frequency is scaled by the VFS technique to adjust the clock speed and performance of the device. Furthermore, reducing the operating voltage of the circuit allows quadratic power savings.

Tuning the frequency and voltage parameters is a common practice extensively exploited in many research areas and commercial processors to reach different levels of computing performance and energy savings [41, 42, 43, 129]. On one hand, frequency scaling is massively employed in High Performance Computing (HPC) systems which require a high level of performance to process computationally intensive tasks in various fields, including computational sciences and big data analysis [130]. On the other hand, voltage scaling is intensively exploited in the Near-Threshold Voltage Computing (NTV/NTC) domain to process applications with a minimal energy consumption and moderate computing power [56, 57, 65].

Nonetheless, frequency increase and supply voltage reduction are both beneficial for future Smart WBSNs enabling, for instance, the execution of biomedical machine learning algorithms or the processing in real-time of a large amount of bio-signal leads (i.e., channels) in parallel, as is the case for Electroencephalogram (EEG) applications [131, 132]. To be processed, both applications require a high computing performance with a limited energy budget due to Smart WBSNs' constraints (see Chapter 1). Hence, the design of WBSNs expresses a strong need for VFS, even if performance and energy efficiency are both conflicting objectives with this technique.

Nevertheless, aside from the benefits offered by this technique, several disadvantages and limitations are resulting from its utilization. When VFS is aggressively employed, the voltage and/or frequency reductions can degrade the performance of the computing platforms beyond the real-time requirements of the applications. In particular, significant propagation delays are stemming from the decrease of the gate supply voltage. Moreover, low voltage operation makes the circuit more sensitive to any threshold voltage shifts and supply voltage deviations compared to the initial or nominal value [57]. In turn, this leads to timing degradations and violations, followed by possible functional errors. Additionally, the use of aggressive circuit operating conditions (i.e., high frequencies, low or high voltages) over an extended period of time, results inevitably into the occurrence of reliability degradations at the silicon level. As the frequency increases and the supply voltage approaches the threshold voltage of the transistors, soft (i.e., transient) and hard (i.e., permanent) errors start to appear in the logic and memory components [55, 56, 133].

To tackle these challenges, the modeling of memory reliability issues has been widely analyzed in the literature [134, 135, 136, 137]. However, the understanding of reliability concerns in the logic, such as Bias Temperature Instability (BTI), is still in its infancy, due to the lack of realistic modeling of its physical phenomena and resulting consequences in logic components [66, 138]. In order to produce an accurate model of BTI aging effects on the logic, a fine-grained and workload-aware transistor-level analysis must be performed (as shown later in Section 3.2.2.3-B). This analysis allows the identification of the BTI-induced degradations at the transistor level, which are subsequently spread to the higher abstraction layers of the system. For all these reasons and as a starting point, a technology-level study is necessary.

From the reliability knowledge gathered at the technology level and following a bottom-up approach, a complete system reliability evaluation is carried out in this chapter, starting from silicon reliability concerns to the assessment of the results produced at the output of well-known biomedical applications.

Moreover, as presented in Section 1.2.2.2 of the introductory chapter, some of the errors occurring during the processing of bio-signals can be tolerated, thanks to the *approximate computing* and *significance-based computing* paradigms [64, 139]. In fact, even if the application is affected by reliability issues occurring at the technology level, the delivered output results may satisfy an acceptable level of quality, thus highlighting the inherent resilience of bio-signal processing applications [56]. This approach is valid for many biomedical applications producing statistical or qualitative results, and for which the quality assessment is directly dependent on human perception. However, more critical biomedical applications (e.g., breathing or dialysis control software) may require an accuracy and dependability of 100 %, even in the presence of outlier degradation effects. Hence, the utmost reliability of the system must be ensured for these applications, by performing deterministic and application-dependent analyses. To this end, in the subsequent sections of this chapter, output-result quality evaluations are systematically performed to guarantee the clinical accuracy of the data produced by the employed biomedical applications, when reliability issues occur at the silicon level of logic and memory components.

3.1.2 Contributions and Outline of the Chapter

In this chapter, I investigate how the modeling of technology reliability issues in logic and memory components, can be exploited to adequately adjust the frequency and voltage parameters of the circuit.

To carry out this exploration of the optimized operating conditions, inline with the physical limitations of the technology, the study has been decomposed into two parts:

In the first half of this chapter, I propose to maximize the computing performance of logic components, by increasing the frequency beyond the traditional safety margins. In parallel, I also analyze the workload-dependent BTI impact on the delivered circuit functionality, when running with a nominal supply voltage at high frequencies. In fact, evaluating the effect of BTI degradations is especially crucial when operating under these conditions, leading also to a high temperature inside the chip.

In the second part of the chapter, I introduce a novel technique to minimize the energy consumption of memory components. To this end, I reduce aggressively their operating voltage, while taking full advantage of the inherent fault tolerance of several bio-signal processing algorithms. This approach is complementary to memory-level functional error mitigation techniques, such as the one proposed by the authors of [140], to mitigate the effect of memory reliability issues with an acceptable energy overhead.

To provide an overview of the most relevant contributions of this chapter, they can be grouped as follows:

Technology-Level Reliability Exploration in Logic Components

- I introduce a comprehensive framework to study the impact of BTI-induced timing degradation effects on functional errors rates. This framework can either be used to improve system performance and correctness (i.e., avoiding unsafe operating points), or to achieve a graceful degradation of its characteristics, in accordance with the requirements of the targeted application. In addition, the structure of this framework is generic enough to be reused for the evaluation of other device-level aging effects (see Section 1.2.2.2), by only changing the adopted aging model.
- I perform a workload-dependent BTI effect analysis for short and extended periods of time, both at the transistor and circuit levels. The considered application workload, generated by a complex ECG processing algorithm, highlights the physical phenomena involved in transistor-level BTI variability.
- I assess and quantify the impact of BTI-induced timing degradations occurring at the circuit level, and propagated as functional errors to the system and application levels. I evaluate the resulting quality degradation of the output data produced by the application, for several operating points.
- I provide insights on the mitigation of BTI-induced functional errors at the hardware and software levels.

Technology-Level Reliability Exploration in Memory Components

- I evaluate the impact of memory failures on the results generated by a representative set of biomedical applications. To this end, I propose a realistic stuck-at fault model, mimicking the effects of pseudo-permanent errors in memories for different voltage levels.
- I introduce the *Dynamic eRror compEnsation And Masking protection (DREAM)*, a new asymmetric Error Mitigation Technique (EMT), consuming 21 % less energy than a traditional Error Correction Code (ECC) with Single Error Correction / Double Error Detection (SEC-DED) memory protection.
- I investigate the energy-efficiency of the resulting significant-base memory protection compared to the SEC-DED ECC for different supply voltages. Additionally, in my experiments I include a space exploration highlighting the trade-offs for the selection of the optimal mitigation technique for a given biomedical application and supply voltage.

This chapter is structured as follows. The first half of the chapter deals with the BTI reliability study in logic components. Section 3.2.1 starts by introducing the origins and current understanding of the physical phenomena involved in BTI-induced transistor degradations. To analyze these degradations, Section 3.2.2 presents and evaluates the proposed framework,

performing short- and long-term workload-dependent BTI effect analysis in the execution stage of a synchronous processor pipeline. Based on the results produced by this analysis at the technology, circuit and application levels, Section 3.2.3 provides insights on the suitable functional error mitigation techniques to tackle BTI reliability concerns with minimal impact on the performance and energy consumption of the circuits.

The second half of the chapter presents the reliability study in memory components operating at low supply voltages. Section 3.3.1 introduces the different reliability issues in current memory technologies. In order to cope with these issues, Section 3.3.2 summarizes the related works and challenges arising from the mitigation of memory reliability concerns with low energy overheads. Then, to characterize the inherent resilience of biomedical applications against errors in memories, a data significance analysis is performed in Section 3.3.3. This analysis is required to identify the critical data that should be protected, a necessary step in order to set up and evaluate the proposed error mitigation technique presented in Section 3.3.4. Finally, Section 3.4 concludes the chapter by summarizing the main achievements.

3.2 BTI-Aware Logic Circuit Design

Increasing the transistor density per chip, thanks to the shrinking of technology nodes over the years, has been the preferred solution to improve the performance of digital multi-processor systems while reducing their energy consumption and production costs. However, several obstacles oppose this trend, such as process variations and device degradations caused by various effects, including Bias Temperature Instability (BTI), Random Telegraph Noise (RTN), Hot Carrier Injection (HCI), low- and high-κ Time-Dependent Dielectric Breakdown (TDDB), and Electromigration (EM) [66]. Those aging effects lead to an increasing number of reliability issues. In particular, they affect the propagation delays of the signals from one register (i.e., flip-flop) to another, which compromises the proper sequential execution of processor pipelines. Among those aging effects, BTI is of paramount interest because of its dominant impact on timing variability in scaled devices and due to the complexity of its physical mechanisms [141]. In the following subsection, the origins of BTI effects on CMOS designs and the current understanding of its physical phenomena are presented.

3.2.1 BTI Effects: A CMOS Reliability Issue

BTI is a transistor aging effect that manifests as an increase of the threshold voltage (V_{th}) in CMOS technologies. The origins of the induced threshold voltage shift (ΔV_{th}) are located at the physical (i.e., atomic) level. Under a constant gate voltage level and stress temperature ranging from ambient temperature to 200 °C, positive charges (i.e., holes) or negative charges (i.e., electrons) get trapped at the Si/SiO_2 interface, which subsequently affects the transistor V_{th} [142]. In addition, the BTI-induced ΔV_{th} can be the consequence of charge carriers trapping in the gate oxide layer. More precisely, if the gate oxide traps gain enough energy, they may capture charge carriers, leading to a reduced amount of carriers in the channel, thus impacting the transistor V_{th} [141]. Depending on the type of charge carrier (i.e., holes for PMOS and electrons for NMOS), BTI effects can be classified as two different phenomena. On one hand, Negative BTI (NBTI) affects PMOS transistors, while on the other hand, Positive BTI (PBTI) degrades NMOS transistors. NBTI has for a long time been considered as the main reliability issue, while PBTI has been neglected due to it relatively low impact in older technologies. Nevertheless, with the current high-κ dielectric materials, PBTI effects are comparable to NBTI ones [141]. As a consequence, both NBTI and PBTI must be considered when evaluating the impact of BTI on logic circuits. In particular, BTI may alter the switching characteristics of the transistors, resulting in timing violations at the circuit level that can ultimately lead to functional errors at the system level. Furthermore, aggressive transistor integrations, higher voltages and clock frequencies lead to increased operating temperatures inside the chip; since BTI has a strong dependency on circuit temperature, its effects will hence become more pronounced with future transistor downscaling [143, 144]. Additionally, BTI-induced transistor V_{th} degradations can be complemented by other aging effects, such as RTN, HCI and high-κ TDDB [66]. However, those effects are not considered in the BTI impact analyses described in Section 3.2.2.3, since they are out of the scope of this thesis.

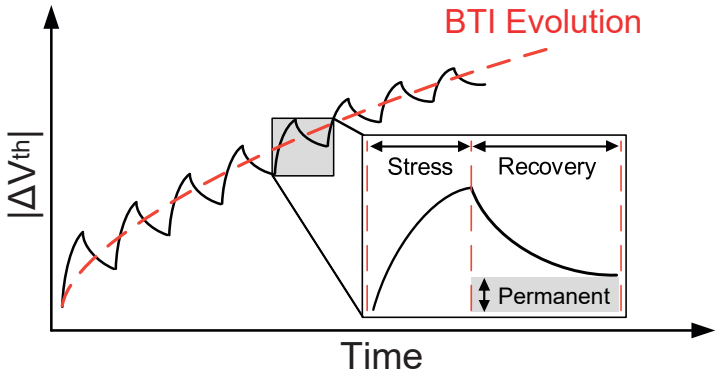


Figure 3.1 – Time-dependent BTI evolution (Modified figure from [133]).

As illustrated in Figure 3.1, the evolution of the BTI degradation is composed of two phases when a time-dependent voltage stress (e.g., square signal) is applied to the transistor's gate. The absolute ΔV_{th} increases during a voltage stress period, and subsequently decreases when the gate voltage is reduced along the recovery (or relaxation) period. The physical mechanisms involved in BTI aging reveal the existence of a recoverable part that disappears when the transistor is switched off, and a permanent one that increases the extent of the previous as the circuit ages. Along time, there has been some controversy in the literature on whether BTI degradation depends on operating frequency [145] or not [146]. Nonetheless, the partially recoverable nature of BTI highlights its strong dependency on the duty factor of each circuit node, i.e., the time that each transistor is in direct (*ON* state) or inverse polarization (*OFF* state).

In addition to temperature and supply voltage, most of the previous research considers the duty factor as one of the main parameters impacting the time-dependent BTI degradation [133, 147, 148]. However, as shown recently in [149, 150], the concrete application workload plays an even more important role in the extent of BTI degradation. In fact, at the circuit level, the application workload modifies the duty factor and stress level of each transistor in the netlist. Also, the duration of the sleeping periods for the complete system (which depends on the amount of work generated in the application to process the current data inputs) affects the relaxation of BTI degradation, producing a partial recovery of the characteristics. Therefore, it is necessary to know the concrete characteristics of the applications that will run on the system to accurately predict the extent of BTI degradation on the long term and determine the actions that should be introduced to guarantee system correctness.

To this end, the next section introduces a long-term and workload-dependent BTI impact analysis framework that incorporates the latest advances in deterministic BTI effects characterization.

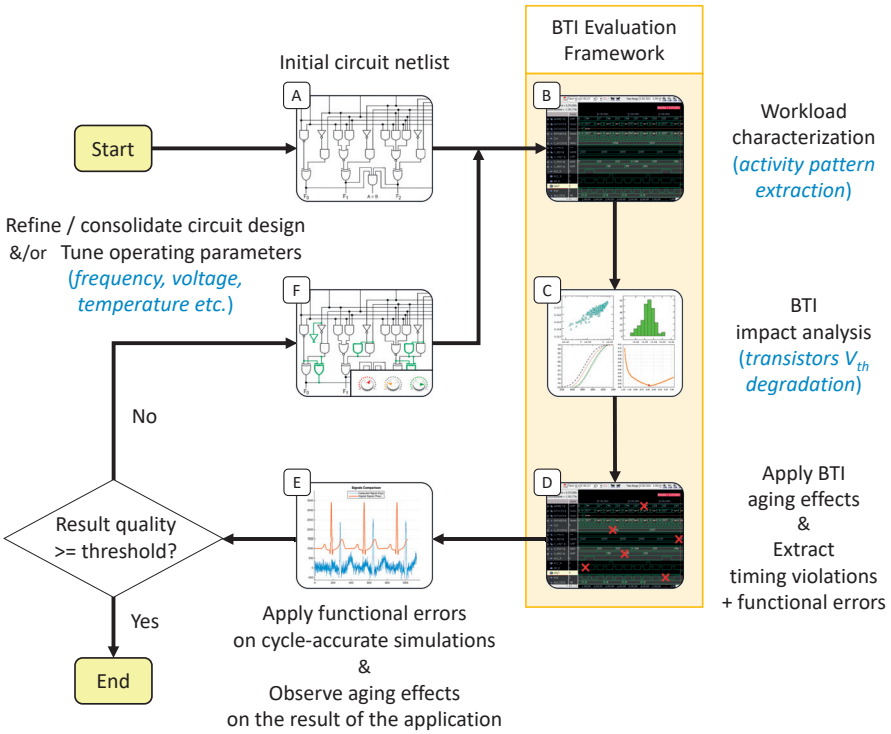


Figure 3.2 – High-level design flow, including the proposed BTI evaluation framework oriented to the analysis of processor pipelines.

3.2.2 Proposed Framework for Workload-Dependent BTI Impact Analysis

In this thesis, I propose the first framework, to the best of my knowledge, that includes workload-based dynamic timing analysis, atomistic trap-based BTI modeling and run-time conditions (e.g., supply voltage, frequency, temperature and aging time) to evaluate their combined impact on circuit functionality. Moreover, this work is the first one to study the high-level effects of BTI-induced functional errors on the output results delivered by a widely-used biomedical application for ECG processing. This framework is applicable to all combinational circuit architectures, such as pipeline stages of single- and multi-core processors, where performance and energy-constraints must be met in scaled technology implementations.

In order to provide a use-case example of the proposed BTI evaluation framework, Figure 3.2 illustrates a high-level representation of the envisioned design flow.

First, the flow starts by extracting the workload activity of each transistor that makes up the initial netlist of the circuit (i.e., *Steps A* and *B*). Second, it analyses the BTI-induced V_{th} degradation based on the aging period and workload stress activity applied to each transistor (*Step C*). Third, the impact of the resulting ΔV_{th} on the timing properties of the circuit is analyzed to extract the possible functional errors induced by timing violations (*Step D*). Fourth, the time-dependent functional errors are injected inside cycle-accurate simulations of the complete DSP platform, to observe the BTI-induced degradations on the output results of

the evaluated application (*Step E*). If the results meet the quality criteria, designers can stop iterating with the flow. However, if it is not the case, the operating parameters and/or the circuit netlist must be modified before starting a new iteration with the BTI evaluation framework (*Step F*). Thanks to the characterization of the circuit reliability concerns (e.g., identification of the faulty bits and operations), designers can take adequate measures to strengthen the circuit by a partial reassessment of the W/L ratios (i.e., Width and Length dimensions) of each transistors or by rebalancing the critical paths in the circuit netlist. However, when these modifications are not sufficient, designers have to rely on the integration of error mitigation techniques to cope with BTI-induced functional errors at run-time (see Section 3.2.3).

The scenario depicted in Figure 3.2 is employed during the experimental evaluation performed in Section 3.2.2.3. Nevertheless, before analyzing the impact of BTI-induced aging degradations, the following sections introduce the necessary background and related work on this topic.

3.2.2.1 Background and Related Work

A) BTI Impact Analysis

To carry out this research, I introduce a novel method to accurately evaluate the long-term effect of BTI-induced transistor degradations on the functionality delivered by the processor pipeline. In contrast with other works, the proposed BTI impact analysis relies on an *atomistic trap-based model* (also termed *defect-centric* or *trapping / de-trapping model*) adapted for deca-nanometer transistors. In addition, the proposed method allows a long-term evaluation of the degradations across the complete transistor netlist (via SPICE electrical simulations), and under the strain of real workloads. This deterministic BTI impact analysis is performed in a reasonable amount of time, and to the best of my knowledge, a BTI evaluation framework which combines accuracy and acceptable execution speed has never been proposed in the literature.

A.1) Atomistic Trap-Based Model

Traditional approaches are based on simplified analytical and probabilistic models to efficiently evaluate the impact of BTI degradation at long-term or even end-of-life (EOL) [146]. Workload impact is considered by simply including the total recovery and stress times in the aging formulas based on the duty factor.
In large transistors, the random differences of individual defects average out, which enables the analytical models to represent accurately the effect of degradation along time. However, the down-scaling of transistors towards tens of nanometers or less reduces the number of defects per transistor responsible for time-dependent effects, making the stochastic nature of each defect (or trap) and its impact on transistor characteristics more relevant [151]. The defect-centric paradigm studies the contribution to transistor degradation of each individual defect through their individual carrier Capture and Emission Times (CET), which can vary

from µs to months [133, 141]. The drawback of the atomistic trap-based model is that it implies a very high computational complexity, which limits its applicability to small sets of transistors or short-term characterizations. This limitation raises an important challenge which must be tackled, especially up to the level of full multi-processor pipelines. In this work, I employ the atomistic trap-based model validated by Kaczer *et al.* [152] to obtain a precise characterization of all the transistors in the circuit during an initial stress period. Then, the results of the model are used to extrapolate the impact of BTI degradation on ΔV_{th} for arbitrarily long periods with affordable computation times.

A.2) Long-Term Extrapolation

Analyzing the degradation over extended periods of time is crucial to truly capture its impact on circuit properties and ultimately highlight functional errors along system lifetime. In particular, the BTI transistor degradation presents a logarithmic behavior with a significant ramp-up period during the first few seconds, corresponding to fast capture events. After longer periods (in the order of months), degradation typically reaches a saturation point. However, the concrete workload, which changes the length of individual stress and rest periods (not only their ratio), may affect the rate of saturation.

A.3) Full Circuit Coverage

As mentioned in Section 3.2.1, BTI affects in different ways P and N-type transistors. In particular, the effect on ΔV_{th}, although not symmetrical, is of opposite sign for each type. This means that the concrete connections of transistors in the circuit netlist determine how NMOS and PMOS transistor degradations interact (i.e., compensating or accumulating) along the signal propagation paths. Therefore, studying the effect of degradation on the complete netlist instead of on individual transistors is essential to observe the real impact on the circuit critical paths. In contrast with previous works based on the defect-centric paradigm and considering individual transistors or small benchmark circuits [153, 154], my research is aligned with studies that tackle the complexity of full circuits representative of real systems such as [155, 156].

A.4) Realistic Workload

Classic analytical studies on BTI degradation used a simple ratio between stress and relaxation times in their formulas to characterize workload activities [146, 157]. While this assumption may be valid for large transistors (i.e., $\geq$ 65 nm) in which the behavior of many effects was averaged, smaller transistors present few defects whose individual impact is much more predominant. Moreover, the use of long and realistic workloads is more than necessary to capture all the relevant effects and outliers contributing to the degradation of the transistor characteristics. In this work, I claim that the actual distribution of stress and relaxation phases for each transistor is relevant to accurately determine its degradation along time with the considered biomedical application workload. In addition, this is inline with previous studies which have underlined the link between the application workload, the duty factor of each

transistor (which is directly determined by the workload and the netlist structure) and the extent of BTI degradation [141, 147, 149, 150]. To this end, I introduce an efficient method to represent pseudo-periodic workloads with flexible granularity allowing trade-offs between accuracy and modeling complexity. In particular, a dynamic granularity varying from few μs to several ms is employed in my experiments, depending on the activity of the processor.

B) Workload Characterization

Rodopoulos *et al.* introduce in [150] a new signal representation, called Compact Digital Waveform (CDW), to reduce the number of simulation steps required by their atomistic model. Their idea is to coalesce periods of input stimuli that have similar characteristics and run the model through them at once. In a related work, Rodopoulos *et al.* introduce accurate BTI degradation estimations, based on pseudo-transient atomistic simulations, but ignoring workload dependencies [158]. In fact, their framework reuses identical stress patterns for every transistor, independently of their relative positions within the netlist and regardless of the activity factors of the nodes. Besides, Stamoulis *et al.* apply CET map modeling to obtain atomistic BTI analysis taking into account full workload dependencies [156]. Their work can be potentially applied to any workload and combinational circuit architecture. However, in their analysis, they have not addressed the aspect of *data latching* on the clock edge in output of pipeline stages, and this is a crucial element to consider as shown further. I build on their expertise by reusing some of the parts of their analysis flow for complete applications. As a major differentiator, I introduce the possibility of accurately evaluating BTI-induced degradation over long periods of time. In their work, to reach that goal, they simply stretch the duration of each CDW point proportionally to cover the total desired period. This trivially enables the analysis over long periods without changing the computational complexity of the analysis. In contrast, I introduce optimizations in the process that produce a more realistic workload during long periods of time. For example, with their proposed model, a periodic workload (such as the ones typical of WBSNs) can be stretched so that one single sample is processed during one year. This is not the equivalent of simulating the system by running the application during that period, but instead it reduces the system working frequency by several orders of magnitude. Indeed, for a periodic application which has 30 % of idle time at the end of each processing period, directly stretching the duration of the CDW points to one year would produce an idle period of more than a hundred days at the end, which does not reflect real working conditions. In a less dramatic example, the fact that BTI degradation can be partially recovered during relaxation periods means that stretching out low activity periods may introduce large recovery effects (i.e., an almost complete release of trapped charges), which are not observed in the actual working conditions. Instead, I tackle the challenge of correctly representing periodic application behaviors and I introduce measures to make the computational complexity of the whole process affordable. Moreover, in comparison to the work of Stamoulis *et al.* relying on the conservative and pessimistic Static Timing Analysis (STA), the proposed framework employs the Dynamic Timing Analysis (DTA) method, providing a better characterization of the workload-dependent timing properties of the circuit without introducing worst-case margins [71]. Lastly, during BTI impact analyses, I also take

into account the effects of data latching by the registers at the output of the considered circuits. As addressed in the next subsection, this establishes a link between BTI-induced timing degradations and the generation of functional errors when the propagated signal does not arrive before the next rising edge of the clock (i.e., processing deadline).

C) Functional Errors Evaluation

To evaluate the BTI-induced impact on functional errors, I extend the framework presented by Stamoulis *et. al* [156]. Their framework starts by tracing the workload at the circuit input, and subsequently propagating it to the inner nodes. Subsequently, through BTI modeling, a workload-dependent degradation is carried out to determine the threshold voltage variation of each single transistor; the obtained ΔV_{th} shifts are then applied to each transistor from the netlist. I extend their model to study BTI effects on the long-term with a reasonable computational cost, as explained in Section 3.2.2.2-B. Additionally, as mentioned in the previous section, instead of relying on the transistor-level STA, whose workload-independent nature represents an important limitation, I perform DTAs via SPICE simulations to evaluate the BTI-induced slack time degradations leading to functional errors.

Similarly, Sivadasan *et. al* propose a workload reliability simulation framework that aims at further minimizing design margins [159]. Nevertheless, their method, based on probabilistic workload indicators to determine path delay margins, exhibits several limitations, such as accuracy losses in the calculation of the aged delay due to worst-case signal duty cycle probabilities. This lack of accuracy has a detrimental effect for the applicability of this method in actual time- and safety-critical designs. On one hand, the probabilistic averaging of the BTI effects removes the possibility to provide any reliability guarantees, because of the ignorance of outlier degradations. As shown in [160], for ultra-scaled technologies, these outlier degradations become more and more dominant and extend even far beyond the 6σ range. On the other hand, this method cannot remove all the unrealistic design margins yet because the workload-dependent effects are also partly averaged out. To remedy these shortcomings, in my work I maximize the accuracy of the workload-induced aging degradation to determine the optimal frequency point of the circuit.

Beyond the accuracy optimization of the proposed BTI-aware analysis framework, I also aim at evaluating the impact of BTI-induced timing violations at the processor pipeline level. In that regard, Chen *et al.* propose a methodology to model at the system level reliability degradations due to BTI effects in microprocessor architectures [161]. However, their proposal is based on analyzing the delays on the most critical paths found with STA; hence, they only give an estimation of the system lifetime, without providing any information at the functional level, such as the type and amount of faulty operations generated by the processor pipeline. In contrast, my framework performs dynamic timing analyses to observe and quantify the rate of BTI-induced functional errors on the circuit outputs over time. Finally, the identified functional errors are injected inside a cycle-accurate simulation framework to evaluate their impact on the quality of the results delivered by the software application.

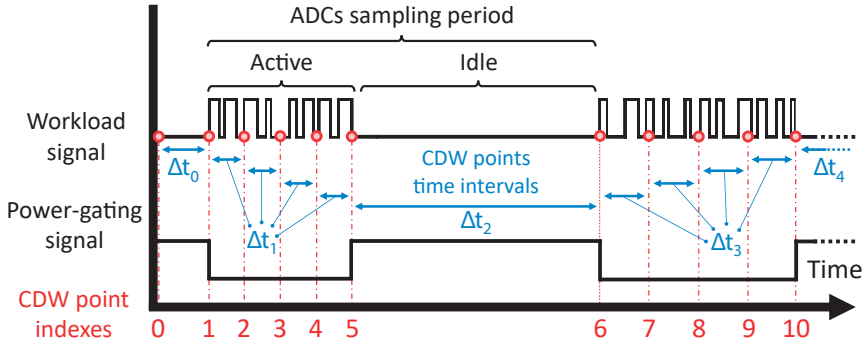


Figure 3.3 – Non-uniform CDW point decomposition.

3.2.2.2 Description of the BTI Evaluation Framework

Based on the related works previously presented, this section describes in detail the different features and modules integrated into the proposed BTI evaluation framework (illustrated in Figure 3.2).

A) Activity-Aware Workload Decomposition

Typical bio-signal processing applications have a *pseudo-periodic* workload composed of alternating active and idle periods to process input samples acquired by Analog-to-Digital Converters (ADCs). The length of each active period varies from one ADC sampling period to another and depends on the concrete actions performed by the application on the sample. Within one active period, the control and data signals of the hardware pipeline toggle more or less intensively, according to the set of instructions executed by the processor. Hence, the duty factor (i.e., the ratio between the *ON* and *OFF* states) of each transistor changes several times within a single active period. Moreover, at a higher level, the duty cycle of the application, characterized by the ratio between the active and sampling periods, depends also on the system operating frequency and ADC sampling frequency. When the system operates at high frequencies, this duty cycle tends to be small which allows partial transistor recovery and large energy savings by power-gating the circuit during idle periods of the processing [50]. This behavior is common in modern circuit designs and has a non-negligible impact on BTI aging trend over time [141].

To account for the dynamic evolution of the BTI-induced V_{th} shift at run-time and to perform an efficient and accurate BTI-aware analysis, the workload (recorded inside a value change dump (VCD) file) is split and represented as a set of continuous CDW points (see Section 3.2.2.1.B) [150, 156]. Each of these points represents a time segment of stimuli signals for which the impact of the workload on the BTI-induced variation is evaluated. A higher number of CDW points increases the accuracy of the BTI-aware analysis at the cost of longer simulation time, while a more coarse analysis averages out significantly the signal activity and the quality of the delivered results [150].

In contrast to earlier literature, in this work I employ a non-uniform decomposition of the workload into CDW points. This method enables a better accuracy of the BTI-aware analysis (for the same simulation effort) by increasing the density of CDW points on the active parts of the workload that produce more stress on the transistors of the circuit. As shown in Figure 3.3, this non-uniform workload decomposition relies on the power-gating control signal of the processor. The traces of this signal are recorded in parallel with the stimuli signals of the workload. The falling and rising edges of the power-gating control signal become the time references used to identify the start and end times of each active computation burst, which are subsequently used for the definition of the CDW point time intervals. Each active period (i.e., computation burst) is uniformly subdivided into several CDW point time segments, whereas a single CDW point is used to characterize the following idle (i.e, non-active) period of the core.

The CDW point decomposition is performed in the *Step B* represented in Figure 3.2. It is particularly relevant in this framework to enable an efficient parallelization of the most complex segments of the workload processing on multi-core servers. More importantly, it reduces the number of BTI model iterations and enables the parallelization of the SPICE simulations over multiple cores during the evaluation of the BTI-induced timing degradations leading to functional errors.

B) Scalable BTI Modeling For Short- and Long-Term Reliability Evaluation

As mentioned in Section 3.2.2.1, the high computational complexity of the considered atomistic trap-based model represents its main drawback. In other words, this model implies lengthy execution times (i.e., from several weeks to several months) when accurately evaluating the BTI-induced V_{th} degradation of a small set of transistors over extended aging periods. To overcome that obstacle and while still retaining accuracy and deterministic modeling as main aims, I have developed a long-term BTI modeling methodology to extend the simulation of realistic workloads from few seconds to several years of transistor aging, without leading to an unacceptable simulation time with high processor and memory requirements. This methodology enables the long-term evaluation of the BTI-induced ΔV_{th} of each transistor from the netlist, based on a user-defined aging time. Its working principle is depicted in Figure 3.4. First, a workload-dependent BTI evaluation is performed with the atomistic trap-based model on all the transistors during a short aging period (e.g., 10 minutes). Then, the time-dependent evolution of the obtained ΔV_{th} curve from each transistor is fitted and

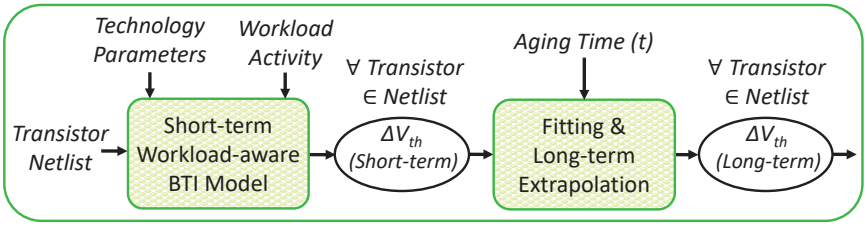


Figure 3.4 – Structural overview of the long-term BTI model, performed in *Step C* in Figure 3.2 and *Step 5* in Figure 3.8.

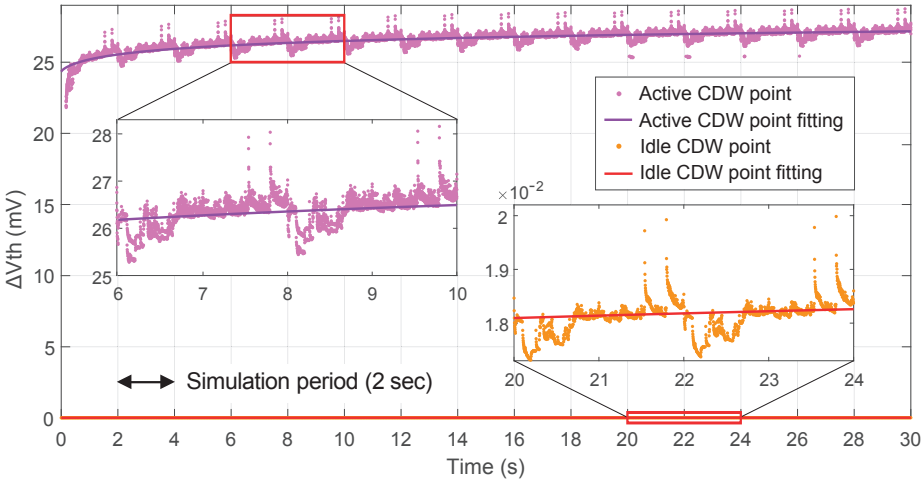


Figure 3.5 – ΔV_{th} evolution illustrating the initial ramp-up phase of BTI-induced degradations from an NMOS transistor. The vertical lines mark each repetition of the application workload (2 seconds).

extrapolated with the help of Equations 3.2 and 3.3. These equations share some similarities with BTI aging analytical models presented in the literature [66, 141]. Moreover, the form of these equations allows a fast fitting and extrapolation operation describing the logarithmic evolution of the workload-dependent ΔV_{th} with the time t.

$$\Delta V_{th}^{fitting}(t) = a \times \left(\log_{10}(t)\right)^{b} + c \tag{3.2}$$

$$\Delta V_{th}^{extrapol}(t) = \left(a + \frac{a \times t}{\Phi}\right) \times \left(\log_{10}(t)\right)^{(b\,+\,(b\times t)\,/\,\Psi)} + \left(c + \frac{c \times t}{\Upsilon}\right) \tag{3.3}$$

The fitting Equation 3.2 is composed of three coefficients a, b and c. In this study, these coefficients are determined for each transistor based on 10 minutes of BTI simulation with the atomistic trap-based model. This duration is long enough to capture the overall trend of the ΔV_{th} curve. Moreover, the BTI simulation is run for all the transistors even when they have the same duty factor to capture the stochastic nature of each individual trap and release event. After the fitting operation, the same three coefficients are reused in Equation 3.3 to extend the ΔV_{th} curve of the transistor to the desired time period (e.g., 10 years).

Nonetheless, three additional calibration factors Φ, Ψ and Υ are required to further adjust the extrapolation. They allow to reach, within a reasonable accuracy margin, the results traditionally obtained by running exhaustive and time consuming BTI simulations with the atomistic trap-based model for prolonged aging periods (see Section 3.2.2.3-B.1). The values of these calibration factors have been determined empirically only once, and are identical for all the equations employed during the analysis. In absence of calibration factors, the estimated ΔV_{th} curve may deviate slightly from its expected value, when the extrapolation extends the ΔV_{th} trend from few minutes to several years. In fact, for each curve, the fitting algorithm is

executed iteratively until it mets a specific quality criteria for the coefficients of the considered fitting operation. A high number of coefficients and a high fitting precision require a long execution time of the algorithm, which is not compatible with the fitting of tens of thousands of ΔV_{th} curves (for all the transistors) in an acceptable time frame. Therefore, a less extreme quality criteria and a reasonable number of fitting coefficients (e.g., three) are used for all the curves. This leads to a tiny fitting imprecision, compensated by the employed calibration factors Φ, Ψ and Υ.

In addition, the three fitting coefficients a, b and c of these equations are determined individually for each transistor of the netlist, but also for the set of CDW points corresponding to the active and idle periods of the workload respectively. Figure 3.5 shows an example of double fitting performed on the short-term evaluation of the BTI-induced ΔV_{th} from an NMOS transistor. The double fitting of the active and idle CDW points allows a more accurate representation of the ΔV_{th} evolution. In particular, the ΔV_{th} values of the idle CDW points are usually small. Thus, a fitting curve with a slow evolution is required to describe this behavior.

Furthermore, compared to the stretching strategy of Stamoulis *et al.* presented in Section 3.2.2.1.B, with the proposed methodology the duration of each CDW point is not expanded to reach the desired aging time. Instead, the real length of each CDW point is conserved and the complete set of CDW points is repeated (NB: In this study, each repetition corresponds to 2 seconds of real workload). The difference is fundamental and reflects better the real behavior of a periodic bio-signal processing application. In this way, active and idle (i.e., stress and relaxation) periods alternate in a realistic manner and represent the workload of WBSNs operating continuously all along the aging duration specified by the user. This solution is also more convenient for designers, since it avoids the generation of extremely long application workload VCD files of several gigabytes or even terabytes of data.

Finally, to assess the accuracy of the proposed long-term BTI modeling methodology, a preliminary study has been carried out by comparing the ΔV_{th} degradations obtained after the execution of the time consuming BTI model, with the ones generated by the proposed methodology. The experimental setup and the results of this preliminary study are presented in Section 3.2.2.3.

C) Analog Signals Discretization Mechanism

To enable and simplify the comparison between the non-aged and aged output signals produced during workload-dependent analyses of the circuit, a conversion from the analog to the digital domain is performed for every SPICE simulation. This conversion relies on a custom *Discretizer* module implemented in Verilog-A and connected to the output ports of the circuit. To avoid any degradation on the accuracy of the conversion, this module is designed in such a way that it has no physical effect (e.g., parasitic load) on the analog waveforms produced during the simulation. For each output signal of the combinational circuit under test, this module mimics the behavior of a flip-flop toggling between two binary states (i.e., “0" and “1"), depending on the value of the analog signal connected to its input. In the context of this

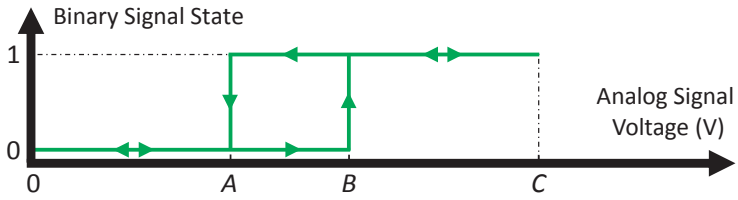


Figure 3.6 – Signal conversion hysteresis from the analog to digital domain, with *A* the transition threshold from “1" (high) to “0" (low), *B* the transition threshold from “0" (low) to “1" (high) and *C* the nominal supply voltage of the circuit.

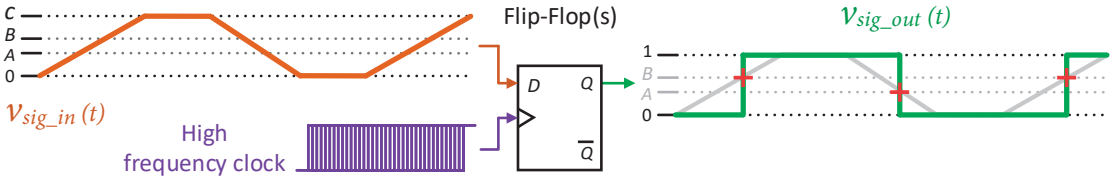


Figure 3.7 – Illustrative example of the transistion thresholds determination. A high frequency clock and a slow input signal (with a small slew rate) are used to determine precisely the transition points *A* and *B* of the flip-flops.

study, this flip-flop plays the role of a register interfaced between two consecutive processor pipeline stages that latches the binary value of the signal at the end of each clock cycle.

As depicted in Figure 3.6, to improve further the accuracy, the conversion performed by this module relies on two transition thresholds (i.e., *A* and *B* on the hysteresis), depending on the direction (i.e., high to low or low to high voltage) of the analog signal to convert. These threshold voltages have been characterized independently of the flow, by using a dedicated circuit and the parameters of the experimental setup presented in Section 3.2.2.3-A.

To accurately determine the values of *A* and *B* for the used technology node, I analyzed a circuit composed of several flip-flops, toggling at a very high frequency (several GHz). An illustrative example is shown in Figure 3.7. At the input of the flip-flop, a slow trapezoidal signal $v_{sig\_in}(t)$ is applied and evolves with a tiny slew rate SR_{sig_in}, compared to the standard one SR_{std} of the binary signals usually propagated through the circuit under test. Equation 3.4 shows the relation between these two metrics.

$$SR_{sig\_in} = \left| \frac{dv_{sig\_in}(t)}{dt} \right| << SR_{std} \tag{3.4}$$

When the input signal exceeds the transition threshold *B*, the binary signal in output of the flip-flop goes from “0" to “1". Conversely, when the input signal drops below the the transition threshold *A*, the output binary signal toggles from “1" to “0". With this setup, I determined the transition threshold voltages *A* (0.399 V) and *B* (0.455 V) with an accuracy of ±1 mV, when the output signals $v_{sig\_out}(t)$ of the considered flip-flops toggle from one state to the other.

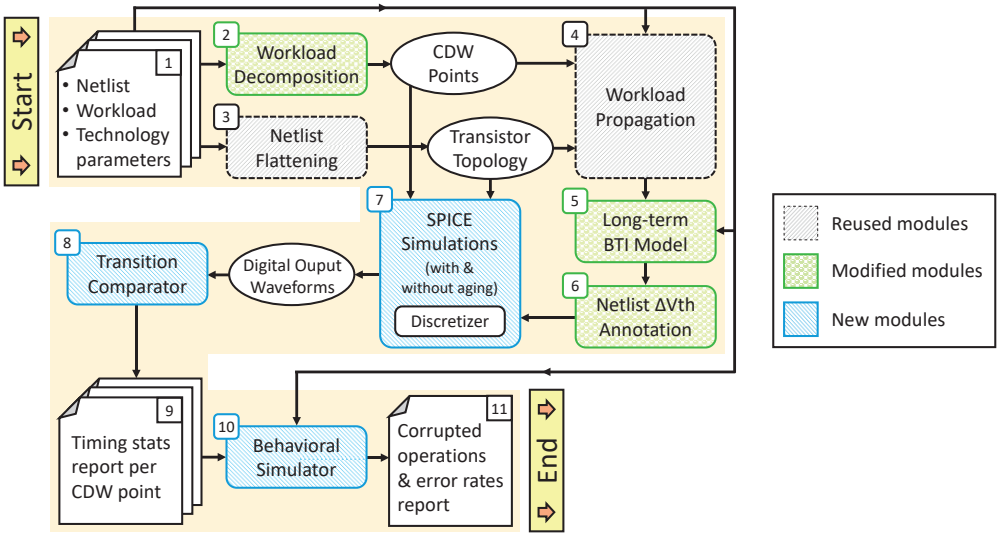


Figure 3.8 – Overview of the proposed workload-aware BTI evaluation framework.

D) Modules Description and Execution Sequence

Figure 3.8 presents the execution sequence of the 11 steps that compose the BTI evaluation framework. Some of these steps or *modules* have been reused from the initial proposal in [156] (i.e., *Steps 3* and *4* in Figure 3.8). Others have been modified (i.e., *Steps 2, 5,* and *6*) or are completely novel (i.e., *Steps 7, 8,* and *10*) to reflect the improvements described above. Compared to the high level representation of the framework in Figure 3.2, the *Step 1* in Figure 3.8 belongs to *Step A* in Figure 3.2, *Steps 2* to *4* are performed in *Step B*, *Step 5* is executed in *Step C*, and *Steps 6* to *11* are integrated in *Step D*.

Once configured and calibrated, the processing performed by the different steps is fully automated. Their execution is performed in-order by following a sequence described inside a Perl script. The design of some of these steps (i.e., *Steps 1, 4, 5* and *7*) is build around commercial software programs and Electronic Design Automation (EDA) tools, as shown in the following paragraphs and in Section 3.2.2.3-A.

Step 1: As an initial step, the complete application workload is stored inside a VCD file, which contains the evolution of all the binary input stimuli signals of the circuit. These stimuli traces can be generated by a testbench or by a behavioral cycle-accurate simulator of the complete system (see Section 3.2.2.3-A.3). They will be applied to the inputs of the circuit under test, described by a gate-level Verilog netlist, and produced by traditional synthesis tools such as Synopsys® Design Compiler (DC) [120].

In addition, in *Step 1*, the technology parameters are provided along with the calibration and configuration parameters of the framework, which specify for instance, the operating temperature of the circuit, the supply voltage, the frequency and the number of CDW points.

Step 2: The workload previously generated is subsequently decomposed into CDW points using the power-gating control signal of the processor, as explained in Section 3.2.2.2-A. In

parallel, in order to be interpreted by a SPICE simulation software, the workload of each CDW point is converted into a standard digital vector (VEC) file.

Step 3: The circuit netlist is flattened from a gate to a transistor netlist, as a preparation for workload propagation and SPICE simulations.

Step 4: To preserve the workload dependency throughout all the nodes of the circuit, the stress activity patterns (which can be characterized in terms of frequency f, duty factor α and duration Δt) are propagated across all the transistors of the netlist. For each CDW point (i.e., workload segments with the activity stress patterns), switch-level simulations of the circuit are performed thanks to the simulator Incisive from Cadence® [162]. These simulations allow to derive the signal activities at each transistor, from which the f and α values are determined [156]. These parameters, which capture the V_{gs} stress voltage of each transistor, are subsequently provided as input for the atomistic BTI modeling stage.

Step 5: The methodology presented in Section 3.2.2.2.B for long-term BTI modeling is used in this step to calculate efficiently the ΔV_{th} of each transistor after the desired aging period. From that point, a full BTI modeling of silicon defects based on Capture and Emission Time (CET) maps is performed using the real workload during the studied application period. Then, the short-term BTI degradation obtained for each transistor and for each CDW point is fitted and extrapolated with the help of MATLAB® [163], to derive the long-term BTI degradation on the V_{th} of the transistors.

Step 6: The resulting workload-dependent ΔV_{th} shifts are annotated for each transistor of the flattened SPICE netlist and for each CDW point.

Step 7: The annotated transistor netlist enables the execution of aging-aware SPICE simulations with Synopsys® CustomSim [164], before evaluating the BTI effects on the timing properties of the circuit with the DTA method. The *Discretizer* module, presented in Section 3.2.2.2.C, is employed in this step to convert the output analog waveforms into digital values throughout the execution of the SPICE simulations.

Step 8: A transition comparator has been implemented and is used to compare the binary values of the non-aged versus aged output signals obtained after the SPICE simulation performed for each CDW point.

Step 9: A set of statistics is produced for each CDW point and signal transition on the rising edge of the clock. For instance, the statistics reports contain the slack time, the propagation time from input to output, the delta time between the next rising edge of the clock and the correct value of the signal when a timing violation occurs, etc. Every mismatch between the non-aged and the aged output signals on the rising edge of the clock is flagged to highlight timing violations.

Step 10: Due to masking effects, each timing violation does not necessarily imply an error at the functional level of the circuit. For example, a timing violation may occur on a signal that is

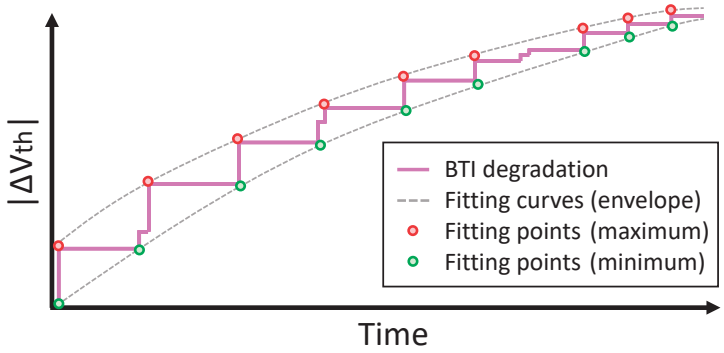


Figure 3.9 – Time-dependent BTI degradation with deeply-scaled transistor technologies (BTI degradation trend inspired from [151]).

not considered as part of the result of the current operation. Therefore, a behavioral simulator has been developed to compare the outputs of the circuit with the correct ones and to evaluate the impact of timing violations on the functionality of the circuit. By mimicking the expected behavior (i.e., all the operations) of the circuit, this simulator provides the *ground-truth* for the output signals corresponding to operations from the input application workload (i.e., *Step 1*). This lightweight behavioral simulator implemented in C++ has a similar working principle with industry-standard design tools such as Mentor Graphics® ModelSim [121].

Step 11: Lastly, the simulator produces a final report with the list of corrupted operations, associated to their timestamps and input operands. Apart from identifying functional errors, this information may also be used to extract the error probability for every bit in the output signals of each operation.

E) Framework Modifications for Advanced Transistor Technologies

In context of future works, to employ the proposed framework with more downscaled transistor technologies (e.g., FinFET 7 nm [165]), few modifications are required in the configuration or design of *Step 1, 5* and *7*. These modifications are highlighted in following paragraphs.

Step 1: Firstly, the parameters (e.g., nominal threshold voltages, Pelgrom's constant) provided in the configuration file of the framework must be updated with the ones provided by the proper technology model card. The content of this new model card will be employed all along the execution of the framework to characterize the transistors of the circuit netlist.

Step 5: Secondly, the parameters of the atomistic trap-based BTI model (e.g., number of traps, calibration parameters) must be tailored to the physical properties of the new technology. In this study, a maximum of 1000 traps per transistor gate (32 nm) is considered. By reducing the W and L dimensions, less traps per transistor gate must be simulated in order to produce accurate and realistic BTI-induced V_{th} shifts with the employed model and technology.

Moreover, as shown in [151], the stochastic nature of a small set of traps in deeply-scaled transistors becomes more apparent, resulting in significant variations of the BTI degradation over time. In particular, as illustrated in Figure 3.9, a ΔV_{th} degradation trend with a large

staircase or *sawtooth* shape is obtained due to the higher contribution of each single charge carrier captured or released by gate oxide traps. Nonetheless, due to the absence of available traps over time leading to a saturation effect of the V_{th} shifts (see Section 3.2.2.3-B.1), the amplitude of each degradation step is progressively reduced and converges to a single curve.

In order to fit and extrapolate such complex evolution with the proposed methodology (see Section 3.2.2.2-B) and based on a short-term degradation trend, a double fitting procedure must be employed for each set of idle and active CDW points. To do so, a sliding analysis window can be used to detect the last point from a given ΔV_{th} degradation step and the first point from the next one. These points characterize the transitions from one step to another, when one or several charge carriers are captured by the gate oxide. A filtering algorithm (e.g., adaptive thresholding, outlier removal) can be employed to discard the small or intermediate steps, which do not contribute to the general trend of the ΔV_{th} degradation. Then, each set of points can be individually fitted and extrapolated to derive the minimum and maximum long-term evolution of the ΔV_{th} degradation trend, based on the same procedure presented in Section 3.2.2.2-B, and after calibration of the constants Φ, Ψ and Υ.

Because of the saturation effect and monotonic evolution of the BTI degradation, the two minimum and maximum (envelope) curves obtained for each set of idle and active CDW points will naturally converge to a single average curve. Therefore, the maximum deviation between the two envelope curves is progressively reduced, which minimizes the error of the long-term BTI model after an extended period of time.

Step 7: Finally, the last modification of the framework concerns the transition thresholds of the *Discretizer* module (see Section 3.2.2.2-C). These thresholds must be reevaluated based on the switching properties of the flip-flops implemented with this new technology.

3.2.2.3 Experimental Evaluation

A) Experimental Setup

Before evaluating the proposed framework and the impact of BTI degradations at the transistor, circuit, functional and application levels, this section introduces the experimental setup and is organized as follows. First, a preliminary analysis is carried out to identify the most sensitive part of a domain-specific processor pipeline to BTI-induced timing degradations. The identified combinational circuit (and sub-circuits) will be used as benchmarks during the experimental evaluation of the framework. Secondly, I present the selected biomedical application employed to generate the input stimuli signals (i.e., stress workload) of the circuits under test. Then, the simulation environment and the experimental parameters are described. Finally, the execution time of the framework is studied to highlight the impact of the experimental conditions (e.g., workload activity and size of the circuit) on the time required to perform the workload-dependent BTI analysis detailed in the following experimental evaluation.

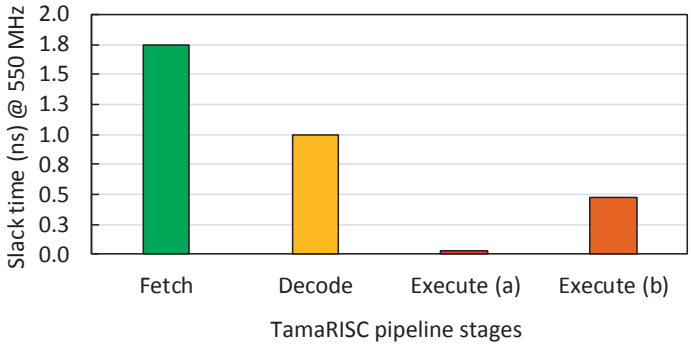


Figure 3.10 – Slack time of the three TamaRISC pipeline stages at a frequency of 550 MHz.

A.1) Domain-Specific Benchmark Circuits

Similarly to the architectural exploration performed in the Chapter 2 of this thesis, I target in this experimental evaluation the *TamaRISC* architecture (see Section 2.3.1.1). This processor is tailored to energy-efficient bio-signal processing applications and the three stages of its pipeline are suitable to the analysis of BTI aging degradations. However, not all the stages are equally sensitive to BTI-induced timing degradations. In particular, a preliminary static timing analysis demonstrates that the execution stage has a smaller slack time (i.e., smaller maximum frequency achievable) compared to the other stages of the pipeline. Figure 3.10 shows the results obtained by performing a transistor-level STA on the different stages with Synopsys® NanoTime [166]. For comparison, the execution stage has been synthesized in two different versions. The original version (a) embeds an Arithmetic and Logic Unit (ALU) with a 32 bits multiplier output, while the modified version (b) features a multiplier with a truncated 16-bit output. In both cases, and according to STA, the execution stage is the first component of the pipeline to be affected by timing degradations that violate the safety time margins. Hence, the ALU component of the execution stage represents an interesting benchmark to perform the envisioned BTI analysis. The original architecture of this component (i.e., with a 32 bits multiplier output) is considered in this study.

The internal structure of the ALU16 circuit is depicted in Figure 3.11. It can perform signed/unsigned integer arithmetic and logic operations widely used in DSP applications, such as: addition/subtraction (Adder16), multiplication (Mult16) and multiply-accumulate (MAC8). The latest performs two operations in series within a single clock cycle. It merges the parallelism of a multiplier architecture with the sequential nature of the adder carry signal, propagated through a long critical path. This complex hybrid organization presents interesting architectural properties, which are employed to evaluate the proposed framework and its ability to handle any type of digital combinational circuit.

In addition to the ALU16, three other circuits (i.e., Adder16, Mult16, and MAC8) are also considered in the first part of the BTI analysis. For more details, Table A.1 in Appendix A lists the different operations supported by the ALU16. In order to optimize the design area and obtain a consistent comparison, the MAC unit reuses the multiplier (Mult16) with an input

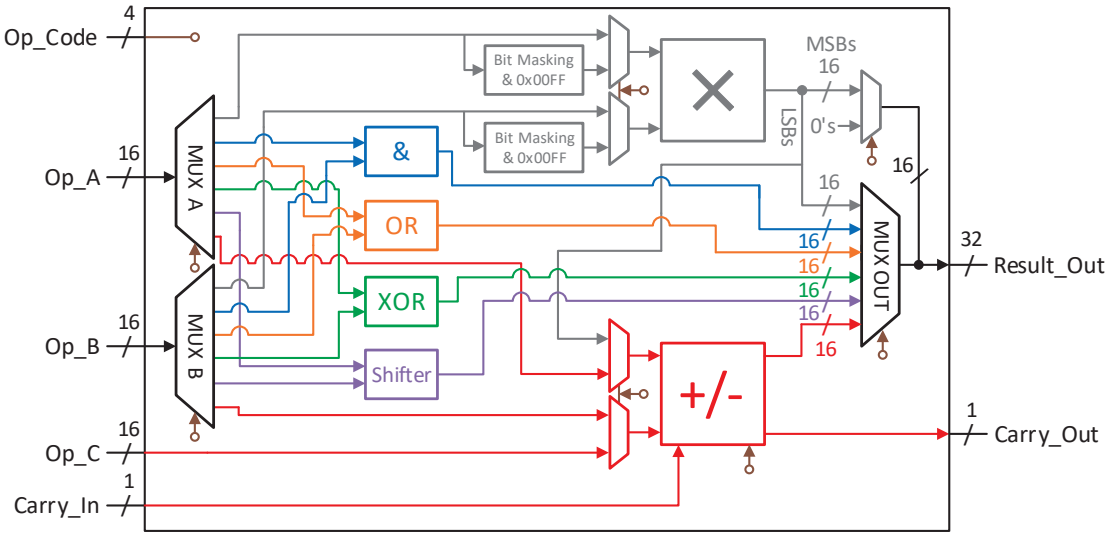


Figure 3.11 – Simplified schematic of the ALU16 circuit. NB: the reset signal has been omitted for clarity reasons.

operand size of 8 bits and considering only 16 bits from its output. A pipeline datapath size of 16 bits has been selected since it is a common size in energy-efficient embedded devices, which process bio-signal samples encoded on 16 bits [24, 122].

Typical SPICE-like analog circuit simulators, such as Synopsys® HSPICE [167], face difficulties to analyze 1000 transistors per netlist in a reasonable amount of time. However, as shown in Table 3.1, I consider medium and large netlists with several thousands of transistors. This is possible with this framework because the division of the workload into CDW points enables easy parallelization of the SPICE simulations. Moreover, I selected an optimized simulator, namely Synopsys® CustomSim [164], to further accelerate the simulations, by taking advantage of the different optimization options offered by this multi-core program.

As concerns the timing properties of each benchmark circuit, Table 3.2 shows the maximum frequency obtained (at a slack time of 0 ps) performing STA with Synopsys® NanoTime for each benchmark circuit.

Table 3.1 – Benchmark circuits from a 16-bit pipeline execution stage.

Circuit	Description	#Gates	#Transistors
Adder16	16-bit adder with carry	65	724
Mult16	16-bit combinational multiplier	1276	13270
MAC8	8-bit combinational multiplier + 16-bit accumulation (i.e., addition)	1368	14046
ALU16	16-bit Arithmetic and Logic Unit	1672	16704

Table 3.2 – Maximum operating frequency as determined with STA (slack time = 0 ps), for each benchmark circuit.

Circuit	Max frequency (MHz)	Period (ps)
Adder16	1645	608
Mult16	562	1778
MAC8	555	1803
ALU16	552	1811

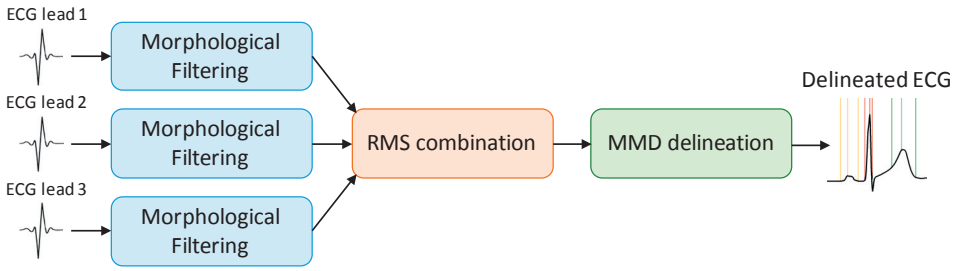


Figure 3.12 – Task graph of the 3L-MMD benchmark.

However, STA is known to have limited accuracy under some scenarios [71]. An alternative is the DTA, which uses input stimuli vectors to drive the activity of the transistors in analog SPICE simulations. The maximum operating frequency obtained with DTA is normally higher than the worst-case values obtained with STA. However, DTA does not take into account circuit degradation due to factors such as BTI-induced effects. Therefore, significant guard bands are introduced in the maximum frequency to guarantee correct operation of the circuits [168, 169]. In my experiments, I show that the workload-aware BTI analysis based on DTA can determine safe maximum operating frequencies without the need for worst-case guard bands.

A.2) Biomedical Application Case Study

To analyze the effects of BTI at different levels in a WBSN platform, a complex real-time bio-DSP application has been considered in this study. This single-core application has been selected for its generic algorithmic structure and wide range of operations. It is representative of most DSP applications, as it executes tasks within fixed time boundaries on a digital synchronous processor under a tight energy budget. Moreover, the pseudo-periodic nature of its workload is an interesting characteristic for this study. In contrast to standard experimental conditions, the pseudo-periodic workload fosters the partial recovery effect of BTI after each active period, which may lead to reduced V_{th} degradations.

The chosen cardiac monitoring application (3L-MMD), depicted in Figure 3.12, performs a three-lead ECG delineation using Multi-scale Morphological Derivatives (MMD) [122]. The different steps which compose this application are commonly employed in the WBSN domain, either as the main processing blocks [14, 170] or associated to additional ECG analysis algorithms [123, 171]. The first part of this application relies on a three-lead Morphological

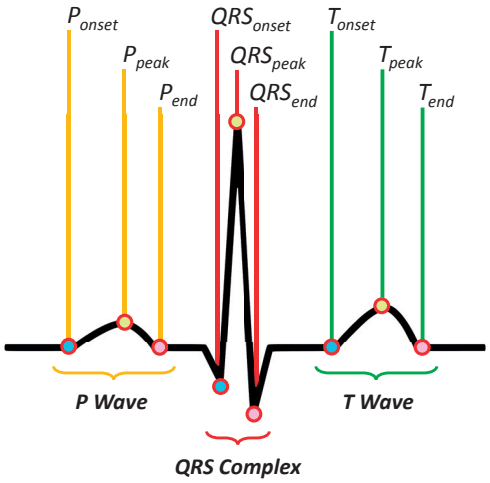


Figure 3.13 – Example of delineated heartbeat (*black line*) with ECG fiducial points.

Filtering (MF) algorithm. It removes artifacts (due to muscle activity, system AC supply interferences and base drift caused by breathing) from an ECG acquisition, according to the algorithm described in [61]. Then, the second processing part merges the filtered streams through a Root-Mean-Square (RMS) combination. Finally, the third and last step performs the delineation of the ECG heartbeats, to identify nine fiducial points represented in Figure 3.13.

The occurrence of BTI-induced functional errors on the data produced by this application may lead to wrong medical results or even to system crashes while being used by patients. Therefore, the proposed framework can prevent these errors by determining appropriate operating frequency limits.

To evaluate the BTI-induced degradation of the results produced by this application, the ECG fiducial points from a P wave, QRS complex or T wave, are classified into three individual categories: *Correct*, *Misplaced* and *Missing*. These categories are exclusive, which means that a single point cannot belong to two or three categories at the same time. An ECG fiducial point is considered as *Correct* when it is present on the heartbeat and when it respects the sequence of points shown in Figure 3.13. In addition, to be considered as *Correct*, the fiducial point must belong to the normal ECG time ranges delimiting the P wave, QRS complex and T wave regions of the heartbeat [172, 173]. If a point is present on the heartbeat within the proper time ranges, but does not respect the sequence shown in Figure 3.13, it will be classified as *Misplaced*. Finally, if the point is absent from the analysis or outside of the allowed time ranges, it will be classified as *Missing*.

The aforementioned quality metrics are used to perform the application-level BTI effect analysis, presented in Section 3.2.2.3-B.4. Moreover, in order to provide a finer-grained analysis of the application results degradation, two additional metrics are employed. First, the average time deviation of the ECG fiducial points classified as *Correct* is computed with respect to

the original position of the points, when no degradations are introduced during the delineation. Secondly, the Percentage Root-mean-square Difference (PRD) is used in this study to characterize the quality degradation of the filtered and combined ECG leads from which the delineation is performed.

$$PRD = \sqrt{\frac{\sum_{i=0}^{n-1}(x_{theo}(i) - x_{exp}(i))^2}{\sum_{i=0}^{n-1} x_{theo}^2(i)}} \times 100 \tag{3.5}$$

The PRD metric is defined in Equation 3.5, where the $x_{theo}(i)$ (theoretical) elements correspond to the error-free samples from the original ECG signal, while the $x_{exp}(i)$ (experimental) elements are the corrupted samples obtained under BTI-induced functional errors injection. This metric is commonly employed to assess the diagnostic quality of compressed ECG records [19]. A link between this metric and the diagnostic distortion has been established in [174]. This work on the weighted diagnostic measure for ECG signal compression, classifies the different PRD values based on the signal quality perceived by cardiologists. In particular, from 0 % to 2 % and from 2 % to 9 % the ECG signal has a *very good* and *good* quality respectively, while above 9 % the signal has a *poor* quality. As shown in Section 3.2.2.3-B.4, these quality classes are particularly interesting to determine the maximum frequency that allows the system to produce biomedical results with an acceptable accuracy. Additionally, in the context of safety-critical biomedical applications, a PRD value of 0 % and an average time deviation of 0 ms demonstrate that a maximum accuracy of 100 % is achieved by the applications even in the presence of BTI-induced transistor degradations.

A.3) Simulation Environment and Parameters

As part of the framework, several knobs are available to users to fit their own needs in terms of speed, accuracy, technology node and circuit operating conditions. In this study 3000 CDW points per simulation period are employed, a worst-case operating temperature of 80 °C and a supply voltage of 1.05 V. This value corresponds to the nominal voltage of the technology library Synopsys® EDK 32 nm with high-V_{th} and Slow-Slow (SS) process corner. The parameters of the transistors are defined by the 32 nm Predictive Technology Model (PTM), including a default V_{th} of −450.0 mV for PMOS transistors and 508.8 mV for NMOS transistors.

For each frequency point, a circuit-specific workload was derived from the execution traces of the 3L-MMD biomedical application. Each workload period (repeated for long-term BTI analysis, see Section 3.2.2.2.B), represents 2 seconds of signal processing (i.e., 3000 CDW points). This allows the generation of sufficiently representative workload traces, characterizing the full range of operations executed by the application (equivalent in time to 1000 ECG samples acquired and processed at a frequency of 500 Hz). The traces were generated with a cycle-accurate SystemC simulator that implements the *TamaRISC* processor architecture, described in [14].

This *TamaRISC* simulator was also employed to perform the BTI impact analysis on the quality of the results produced by the 3L-MMD application. The Listing A.1 in Appendix A provides an example of a BTI-induced faulty operations record generated by the framework. This report lists for each CDW point the different faulty operations along with their input operands and corrupted output data. In the proposed application-level analysis (see Section 3.2.2.3-B.4), the faulty results are directly injected in output of the ALU16 circuit during the cycle-accurate execution of the 3L-MMD application on the simulator. The effect of the corrupted operations is then automatically propagated through the logic of the entire system and has an impact at higher level on the quality of the processing performed by the application. The metrics presented in Section 3.2.2.3-A.2 are subsequently employed to characterize the degradations occurring on the biomedical results delivered by the application.

As concerns the generation of the circuit netlists, they were synthesized with Synopsys® Design Compiler [120] and by using a minimum area constraint to reduce the number of logic gates. This constraint is commonly employed to lower the manufacturing costs of the chip (i.e., occupied silicon surface) and at the same time, minimize the energy consumed by the circuit, to meet the tight energy budget of WBSNs (see Chapter 1). As a reference point, each timing analysis is started with a frequency equal to the maximum frequency obtained with the STA method for the considered benchmark circuit. In all the experimental section, when the frequency is not specified, this reference is always employed.

Lastly, the proposed long-term BTI modeling methodology (see Section 3.2.2.2.B) was validated by using the 8 most active, 8 least active and 8 randomly selected transistors from the ALU16 circuit. The ΔV_{th} shifts produced by the exhaustive 1 year BTI simulation of these transistors were compared to the degradations obtained after the fitting and extrapolation of 10 minutes BTI simulations with the proposed methodology.

A.4) Framework Execution Time and Workload Activity

The proposed BTI evaluation framework consists of eleven consecutive execution steps (see Figure 3.8) that are globally serialized and internally parallelized for each CDW point in the proposed implementation. Thus, it is important to examine the temporal contribution of each single step based on the chosen setup, to efficiently plan the simulations in advance during the design flow. Their execution time depends mainly on three factors:

- Circuit size (number of transistors) affects linearly the BTI modeling (*Step 5*) and sub-linearly (for the type of circuits studied) the SPICE simulation time (*Step 7*) (i.e., the most expensive step).
- The number of signal transitions and the length of the workload affect linearly the BTI modeling and the SPICE simulations.
- Design characteristics (number of input signals, paths and topology) impacts the SPICE simulation time in a complex way: Circuits with long transistor chains, rather than short and parallel paths, require longer simulation times to propagate transitions across levels.

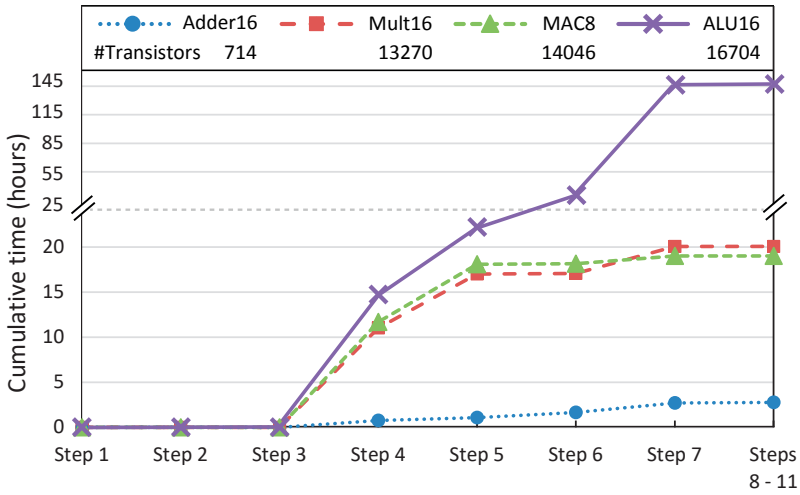


Figure 3.14 – Cumulative framework run-time averaged over all the experiments.

To highlight these dependencies, Figure 3.14 reports the average cumulative execution time of the framework over all the experiments and for the different benchmarks. The most time-consuming steps are the switch-level simulations for workload propagation (*Step 4*), the ΔV_{th} computation in the BTI model evaluation (*Step 5*) and the transistor-level SPICE simulations (*Step 7*).

Table 3.3 – Workload activity summary for each benchmark circuit for the considered 3L-MMD biomedical application.

Circuit	Number of input (bit) signals	Number of output (bit) signals	Bit togglings (transitions) / 2000 ms
Adder16	34	17	14 104 544
Mult16	37	33	541 008
MAC8	54	17	95 614
ALU16	54	33	18 001 541

To supplement Figure 3.14, Table 3.3 shows the number of transitions from the workload of each circuit, which are specific to the studied biomedical application. This table justifies the differences in execution time observed in Figure 3.14. For instance, the multiplier Mult16 receives 5.7x more input transitions than the MAC8, which increases by 1.06x its analysis time with a smaller circuit size. Similarly, the higher workload and bigger circuit of the complete ALU16 makes it harder to analyze.

Nonetheless, with the proposed long-term BTI evaluation methodology, significant execution time savings are achieved. In particular, for a single transistor, the 1 year ΔV_{th} evaluation based on the exhaustive BTI model takes approximately 5.3 days, while with the proposed methodology only 7.4 hours are required to perform the long-term BTI evaluation of all the transistors that compose the ALU16 circuit.

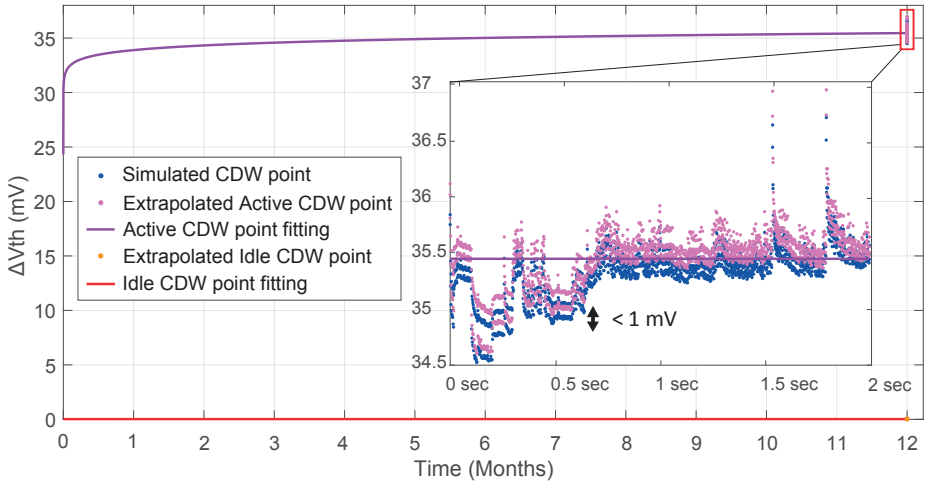


Figure 3.15 – Fitting extended to 1 year (*DC offset*), with final *AC* activity corresponding to one application workload of 2 seconds.

Moreover, at the aggregate level of the framework, further time optimizations can be performed, but at the cost of several degradations on the accuracy of the simulations (e.g., the time resolution of SPICE simulations can be reduced or the workload can be approximated with less CDW points to perform faster analyses). In some circumstances, a lower accuracy may not be acceptable, especially when evaluating the effects of small BTI-induced timing violations. However, without any accuracy penalty, thanks to the decomposition of the workload into CDW points, an efficient parallelization of the different steps can be further achieved by executing the framework on parallel machines. This is usually not an easy task with conventional timing analysis flows. The aforementioned exploration of trade-offs between accuracy and execution time of the framework is reserved for future works.

B) Experimental Results

Based on the bottom-up approach introduced in Section 3.1.1 and following the presentation of the experimental setup, I evaluate hereafter the resulting effects of BTI-induced degradations at four different abstraction levels. More precisely, I start this reliability evaluation at the transistor level, followed by an assessment of the propagated BTI effects at the circuit and functional levels. Finally, I conclude this multi-level evaluation by analyzing and quantifying the degradations obtained at the output of the considered application.

B.1) Transistor-Level BTI Effects Analysis

As mentioned in Section 3.2.2.2-B, a preliminary study has been carried out to validate the proposed BTI modeling methodology for long-term transistor ΔV_{th} evaluations. An example showcasing the quality of the fitting and extrapolation performed for one transistor is illustrated in Figure 3.15. This figure depicts the ΔV_{th} evolution of an NMOS transistor aged for 1

year with the considered application workload. More particularly, a zoom is performed on the 2 seconds following the 1 year aging of the transistor. Two methods have been employed to compute the ΔV_{th} value associated to each CDW point represented on this curve. On one hand, the simulated (active and idle) CDW points have been produced by executing the exhaustive and time consuming BTI model for 1 year plus 2 seconds of transistor aging. On the other hand, the extrapolated CDW points have been generated with the proposed scalable BTI model (see Section 3.2.2.2-B), and then by adding 2 seconds of full BTI simulation at the end of the aging period. By superimposing the two series of points generated by each method, a small deviation of 0.15 mV can be observed in this example. However, this deviation remains negligible as it represents only 0.41 % of the maximum ΔV_{th} degradation. Similar results have been obtained with the other most active, less active and randomly selected transistors used for this preliminary study. In particular, the absolute deviation is always inferior to 1 mV, which is more than acceptable compared to the simulation time savings resulting from the proposed long-term BTI evaluation methodology and based on the stochastic nature of the atomistic trap-based BTI model.

Table 3.4 – Maximum transistor V_{th} degradation measured after 2 seconds of circuit operation with accumulated BTI aging of 0 seconds, 1 year and 10 years, for the ALU16.

Transistor Degradations \ Aging Duration		t = 0 sec (New circuit)	t = 1 year (Aged circuit)	t = 10 years (Aged circuit)
PMOS	**Max $\lvert\Delta V_{th}\rvert$ (mV)**	37.1	54.6	58.3
	V_{th} increase (%)	8.1	12.1	13.0
NMOS	**Max $\lvert\Delta V_{th}\rvert$ (mV)**	36.3	48.1	51.4
	V_{th} increase (%)	7.3	9.5	10.1

Thanks to this methodology, the threshold voltage degradations from all the transistors which compose the complete ALU16 circuit have been evaluated. Table 3.4 summarizes the main ΔV_{th} statistics recorded with a stress workload of 2 seconds and after 0, 1 and 10 years of accumulated BTI aging. The evolution of the maximum PMOS and NMOS ΔV_{th} confirms the degradation trend depicted in Figure 3.15. In fact, NBTI and PBTI are *front-loaded* phenomena characterized by a steep V_{th} degradation immediately after time zero, and which slows down or saturate rapidly [157]. The steep increase is the result of a large amount of charge carriers that get trapped in the gate oxide of new transistors (i.e., at the beginning of the circuit's life). However, when the circuit ages, only few available traps manage to capture some extra charge carriers, thus leading to a progressive saturation of the time-dependent ΔV_{th} increase.

Moreover, the absolute maximum ΔV_{th} obtained after 10 years, such as 58.3 mV for a PMOS transistor, is consistent with commonly observed BTI-induced degradations for the same aging time [153]. As shown in the literature, the temporal evolution of V_{th} degradations is also dependent on the supply voltage, frequency, temperature, technology and transistor stress activity (e.g., duty cycle), which opens up a wide design space exploration [66, 133, 147, 175].

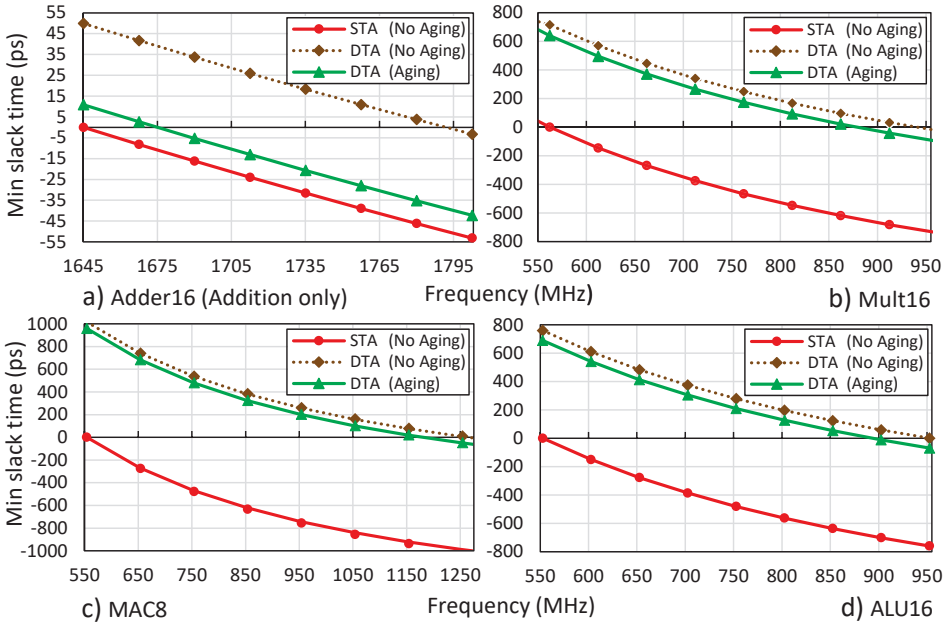


Figure 3.16 – Minimum slack time for different operating frequencies and benchmark circuits, measured after 2 seconds of operation without and with accumulated BTI aging of 0 seconds (i.e., new circuits). The X axis (slack time = 0 ps) marks the maximum safe frequency determined with each technique.

However, in order to keep a reasonably-sized study of BTI reliability concerns in logic circuit, I focus more specifically in the next sections, on the effects of aging time and frequency with the considered bio-signal processing application.

B.2) Circuit-Level BTI Effects Analysis

Slack Time Evaluation: Following the V_{th} degradation at the transistor level, the first observable effect of BTI-induced degradations at the circuit level is the increase of the gate switching time. To quantify this effect, the slack time can be employed as a metric to characterize the reduction of the minimum time margin left after the execution of an operation within a single clock cycle.

Figure 3.16 shows the slack time evolution of the different benchmark circuits, according to the STA, DTA, and DTA with BTI aging at time = 0 seconds (i.e., new circuits). The graphs indicate that, at the studied technology node (i.e., 32 nm), the gap between the maximum frequency determined with DTA and the frequency that guarantees correct operation even when considering BTI-induced effects is moderate given quantitative measure. Therefore, tight boundaries on frequency guard bands can be defined. Moreover, the comparison between the curves for the Adder16 (Figure 3.16.a) and the rest shows that the distance between DTA and the safe frequency cannot be determined beforehand as it depends on the characteristics of the concrete workload and circuit under study. Therefore, general approximations to establish universal guard bands on the maximum frequency of the circuit are suboptimal as they would

lead to performance levels similar to those obtained with STA.

As an example, Figure 3.16.a shows that the Adder16 circuit can operate safely with a frequency up to 1.8 % higher than the maximum frequency determined through STA for a slack time of 0 ps (see Table 3.2), whereas relying on aging-oblivious DTA would potentially drive the system into functional errors (this is explored in Section 3.2.2.3-B.3). However, the other benchmarks can safely operate at frequencies closer to DTA: 116 % over STA for MAC8 and 57 % over STA for Mult16, in both cases with a slack time of 0 ps and when the circuit is new (i.e., aging time = 0 seconds). Hence, relying on worst-case guard bands would impose unnecessary limits on the performance of the latter ones, potentially leading to increased energy consumption through a longer execution time. Alternatively, the knowledge provided by the framework enables the selection of lower supply voltages for the same latency target, thanks to a more accurate maximum frequency determination for each circuit, workload and operating voltage combination.

Even if for the studied 32 nm technology node BTI degradation is moderate, in smaller nodes its effect is expected to have a much deeper impact on the maximum operating frequency of the circuits [143, 144]. Moreover, as the gap from the frequency determined by non-aging DTA increases, the proposed framework can help designers to determine more accurately the safe operating point of their circuits.

Timing Violations Evaluation: In addition, apart from the evolution of the slack time, the framework can be employed to derive the maximum timing violation for a specific frequency point, considering BTI degradation effects. Figure 3.16 also illustrates the variation of the maximum timing violation obtained by increasing the frequency above the STA reference for a slack time of 0 ps. In fact, when the slack time curve becomes negative, an extra amount of time is required to finish the execution of an operation after the violated clock cycle deadline. In particular, Figure 3.16 shows that the Mult16 circuit starts to have timing violations above 57 % over STA, while the MAC8 circuit (which, as explained in Section 3.2.2.3.A, shares the same multiplier Mult16) suffers from timing violations above 116 % over STA with a slack time of 0 ps. Even if the MAC8 circuit is structurally bigger than Mult16 (see Table 3.1), MAC8 features shorter critical paths. Therefore, as depicted in Figure 3.16, the framework suggests the safe use of higher operating frequencies with the MAC8 benchmark. Moreover, the workload processed by the Mult16 operator is heavier than the one applied on the MAC8 circuit (see Table 3.3). As the MAC8 operator is less often used by the application, it suffers less BTI-induced degradation. In addition, this benchmark operates on two 8-bit operands and one 16-bit operand with an output of 16 bits, whereas the Mult16 benchmark processes only 16-bit operands, with an output of 32 bits. For the ALU16 circuit, this effect is masked because both operators share parts of their structure and are hence partially subject to similar degradations: MAC8 circuit includes approximately half of the Mult16 critical path and adds its own 16-bit adder on top of that. The hybrid structure of MAC8 benchmark (see Section 3.2.2.3.A) also demonstrates the possibility of using the proposed framework with a wide range of combinational circuit architectures built with several interconnected operators in a single pipeline stage.

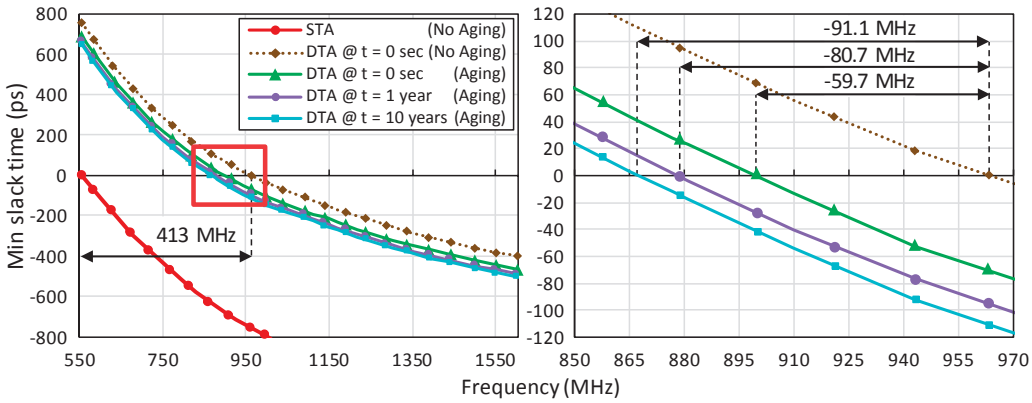


Figure 3.17 – Minimum slack time at different working frequencies for the ALU16 (32 bits output multiplier). The X axis (slack time = 0 ps) marks the maximum safe frequency determined with each timing analysis technique and aging time.

Following the individual evaluation of the Adder16, Mult16 and MAC8 benchmark circuits, Figure 3.17 provides a closer look to the time-dependent BTI-induced degradations affecting the complete ALU16 circuit. Similarly to Figure 3.16, Figure 3.17 represents the evolution of the slack time according to the STA and DTA methods. However, several transistor aging durations are now considered. In particular, the BTI effects on the timing properties of the ALU16 circuit are evaluated after 2 seconds of circuit operation with accumulated BTI aging of 0 seconds, 1 year and 10 years. As shown in Figure 3.17, the observations made previously at the transistor level (see Section 3.2.2.3-B.1) are also valid at the circuit level. Most of the BTI-induced slack time degradation is occurring during the first months of the circuit operating life (see Figure 3.15). Then, the slack time degradation starts to progressively saturate, leading up to 8.4 % frequency reduction (i.e., -80.7 MHz) after 1 year, and up to 9.5 % frequency reduction (i.e., -91.1 MHz) after 10 years, compared to the maximum operating frequency obtained with DTA and no BTI aging (i.e., 963 MHz).

These results showcase that BTI aging has an impact of approximately 10 % on the maximum operating frequency with the considered experimental setup. As a consequence, this performance degradation must be taken into account during the design of the biomedical device. In particular, the timing violations resulting from slack time or frequency reductions have adverse consequences on the functionality delivered by the circuit. To demonstrate this point, the results presented in the next section characterize and quantify the effect of BTI-induced functional errors on the operations performed by the ALU16 circuit.

B.3) Functional-Level BTI Effects Analysis

Functional Errors Evaluation: The application of the proposed BTI evaluation framework enables safe circuit operation without worst-case frequency guard bands. However, this framework can also accurately quantify and identify the faulty bits and operations that produce erroneous results. In the following paragraphs, I explore the impact of timing violations on the circuit functionality at two different levels.

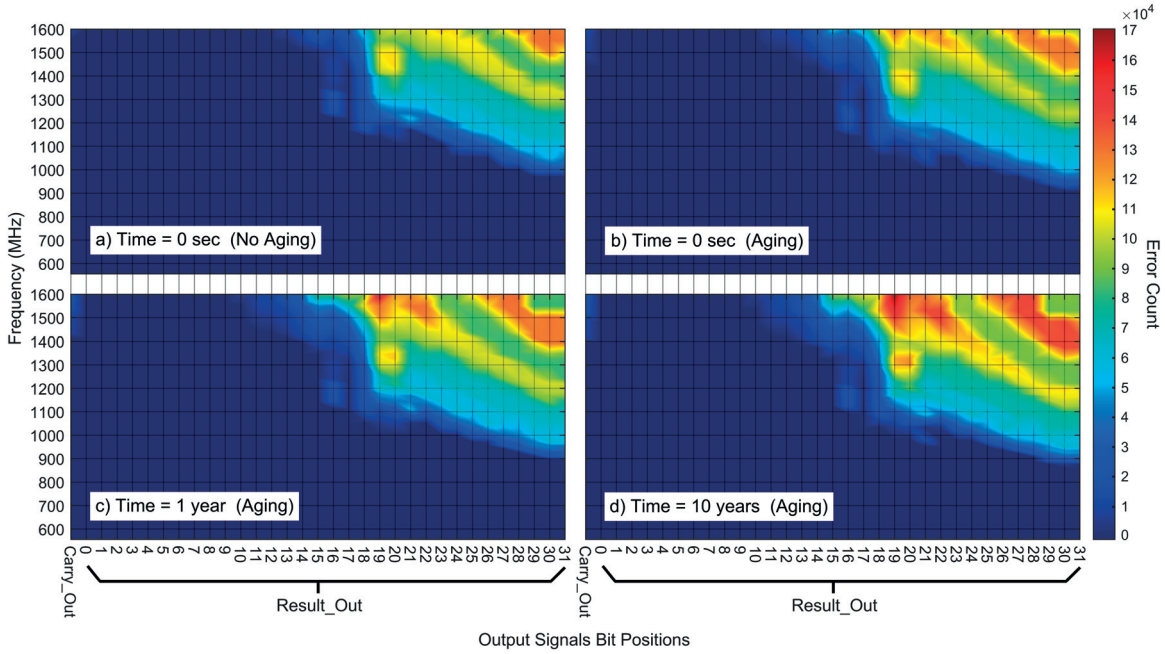


Figure 3.18 – Heatmaps: BTI degradation of the ALU16 output signals (*Carry_Out*, *Result_Out*) with the operating frequency, measured after 2 seconds of circuit operation without and with accumulated BTI aging of 0 seconds, 1 year and 10 years.

First, a bit-level analysis is performed on the output signals of the ALU16, by using the DTA method with and without BTI aging. Figures 3.18 and 3.19 represent respectively in two and three dimensions, the output signals degradations with the operating frequency, measured after 2 seconds of circuit operation without and with accumulated BTI aging of 0 seconds, 1 year and 10 years. Warmer is the color represented on these graphs, higher is the number of faulty bits in output of the circuit with the frequency.

As shown in Section 3.2.2.3.A, the 16 Most Significant Bits (MSBs) of the *Result_Out* signal are only employed by the Mult16 circuit producing a 32 bits result with the current architecture. Furthermore, the *Carry_In* and *Carry_Out* signals of the ALU are only used when the operation to execute employs the Adder16 circuit. Thus, it is possible to observe errors on the *Carry_Out* signal when the frequency is higher enough to also notice errors on the MSBs of the output result delivered by the Adder16 (i.e., bits 15, 14, 13 and so forth). In fact, the *Carry_Out* signal is generated at the end of a chain of 1-bit Full Adders (FAs), producing each bit of the output result and propagating an internal carry signal from one full adder to the next one, until the end of the chain.

In addition, Figures 3.18 and 3.19 show several valleys with bit error counts close to 8×10^4. These valleys are surrounded by ridges with bit error counts between 10×10^4 and 17×10^4. The observed reliefs are not only a consequence of the workload and aging of the circuit. In fact, they are mainly a signature of the internal circuit topology and the operational behavior of the multiplier (Mult16). Their origins stem from the architecture of this specific circuit and

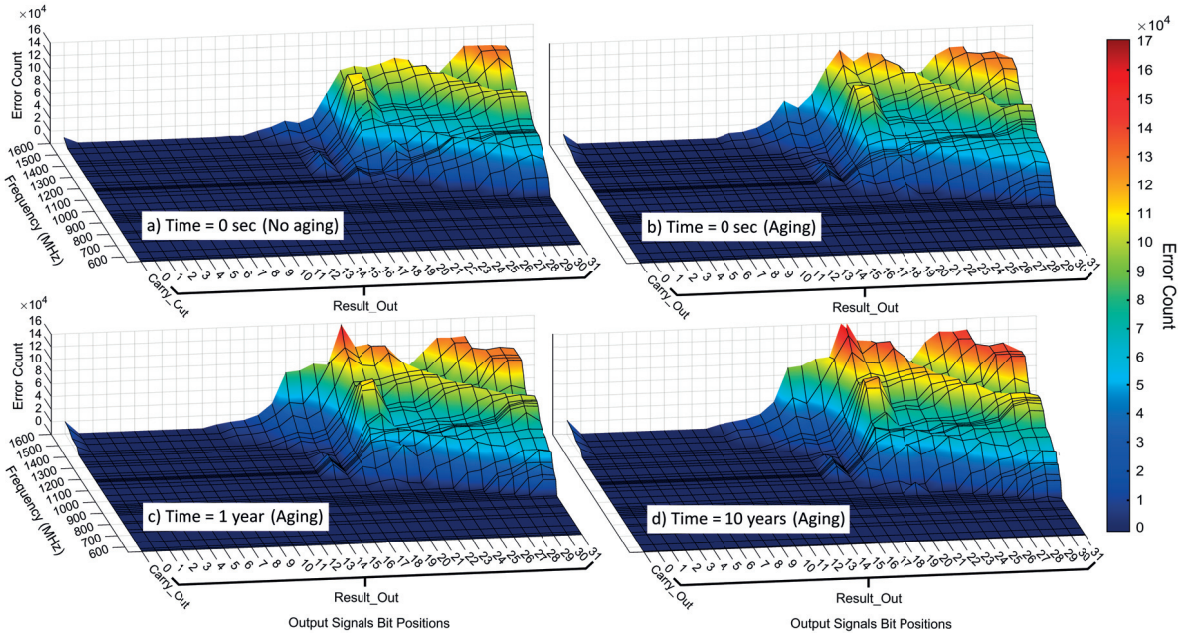


Figure 3.19 – 3D Surfaces: BTI degradation of the ALU16 output signals (*Carry_Out*, *Result_Out*) with the operating frequency, measured after 2 seconds of circuit operation without and with accumulated BTI aging of 0 seconds, 1 year and 10 years.

from the way the signals are propagated through it. In particular, the synthesis tool has tried to balance the critical paths leading to a type of *wave* of bits streaming through the parallel matrix of 1-bit adder cells. The carry ripples lead to a diagonal skewing of the wave which is clearly visible in the final results generated in a gradual manner.

Moreover, due to the propagation chain of the circuit, once a bit in a position is erroneous, the bits with higher weights must all be considered as undefined or potentially erroneous. Therefore, the important area of the heatmaps and 3D surfaces is the frontier where the first erroneous bit appears. Once the frontier is crossed, some bits may be correct since they are possibly subject to an even number of flips. To support these explanations, the Listing A.2 of Appendix A gives an example of signed multiplication operation with all the intermediate results delivered at the output of the Mult16 circuit. In this simple example, the final result of the multiplication is composed of 32 × 0's in a row and takes 731.57 ps to be computed. If the clock period is inferior to this duration or if the circuit takes more time to perform the operation due to aging, the result of the multiplication will be erroneous because of timing violations. Its value will be randomly composed of 0's and 1's, similarly to one of the intermediate results shown on the Listing A.2. In order to produce the Figures 3.18 and 3.19, a comparator is employed to evaluate bit per bit the correctness of the operations performed by the ALU16. The 0's of the faulty result will be consider as correct, since the expected multiplication result is only composed of 0's in this example. Thus, the error count of these bits will be lower compared to the other bits consider as faulty. This leads to the creation of valleys or colored patterns in Figures 3.18 and 3.19 when the frequency is increased.

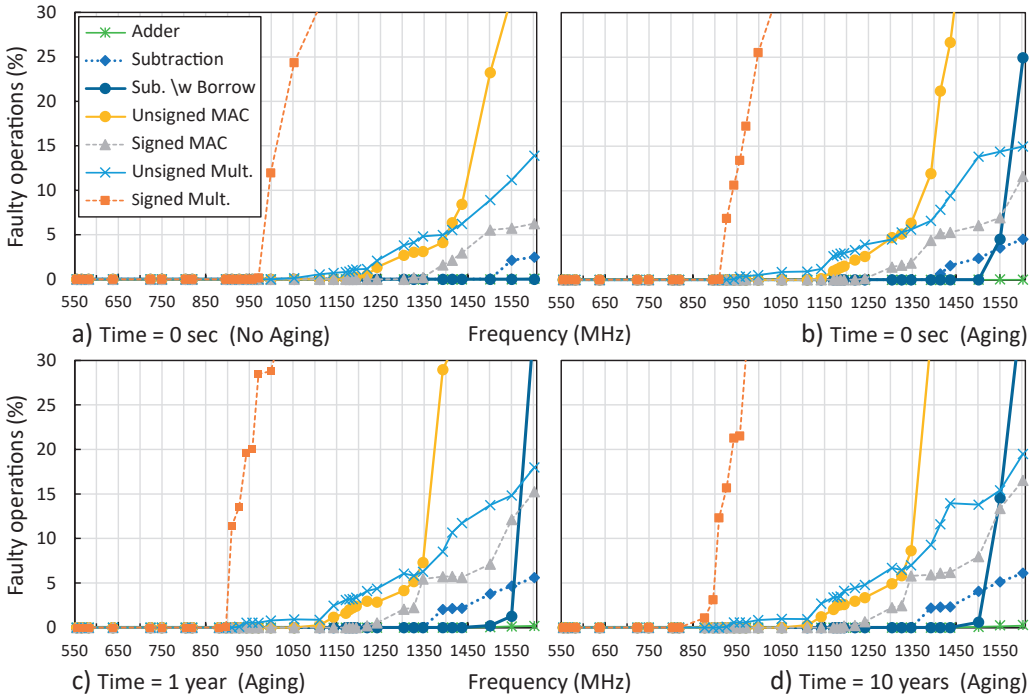


Figure 3.20 – Evolution of the percentage of faulty operations for the ALU16 (32 bits output multiplier), at time = 0 seconds, 1 year and 10 years.

Furthermore, the actual workload (i.e., the precise sequence of 0's and 1's presented in input of the multiplier) has a smaller impact on these reliefs, and apart from tiny differences, the diagonal wave pattern will remain, as discussed previously. In this way, the general position of the ridges and valleys in the figures is mainly defined by the internal circuit structure and adder cell operations.

In addition, the aging is not changing drastically the general shape of these ridges and valleys, but in fact, they are progressively moved in a diagonal direction towards the lower frequencies due to BTI-induced timing degradations. This evolution highlights the occurrence of more and more BTI-induced functional errors with the time. In particular, if we compare the frontiers where the first functional errors start to appear without aging (a) and after 10 years of BTI aging (d), we observe a shift of approximately 100 MHz after 10 years, which is in line with the evolution of minimum slack time observed in Figure 3.17.

Faulty Operations Evaluation: From a higher perspective, the effect of timing violations is also visible at the operation level. Figure 3.20 represents the percentage of faulty operations when the system frequency exceeds the maximum safe point and taking into account BTI timing degradations for several aging durations. As depicted by the four graphs of this figure, the percentage of faulty operations performed by the ALU16 increases with the frequency, highlighting the presence of more and more timing violations when this operating parameter is set beyond the maximum safe point (i.e., the highest possible frequency with a faulty operation

percentage equal to zero). However, each timing violation does not necessarily imply a faulty operation, since they may affect signals which are not part of the final result delivered by the operation. To this end, the proposed framework can accurately identify the situations that lead to incorrect results. Additionally, when the circuit lifetime (i.e., aging time) is extended, faulty operations start to appear at lower frequencies. This observation is consistent with the time-dependent displacement of the ridges and valleys illustrated in Figures 3.18 and 3.19 and with the slack time evolution previously presented in Figure 3.17. More precisely in Figure 3.20, the signed and unsigned multiplications are failing first when the frequency is increased above 850 MHz. In fact, as discussed in the previous section, the multiplier Mult16 is the first component to be affected by timing violations due to its longer critical path and more intensive workload.

Based on the results provided by these bit- and operation-level analyses, two observations can be derived.

First of all, with the proposed framework, it is possible to predict the degradations in output of the ALU16 circuit for a given set of operating parameters. As shown by the different figures, the BTI degradation is moderate with the current setup (i.e., approximately 100 MHz of maximum safe frequency deviation after 10 years). In fact, inside a specific CDW point period, some of the transistors are affected with high V_{th} shifts compared to the others. However, with the considered bio-signal processing workload and when operating at high frequencies (i.e., >50 MHz), the computing platform spends most of its execution time in sleep mode (i.e., long idle time periods after each processed sample). Thanks to the small active/idle time ratio of the platform, the most degraded transistors have enough time to recover from the stress accumulated during the active periods of the processing. These long relaxation periods lead to a smaller BTI impact, since only the permanent part of the BTI degradation will be conserved all along the extended execution of the platform. In this context, the observed behavior will be similar for smaller technologies. In particular, it underlines the opportunity of using more advanced technology nodes at lower voltages and with a lower energy consumption (and/or higher frequencies), but without being impeded by significant BTI degradations.

Secondly, because of the weak and progressive evolution of the BTI degradation after the first months of execution (see Figure 3.15), the operating parameters of the circuit can be extended to a region where the BTI impact is not null. The next section explores the inherent resilience of the considered biomedical application to functional errors induced by timing violations.

B.4) Application-Level BTI Effects Analysis

The visible impact of timing-induced functional errors on the output results produced by the application is illustrated in Figure 3.21. Similarly to previous experiments and except for the operating frequency, all the parameters presented in the experimental setup (see Section 3.2.2.3) are kept constant for each of the three simulations. As shown by this figure, the quality degradation of the delineation follows the operating frequency increase. Moreover, when a significant number of functional errors occur during the processing of the signal, it

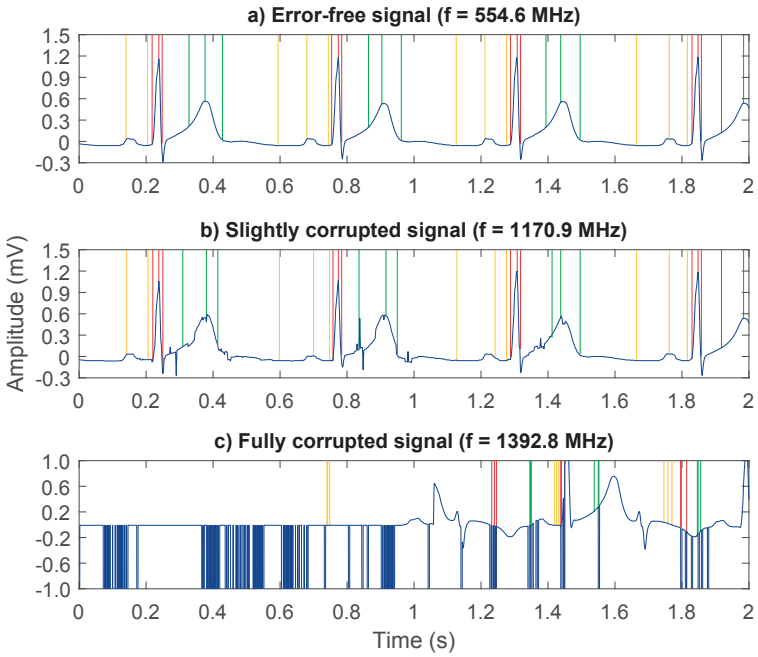


Figure 3.21 – Example of signals impacted by BTI and overclocking functional errors relative to an error-free ECG after 10 years. NB: These graphs have been produced by simulations not impacted by functional errors leading to a crash of the complete system.

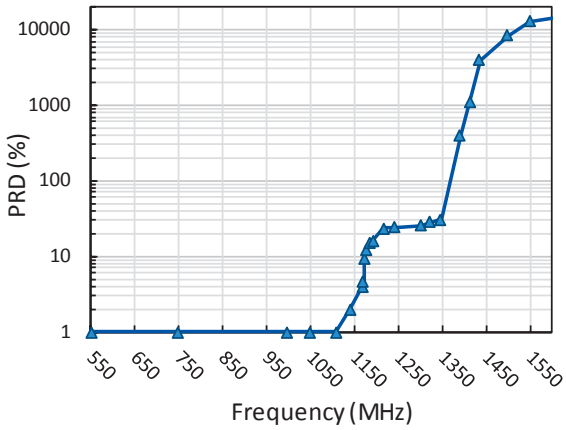
becomes possible to observe saturation effects on the ECG amplitude, similar to the ones in Figure 3.21.c. In the worst-case, the most critical functional errors may even lead to an entire crash of the system. As a consequence, no output data are produced and the system must be rebooted. These situations can be identified at run-time thanks to error detection and recovery mechanisms such as the one proposed in [56].

To quantify these degradations, Figure 3.22 depicts the evolution of the selected metrics (see Section 3.2.2.3-A.2) assessing the quality of the results produced by the ECG delineation application for several operating frequencies and after 10 years of BTI aging. A more detailed record of the different points is provided in Table A.2 of Appendix A.

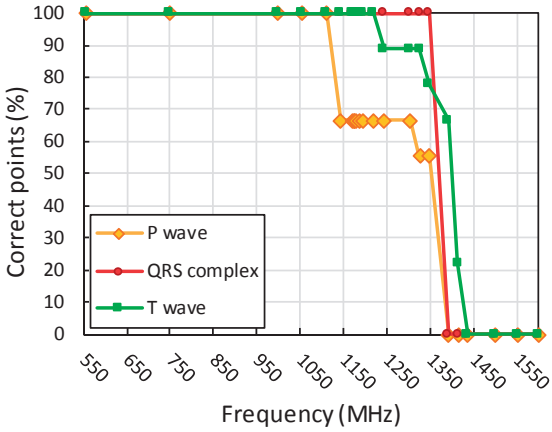
The graphs in Figure 3.22 confirm the observations made previously in Figure 3.21, by highlighting a clear dependency of the delineation quality with the operating frequency above 1110 MHz. However, the early timing violations occurring on the MSBs of the multiplier at lower frequencies, are not affecting the quality of the delivered results. In fact, this is due to a peculiarity of the delineation application which only uses the 16 Least Significant Bits (LSBs) of the results produced by the multiplier. In other words, the application results are only degraded when the *Carry_Out* or one of the 16 LSBs from the *Result_Out* signal are corrupted by functional errors (see Figure 3.20). In this way, the knowledge of the application requirements offers the opportunity to maximize the frequency beyond the maximum safe point determined by the STA and DTA methods.

Additionally, based on the quality classes associated to the PRD metric, Figure 3.22.a underscores the possibility to operate with a frequency up to 1172 MHz (112 % over the STA frequency) and with an acceptable ECG signal quality (i.e., PRD < 9 %). However, as depicted in Figure 3.22.b, a degradation by one-third of the percentage of P wave points correctly delineated must be tolerated when operating at such high frequency. Conversely, this degradation increases the number of misplaced and missing delineated points, as represented in Figure 3.22.c and 3.22.d. When the frequency is further increased, the misplaced points are replaced by missing points, thus generating a *bump* in Figure 3.22.c. Furthermore, as shown by Figure 3.22.e, an average time deviation by 11 ms (i.e., 5.5 sampling periods) and 4 ms (i.e., 2 sampling periods) of the correct P wave and T wave delineated points respectively, is observed when the system operates at 1172 MHz. Nonetheless, these time deviations remain relatively small compared to the normal duration of the P wave (~95 ms) and T wave (~190 ms) [172, 173]. In addition, all these issues are not critical for many high-level biomedical applications which only require the accurate delineation of the QRS complex from each heartbeat, such as for instance, epilepsy detection and obstructive sleep apnea detection [176, 177]. The identification of the QRS complex from each heartbeat is usually easier compared to the P wave and T wave points, since the amplitude of R peaks is significantly bigger compared to the other points represented on ECG signals (see Figure 3.13).

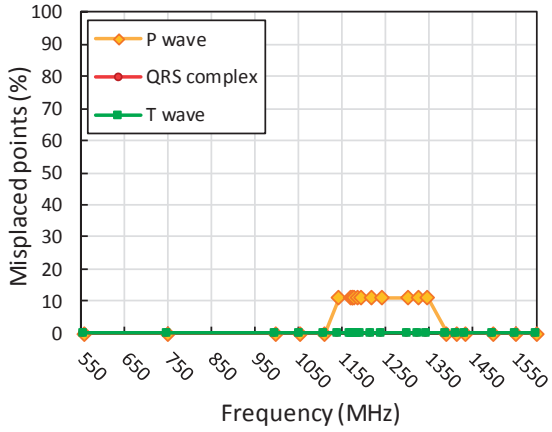
As a consequence, the precise knowledge of application-level degradations identified by the proposed framework enables the design of *approximate systems* that knowingly trade-off accuracy for performance (or energy efficiency) in the context of WBSN systems with limited resources. Finally, for *safety-critical biomedical systems* that rely on exact computing, a lower frequency must be employed (e.g., 1110 MHz corresponding to 101 % over the STA frequency with the current setup) to guarantee error-free computations, even in the presence of outlier degradations. However, to conserve high operating frequencies, an orthogonal strategy consists of implementing specific hardware and software techniques to detect and mitigate the BTI-induced functional errors impacting the reliability of the system. This is in fact the only safe solution for future scaled technology nodes to guarantee high system dependability, since the effects of outliers will lead to too frequent functional failures, despite a low average degradation. The following section provides several insights on the potential techniques to mitigate those issues at different levels.



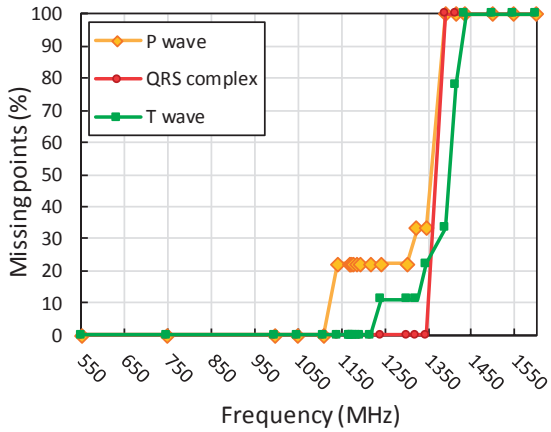
a) ECG signal quality evaluation.



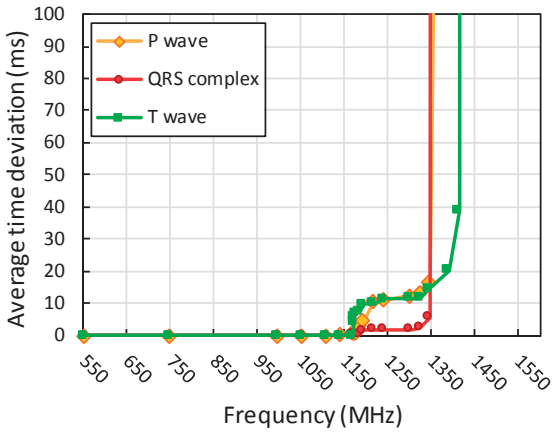
b) Correct ECG fiducial points obtained during the delineation.



c) Misplaced ECG fiducial points obtained during the delineation.



d) Missing ECG fiducial points not found during the delineation.



e) Average time deviation of the correct ECG fiducial points obtained during the delineation.

Figure 3.22 – Quality assessment of the output results generated by the MMD delineation application, impacted by BTI functional errors after 10 years. NB: As a time reference in Figure e), with a sampling frequency of 500 Hz, a new ECG sample is acquired every 2 ms, which corresponds to the time distance between two consecutive samples.

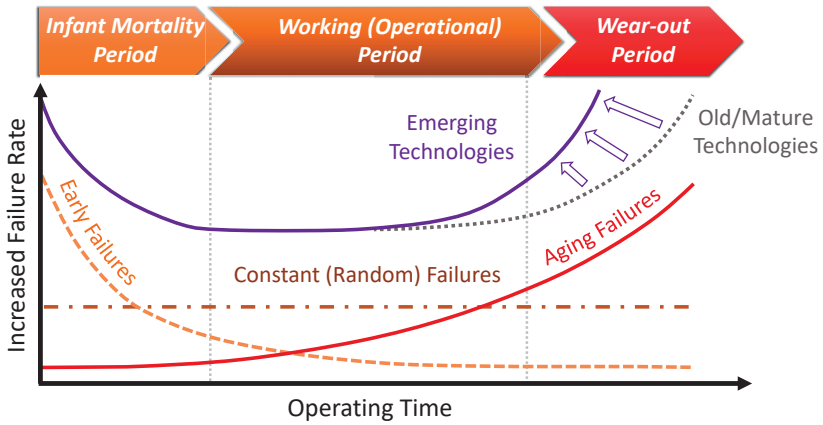


Figure 3.23 – Devices failure rate evolution with mature versus emerging silicon technologies.

3.2.3 Insights on BTI-Induced Functional Error Detection and Mitigation

As demonstrated in previous sections, BTI has a non-negligible impact on the functionality delivered by the system when operating with extreme frequencies. More importantly, BTI is becoming a key limiting factor of circuit lifetime with emerging CMOS technologies. In particular, it has a visible impact on the *bathtub* curve typically used to model systems reliability [178]. This curve represents the evolution of the failure rate over time as depicted in Figure 3.23. Traditional design strategies apply expensive guard bands to tolerate BTI-induced degradations leading to aging failures during the wear-out period of the circuit. This conservative approach employed with old and mature technologies results in non-optimized worst-case based designs with limited computing performances and possible energy/cost penalties. In order to achieve a higher computing power, modern designs operate at the edge of the safe frequency region, which in turn, increases the risk of system failure caused by aging effects. As a consequence, the evolution of the failure rate due to aging issues is shifted toward the left on the bathtub curve, thus reducing further the statistical lifespan of the system.

To address this challenge, designers are now forced to integrate energy-efficient and low-cost error recovery schemes to provide robustness against BTI degradations and meet system reliability requirements. The following section surveys the possible state-of-the-art solutions to tackle BTI-induced timing and functional errors in modern designs.

3.2.3.1 State-of-the-Art

A) Timing Violations Detection and Mitigation

A plethora of techniques and methodologies have been proposed in the literature to anticipate or mitigate the effects of timing violations in logic designs. In particular, performing a frequency binning (i.e., classification) of the dies during manufacturing is a traditionally employed technique in industry to increase production yield and avoid timing violations due

to Process Voltage Temperature (PVT) variations [53]. Nonetheless, this *time-zero technique* does not account for time-dependent BTI-induced timing violations increasing throughout the life-cycle of the chip [69, 179, 180]. Therefore, it cannot be applied in the current context to mitigate BTI aging degradations. To anticipate those future reliability issues, designers apply large frequency margins at design time. Despite its simplicity, this solution leads to non-optimized designs (as mentioned previously), which does not take into account the workload and time-dependency of BTI degradations.

To avoid worst-case scenarios and pessimistic estimations of BTI degradations at design time, BTI-aware error detection and mitigation techniques must be developed to be able to deal with circuit aging at run-time, based on the current operating conditions of the system [157]. To this end, aging detectors and *canary* circuits have been conceived to monitor circuit aging, performance and timing degradations [53, 181]. However, those solutions only evaluate the aging degradation of a small portion of the circuit (e.g., ring oscillator, pass-transistor chains, one or few critical paths identified at design time) which may not reflect the real BTI degradation of the whole circuit. Moreover, all along the utilization of the circuit, the critical paths can move due to BTI degradations [182]. In fact, in an ALU circuit for instance, some of the operators can be more degraded than others due to more intensive utilizations (i.e., higher activity stress applied on the transistors performing the operation). Hence, a non-uniform degradation of the circuit occurs, making local aging sensors ineffective to detect all the BTI-induced timing violations.

In order to take into account the overall circuit degradation and avoid timing violations, time borrowing techniques, such as TIMBER flip-flops and latches [183], have been designed to borrow the time left from successive pipeline stages. These on-line error masking techniques can recover timing margins without rollback support or instruction replay. Nonetheless, time borrowing techniques require a significant modification of the design and may strongly depend on the underlying architecture. For example, to support TIMBER latches, flip-flop-based designs must be re-timed to a two-phase latch-based design. Besides, time borrowing approaches are limited by the distribution of the critical paths and may fail when two critical paths are connected through adjacent pipeline stages heavily stimulated [184]. Also, as mentioned previously, the critical paths can be displaced due to non-uniform BTI aging, which makes this technique inadequate, even more in the case of deeply-scaled technology nodes.

Similarly to time borrowing techniques, other hardware solutions rely on the modification of the flip-flops to detect timing failures. In particular, Razor-based registers (e.g., Razor [185], Razor-II [186], Bubble Razor [187], Razor-Light [188] or iRazor [53]) provide an efficient mechanism to detect signal metastability on clock edges. The resulting error flag signal can subsequently be used to restore the pipeline and replay the instruction. Alternatively, it can also activate the stretching of the current clock cycle (i.e., time dilatation) or enable an extra processing cycle (i.e., bubble insertion) required for the completion of the operation after the initial clock deadline. However, the main drawbacks of Razor-based strategies reside in the extra delay, energy and area required to integrate these mechanisms into the design.

Furthermore, not all the timing violations occurring at the circuit level will lead to reliability issues at the application level. As shown in the previous sections, depending on the software application employed, some of the output data bits generated by a hardware component (e.g., multiplier) may not be required to perform the desired task. Hence, an error recovery technique should not be triggered each time a timing violation occurs on signals (or bits) which are not part of the final result. In fact, an efficient error detection and recovery technique must evaluate the impact of timing violations on the expected result, before the execution of recovery mechanisms involving throughput and energy losses.

For this reason, an evolution from *timing violation detection* to *functional error detection* is highly recommended to mitigate efficiently BTI-induced reliability issues.

B) Functional Errors Detection and Mitigation

In the domain of functional error analysis, Concurrent Error Detection (CED) schemes have been widely employed to improve system dependability [189]. These schemes rely on *redundancy techniques* to perform functional error detections. On one hand, temporal redundancy solutions such as *alternate data-retry* [189] and *re-computation with shifted or permuted operands* [190, 191] are efficient techniques to detect functional errors in processor pipelines. Nevertheless, they directly affect energy consumption and system performance by introducing delays when recomputing several times the same instruction differently. On the other hand, hardware redundancy techniques (e.g., Dual or Triple Modular Redundancy DMR/TMR [192] or dual-rail checkers [193]) may result in substantial energy and area overheads, when the computing resources are replicated. Moreover, in the context of BTI aging, if the replicated computing resources are stressed or aged in the same way for each operation executed, the *majority voter* performing the comparison of the final results may fail to detect the corrupted operation(s). In addition, redundant error detectors based on Residue Codes or Parity Prediction can only detect single bit errors [193, 194], which is not sufficient to identify multiple erroneous bits resulting from BTI-induced degradations. To provide a better coverage of all multiple bit errors, Berger and Bose-Lin Codes have been developed [189], but their implementation introduces large area and energy overheads.

Alternatively, in the context of multi-processor systems, interesting opportunities can be leveraged to increase resilience and mitigate functional reliability errors at the system level. In particular, heterogeneous and reconfigurable multi-core platforms, similar to the one proposed in Chapter 2, offer the ability to perform dynamic and non-uniform task allocations on a subset of computing resources (e.g., cores, reconfigurable cells) based on aging estimations. As discussed in [195], which investigates various methods to mitigate functional errors, the aging-aware task allocation strategy can rely on primary and spare processing resources employed individually at different moments in time, to distribute non-uniformly the application workload within the multi-core system. In this way, BTI effects are not always degrading the timing characteristics of the same computing resources. Some of these computing resources (i.e., spare ones) can be in standby mode (i.e., power-gated) to prevent any aging degradations

while not in use. The others (i.e., primary resources) can execute moderate or intensive tasks leading to degradations proportional to their activity. By moving or alternating the burden of execution between primary and spare units, the system lifetime can be substantially expanded at the cost of overall performance penalties and area overheads. Alternatively, aging-aware task allocations allow also to map and execute highly vulnerable application tasks on cores that are more robust to aging issues. Conversely, the less critical tasks can be executed on cheaper and more energy-efficient cores without error recovery mechanisms. Such strategy allows the heterogeneous multi-core platform to reach its maximum throughput/performance, but on the other hand the full system dependability can not be guaranteed, which is not compliant with the requirements of safety-critical applications. Additionally, both aging-aware strategies require a proper time-dependent reliability analysis (similar to the one proposed in this thesis), or an efficient aging detection mechanism to discard error-prone computing resources at run-time when the degradation exceeds a certain level.

Independently from aging analyses and estimations, other solutions have been developed to detect the occurrence of functional errors by monitoring the application execution flow and the possible divergences. In particular, the strategy proposed in [56] integrates a lightweight hardware execution monitor used to control at run-time the sequence of synchronization events, the state of the First-In First-Out (FIFO) memory buffers and the accesses to the instruction and data memories. By monitoring various sensitive points of the platform at the hardware and software level, this solution ensures the reliable operation of the complete system, transparently for the application and with little energy/area overheads. However, this technique may fail to detect the functional errors early enough in the context of time-critical applications. For instance, a real-time error detection may be difficult to achieve if the synchronization points of the program are too distant in time, or if the FIFOs are not overflowing fast enough to detect an issue in the application execution flow. To address this limitation a watchdog and static task scheduling (i.e., with known execution deadlines for each task) can be employed. In this case, when the task's execution exceeds the specified time limit, the watchdog can raise an error flag to highlight the presence of functional errors [195]. Nonetheless, this solution can only detect BTI-induced functional errors leading to an increase of the task's execution time within a deterministic environment. Therefore, other solutions must be adopted to enhance the coverage of this software-level functional error detection mechanism.

As an example, the DMR and TMR techniques previously presented at the hardware level can also be used at the software level. An interesting strategy relies on the sequential or concurrent execution of several algorithms (or tasks) implemented differently, but producing identical output results in absence of BTI-induced functional errors [133, 195]. However, despite its simplicity, this solution induces significant performance and energy penalties due to the execution on the same processor of redundant algorithms combined with a majority voter to deliver a single set of results.

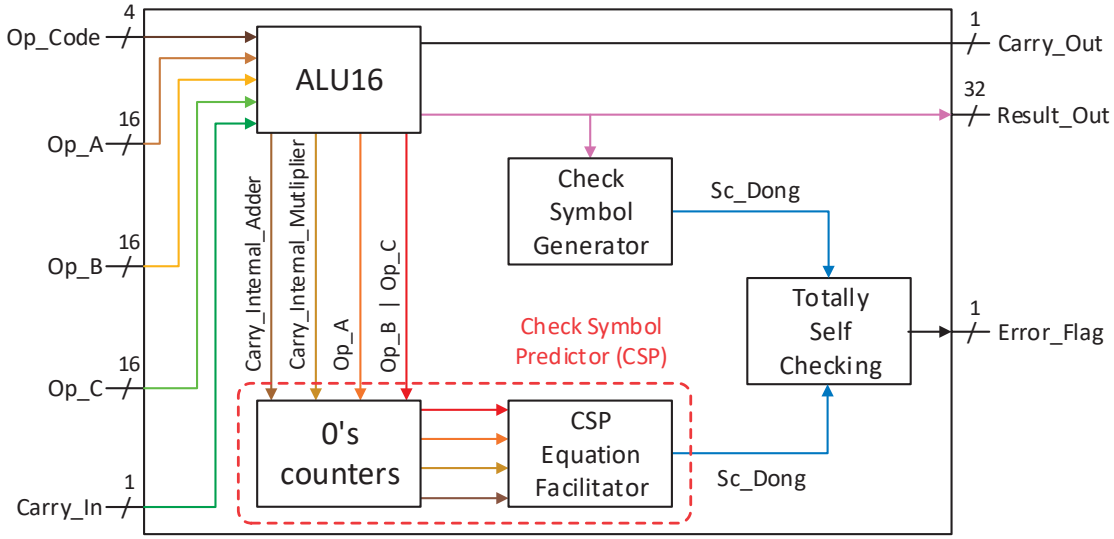


Figure 3.24 – ALU16 circuit with concurrent error detection scheme using Dong's code. NB: the reset signal has been omitted for clarity reasons.

Following this survey and to the best of my knowledge, the current state-of-the-art does not provide an optimal solution to detect and mitigate the effect of BTI-induced functional errors. Therefore, a combination of different techniques must be studied at both hardware and software levels to address this challenge.

3.2.3.2 Hardware-Level Technique for BTI-Induced Functional Error Detection

As observed in Section 3.2.2.3, the ALU16 circuit is the most critical part of the considered DSP processor pipeline, and it is the first component affected by BTI-induced timing violations when operating at high frequencies. However, its deterministic behavior (based on known arithmetic and logic operations) is suitable for the detection of BTI-induced functional errors. For that purpose, a combinational information redundant CED scheme based on Dong's Code has been developed and applied to the design of self-checking RISC processors [194]. Dong's Code is a lightweight alternative to Berger Code, allowing the detection of multiple bit errors with minimized area and energy overheads. It is a separable code with a clear distinction between data bits and check bits, which enables a simplified implementation without decryption mechanisms. Its error coverage can be tailored to a given application, and compared to a Bose-Lin Code (for example), Dong's Code provides a better error correction capability with the same number of check bits [196].

To integrate such CED scheme in the ALU16 component, a set of logic blocks must be interfaced to the circuit as illustrated in Figure 3.24. The Check Symbol Predictor (CSP) is employed to compute the check symbols (i.e., bits) from the input and internal carries of the ALU16 in parallel of its execution. In output of the ALU16, a Check Symbol Generator (CSG) determines

the check symbols from the *Result_Out* signal of the ALU16. After each operation, the Totally Self Checking (TSC) block compares the check symbols generated by the CSP and CSG blocks and raises an error signal *Error_Flag* if they are different. Then, the operation can be replayed with a lower frequency or an extra cycle can be provided for its execution with a pipeline stall, similarly to other in situ error detection and correction strategies previously presented. Further details regarding the implementation of this CED scheme are provided in [194].

In the context of this thesis, a preliminary study of this CED strategy has highlighted some restrictions regarding the architecture of the enhanced ALU16. In particular, the employed Booth Multiplier (Mult16) must be replaced by a Braun Array Multiplier to allow the extraction of the internal carries required for the computation of the check symbols by the CSP module. The structure of the Braun array multiplier increases slightly the critical path of the ALU16, thus a degradation inferior to 1 % of the maximum frequency is obtained with STA for the complete ALU16 component. This performance degradation is minimal in the case of a 16-bit array multiplier, but will be much higher for a 32-bit or 64-bit array multiplier [197]. Moreover, significant overheads are resulting from the integration of the envisioned CED strategy in the ALU16 component. After the synthesis of the new design, performance has decreased by 28 %, while the area and energy consumption have increased by 72 % and 85 % respectively, compared to the ALU16 circuit without Dong's Code error detection. Those overheads and performance penalties are mainly introduced by the circuit performing the computation of the multiplier check symbols. In fact, the CSP module relies on a 0's counter (*popcount*) of 240 bits (i.e., 16 × (16-1) bits) to determine the number of zeros in the internal carries of the array multiplier. Hence, the complexity of this bit counter increases dramatically the size of the ALU16. As a proof of its impact on the design, by synthesizing the new ALU16 without Dong's Code error detection in the multiplier, the performance is only decreased by 10 % and smaller overheads of 25 % and 41 % are observed for the area and energy consumption respectively.

Therefore, removing the error detection from the multiplier component is an interesting design choice to minimize the impact of the CED integration in the complete circuit. However, as shown in Section 3.2.2.3-B.2, the multiplier operator represents the *Achilles heel* of the ALU16, since it is the first component to be affected by BTI-induced functional errors when the operating frequency exceeds the safe frequency point. Thus, to ensure the overall reliability of the pipeline execution stage, another error mitigation technique must be associated with the hardware CED scheme, to combine both techniques into a hybrid and synergistic solution. More precisely, the following section proposes to complement the hardware-level approach with a software-level technique to assess the correctness of the results produced by the multiplier under specific operating conditions.

Algorithm 4: BTI-aware dynamic voltage and frequency scaling algorithm

Initialize system's supply voltage & frequency levels (only once):

```
System_Volt ← MIN_VOLT
System_Freq ← MAX_FREQ
```

Periodically executed at run-time:

```
// Run the longest signed multiplication with current voltage & frequency levels
Result_Out ← (−1)10 × (−1)10
if Result_Out ≠ (1)10 then
    // Faulty operation => Update System_Volt & System_Freq
    System_Volt ← MAX_VOLT
    for f ← System_Freq down to MIN_FREQ do
        System_Freq ← f
        Result_Out ← (−1)10 × (−1)10
        if Result_Out = (1)10 then
            // Correct operation => Keep System_Freq & update System_Volt
            for v ← (MAX_VOLT − 1) down to MIN_VOLT do
                System_Volt ← v
                Result_Out ← (−1)10 × (−1)10
                if Result_Out ≠ (1)10 then
                    System_Volt ← (v + 1)
                    // Correct operation with a higher supply voltage level
                    return Operating Parameters: OK
                end
            end
            // Correct operation with System_Volt = MIN_VOLT
            return Operating Parameters: OK
        end
    end
    // Error: MAX_VOLT & MIN_FREQ inappropriate to perform a correct operation
    return Operating Parameters: ERROR
else
    // Correct operation => Keep System_Volt & System_Freq
    return Operating Parameters: OK
end
```

3.2.3.3 Software-Level Technique for BTI-Induced Functional Error Detection

The aforementioned hardware strategy can be used to trigger correction or recovery mechanisms when functional errors are detected in the pipeline execution stage. However, when the critical path is aged by permanent BTI degradations, the same functional errors may occur repeatedly, leading to an over-utilization of the recovery mechanism. In fact, no actions are taken to adjust the operating parameters of the system and prevent those errors from happening again in the future. To address this limitation, I propose an aging-aware Dynamic Voltage and Frequency Scaling (DVFS) technique able to tune the parameters of the system and achieve reliable operation under the influence of BTI effects. Similarly to the BTI-aware design presented in [198], the proposed DVFS technique ensures the correctness of the operations performed by the multiplier, which represents the most critical and error-prone component in

the ALU. More importantly, this technique aims at minimizing the BTI-induced performance degradations and energy overheads by selecting the maximum frequency and minimum voltage allowing reliable operations. The software routine of the proposed aging-aware DVFS technique is described in Algorithm 4.

In this software routine, the longest operation performed by the ALU (i.e., signed multiplication $(-1)_{10} \times (-1)_{10}$) is used as an *aging detector*. As previously shown in Figure 3.20 and based on the obtained critical paths distribution after 10 years, signed multiplications are the first operations to fail, even well ahead of the other arithmetic and logic operations when the frequency is increased. In other words, with the considered experimental evaluation, if this operation does not fail, the other operations performed by the ALU are assumed to be still correct. Besides, the multiplication is a commonly employed operation in DSP applications. As a consequence, its BTI-induced degradation is comparable to other operators often used in the circuit such as adders and subtracters.

Furthermore, the aging-aware DVFS algorithm can be executed at system startup, but also at different time intervals to evaluate the level of BTI degradation and reassess the value of the circuit operating parameters. Since the execution of this algorithm results in a small time penalty at the application level, it opens up the exploration of trade-offs between the frequency of routine calls and the amount of BTI-induced functional errors not detected early enough by the software routine. However, in the context of mission-critical systems, additional solutions must be exploited and in particular error mitigation mechanisms as shown in the next section.

3.2.3.4 Hardware and Software Techniques for BTI-Induced Functional Error Mitigation

To guarantee the exactness of the operations performed by the ALU, the hardware and software techniques previously presented must be combined together to produce an efficient hardware/software functional error mitigation technique. Also, in order to bridge gaps and increase the coverage of these techniques, additional solutions can be employed. For instance, data checksums and signature-based solutions are commonly integrated into critical software applications to ensure the correctness of the results produced [199]. Furthermore, when BTI-induced functional errors are detected either at the pipeline or application level, an execution rollback strategy can be triggered to restart the application from a safe point, such as checkpoints recorded at fixed time intervals [140]. In a simpler way, the whole system can also be reset, similarly to lightweight error recovery techniques [56], and re-executed with appropriate (i.e., less aggressive) operating parameters inline with the aging-level of the circuit.

All these techniques raise several design challenges to be explored in future works. In particular, they stimulate the exploration of several trade-offs in terms of energy consumption, silicon area and performance penalties on the real-time behavior of the application. In addition, other reliability issues may degrade the functionality of the system. To complement this study, the second part of this chapter deals with reliability concerns in memory components.

3.3 Reliability Analysis and Significance-Based Data Protection in Memories

As shown in the previous sections of this chapter, logic components may encounter non-negligible timing degradations leading to functional errors compromising the reliability of the entire platform. However, the fragility and vulnerability of logic blocks to reliability issues remain significantly lower compared to those of memory components. In fact, memories represent the largest energy and area portions of modern designs [14, 50, 200]. Their optimized and constrained structures targeting high-density and high-capacity, increase the probability of having memory cell failures induced by physical-level issues [201]. Therefore, in the following sections I complete this technology-level exploration by analyzing the reliability of memory components in WBSN devices. In addition, I apply aggressive voltage scaling on these components to minimize their energy consumption and showcase possible trade-offs between energy efficiency and accuracy of the results produced by a representative set of biomedical applications.

In order to provide the necessary background on this topic, the next section starts by presenting the different reliability issues and resulting challenges in current memory technologies.

3.3.1 Reliability Concerns and Challenges in Memories

The continuous shrinking of CMOS transistors enables the development of increasingly complex computing systems including more computing capabilities within the same chip-size. Unfortunately, smaller transistor sizes have adverse consequences on reliability, as technology shrinking increases their fragility and brings sensitive nodes closer to each other. In particular, specific components such as embedded memories are more prone to soft and hard errors [55, 141, 202].

Soft errors can be induced by signal noises, radiated electromagnetic interferences or on-chip crosstalks between two circuits or design elements (i.e., wires and interconnects) [66]. Additionally, soft errors may also be caused by Single Event Upsets (SEUs) such as ionizing particle strikes (e.g., heavy ions, electrons, photons) from outer space (i.e., cosmic rays) or emanating from packaging, bonding, and die materials [54, 55]. However, these SEUs are not considered as permanently damaging the functionalities of the transistors, unlike other radiation effects including: Single Event Latchup (SEL), Single Event Gate Rupture (SEGR), Single Event Dielectric Rupture (SEDR), Single Event Burnout (SEB), and so forth [203, 204]. The resulting permanent errors induced by these Single Event Effects (SEE) may also be complemented by different technology- and process-related phenomena such as manufacturing defects, electromigration, device wear-out, and aging effects [66, 205].

All these issues constitute reliability challenges to address when designing modern systems. In fact, the reliability of current designs is mainly compromised by the soft and hard error rates

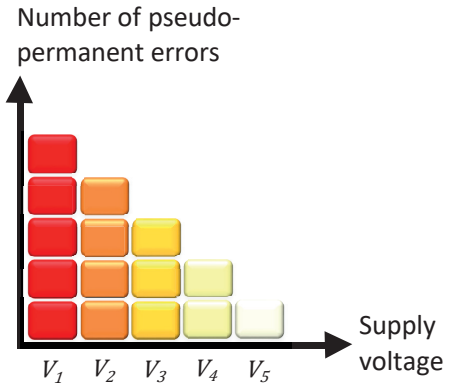


Figure 3.25 – Illustrative example of the accumulation of pseudo-permanent errors when the memory supply voltage is reduced.

of memories, stressing the need for memory protections. On top of that, embedded systems designers must deal with the high energy demand of memories. An effective technique to reduce it is to diminish the memory supply voltage, thanks to VFS, which translates into quadratic energy savings (see Section 3.1.1).

However, VFS has adverse consequences on the reliability of the circuit. More precisely, as the supply approaches the transistors' threshold voltage, the sensitivity of memories to radiations and voltage droops increases proportionally [54, 56]. In addition, aggressive voltage reductions lead to pseudo-permanent errors in memory cells in the form of stuck-at faults [206]. As depicted in Figure 3.25, these errors are gradually accumulated as the supply voltage decreases. Nevertheless, the pseudo-permanent nature of these errors enables the memories to recover by re-increasing the supply voltage, at the cost of higher energy consumptions.

Dealing with these issues is a real challenge for designers forced to craft reliable devices by implementing new Error Mitigation Techniques (EMTs) which reduce or eliminate the occurrence of failures at acceptable costs: performance penalty, energy, and area overheads. To address this challenge, in the remainder of this chapter, I propose and evaluate a novel hardware/software technique, called *DREAM*. This technique is designed to mitigate the pseudo-permanent errors in memories with low energy overheads. Furthermore, its design takes advantage of the resilient nature or fault tolerance of several biomedical applications, to perform an aggressive voltage scaling leading to a better minimization of the energy consumption.

3.3.2 Related Works

Since the invention of the first Error Correction Code (ECC) by Richard Hamming in 1950 [207], hardware-based EMTs for system reliability and energy savings enhancement have been widely explored in the literature. Multiple improved versions of the ECC were developed to detect and protect in hardware more and more erroneous memory data bits while minimizing the impact on performance and resources [208, 209].

Aside from ECC memory protections, Schechter *et al.* [210] introduce another technique called Error Correcting Pointers (ECP) to tackle defective memory cells (i.e., permanent faults). This EMT allows to redirect writes in faulty data cells to brand new spare cells. Moreover, this technique has some similarities with the Built-In Self-Repair (BISR) scheme introduced by Sridhar *et al.* [211], which is able to replace faulty memory words by redundant ones. The efficiency of both techniques is mainly dictated by the available number of spare memory locations. On one hand, a high number of spare memory addresses leads to higher energy and area overheads. On the other hand, a small amount of them is not enough to fix memories with many faulty locations, which is particularly the case when aggressive voltage scaling is employed (see Section 3.3.4.2).

Alternatively, Ejlali *et al.* [212] present an energy overhead minimization technique in fault-tolerant hardware redundant systems. It relies on parallel execution of primary and spare processing units, used in combination with Dynamic Voltage Scaling (DVS) and Dynamic Power Management (DPM) to reach different levels of energy reduction. However, this solution is not suitable for area- and energy-sensitive systems since it requires a full duplication of the processing unit.

In a similar way to my proposal, Kai-Chiang *et al.* [213] propose to use several supply voltages on different parts of the circuitry, to achieve higher energy savings. Their energy-aware Soft Error Rate (SER) reduction framework assigns lower supply voltage to the gates that have a weak error impact and contribute less to the overall SER. This solution requires a fine grain and probabilistic study of the circuit to determine the criticality of each gate, and a modification of the layout of the transistors. DREAM on the other hand, uses standard libraries for logic design and does not require custom design, which greatly reduces its complexity, increasing its applicability.

Furthermore, as a general rule, the lower the voltage is, the more energy, delay, and area overheads have to be introduced in hardware, to guarantee error-free operations [135]. To overcome the need for powerful hardware error correction mechanisms at near-threshold supply voltage, embedded systems designers now focus on error mitigation techniques operating both at the hardware and software levels. These techniques such as the one proposed by Sabry *et al.* [140] rely on mechanisms to detect the errors at the hardware level, and either activate some countermeasures to prevent it, or restart the execution of the application from a safe state thanks to checkpoint and rollback mechanisms.

Other solutions operating at the software level, explore the inherent robustness of applications. As an example, the approximate computing paradigm exploits the fault tolerance of many embedded applications, and more specially image- and signal-processing applications [63]. Moreover, significance-based computing has been recently proposed as a variation of the approximate computing paradigm to selectively protect the execution of significant computations or data while allowing controlled errors to occur in other parts of the program [64]. To illustrate this last paradigm, recent work on JPEG encoding/decoding by Karakonstantis *et*

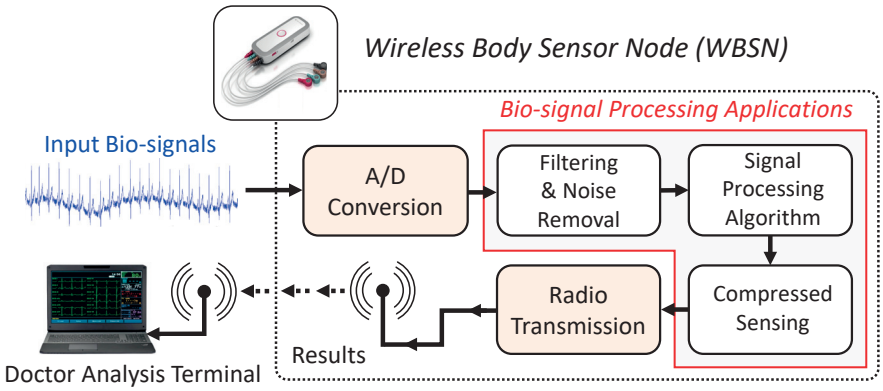


Figure 3.26 – Block scheme of a typical WBSN.

al. [214], proposes to tune the supply voltage of the blocks with less-crucial computations, leading to minimal quality deterioration and better energy savings. This technique can be applied both to logic and memory, and highlights the necessity of algorithmic transformations to ensure minimum quality degradation under delay errors induced by voltage scaling. On the hardware side, the authors propose an unequal memory protection strategy in order to find an acceptable trade-off between application output quality, area overhead and energy consumption. To that end, their strategy combines different levels of protection using two well-established technologies (6- and 8-transistor SRAM cells), to lighten or harden respectively the memory cells according to the significance of the data bits recorded.

In comparison to this strategy, I propose DREAM as a hybrid strategy to mitigate pseudo-permanent errors based on the knowledge of the applications running on the system and without modifying the structure of the memory cells. More importantly, to the best of my knowledge, I am the first one to apply such scheme to applications that fall outside the multimedia domain. As I show in the following sections, biomedical applications are very different in terms of requirements, and offer different trade-offs, thus requiring a different approach for data protection.

3.3.3 Analysis of the Inherent Resilience of Biomedical Applications

Bio-signal analysis applications aim at acquiring and processing pharmaceutically, clinically or biochemically relevant information from single- or multiple-input signals, in order to establish or improve medical diagnoses [215]. The common flow of all of these applications consists of filtering the signal of interest out from the noisy background and then reducing the redundant data stream to only few characteristic features. WBSN devices are typical examples of systems implementing these applications. A schematic of their structure is depicted in Figure 3.26.

3.3.3.1 Representative Set of Biomedical Applications

In this work I have selected five representative applications that are widely used in the field of ECG processing, either as standalone applications, or as the core components of more complex monitoring systems such as WBSN devices (see Figure 3.26).

Similarly to the state-of-the-art [49, 61, 62, 216], to minimize the memory footprint and energy consumption of these applications, the employed data format has been reduced to 16 bits integer. This resolution provides a reasonable encoding accuracy of the ECG signals (i.e., covering the full dynamic range of the information), while preventing saturation effects during their processing. Furthermore, by employing more compact data formats, the amount of bits per data word that needs to be protected is also reduced, thus simplifying the architecture and reducing the energy consumption of a future memory protection technique.

The selected applications have been tested using ECG traces from the MIT-BIH Arrhythmia database, which contains recordings from healthy patients and with different cardiovascular pathologies [29]. Based on these signals, this study aims at understanding the computing and memory requirements of the applications. More importantly, an evaluation of the significance of each manipulated data bit is performed to identify which part of the data words must be protected first by the proposed EMT, in order to minimize the impact of errors on the final results. The following paragraphs present each of these applications:

- **Discrete Wavelet Transform (DWT):** The DWT application performs a multi-lead ECG signals analysis widely employed in commercial WBSNs [122]. In my implementation, the DWT takes as input a vector of 1024 ECG samples and performs on it several scales of low-pass and high-pass filtering. As stated previously, each of these samples has a size of 16 bits.
- **Matrix Filtering:** A signal processing algorithm that applies a given transformation (e.g., low-pass or high-pass filtering) to a set of bio-signal samples [62]. At the lower level, this operation consists of a series of matrix multiplication operations $[A] \times [B] = [C]$ repeated (iterations of the algorithm) until the quality of the result meets the desired level. This computationally-intensive processing can easily be parallelized splitting the original set of samples into chunks, which can be then processed independently by each core instantiated in the system. Once the data crunching is finished, the main processor is in charge of reassembling the resulting matrix. However in the current setup (see Section 3.3.4.2), to be consistent with the other applications running on a single core, the full matrix filtering application is also executed on a single processor. The size of the three matrices has been defined as a kernel of 32×32 elements (1024 samples), and the number of processing iterations has been set to 5.
- **Compressed Sensing (CS):** Data compression is a well-known method to encode the information using fewer bits or samples than the original representation. In digital computing systems, this method is used to address problems of narrow transmission bandwidth and small data memory storage. Within the context of embedded ECG

monitoring systems, the Compressed Sensing (CS) signal acquisition/compression technique has been introduced as a new paradigm for energy-aware WBSNs. It helps to reduce airtime over energy-hungry wireless links. In the context of this study, a similar version to the algorithm presented in [217] has been implemented. It takes as input a vector of ECG samples and applies a 50 % lossy compression algorithm to convert it into a smaller one (half the size of the input vector). To evaluate the deterioration of the signal after compression and memory error injection, a decompressed sensing application has been developed to regenerate at the output the original vector from the compressed vector. This decompression phase runs outside of the employed platform; i.e., it is not corrupted by the errors present in the data memory of the platform simulator.

- **Morphological Filtering:** ECG signals are often degraded with different types of noise from various sources, such as: patient's muscles activity and breathing, motion artifacts from the electrode-skin contact and system AC supply interferences. Morphological Filtering is a special type of filtering algorithm developed to clean, thanks to different erosion and dilation steps, the raw ECG signals attending to the shape or morphology of certain expected features. These filtering steps are widely used in the image- and biomedical signal-processing domains [122]. My implementation requires 1024 ECG samples in input and output, plus a set of 10 internal buffers with a size ranging from 5 to 150 samples to perform the different erosion and dilation steps inherent to this algorithm.

- **Wavelet Delineation:** This algorithm is typically employed to perform an analysis of ECG signals to detect heartbeat fiducial points [122]. Therefore, the implemented version of this algorithm is based on the DWT application previously presented and a set of 9 intermediate buffers plus 1 input buffer with sizes ranging from 4 to 256 samples. This application is executed each time on a vector of 1024 input ECG samples and generates, as output, the list of P, Q, R, S, and T heartbeat points found (see Figure 3.13). The output quality assessment is done through a comparison between the theoretical and experimental values that compose the five dyadic scales recorded inside the intermediate buffers [35].

3.3.3.2 Data Significance Analysis and Characterization of Biomedical Applications

In order to efficiently apply the significance-based computing paradigm, I first analyze and explore the nature of biomedical applications. I conduct a significance analysis for each bit of information, with the goal of determining which data bits (from the input, intermediate and output application buffers) in the main data memory must be protected first by an EMT. To do so, I perform a quality analysis of the computation results under the impact of pseudo-permanent errors in the memory for each application. I use the Signal-to-Noise Ratio (SNR) metric, as defined in Equation 3.7, based on the Mean Square Error (MSE) metric, which is defined as the average of the squares of the difference between all the error-free $x_{theo}(i)$ (theoretical) and corrupted $x_{exp}(i)$ (experimental) output data.

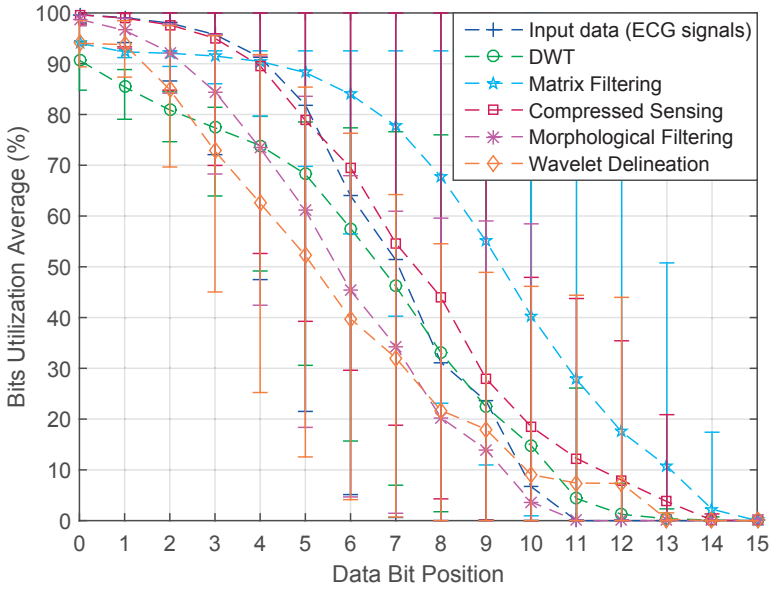


Figure 3.27 – Data bits utilization percentages evolution. NB: The highest data bit positions (i.e., starting from 15 downwards) correspond to the Most Significant Bits (MSBs), while the lowest data bit positions (i.e., starting from 0 upwards) designate the Least Significant Bits (LSBs).

$$MSE = \frac{1}{n}\sum_{i=0}^{n-1}(x_{theo}(i) - x_{exp}(i))^2 \tag{3.6}$$

$$SNR = 20 \times \log_{10}\frac{\sqrt{\frac{1}{n}\sum_{i=0}^{n-1} x_{theo}^2(i)}}{\sqrt{MSE}} \tag{3.7}$$

Figure 3.27 shows the average utilization percentage of each data bit according to its position for the different applications. In this figure, all the bits encoding the sign extensions are not taken into consideration for redundancy reasons. The Least Significant Bits (LSBs) are used most of the time to encode the main part of the information, while the Most Significant Bits (MSBs) only repeat the sign bit. Indeed, bits in positions 13, 14, and 15 have utilization percentages close to zero, which demonstrates that they are used most of the time to encode sign extensions (several 0's or 1's in a row, respectively, for a positive or negative number).

As a complement to this analysis, Figure 3.28 depicts the evolution of the output SNR as a function of the erroneous bit position in the data words for each application. To generate output data corrupted by pseudo-permanent errors, I successively set to "1" and "0" each bit located on the positions 0 to 15 of each 16-bit data buffers. In other words, for every run only one specific bit per memory word is permanently corrupted by a "1" or a "0" all along the execution of the application. Moreover, several ECG signals with different pathologies are used to produce each averaged point of Figure 3.28. The gap between the SNR curve of the Matrix

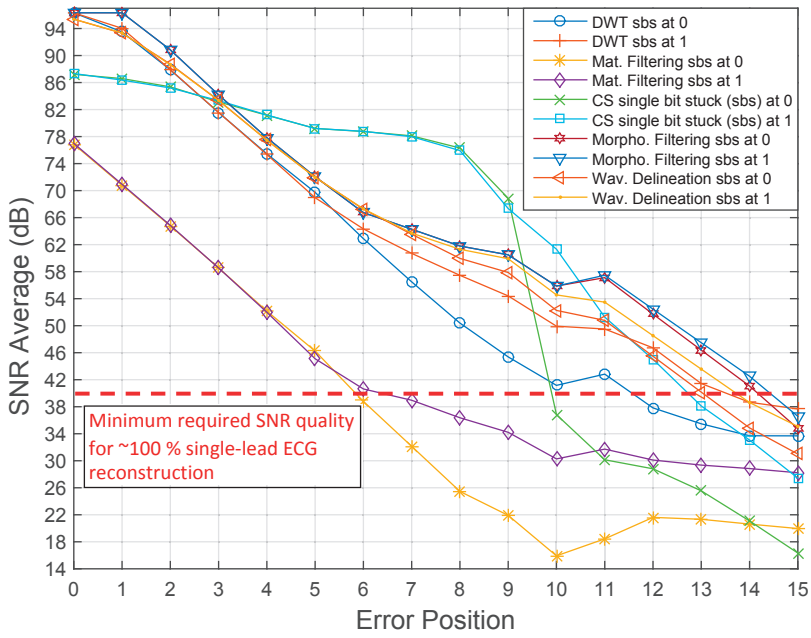


Figure 3.28 – SNR vs. data bit positions of injected errors.

Filtering and the other curves stems from the fact that, when operating with matrices, each element of the resulting matrix depends on 64 elements (one full row and one full column) of the input matrices. As a consequence, a single error affects several positions in the output. Also, Figure 3.28 outlines that, for the Matrix Filtering and CS applications, erroneous bits set to “1" on MSB positions have a smaller impact on the SNR than erroneous bits set to “0". This is due to the fact that most of the bio-signal samples employed during the experiments are negative; thus, when a permanent error sets to “1" a bit on the MSB positions of a negative number, the effect is often hidden.

In addition, for all the applications, Figure 3.28 highlights the continuous decrease of the SNR as the erroneous bit is shifted towards the MSB positions, which demonstrates that errors on the MSBs have a stronger impact on the application results, whereas errors on the LSBs often have a small or negligible effect. This finding demonstrates the possibility of dealing with some degree of inexactness on the LSBs positions, as it is the case for input data samples acquired in real-life conditions, namely, from noisy analog sources (see *Morphological Filtering* in Section 3.3.3.1).

Finally, some applications, such as the Heartbeat Classifier [62] (based on Wavelet Delineation + CS), produce statistical or qualitative results. After delineation, heartbeats are sorted out according to different classes of morphologies to detect patients' pathologies, and this task usually requires fine-tuning with human feedback to adjust the margin inherent to the classification algorithm. This classification is often performed visually by doctors and its precision depends on human interpretation with coarse-grained boundaries between classes. This process enables a relaxation of the traditional reliability requirements (i.e., 100 % computational precision is not needed), which can be exploited by the CS application. In particular,

the minimum required output SNR to get almost a 100 % reconstruction quality is only 35 dB in the case of multi-lead ECG [217] and 40 dB in the case of a single lead ECG [19]. Thus, as Figure 3.28 shows, CS can tolerate errors on the bit positions from 0 to 10, for bits stuck-at “0"; and from 0 to 12, for bits stuck-at “1".

3.3.4 Proposed Significance-Based Error Mitigation Technique for Memory Components

Based on the characterization of the biomedical applications previously achieved, this section introduces *DREAM*, the memory protection technique developed in this work to mitigate pseudo-permanent errors induced by supply voltage scaling in memories. The presentation of DREAM is then followed by an experimental evaluation, to assess the energy benefits provided by this technique.

3.3.4.1 Presentation of the DREAM Memory Protection Technique

In embedded biomedical applications, most of the samples produced by the ADCs contain series of bits with the same value on the MSB positions (see Section 3.3.3.2). The reason is that not all the samples need the full range of bits allowed by the system to encode the information extracted from the electric signal. For instance, when a 16-bit ADC produces the value $(511)_{10}$, only 9 bits are required to encode this value. The remaining 7 bits on the most-significant bit positions are just zeros (i.e., sign extension) and can be considered as constant for all the sample values ranging from 0 to $(511)_{10}$. Furthermore, as analyzed in the previous section, errors occurring on the MSB positions have a stronger impact on the final result of each application. This implies to protect first these bits to reduce the effect of errors on the final result.

Based on these two observations, I propose in this work the *Dynamic eRror compEnsation And Masking (DREAM)* technique to dynamically preserve the value of the constant MSBs of each memory word. DREAM relies on bit masks to keep track of the series of MSBs with the same value in each sample. These bits must be kept constant all along the storage of the sample into the faulty memory. To do so, similarly to ECC, some extra logic is required to determine and apply dynamically the mask on data words. Also, a small error-free memory is needed to store the mask identifier (mask ID) and sign bit of each data word. This memory overhead per data word can be determined using Equation 3.8. In this study, the number of masked bits from two consecutive masks only differ by one bit (i.e., one additional bit is covered/protected by the following mask when the mask ID is increased by one position). Therefore, the number of masks (or mask IDs) employed by DREAM is equivalent to the maximum number of bits to protect in the data word (i.e., 16 bits), since the different masks can protect between 1 and 16 bits of data.

$$\begin{aligned} Extra\,Bits\,/\,Memory\,Word &= Sign\,Bit + Mask\,ID\,Bit\,Size \\ &= 1 + \log_2(Number\,of\,Masks) \end{aligned} \tag{3.8}$$

Table 3.5 – DREAM error correction example for 16-bit data

	Case 1 ***Original Data*** ≥ 0	**Case 2** ***Original Data*** < 0
Original Data	$(0400)_{16} = (1024)_{10}$	$(FC00)_{16} = (-1024)_{10}$
Corrupted Data	$(2400)_{16}$	$(DC00)_{16}$
Data Sign Bit	$(0)_2$	$(1)_2$
Mask	$(F800)_{16}$	$(F800)_{16}$
Data after Masking	$(2400)_{16}$ & !$(F800)_{16} = (0400)_{16}$	$(DC00)_{16}$ || $(F800)_{16} = (FC00)_{16}$

Table 3.5 gives a concrete example of how this technique works. The first row of this table shows the original data and then in the second row, the value corrupted, where some erroneous bits (induced by faulty data cells) have mutated the value. The third and fourth rows of this table contain respectively the original data sign bit and mask value. The MSBs of the mask set to “1" are used to highlight the bits of the data word that must be set to a constant value equal to the sign bit. During the read operation, if the data sign bit is equal to “0" (positive

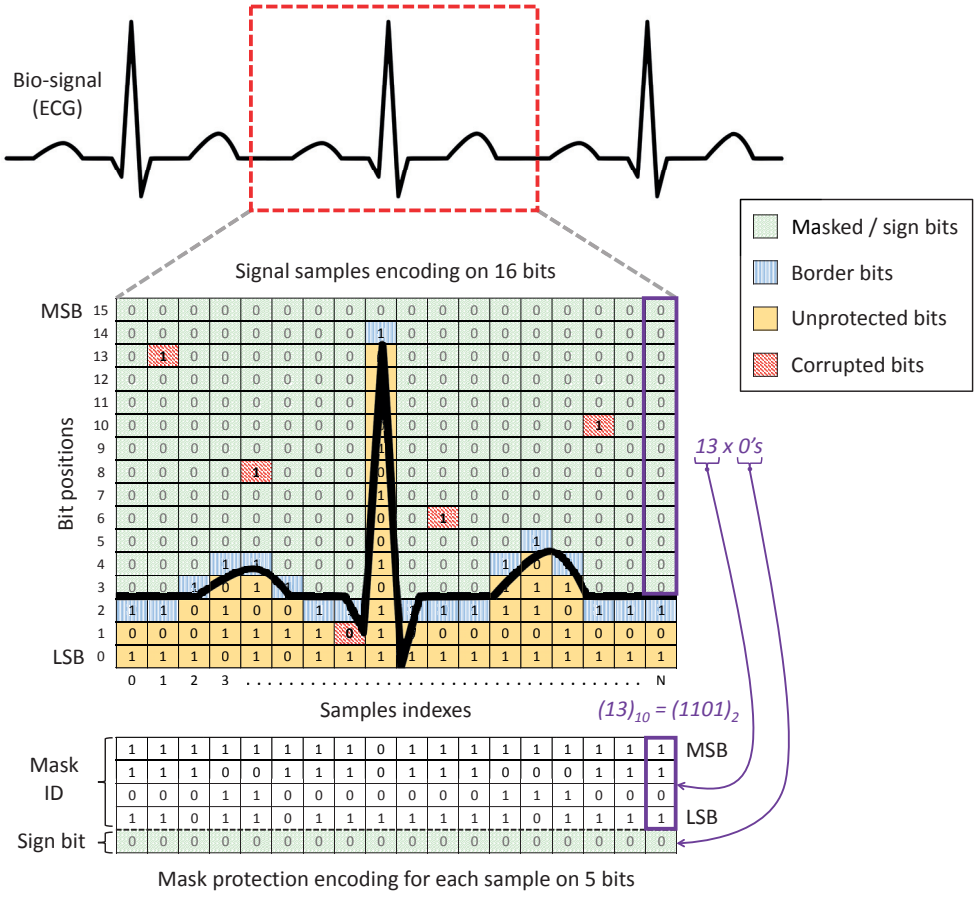
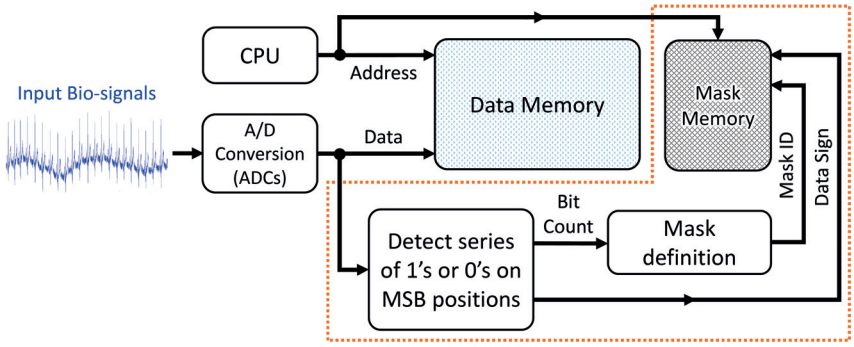
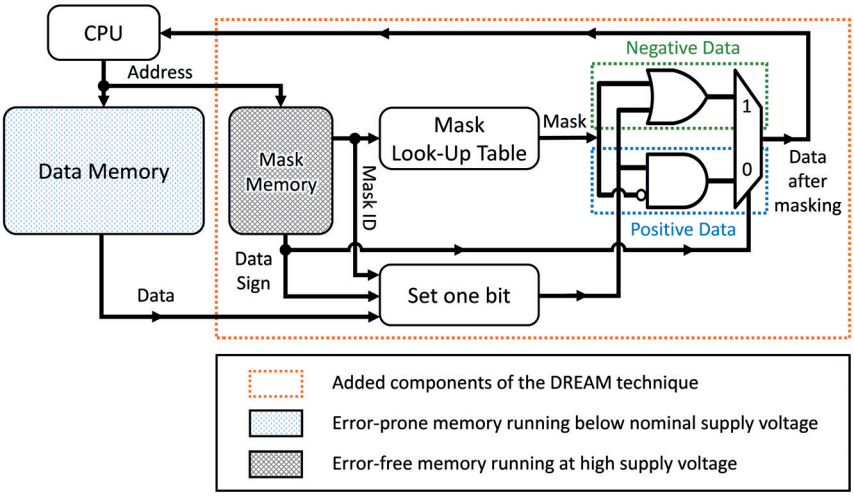


Figure 3.29 – DREAM technique - Simplified representation of the protection mask encoding (assuming an ECG signal with a positive DC offset).



a) Write mode diagram



b) Read mode diagram

Figure 3.30 – DREAM technique operating modes.

data), the corrupted data from the memory is bitwise ANDed with the mask complemented. Conversely, if the data sign bit is equal to “1" (negative data), the corrupted data from the memory is bitwise ORed with the mask, in order to retrieve the original data.

DREAM is able to correct multiple errors located in the series of MSBs highlighted by the mask. In fact, an additional data bit is always protected because the most-significant bit of the data part not covered by the mask is always set to the inverted value of the data sign bit; therefore, the position of this bit in the data word can be specified by the mask ID and then inverted by a simple NOT gate (see *Set one bit* block in Figure 3.30.a). To complement these explanations, Figure 3.29 illustrates in a more visual way the encoding of the protection mask for each bio-signal sample. Also, it can be noted that, the smaller the data encoded inside the data word is, the bigger the number of MSBs set to the same value. In other words, this EMT offers great error correction capabilities in the scenario of biomedical applications, since they typically manipulate signals with small dynamic ranges and a distribution of the values centered around zero. In the following, more details are provided on the operating modes of DREAM at the circuit level.

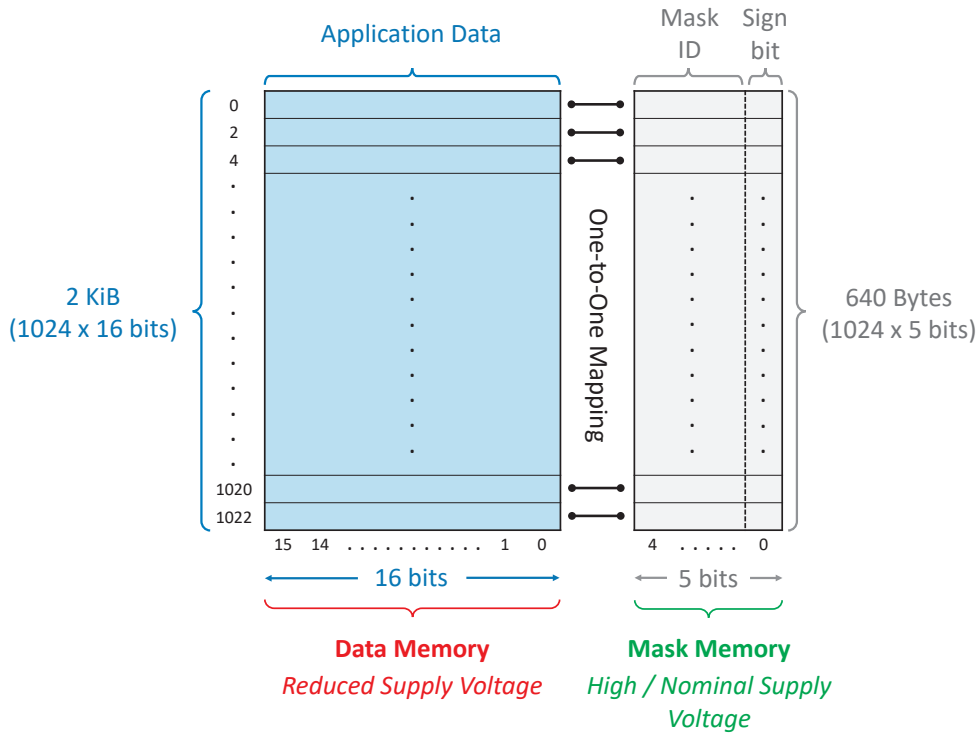


Figure 3.31 – Example of memory mapping with DREAM support for a 2 KiB data bank.

A) Write Operating Mode

During write access to the data memory, both sample storage and mask ID determination are done in parallel. This can be observed in Figure 3.30.a. Each original or *error-free* sample produced by the ADC, for instance, is stored into the error-prone data memory and, at the same time, passes through a logical block used to determine the sign and the number of MSBs set at the same value in the data word. Subsequently, the bit count (number of MSBs) is used to determine a mask ID, that will be concatenated with the sign bit. Then, these data are stored in an independent error-free memory (i.e., *Mask Memory*) running at a high supply voltage level to prevent the occurrence of pseudo-permanent errors induced by voltage scaling. Compared to the (small) error-free memory, the main (big) *Data Memory* operates below the nominal supply voltage to minimize the energy consumption, despite the fact that some pseudo-permanent errors may occur. Moreover, because of different supply voltages, both memories are structurally independent. However, they share the same *one-to-one* address mapping. An example of memory mapping including both memories is illustrated in Figure 3.31.

B) Read Operating Mode

In read mode, the system works in the reverse way compared to the write mode, as it is illustrated in Figure 3.30.b. First of all, the data stored in the memory (possibly corrupted) and the mask ID concatenated with the data sign bit, are fetched from the memories. Secondly, the mask ID is converted into a full mask thanks to a mask decoder. Then, two logical operations

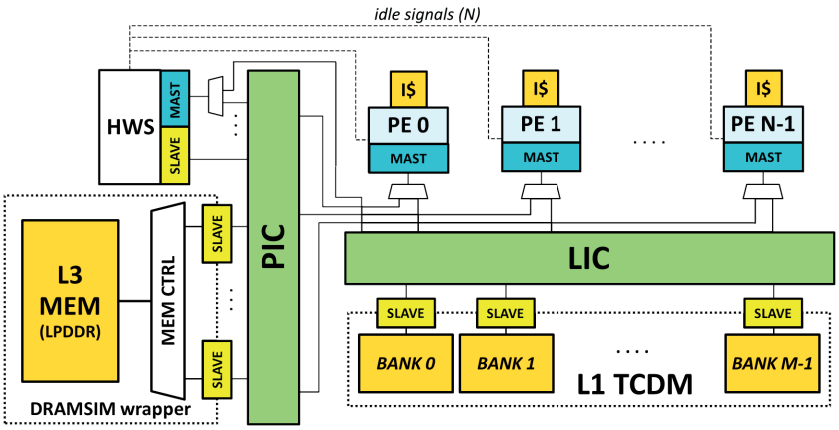


Figure 3.32 – High-level view of the VirtualSoc platform.

(AND and OR) are performed with the mask and the corrupted data. After that, the corrected data is selected by a 2 to 1 multiplexer controlled by the data sign bit, before being forwarded to the processor.

As shown in Figure 3.30.b, the corrected data word is not directly written back to the main data memory. In fact, it is not useful to write the corrected data to the memory, since the pseudo-permanent error(s) present at this voltage level and memory address will corrupt again the data word at the same bit position(s). As a consequence, when operating at low supply voltage levels, an error mitigation technique (e.g., DREAM, SEC-DED ECC) must be employed every time a data word is read from the main memory in order to correct the potentially corrupted data.

3.3.4.2 Experimental Evaluation

A) Experimental Setup

To evaluate the proposed approach using the DREAM technique, in terms of correction capabilities and energy consumption, I model the architecture of the biomedical computing device INYU [218] by extending VirtualSOC [219], an existing multi-processor cycle-accurate simulator. The architecture of VirtualSOC is illustrated in Figure 3.32. It can instantiate up to 16 ARM v6 cores interconnected to private and shared memories, accessed at a clock frequency of 200 MHz. This platform embeds also several interesting features, such as access to peripherals (e.g., external main memory (L3 MEM)) through a logarithmic Peripheral Interconnect (PIC), and a Hardware Synchronizer (HWS) module based on semaphores to enable multi-core synchronization. In the context of this study, the simulated architecture has been instrumented with the necessary mechanisms to uniformly inject pseudo-permanent errors into the shared Tightly Coupled Data Memory (TCDM) and detect data corruption at run-time for each considered voltage level.

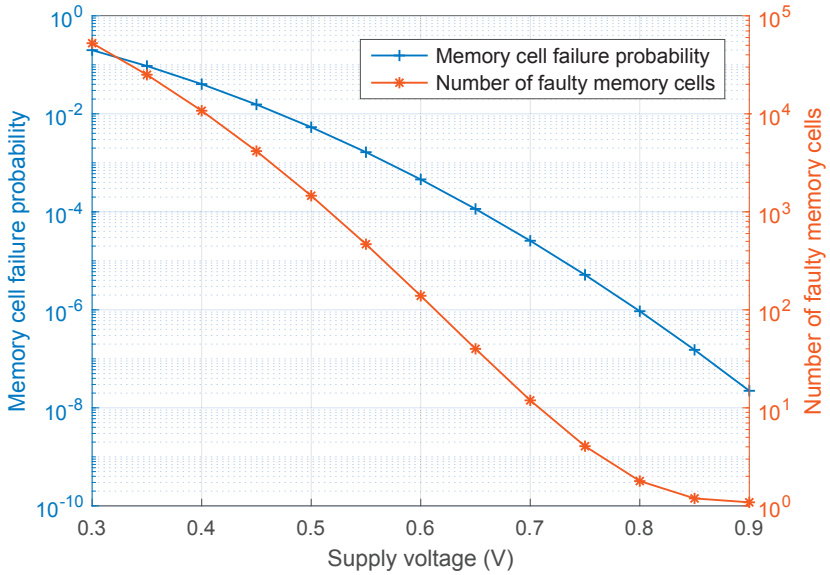


Figure 3.33 – Evolution of the memory cell failure probability and number of faulty memory cells impacted by voltage-dependent stuck-at faults (leading to pseudo-permanent errors), for a 32 KiB SRAM memory.

The biomedical applications operate with 16-bit data words stored into the TCDM of 32 KiB and divided into 16 banks accessible by the cores through a parametric Mesh-of-Trees (MoTs) (i.e., Local Interconnect (LIC)). In order to compare the different EMTs, the memory has been enhanced to fully support not only the DREAM technique, but also the well-known SEC-DED ECC [208]. To protect a 16-bit data word, the memory overhead of these two EMTs are $1 + log_2(16) = 5$ extra-bits for the DREAM technique (see Figure 3.31) and $2 + log_2(16) = 6$ extra-bits for the SEC-DED ECC. These EMTs have to cope with data corruption induced by pseudo-permanent errors that occur at random positions and set the affected memory bits to “1" or “0". During system execution, certain error positions and values are more critical than others and degrade differently the final result of the application. That is why, in order to provide fair comparisons between the different EMTs, pre-generated tables are used to record the random error locations/mappings employed for the different set of simulations. Hence, for each set of simulations performed for one EMT, the same set of error mappings is reused to run the next set of simulations with a different EMT.

The amount of pseudo-permanent errors injected $N_{perm\_err}(V_n)$ is directly linked to the size S_{mem} and supply voltage level V_n of the memory through a Bit Error Rate $BER(V_n)$ as shown by Equation 3.9 [135, 136].

$$N_{perm\_err}(V_n) = S_{mem} \times BER(V_n) \tag{3.9}$$

This bit error rate is obtained for each voltage level by simulating the SRAM memory for the selected technology node [137, 220]; in this case, Synopsys® EDK 32 nm with low-power

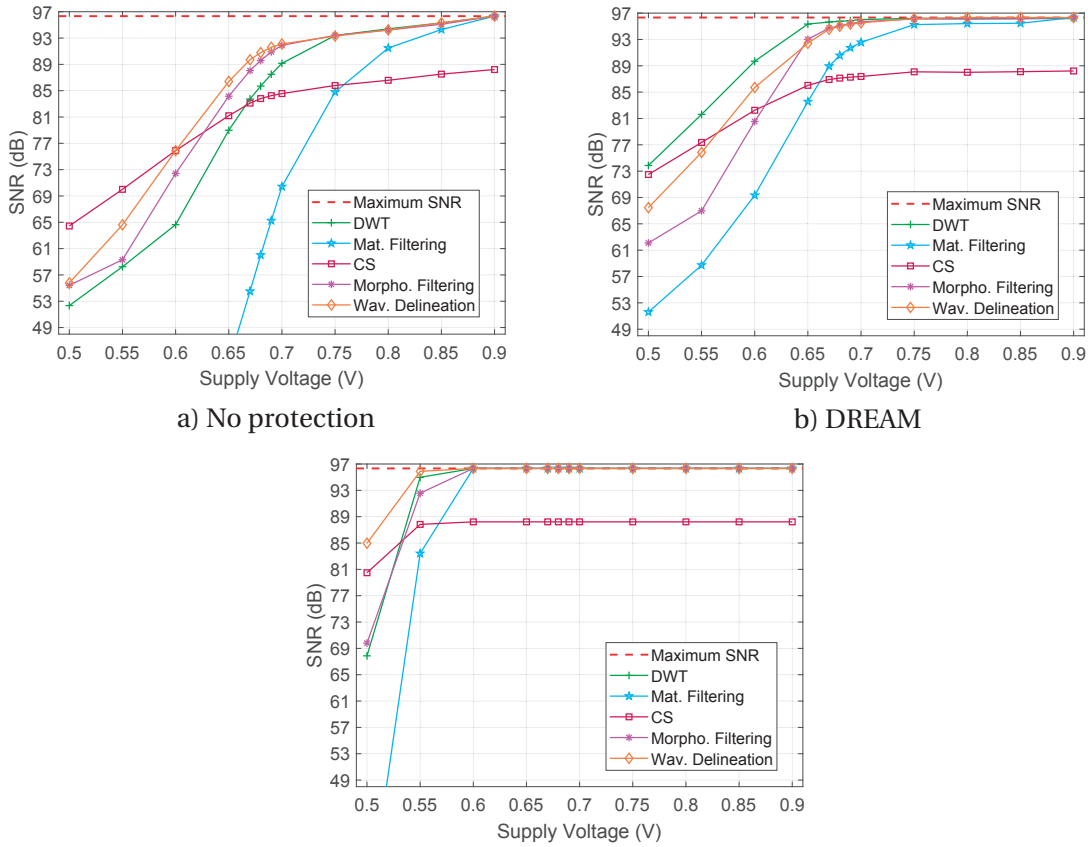
memory cells. Figure 3.33 depicts the results extracted from the characterization of the memory, performed in collaboration with the Telecommunication Circuits Laboratory (EPFL, Switzerland). This figure represents the evolution of the memory cell failure probability (i.e., BER) and number of faulty memory cells (i.e., stuck-at faults) per voltage level. As illustrated in Figure 3.33, the number of pseudo-permanent errors increases steeply when the supply voltage of the memory is aggressively reduced. In order to get a representative sample, 200 simulations per voltage level were executed. For every execution, a different set of memory error locations was generated, and all the biomedical benchmarks of Section 3.3.3.1 were evaluated. Similarly to Section 3.3.3.2, the SNR is used again as a metric to assess the quality of the final result computed by the applications for all the simulations. Then, an average of the 200 SNRs in dB is performed for every point. The dynamic and static energy consumption of the memory banks (Data + Mask Memory) have been determined through energy estimation figures produced by CACTI 6.5 [221] and the synthesis energy reports from Synopsys® Design Compiler [120] for both the encoder and decoder of the different error correction mechanisms, assuming an operating temperature of 70 °C. Lastly, the timing constraints applied to the synthesis of the designs are identical to provide comparative propagation delay penalties.

B) Experimental Results

While some error mitigation techniques may require more logic, memory, and energy, they might still have better correction abilities compared to other techniques. The objective of this experimental evaluation is to study this trade-off and determine which techniques work best for the targeted application and supply voltage. Later in this section, I demonstrate the impossibility to have an energy-efficient *one-fits-all* technique for the complete range of memory operating voltages. In more detail, this section is structured as follows. It first analyzes the degradations in output of the applications due to voltage scaling for different memory protections. Then, the resulting energy consumptions of each technique are quantified, and finally the section concludes by presenting different design trade-offs between result quality and energy savings.

B.1) Output Degradation Analysis

Figure 3.34 illustrates the evolution of the output SNR as a function of the memory supply voltage for all the applications and for different error protections. In particular, Figure 3.34.a for no protection and, respectively, Figure 3.34.b and 3.34.c for the DREAM and SEC-DED ECC protections. In all the graphs, the decrease of the SNR during voltage scaling is caused by the increase in the Bit Error Rate (BER) as the data memory supply voltage is reduced from 0.9 V to 0.5 V. At the same time, in the small memory which contains the protection bits of the error mitigation techniques, the BER remains constant and close to 0. This is achieved thanks to a constant supply voltage of 0.9 V to avoid the occurrence of pseudo-permanent errors. The dashed line on the graphs shows that the maximum SNR is equal to 96.3 dB and reached by almost all the applications when the memory supply voltage is equal to 0.9 V. This SNR value corresponds to the maximum resolution upper bound that it is possible to get with a data word



a) No protection

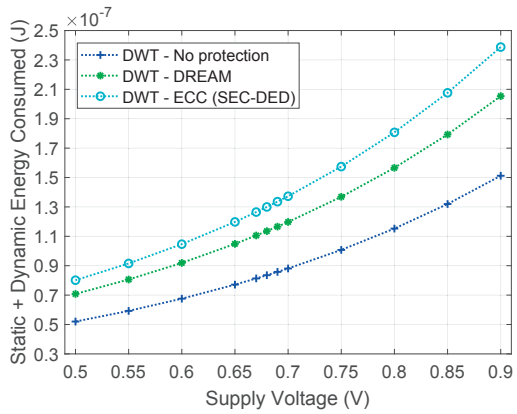
b) DREAM

c) SEC-DED ECC

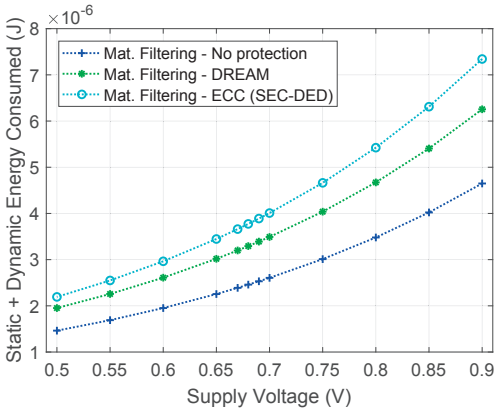
Figure 3.34 – SNR evolution vs. supply voltage for each application.

size of 16 bits when no errors affect the output. It is not infinite because, by construction, the calculation algorithms used by the applications have a finite precision. For the CS application, however, the maximum SNR is around 88 dB, because CS is, by construction, a lossy data compression algorithm that deteriorates the data even in the case of an error-free execution.

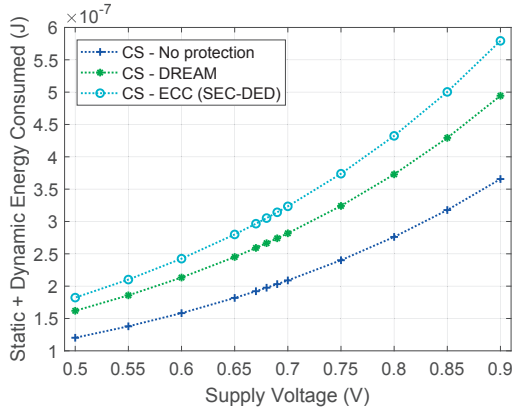
By comparing for instance the different curves of Figure 3.34, it can be observed that some applications are more robust against errors than others: the higher the SNR curve, the better the robustness of the application. Figures 3.34.b and Figure 3.34.c show that SEC-DED ECC offers a slightly better protection overall than DREAM in the range 0.55 V to 0.65 V. Below 0.55 V (with multiple errors in the same data word) SEC-DED ECC under performs for some applications, as it will detect but not correct the errors as DREAM does on MSB positions. However, in order to perform a complete and fair comparison between these two techniques, an evaluation of their energy consumptions must be done.



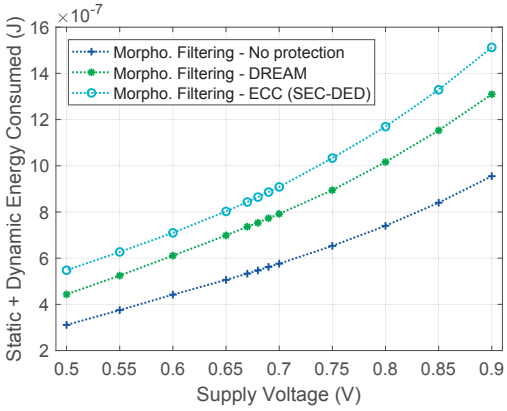
a) DWT



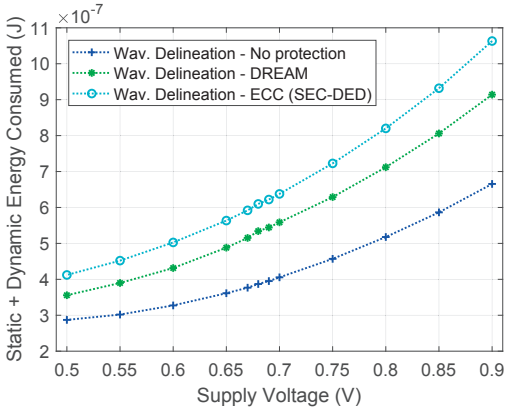
b) Matrix Filtering



c) Compressed Sensing



d) Morphological Filtering



e) Wavelet Delineation

Figure 3.35 – Memory energy consumption vs. supply voltage, for different error mitigation techniques and biomedical applications.

B.2) Energy Consumption Analysis

Figure 3.35 presents the energy consumption of all the applications for different voltage scaling levels and error protections as the sum of the static and dynamic energies consumed

by both the under-powered data memory and the highly-powered protection bits memory. The graphs show similar exponential growths in energy when the supply voltage increases; with an additional offset from one application to another, caused by its specific energy requirements (execution time and number of read/write accesses).

As presented in Figure 3.35, when SEC-DED ECC is used to protect the under-powered data memory, the experimental results show that the system consumes in average 55 % more energy for each voltage and applications compared to the case with no error protection. With the DREAM technique, the overall energy overhead is only 34 %, reducing by 21 % the overhead of ECC. The energy gap between the three curves depends on the overhead needed to apply the error protection. As shown in Section 3.3.4.2, DREAM requires 5 extra-bits per 16 bits data word, while the SEC-DED ECC protection requires 6 bits. This implies that to protect a memory the SEC-DED ECC needs an overhead of 6.3 % memory cells compared to the DREAM technique.

Moreover, after the synthesis of both techniques with Design Compiler (DC) from Synopsys® [120], it can be noted that encoders are implemented with an equivalent number of 50 logic gates. However, the decoder of the SEC-DED ECC protection requires double the amount of logic gates in comparison to the decoder of DREAM technique (70 gates for DREAM and 145 gates for the SEC-DED ECC). These observations justify also the difference between the energy overhead of both techniques.

B.3) Exploration of Trade-Offs between Result Quality and Energy Consumption

In these experimental results, ECC demonstrates slightly better correction capabilities than DREAM in the range from 0.55 V to 0.65 V of memory supply voltages. On the contrary, DREAM shows a smaller energy consumption than ECC. To get the benefits from both techniques and use the smallest amount of energy, I propose to trigger, selectively, one or the other according to the memory supply voltage and level of protection required. For example, in a real scenario where the DWT application can run with an output degradation tolerance of -1 dB, I propose the following scheme:

First, for voltages greater than 0.85V, the output distortion for the non-protected data memory is very small or even non-existent. In this range, the results indicate that no real need exists to apply error mitigation techniques on top of the data memory. From 0.85 V to 0.65 V, the output degradation becomes significant; thus, to maintain an acceptable diminution of the maximum SNR by -1 dB with a low energy and area overhead, I advocate the utilization of the DREAM technique. Then, from 0.65 V to 0.55 V, I propose to implement the SEC-DED ECC protection in order to cope with the large amount of errors occurring. According to these three ranges of voltages, I can respectively save up to 12.7 % with no protection, 30.6 % with DREAM and 39.5 % with SEC-DED ECC, compared to a system running at the nominal supply voltage (i.e., 0.9 V) with no protection.

Furthermore, for memory supply voltages lower than 0.55 V, techniques for correction of multiple errors must be employed in biomedical applications to guarantee a reliable medical output, such as an ECC with Double Error Correction / Triple Error Detection (DEC-TED ECC), that comes with a huge area and energy overhead. Although in this situation embedded systems designers would normally opt for another technique running at higher voltage with less area and energy overhead, it could be the case that the platform does not have higher supply voltages available on-board (e.g., when the battery is running low and system operation must be maintained). Therefore, embedded systems designers have no other option but to use a resource-intensive solution. Moreover, in some circumstances, energy-hungry error correction strategies, such as ECC protections may be required to improve the resilience of the memories even at high operating voltages. This is particularly the case for safety-critical biomedical applications which do not tolerate any degradation in contrast to the considered bio-signal processing algorithms of this study with relaxed reliability constraints.

In addition, as an example of improvement at the hardware level, a power management controller may be added to switch dynamically from one error mitigation technique to another when the data memory supply voltage surpasses a certain threshold. This controller can also increase the error protection when users or applications ask for better output quality and system robustness even at high or moderate memory supply voltages.

Apart from its benefits, the integration of such hardware component may induce delays and performance penalties, which must be taken into account when changing the EMT dynamically. In fact, the execution of the processors must be stalled during a certain amount of cycles to provide enough time to the controller to enable or disable an EMT and its extra memory banks (e.g., the Mask Memory in the case of DREAM). Similarly, based on the available battery charge, reducing or increasing the supply voltage of the memories is not an instantaneous operation, as it requires a transition time to be able to move from one voltage level to another.

Finally, when switching to another EMT at run-time, data protection policies must be adopted to protect (or not) the data already present in the memories. As an example, when the system switches from no protection to DREAM, the data words already saved in the memory banks are not yet associated to the proper check bits (e.g., mask ID, sign bit). Therefore, system designers can adopt different strategies to protect the previously stored data words. For instance, before lowering the supply voltage of the main data memory, the power management controller can trigger an analysis of the complete memory space, to compute and store the new check bits associated to each data word already recorded. However, in spite of ensuring the *correctness* of the data when moving from one EMT to another, this solution induces significant energy overheads and time penalties, especially with large memories. To address this issue, a low-cost strategy can be to tolerate errors in these data words, until the next writing operation to the same memory address with the new check bits generated by the current EMT. The development of the proposed power management controller supporting both strategies is left for future works and open ups the exploration of additional trade-offs between energy consumption, performance penalties and data correctness.

3.4 Summary and Concluding Remarks

In the context of personalized healthcare, emerging Smart WBSN appliances require a better energy efficiency and computing performance to operate over longer periods of time, while performing increasingly complex bio-signal processing.

In the first part of this chapter, I have addressed the problem of performance degradation induced by BTI reliability issues in the current and future technology nodes of embedded biomedical devices. BTI effects deteriorate significantly the timing properties of circuits ultimately leading to functional errors that compromise the reliability of the whole system, and in some cases, the safety of its users. In response to the formulated problem, I have introduced a complete framework able to perform short- and long-term workload-dependent BTI-aware analyses to spot and quantify functional error rates in processor pipeline stages. The proposed framework is applicable to both homogeneous and heterogeneous computing systems, with a single- or multi-core processor. The eleven steps of this comprehensive framework can either be used to improve system performance and correctness (i.e., avoiding non-optimized and unsafe operating points), or to achieve a graceful degradation of its characteristics, in accordance with the requirements of the targeted application.

As a case study, benchmark circuits from a pipeline execution stage have been selected and stimulated with a realistic pseudo-periodic workload, characteristic of real-time ECG processing applications running on energy-efficient bio-DSP platforms. Furthermore, the considered application workload presents several interesting features perfectly suited to highlight the physical phenomena involved in transistor-level BTI variability.

To carry out this reliability study, a bottom-up approach has been followed to evaluate the effects of BTI-induced degradations at four different levels. First, at the transistor level, experimental results have shown a maximum threshold voltage deviation of -58.3 mV and 51.4 mV for PMOS and NMOS transistors respectively, after 10 years of continuous system operation. Secondly, at the circuit level, the proposed Dynamic Timing Analysis (DTA) has highlighted a reduction in the maximum safe operating frequency by 9.5 % after 10 years, compared to a degradation-free (i.e., non-aged) circuit. This observation has been confirmed at the functional level by showing the growing evolution of faulty operation rates with the frequency and age of the circuit. Moreover, thanks to the aging-aware DTA method, this framework is able to determine the range of safe circuit operating frequencies without introducing worst-case guard bands compared to commercial Static Timing Analysis (STA) tools. In particular, at the application level, a quality assessment of the output results has showcased the possibility to safely raise the frequency up to 101 % above the maximum obtained with the classical STA method.

To conclude the first part of the chapter, based on the results and observations gathered during the experimental evaluation, I have provided several insights on suitable error detection and correction techniques to mitigate the effects of BTI-induced functional errors at the hardware and software levels.

Then, in the second part of this chapter, through the study of several biomedical applications designed for embedded health monitoring systems, I have proposed an approach that allows energy savings by dynamically and unequally protecting an under-powered data memory in a new way compared to regular error protection schemes. This approach relies on the *Dynamic eRror compEnsation And Masking (DREAM)* technique that reduces by approximately 21 % the energy budget consumed by traditional SEC-DED ECC.

Moreover, experimental results have highlighted that different error mitigation techniques must be used, depending on the level of voltage scaling, in order to find a trade-off between the error correction capability required and the energy budget.

As an example, in the case of the Discrete Wavelet Transform (DWT) application and according to three ranges of voltages (i.e., [0.9 ; 0.85], [0.85 ; 0.65] and [0.65 ; 0.55] Volts), I have demonstrated that it is possible to achieve different levels of energy savings at the cost of some overheads and small accuracy degradations (i.e., at the maximum -1 dB degradation on the SNR quality metric). More precisely, I have shown that the energy consumed by the memories for each supply voltage range can be minimized by 12.7 % with no protection, 30.6 % with DREAM and 39.5 % with SEC-DED ECC respectively, compared to a system running at the nominal supply voltage (0.9 V) with no protection.

4 Conclusions and Future Work

As a conclusion of this thesis, the first part of this chapter highlights the principal contributions of my research work beyond the state-of-the-art, as well as the resulting impact on the international research community. Additionally, in its last part, I present several research lines for future work based on the results provided in this thesis.

4.1 Summary and Contributions

In this thesis, I have proposed a set of architectural and technological solutions to further enhance the energy efficiency and reliability of current wearable health monitoring devices. First, following a top-down approach, driven by the software characteristics of bio-signal processing algorithms, I have studied the benefits offered by the development of domain-specific and energy-efficient computing platform architectures. Finally, I have investigated how the modeling of technology reliability issues in logic and memory components can be exploited to adequately adjust the frequency and voltage parameters of the circuit, with the aim of optimizing its computing performance and energy savings.

The following sections provide a more detailed summary of the contributions introduced in each chapter, and discuss the outcome of the different experimental evaluations performed in this thesis:

Heterogeneous and Reconfigurable Energy-Efficient Bio-Signal Processing Architectures

In Chapter 2, I have performed an architectural exploration of a heterogeneous and reconfigurable platform devoted to bio-signal processing. This domain-specific platform is able to achieve higher performance and energy savings, beyond the capabilities of a baseline multi-processor system. In particular, I have introduced several architectural solutions to exploit and maximize the parallel computations of commonly employed biomedical applications.

In the medical field, the application workload is often divided between control-dominated phases and computationally-intensive phases in the form of compact loops (i.e., kernels).

The proposed computing platform can efficiently support both: the former on multiple ultra-low power processing cores, the latter by employing a Coarse-Grained Reconfigurable Array (CGRA) accelerator, interfaced to the cores as a shared acceleration resource. This strategy, relying on a hardware accelerator, can result in orders-of-magnitude energy reductions by allowing a precise optimization of the software application execution.

In this work, I have proposed three different architectural versions of the CGRA accelerator, which contribute differently to the maximization of the application parallelization. Each version of the CGRA is interfaced with a multi-core system, composed of eight TamaRISC cores, interconnected to multi-banked data and instruction memories through combinational crossbars.

Firstly, I have introduced a *Single-Datapath CGRA* architecture that is shared by multiple cores and able to process several acceleration requests in parallel on the different computing resources (i.e., reconfigurable cells) of the CGRA mesh. It is the simplest CGRA design proposed in this work, with the minimum area and energy overheads. It allows system-wide energy savings of up to 18.6 %, when executing complex ECG processing applications, in comparison to an equivalent multi-core solution without CGRA acceleration of kernels.

Secondly, I have proposed a *Multi-Datapath CGRA* design, allowing the platform to support single-instruction multiple-data (SIMD) execution modes, both at the processor and CGRA levels. In this way, the platform leverages SIMD in order to (1) merge the memory accesses from different processors running in lock-step, (2) minimize the number of reconfigurations and the energy consumption required to execute accelerated kernels on the CGRA, and (3) reduce the access contention to the CGRA resource by coalescing multiple acceleration requests into the same reconfigurable cells. Thus, the proposed platform is perfectly tailored to the acceleration of heavily-parallel biomedical applications. Thanks to its optimized architecture, energy savings of up to 37.2 % are achievable when executing real-world bio-signal processing applications, compared to a multi-core system without hardware acceleration capabilities.

Lastly, I have showcased the efficiency of an *Interleaved-Datapath CGRA*, which is able to parallelize the execution of an acceleration request from a single core over multiple processing datapaths and inside the same computing resources. Its design is particularly suitable for the acceleration of kernels with a large number of iterations and/or called by a single core in the whole application. The interleaved run-time scheme of this CGRA architecture maximizes the utilization of resources and available bandwidth between the CGRA and the data memory. This leads to notable run-time and energy efficiency gains. In particular, this architecture enables a reduction of up to 69.4 % of the energy consumed by the accelerated kernels, compared to their execution on the processors, without hardware accelerator.

In a nutshell, the three proposed CGRA datapath architectures are complementary. They can be employed individually or in combination to address different processing needs. More precisely, the *Single-Datapath* version represents the generic CGRA architecture of this study with a minimum area overhead. It can bring reasonable performance and energy improvements

for both single- and multi-core systems. The second CGRA design featuring a *Multi-Datapath* architecture is suitable for multi-core biomedical applications requesting the concurrent execution of many identical kernel accelerations on the same CGRA resources, thus minimizing the energy overheads resulting from reconfiguration. Finally, the *Interleaved-Datapath* architecture is adapted for the acceleration in parallel of individual kernels with a large amount of processing iterations and requested by a single core. Therefore, compared to the *Multi-Datapath* CGRA, the *Interleaved-Datapath* design is more suited for single-core biomedical applications or multi-core systems with disjoint executions of the processors.

Publications: All of the work presented in Chapter 2 has been highly appreciated by the circuits and systems community, and has lead to several publications. First of all, the work based on a single-datapath CGRA architecture shared by a multi-core system has been presented at the *Biomedical Circuits and Systems (BioCAS) Conference* [222]. Then, the work on the multi-datapath SIMD-CGRA architecture has been accepted for publication in the journal *IEEE Transactions on Circuits and Systems I (TCAS-I)* [51]. Thirdly, the work on the interleaved-datapath CGRA has been published in the journal *IEEE Embedded Systems Letter (ESL)* [223]. Finally, the developed experimental framework has been also used for a joint project on inexact/near-threshold computing with *Soumya Basu*, and was presented at the *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)* [56], and at the *International Symposium on Circuits and Systems (ISCAS)* [65].

Technology-Level Reliability Exploration in Energy-Efficient Biomedical Systems

A) Technology-Level Reliability Exploration in Logic Components: Deeply-scaled technology nodes suffer more and more from variability and aging issues, such as the Bias Temperature Instability (BTI). The BTI-induced degradation affects dramatically the timing properties of circuits ultimately leading to functional errors that jeopardize the reliability of the whole system. To tackle this issue, in the first part of Chapter 3, I have introduced a complete framework able to perform workload-dependent BTI-aware analysis to identify and quantify functional error rates in processor pipeline stages. The proposed framework is applicable to both homogeneous and heterogeneous computing systems, with a single- or multi-core processor. The eleven steps of this comprehensive framework can either be used to improve system performance and exactness (i.e., avoiding unsafe operating points), or to achieve a graceful degradation of its characteristics, in accordance with the requirements of the targeted application. Furthermore, at the heart of this framework, a methodology has been developed to derive the long-term and workload-dependent BTI degradation from a precise atomistic trap-based model. The considered application workload, produced by a widely employed ECG processing algorithm, presents several interesting features perfectly suited to highlight the physical phenomena involved in transistor-level BTI variability.

To carry out this reliability study, a bottom-up approach has been followed to evaluate the effects of BTI-induced degradations at four different levels. First, at the transistor level, experimental results have shown a maximum threshold voltage deviation of -58.3 mV and 51.4 mV

for PMOS and NMOS transistors respectively, after 10 years of continuous system operation. Secondly, at the circuit level, the proposed Dynamic Timing Analysis (DTA) has highlighted a reduction in the maximum safe operating frequency by 9.5 % after 10 years, compared to a degradation-free (i.e., non-aged) circuit. This observation has been confirmed at the functional level by showing the growing evolution of faulty operation rates with the frequency and age of the circuit. Moreover, thanks to the aging-aware DTA method, this framework is able to determine the range of safe circuit operating frequencies without introducing worst-case guard bands compared to commercial Static Timing Analysis (STA) tools. In particular, at the application level, a quality assessment of the output results has showcased the possibility to safely raise the frequency up to 101 % above the maximum obtained with the classical STA method.

Lastly, based on the results and observations gathered during the experimental evaluation, I have provided several insights on suitable error detection and correction techniques to mitigate the effects of BTI-induced functional errors at the hardware and software levels.

Publication: The work on BTI-aware logic circuit design, presented in Chapter 3, will be submitted shortly to a prestigious journal from the computer-aided circuit design community.

B) Technology-Level Reliability Exploration in Memory Components: Aggressive voltage scaling is a widely-employed technique to drastically reduce the energy consumption of the most energy-hungry components, such as memories. The lower bound is given by the fact that reliability issues appear as the supply voltage approaches the threshold voltage of the transistors. While embedded memories often rely on Error Correction Codes (ECC) for error protection, in the second part of Chapter 3, I have explored how the characteristics of biomedical applications can be exploited to develop new techniques with lower energy overheads. In particular, my initial study of a representative set of bio-signal processing applications has suggested that their intrinsic resilience can be coupled with significance-based computing, enabling new possibilities for energy savings.

Based on this initial study, I have proposed the *Dynamic eRror compEnsation And Masking (DREAM)* technique, a new asymmetric error mitigation technique that provides partial memory protection with less area and energy overheads than a traditional ECC with Single Error Correction and Double Error Detection (SEC-DED) capabilities [81]. Different trade-offs between the pseudo-permanent error correction ability of the techniques and their energy consumption were examined to conclude that, when properly applied, DREAM consumes 21% less energy than the SEC-DED ECC. Additionally, I have introduced a methodology to minimize the energy consumption of ultra-low power architectures by dynamically adjusting the level of protection of the system memories in order to get the desired output result quality. It combines the aforementioned multiple error mitigation techniques (e.g., DREAM and SEC-DED ECC), to obtain the best trade-off between error correction ability and energy consumption, for a given memory supply voltage and biomedical application.

By demonstrating the ability to produce acceptable application results under extreme operating and reliability conditions, I paved the way for the development of promising wearable biomedical devices with reduced energy consumptions and increased computing capabilities.

Publication: The work on energy versus reliability trade-offs exploration in memories, presented in Chapter 3, has been well received by the design automation community and accepted for publication at the *Design Automation and Test in Europe (DATE) Conference* [52].

4.2 Future Work

The research fields covered in the different chapters of this thesis are continuously explored by their respective communities. Following the presentation of my research results in the previous section, I provide below some of the possible short- and long-term research lines that can stem from my work.

First, I underline a set of research directions to complete the architectural exploration carried out in this thesis:

- **Automatic kernel mapping framework for advanced CGRA architectures:** As mentioned in Section 2.3.2.3, the identification and selection of the kernels from the applications have been assisted by a profiling pass, built on top of the LLVM compiler infrastructure [110]. Additionally, the scheduling and mapping of the kernels have been manually done in this work. While automated strategies to individually perform these tasks have already been presented in the literature [97, 109, 113], they are still not optimized for the proposed heterogeneous and reconfigurable platform. A short-term research line will be to explore the development of a complete compiler framework, able to decompose a single-thread application into a multi-threaded representation of the program, from which computationally-intensive kernels can be identified inside each thread. The selection of the kernels can be based on user-defined rules matching with the architecture of the employed accelerator (e.g., single, multi or interleaved-datapaths SIMD-CGRA). This framework will open the exploration of several scheduling and mapping trade-offs, such as the energy consumption and the quantity of allocated resources (e.g., CGRA columns, number of configuration registers and datapaths per cell) versus the execution speed of the kernel. It will also enable the comparison between the achievable performance of a manual versus automatic scheduling and mapping of the kernels. By proposing a generic and reconfigurable approach, this compiler framework will be also applicable to other types of parallel computing platforms, interfaced with one or several hardware accelerators. Nowadays, these types of platforms are becoming more common to address the demise of Moore's law.

- **Methodology for selection of optimized architectural parameters:** In this thesis, the main constraint imposed on the design of the heterogeneous and reconfigurable platform was to improve its energy efficiency, compared to a baseline multi-core system.

Similarly to RTL synthesis tools [120] operating at the gate level, an interesting area of research will be to develop a methodology allowing an automated selection of the appropriate design parameters at the architectural level. This selection can be based on user-defined constraints, specifying for instance, the required run-time performance, the maximum energy consumption, and the maximum area of the design. This methodology may also enable the exploration of the knobs employed to tune the parameters of the system with respect to a specific application workload. For instance, with the proposed platform, the selection of the optimal operating point will affect the computing architecture by increasing or decreasing the number of cores, including or not the CGRA, modifying the size of the CGRA mesh, enabling different datapath configurations and so forth. This short-term research line will enable embedded systems designers to determine the configuration ranges or boundaries of the space exploration, providing benefits to the overall system for a domain-specific application.

- **Accelerator-based computing platforms for the edge computing era:** Nowadays, the computing performance of Internet-of-Things (IoT) devices, such as Wireless Body Sensor Nodes (WBSNs), is dictated by the edge computing paradigm. In fact, to face the network latencies, bandwidth limitations, and permanent coverage requirements, the data produced by these devices is intensively analyzed at the edge of the network before being sent to a cloud. In the context of health monitoring devices, this paradigm is putting even more pressure on the design of the resource-constrained computing architecture, devoted to bio-signal analysis. To support even more computationally demanding applications, such as ECG heartbeat classification based on neural networks [224] and machine learning algorithms for medical diagnoses [131, 132], some of the research avenues under exploration consider hardware solutions based on SIMD multi-processing architectures and many-core systems supporting custom accelerators. Incorporating the proposed CGRA components into these edge computing platforms is a natural extension of the architectural exploration performed in this thesis.

Additionally, I introduce several future research directions to complement the technology-level exploration performed in this thesis:

- **Extension of the BTI evaluation framework:** As shown in Chapter 3, intensive efforts have been carried out to significantly increase the accuracy of the workload-dependent BTI-aware analysis. However, several enhancements can be made to further extend the scalability and the capabilities of the framework. The suggested improvements are presented in the following lines.

  In this thesis, the framework has been employed to evaluate the BTI-induced functional errors exclusively inside fully combinatorial circuits. Nonetheless, an interesting long-term research direction will be to adapt the current framework for the analysis of bigger circuits including sequential components, such as complete multi-stage processor pipelines or CGRA meshes. To do so, several challenges must be addressed. In particular,

the abstraction level of the BTI modeling must be raised from the transistor to the gate level to speed-up the analysis without sacrificing its accuracy.

In addition, a higher abstraction of the BTI modeling will also unlock the possibility to perform a *closed-loop* BTI degradation analysis inside a complete simulation framework, involving a benchmark application running on top of a cycle-accurate simulator. In fact, as mentioned in Chapter 3, BTI degradations are dependent of the time, application workload, operating temperature, voltage, and frequency of the circuit. In this work, the operating parameters (i.e., temperature, voltage, and frequency) are statically defined before each BTI analysis. Conversely, in a close-loop simulation environment, the operating parameters may change dynamically at run-time based on the application workload. This is often the case for systems trying to reach different levels of performance and/or energy savings, thanks to Dynamic Voltage and Frequency Scaling (DVFS). By employing a closed-loop simulation framework, the level of BTI degradations can evolve accurately with the parameters of the circuit. In this way, the reliability of the computing platform is compromised differently all along the execution of the application. This feature enables the deep study of several error detection and recovery techniques for counteracting BTI reliability issues, by adjusting the parameters of the system at run-time, similarly to the technique proposed in Section 3.2.3.

Furthermore, as presented in Section 3.2.2.3, the obtained BTI degradation is relatively small for the targeted bio-signal processing application. In fact, when operating at high frequencies, the application highlights long idle periods for all the processed samples. These relaxation periods provide enough time to the transistors to recover from the stress accumulated during the active periods of the processing. Following this observation, it will be interesting to evaluate the BTI degradation induced by intensive application workloads from another domain (e.g., smartphone or high-performance cloud computing applications), and characterized by higher active/idle time ratios (i.e., processor utilization rates) [225]. Moreover, this study can also be used to showcase the flexibility of the BTI evaluation framework and its ability to adapt the analysis to different types of workloads, either from the embedded processing or high-performance computing domain.

Finally, as presented in the first part of Chapter 3, the BTI degradation is even more prominent with ultra-scaled technology nodes, such as 7 nm chips currently produced. This observation motivates the importance to evaluate the impact of BTI on these new devices, but also when it is combined with other aging effects thanks to the reusable and generic structure of the framework (see Section 3.2).

- **Advanced multi-level technique for joint parametric timing failures and BTI-induced functional errors mitigation:** Beyond the hardware/software BTI error mitigation approach presented in Section 3.2.3, other techniques can be developed to solve the reliability concerns occurring at the physical level, and propagated in a bottom-up way, from the transistors to the software application [140]. An emerging strategy is to detect and mitigate the errors in parallel at different abstraction levels (e.g., physical, circuit,

functional and application levels), with reduced area/energy costs. In this scenario, the area/energy overheads can be minimized by not imposing a 100 % error correction constraint at each level. In fact, if an error is not detected at a certain level, it will be caught by the higher ones. The exploration of this strategy can exploit the combined efficiency of different error detection/correction techniques to address the timing and functional issues, caused by process variations or BTI aging effects.

- **Reliability exploration in the current and future semiconductor technologies:** Over the years, the increased transistor density per chip has been accompanied with a variety of technological challenges. By shrinking technology nodes towards atomic dimensions, a plethora of reliability concerns are becoming more dominant, thus jeopardizing the performance and proper functioning of the current and next generation of integrated circuits. As shown in this thesis, energy-efficient circuit architectures need to address these issues, especially when the operating parameters (e.g., frequency and voltage) are pushed beyond the traditional safety limits, to achieve higher performance or energy-savings. In addition to BTI degradations in the logic and stuck-at faults in the memories (see Chapter 3), CMOS circuit designers must also cope with other types of spatial and temporal unreliabilities, such as process variations, Random Telegraph Noise (RTN), Hot Carrier Injection (HCI), low- and high-κ Time-Dependent Dielectric Breakdown (TDDB), Electromigration (EM), Electromagnetic interference, power supply signal noise and so forth [66]. The development of run-time detection and mitigation techniques to face the functional errors induced by these silicon level failures represents a long-term research line, which will be beneficial not only for domain-specific and energy-efficient architectures, but also for the whole digital circuit design community.

A Appendix: BTI Degradations Evaluation

This appendix provides additional information on the experimental setup and results presented in the Section 3.2.2 of Chapter 3. In particular, it contains the following elements:

- Table A.1 enumerates all the arithmetic and logic operations performed by the employed benchmark circuit *ALU16*.
- Listing A.1 gives an example of statistics recording with the faulty operations occurring during two consecutive CDW point periods. For clarity reasons, the faulty operations with the same parameters (i.e., operation code, input operands and degraded output results) are only listed once per CDW point period. Moreover, the corrupted *Result_Out* signal is represented in hexadecimal on 16 bits, since the considered biomedical application uses only the 16 least significant bits of the 32 bits result generated by the multiplier *Mult16*.
- Listing A.2 shows an example of signed multiplication operation with all the intermediate results delivered in output of the multiplier *Mult16*.
- Table A.2 provides a detailed record of the different experimental points obtained during the quality assessment of the results produced by the ECG delineation application after 10 years of BTI aging.

Table A.1 – Operation code (*Op Code*) from the 16 bits pipeline execution stage (*ALU16*).

Op Code (Hex)	Op Code (Binary)	Operation description	Sub-circuit name
0x0	0000	Addition without carry in	*Adder16*
0x1	0001	Addition with carry in	*Adder16*
0x2	0010	Subtraction	*Adder16*
0x3	0011	Subtraction with borrow	*Adder16*
0x4	0100	Logical AND operation	-
0x5	0101	Unused operation code	-
0x6	0110	Logical OR operation	-
0x7	0111	Unused operation code	-
0x8	1000	Logical XOR operation	-
0x9	1001	Logical left shift (zeros shifted in)	-
0xA	1010	Logical right shift (zeros shifted in)	-
0xB	1011	Arithmetic right shift (sign bit shifted in)	-
0xC	1100	Unsigned multiplication with accumulation	*MAC8*
0xD	1101	Signed multiplication with accumulation	*MAC8*
0xE	1110	Unsigned multiplication	*Mult16*
0xF	1111	Signed multiplication	*Mult16*

Listing A.1 – Excerpt from a record of faulty operations induced by BTI degradations and performed by the *ALU16* circuit.

```
<List of input/output signals>
<Op_Code Op_A Op_B Op_C Carry_In Carry_Out_With_Aging Result_Out_With_Aging>

...

<CDW_ID::1745>
<CDW_Start_Time(ps): 1162005689626.000>
<CDW_End_Time(ps):   1162011734184.000>
0x0     0x1     0xFFFF 0x0     0x0     0x0     0xC000
0x2     0x0     0x1    0x0     0x1     0x1     0x1FFF
0x2     0x0     0x2    0x0     0x1     0x1     0x3FFE
0x2     0x1     0x2    0x0     0x1     0x1     0x3FFF
0x2     0x1     0x3    0x0     0x1     0x1     0x3FFE
0x2     0x2     0x3    0x0     0x1     0x1     0x1FFF
0x2     0x3     0x6    0x0     0x1     0x0     0x7FFD
0x2     0x8     0x9    0x0     0x1     0x1     0x1FFF
0xC     0xFFFE  0xFFED 0x0     0x0     0x0     0xBF26
0xC     0xFFFF  0xFFEE 0x0     0x0     0x0     0xFD12
0xD     0xFFFE  0xFFED 0xEB26  0x0     0x1     0xEB4C
0xD     0xFFFF  0xFFEE 0xED12  0x0     0x0     0x6D24
0xF     0x2     0x64   0x0     0x0     0x0     0x80C8
0xF     0x2     0x9    0x0     0x0     0x0     0xE012
0xF     0x2     0x96   0x0     0x0     0x0     0x812C

<CDW_ID::1746>
<CDW_Start_Time(ps): 1163999644455.000>
```

```
<CDW_End_Time(ps):    1164005023705.000>
0x2     0x0     0x1     0x0     0x1     0x1     0x3FFF
0x2     0x1     0x2     0x0     0x1     0x0     0x7FFF
0x2     0x1     0x3     0x0     0x1     0x0     0x7FFE
0x2     0x2     0x3     0x0     0x1     0x1     0x3FFF
0x2     0x8     0x9     0x0     0x1     0x1     0x3FFF
0x2     0xFFF0  0xFFF1  0x0     0x1     0x0     0x7FFF
0xF     0x1     0x9     0x0     0x0     0x0     0x4009
0xF     0x1     0x96    0x0     0x0     0x0     0x2896
0xF     0x2     0x96    0x0     0x0     0x0     0x812C

...
```

Listing A.2 – Example of signed multiplication operation with its intermediate results in output of the *Mult16* circuit.

```
At time = 0 ps (starting of the clock period):
Operand A (Op_A) = 0x0000 (16 bits)
Operand B (Op_B) = 0x0064 (16 bits)

-----------------------------------------------

Normalized   Multiplier output result (32 bits)
timestamp       (Op_A x Op_B => Result_Out)
   (ps)

  166.49       00000000000000000000000000000001
  220.20       00000000000000000000000100000000
  231.65       00000000000000000000001100000000
  244.20       00000000000000000000011100000000
  260.21       00000000000000000000111100000000
  291.74       00000000000000000001111100000000
  300.85       00000000000000000011111100000000
  308.98       00000000000000000111111100000000
  344.12       00000000000000000111111100000000
  345.17       00000000000000001011111100000000
  357.71       00000000000000001101111100000000
  360.99       00000000001000011011111100000000
  361.79       00000000001100011011111100000000
  362.72       00000000001100111011111100000000
  362.92       00000000001101111011111100000000
  363.33       00000000001111111011111100000000
  376.15       00000000001111111011111000000000
  376.45       00000000011111111011111000000000
  378.29       00000000011111111111111000000000
  402.32       00000000011111111011111000000000
  403.12       00000000011110111011111000000000
  407.40       00000000111110111011111000000000
  411.62       00000000111110111011110000000000
  424.23       00000001111110111011110000000000
  435.95       00000001111110111011100000000000
  456.24       00000011111110111011100000000000
  456.86       00000011111110111011000000000000
  458.00       00000111111111111011000000000000
  463.61       00000111111111111110000000000000
```

```
  471.65       00000111111111111111000000000000
  493.12       00000111111111111110000000000000
  503.22       00001111111111111110000000000000
  518.23       00011111111111111110000000000000
  525.52       00011111111111111100000000000000
  548.77       00011111110111111100000000000000
  553.51       00111111110111111100000000000000
  557.03       01111111110111111100000000000000
  560.34       01111111110111111000000000000000
  561.98       01111111110011111000000000000000
  567.27       11111111110011111000000000000000
  568.11       11111111100011111000000000000000
  590.17       11111111000011111000000000000000
  591.57       11111111000011110000000000000000
  608.23       11111111000010110000000000000000
  608.59       11111110000010110000000000000000
  611.73       11111110001010110000000000000000
  621.45       11111110001010100000000000000000
  626.24       11111110001010000000000000000000
  627.89       11111110011010000000000000000000
  629.92       11111110011000000000000000000000
  630.76       11111100011000000000000000000000
  649.79       11111000011000000000000000000000
  650.30       11111000010000000000000000000000
  662.56       11111000000000000000000000000000
  672.05       11110000000000000000000000000000
  690.86       11100000000000000000000000000000
  718.07       11000000000000000000000000000000
  727.55       01000000000000000000000000000000
  731.57       00000000000000000000000000000000 <= Final result
```

Table A.2 – Quality assessment of the output results generated by the MMD ECG delineation application under BTI degradations (after 10 years) for different circuit operating frequencies.

Circuit Operating Frequency (MHz)	ECG Quality	ECG Fiducial Points Positions											
		P Wave				QRS Complex				T Wave			
	PRD (%)	Correct (%)	Time Deviation (ms)	Misplaced (%)	Missing (%)	Correct (%)	Time Deviation (ms)	Misplaced (%)	Missing (%)	Correct (%)	Time Deviation (ms)	Misplaced (%)	Missing (%)
554.57	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
749.66	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
998.22	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
1052.57	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
1110.24	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
1142.41	1.97	66.67	0.32	11.11	22.22	100.00	0.00	0.00	0.00	100.00	0.00	0.00	0.00
1170.13	3.98	66.67	0.67	11.11	22.22	100.00	0.44	0.00	0.00	100.00	0.00	0.00	0.00
1172.33	4.60	66.67	0.75	11.11	22.22	100.00	0.44	0.00	0.00	100.00	4.22	0.00	0.00
1173.71	9.34	66.67	0.87	11.11	22.22	100.00	0.44	0.00	0.00	100.00	6.00	0.00	0.00
1178.78	11.91	66.67	1.00	11.11	22.22	100.00	0.44	0.00	0.00	100.00	7.11	0.00	0.00
1186.71	14.97	66.67	3.01	11.11	22.22	100.00	0.44	0.00	0.00	100.00	7.56	0.00	0.00
1194.16	16.07	66.67	4.67	11.11	22.22	100.00	1.56	0.00	0.00	100.00	9.75	0.00	0.00
1218.20	23.36	66.67	10.80	11.11	22.22	100.00	1.78	0.00	0.00	100.00	10.00	0.00	0.00
1242.23	24.01	66.67	11.00	11.11	22.22	100.00	1.78	0.00	0.00	88.89	11.33	0.00	11.11
1303.23	25.66	66.67	12.40	11.11	22.22	100.00	2.00	0.00	0.00	88.89	11.50	0.00	11.11
1325.41	28.37	55.56	13.20	11.11	33.33	100.00	2.44	0.00	0.00	88.89	11.50	0.00	11.11
1347.59	30.59	55.56	16.67	11.11	33.33	100.00	5.78	0.00	0.00	77.78	14.57	0.00	22.22
1391.96	401.07	0.00	-	0.00	100.00	0.00	-	0.00	100.00	66.67	20.33	0.00	33.33
1414.14	1127.42	0.00	-	0.00	100.00	0.00	-	0.00	100.00	22.22	39.00	0.00	77.78
1436.32	3981.58	0.00	-	0.00	100.00	0.00	-	0.00	100.00	0.00	-	0.00	100.00
1500.10	8354.77	0.00	-	0.00	100.00	0.00	-	0.00	100.00	0.00	-	0.00	100.00
1550.01	13067.66	0.00	-	0.00	100.00	0.00	-	0.00	100.00	0.00	-	0.00	100.00
1600.48	14034.70	0.00	-	0.00	100.00	0.00	-	0.00	100.00	0.00	-	0.00	100.00

Bibliography

[1] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2688–2710, October 2010.

[2] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, January 2010.

[3] World Health Organization, "Health in 2015: from MDGs to SDGs," http://www.who.int/gho/publications/mdgs-sdgs/en/, 2015, [Last accessed on May 1st 2018].

[4] World Health Organization, "Global health risks," http://www.who.int/healthinfo/global_burden_disease/global_health_risks/en/, 2009, [Last accessed on May 1st 2018].

[5] World Health Organization, "Cardiovascular diseases," http://www.who.int/topics/cardiovascular_diseases/en, August 2014, [Last accessed on May 1st 2018].

[6] MEP Heart Group, "Cardiovascular diseases facts and figures," http://www.mepheartgroup.eu/index.php/facts-a-figures, 2015, [Last accessed on May 1st 2018].

[7] P. Bonato, "Wearable sensors and systems," *IEEE Engineering in Medicine and Biology Magazine*, vol. 29, no. 3, pp. 25–36, May 2010.

[8] E. Jovanov, A. Milenkovic, C. Otto, and P. De Groen, "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," *Journal of NeuroEngineering and rehabilitation*, vol. 2, no. 1, p. 6, 2005.

[9] J. Espina, T. Falck, J. Muehlsteff, and X. Aubert, "Wireless body sensor network for continuous cuff-less blood pressure monitoring," in *2006 3rd IEEE/EMBS International Summer School on Medical Devices and Biosensors*, Sept 2006, pp. 11–15.

[10] T. Torfs, V. Leonov, R. F. Yazicioglu, P. Merken, C. V. Hoof, R. J. M. Vullers, and B. Gyselinckx, "Wearable autonomous wireless electro-encephalography system fully powered by human body heat," in *2008 IEEE Sensors*, Oct 2008, pp. 1269–1272.

[11] R. Vecht, M. A. Gatzoulis, and N. Peters, "ECG diagnosis in clinical practice," London, UK, 2009.

[12] S. Dee Unglaub, *Human Physiology: An Integrated Approach*. Pearson, January 2018.

[13] Texas Instruments, S. Kamath and J. Lindh, "Measuring Bluetooth Low Energy Power Consumption," http://www.ti.com/lit/an/swra347a/swra347a.pdf, 2012, [Last accessed on May 1st 2018].

[14] R. Braojos, A. Y. Dogan, I. Beretta, G. Ansaloni, and D. Atienza, "Hardware/software approach for code synchronization in low-power multi-core sensor nodes," in *Proc. of the 2014 Design, Automation Test in Europe Conference Exhibition (DATE)*. IEEE, March 2014, pp. 1–6.

[15] F. Zhang, J. Holleman, and B. P. Otis, "Design of ultra-low power biopotential amplifiers for biosignal acquisition applications," *IEEE transactions on biomedical circuits and systems*, vol. 6, no. 4, pp. 344–355, January 2012.

[16] American Heart Association, "Holter Monitor," http://www.heart.org/HEARTORG/Conditions/HeartAttack/DiagnosingaHeartAttack/Holter-Monitor_UCM_446437_Article.jsp#.Woqjd302Xqk, January 2018, [Last accessed on May 1st 2018].

[17] R. Braojos, H. Mamaghanian, A. D. Junior, G. Ansaloni, D. Atienza, F. J. Rincón, and S. Murali, "Ultra-low power design of wearable cardiac monitoring systems," in *Proc. of the 51st Annual Design Automation Conference (DAC)*. ACM, June 2014, pp. 1–6.

[18] F. Chen, F. Lim, O. Abari, A. Chandrakasan, and V. Stojanovic, "Energy-aware design of compressed sensing systems for wireless sensors under performance and reliability constraints," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 3, pp. 650–661, March 2013.

[19] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Compressed sensing for real-time energy-efficient ECG compression on wireless body sensor nodes," *IEEE Transactions on Biomedical Engineering (T-BME)*, vol. 58, no. 9, pp. 2456–2466, September 2011.

[20] D. Daly and A. Chandrakasan, "A 6-bit, 0.2 V to 0.9 V highly digital flash ADC with comparator redundancy," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 11, pp. 3030–3038, November 2009.

[21] E. Nemati, M. Deen, and T. Mondal, "A wireless wearable ECG sensor for long-term applications," *IEEE Communications Magazine*, vol. 50, no. 1, pp. 36–43, January 2012.

[22] R. Braojos, D. Atienza, M. Aly, T. F. Wu, H. Wong, S. Mitra, and G. Ansaloni, "Nano-engineered architectures for ultra-low power wireless body sensor nodes," in *Proc. of the 2016 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, October 2016, pp. 1–10.

[23] Y. Hao and R. Foster, "Wireless body sensor networks for health-monitoring applications," *Physiological measurement*, vol. 29, no. 11, p. R27, October 2008.

[24] A. Y. Dogan, J. Constantin, D. Atienza, A. Burg, and L. Benini, "Low-power processor architecture exploration for online biomedical signal analysis," *IET Circuits, Devices Systems (IET-CDS)*, vol. 6, no. 5, pp. 279–286, September 2012.

[25] Y. He, Y. Pu, R. Kleihorst, Z. Ye, A. A. Abbo, S. M. Londono, and H. Corporaal, "Xetal-Pro: An Ultra-Low Energy and High Throughput SIMD Processor," in *Proc. of the 47th Design Automation Conference (DAC)*. ACM, June 2010, pp. 543–548.

[26] J. Kwong and A. P. Chandrakasan, "An energy-efficient biomedical signal processing platform," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 7, pp. 1742–1753, June 2011.

[27] J. Kalyanasundaram and Y. Simmhan, "Arm wrestling with big data: A study of arm64 and x64 servers for data intensive workloads," *arXiv preprint arXiv:1701.05996*, 2017.

[28] ARM, "Cortex-M0 Processor," https://www.arm.com/products/processors/cortex-m/cortex-m0.php, [Last accessed on May 1st 2018].

[29] "Physiobank," http://www.physionet.org/physiobank/, August 2016, [Last accessed on May 1st 2018].

[30] J. Schlachter, V. Camus, K. V. Palem, and C. Enz, "Design and applications of approximate circuits by gate-level pruning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 5, pp. 1694–1702, 2017.

[31] Texas Instruments, "MSP430 ultra-low power sensing & measurement MCUs," http://www.ti.com/microcontrollers/msp430-ultra-low-power-mcus/overview.html, May 2018, [Last accessed on May 1st 2018].

[32] A. Y. Dogan, "Energy-aware processing platform exploration for embedded biosignal analysis," Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland, 2013.

[33] P. Gautham, R. Parthasarathy, and K. Balasubramanian, "Low-power pipelined mips processor design," in *Integrated Circuits, ISIC'09. Proceedings of the 2009 12th International Symposium on*. IEEE, 2009, pp. 462–465.

[34] H.-G. Han, S. Zhang, and J.-F. Qiao, "An adaptive growing and pruning algorithm for designing recurrent neural network," *Neurocomputing*, vol. 242, pp. 51–62, 2017.

[35] R. Braojos, G. Ansaloni, D. Atienza, and F. J. Rincón, "Embedded real-time ECG delineation methods: A comparative evaluation," in *Proc. of the IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE)*, November 2012, pp. 99–104.

[36] L. Polania, R. Carrillo, M. Blanco-Velasco, and K. Barner, "Compressed sensing based method for ECG compression," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 761–764.

[37] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Real-time compressed sensing-based electrocardiogram compression on energy-constrained wireless body sensors," in *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2011, pp. 1744–1747.

[38] R. Braojos, "Hardware/software co-design of ultra-low power biomedical monitors," Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland, 2016.

[39] D. Bortolotti, M. Mangia, A. Bartolini, R. Rovatti, G. Setti, and L. Benini, "Rakeness-based compressed sensing on ultra-low power multi-core biomedical processors," in *Proc. of the 2014 Conference on Design and Architectures for Signal and Image Processing (DASIP)*, October 2014, pp. 1–8.

[40] G. Ansaloni, K. Tanimura, L. Pozzi, and N. Dutt, "Integrated kernel partitioning and scheduling for coarse-grained reconfigurable arrays," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 31, no. 12, pp. 1803–1816, Dec 2012.

[41] M. Seok, S. Hanson, Y.-S. Lin, Z. Foo, D. Kim, Y. Lee, N. Liu, D. Sylvester, and D. Blaauw, "The phoenix processor: A 30pw platform for sensor applications," in *Proc. of the 2008 IEEE Symposium on VLSI Circuits*. IEEE, June 2008, pp. 188–189.

[42] M. Ashouei, J. Hulzink, M. Konijnenburg, J. Zhou, F. Duarte, A. Breeschoten, J. Huisken, J. Stuyt, H. de Groot, F. Barat, J. David, and J. V. Ginderdeuren, "A voltage-scalable biomedical signal processor running ECG using 13pJ/cycle at 1MHz and 0.4V," in *Proc. of the 2011 IEEE International Solid-State Circuits Conference (ISSCC)*, February 2011, pp. 332–334.

[43] S. R. Sridhara, M. DiRenzo, S. Lingam, S.-J. Lee, R. Blazquez, J. Maxey, S. Ghanem, Y.-H. Lee, R. Abdallah, P. Singh *et al.*, "Microwatt embedded processor platform for medical system-on-chip applications," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 46, no. 4, pp. 721–730, April 2011.

[44] A. Y. Dogan, R. Braojos, J. H.-F. Constantin, G. Ansaloni, A. Burg, and D. Atienza, "Synchronizing code execution on ultra-low-power embedded multi-channel signal analysis platforms," in *Proc. of the 2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 396–399.

[45] P. Magarshack, P. Flatresse, and G. Cesana, "Utbb fd-soi: A process/design symbiosis for breakthrough energy-efficiency," in *Proc. of the 2013 Design, Automation Test in Europe Conference Exhibition (DATE)*. San Jose, CA, USA: EDA Consortium, 2013, pp. 952–957.

[46] J.-P. Colinge *et al.*, *FinFETs and other multi-gate transistors*. Springer, 2008, vol. 73.

[47] A. Singh, M. Khosla, and B. Raj, "Comparative Analysis of Carbon Nanotube Field Effect Transistor and Nanowire Transistor for Low Power Circuit Design," *Journal of Nanoelectronics and Optoelectronics*, vol. 11, no. 3, pp. 388–393, 2016.

[48] E. Sicard, "Introducing 7-nm FinFET technology in microwind," https://hal.archives-ouvertes.fr/hal-01558775/document, 2017, [Last accessed on May 1st 2018].

[49] N. Boichat, N. Boichat, D. Atienza, and N. Khaled, "Wavelet-based ECG delineation on a wearable embedded sensor platform," in *Proc. of the 6th International Workshop on Wearable and Implantable Body Sensor Networks*, June 2009, pp. 256–261.

[50] R. Braojos, D. Bortolotti, A. Bartolini, G. Ansaloni, L. Benini, and D. Atienza, "A Synchronization-Based Hybrid-Memory Multi-Core Architecture for Energy-Efficient

Biomedical Signal Processing," *IEEE Transactions on Computers (TC)*, vol. 66, no. 4, pp. 575–585, April 2017.

[51] L. Duch, S. Basu, R. Braojos, G. Ansaloni, L. Pozzi, and D. Atienza , "HEAL-WEAR: An ultra-low power heterogeneous system for bio-signal analysis," *IEEE Transactions on Circuits and Systems I (TCAS-I)*, vol. 64, no. 9, pp. 2448–2461, September 2017.

[52] L. Duch, P. Garcia Del Valle, D. Atienza, S. Ganapathy, and A. Burg, "Energy vs. reliability trade-offs exploration in biomedical ultra-low power devices," in *Proc. of the 2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 838–841.

[53] Y. Zhang, M. Khayatzadeh, K. Yang, M. Saligane, N. Pinckney, M. Alioto, D. Blaauw, and D. Sylvester, "irazor: Current-based error detection and correction scheme for pvt variation in 40-nm arm cortex-r4 processor," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 2, pp. 619–631, Feb 2018.

[54] J. Zielger and H. Puchner, "SER-history, trends and challenges," *Cypress Semiconductor Corporation*, 2004.

[55] V. Chandra and R. Aitken, "Impact of technology and voltage scaling on the soft error susceptibility in nanoscale CMOS," in *Defect and Fault Tolerance of VLSI Systems, 2008. DFTVS '08. IEEE International Symposium on*, Oct 2008, pp. 114–122.

[56] S. Basu, L. Duch, R. Braojos, G. Ansaloni, L. Pozzi, and D. Atienza, "An inexact ultra-low power bio-signal processing architecture with lightweight error recovery," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 5s, pp. 159:1–159:19, September 2017.

[57] R. G. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, 2010.

[58] F. Catthoor, M. Sabry, Z. Ma, and D. Atienza, "Method and system for real-time error mitigation," September 2014, uS Patent 8,826,072. [Online]. Available: https://www.google.com/patents/US8826072

[59] M. Sabry, D. Atienza, and F. Catthoor, "A hybrid HW-SW approach for intermittent error mitigation in streaming-based embedded systems," in *Proc. of the 2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 1110–1113.

[60] H. Mamaghanian, G. Ansaloni, M. M. S. Aly, D. Atienza Alonso, and P. Vandergheynst, "Hardware-software inexactness in noise-aware design of low-power body sensor nodes," in *Designing with Uncertainty-Opportunities & Challenges*, no. EPFL-CONF-198291, 2014.

[61] Y. Sun, K. L. Chan, and S. M. Krishnan, "ECG signal conditioning by morphological filtering," *Computers in biology and medicine*, vol. 32, no. 6, pp. 465–479, November 2002.

[62] R. Braojos, I. Beretta, G. Ansaloni, and D. Atienza, "Early Classification of Pathological Heartbeats on Wireless Body Sensor Nodes," *Sensors*, vol. 14, no. 12, pp. 22 532–22 551, December 2014.

[63] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Proc. of the 2013 18th IEEE European Test Symposium (ETS)*, May 2013, pp. 1–6.

[64] D. Nikolopoulos, H. Vandierendonck, N. Bellas, C. Antonopoulos, S. Lalis, G. Karakonstantis, A. Burg, and U. Naumann, "Energy efficiency through significance-based computing," *Computer*, vol. 47, no. 7, pp. 82–85, July 2014.

[65] S. Basu, L. Duch, M. Peón-Quirós, G. Ansaloni, L. Pozzi, and D. Atienza, "Heterogeneous and inexact: Maximizing power efficiency of edge computing sensors for health monitoring applications," in *Proc. of the International Symposium on Circuits and Systems (ISCAS)*, May 2018.

[66] E. Maricau and G. Gielen, "CMOS reliability overview," in *Analog IC Reliability in Nanometer CMOS*. Springer, 2013, pp. 15–35.

[67] T. K. Wong, "Time dependent dielectric breakdown in copper low-k interconnects: Mechanisms and reliability models," *Materials*, vol. 5, no. 9, pp. 1602–1625, 2012.

[68] A. Zaka, Q. Rafhay, P. Palestri, R. Clerc, D. Rideau, L. Selmi, C. Tavernier, and H. Jaouen, "On the accuracy of current tcad hot carrier injection models for the simulation of degradation phenomena in nanoscale devices," in *2009 International Semiconductor Device Research Symposium*, Dec 2009, pp. 1–2.

[69] H. Kukner, P. Weckx, P. Raghavan, B. Kaczer, F. Catthoor, L. R. van der Perre, and G. Groeseneken, "BTI reliability from Planar to FinFET nodes," in *Proc. of the 3rd Workshop on Manufacturable and Dependable Multicore Architectures at Nanoscale (MEDIAN'14)*, 2014, pp. 11–14.

[70] L. Chen and T. Mitra, "Shared reconfigurable fabric for multi-core customization," in *Proc. of the 48th ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2011, pp. 830–835.

[71] J. Constantin, L. Wang, G. Karakonstantis, A. Chattopadhyay, and A. Burg, "Exploiting dynamic timing margins in microprocessors for frequency-over-scaling with instruction-based clock adjustment," in *Proc. of the 2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015, pp. 381–386.

[72] L. Gavrilov and P. Heuveline, "Aging of population," *The encyclopedia of population*, vol. 1, pp. 32–37, 2003.

[73] D. Farhud, "Impact of lifestyle on health," *Iranian journal of public health*, vol. 44, no. 11, p. 1442, 2015.

[74] B. Rajoub, "An efficient coding algorithm for the compression of ECG signals using the wavelet transform," *Biomedical Engineering, IEEE Transactions on*, vol. 49, no. 4, pp. 355–362, April 2002.

[75] S. Farshchi, A. Pesterev, P. Nuyujukian, I. Mody, and J. Judy, "Bi-fi: An embedded sensor/system architecture for remote biological monitoring," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 11, no. 6, pp. 611–618, Nov 2007.

[76] J. Martinez, R. Almeida, S. Olmos, A. Rocha, and P. Laguna, "A wavelet-based ECG delineator: evaluation on standard databases," *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 4, pp. 570–581, april 2004.

[77] Y. Sun, K. Luk Chan, and S. Muthu Krishnan, "Characteristic wave detection in ECG signal using morphological transform," *BMC Cardiovascular Disorders*, vol. 5, pp. 1–7, Sep 2005.

[78] R. Ceylan and Y. Ozbay, "Comparison of FCM, PCA and WT techniques for classification ECG arrhythmias using artificial neural network," *Expert Systems with Applications*, vol. 33, no. 2, pp. 286–295, 2007.

[79] S.-N. Yu and K.-T. Chou, "Integration of independent component analysis and neural networks for {ECG} beat classification," *Expert Systems with Applications*, vol. 34, no. 4, pp. 2841–2846, May 2008.

[80] A. Y. Dogan, D. Atienza, A. Burg, I. Loi, and L. Benini, "Power/performance exploration of single-core and multi-core processor approaches for biomedical signal processing," in *Proc. of the 21st International Conference on Integrated Circuit and System Design: Power and Timing Modeling, Optimization, and Simulation*. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 102–111.

[81] D. Rossi, F. Conti, A. Marongiu, A. Pullini, I. Loi, M. Gautschi, G. Tagliavini, A. Capotondi, P. Flatresse, and L. Benini, "Pulp: A parallel ultra low power platform for next generation iot applications," in *2015 IEEE Hot Chips 27 Symposium (HCS)*, Aug 2015, pp. 1–39.

[82] V. F. Annese, M. Crepaldi, D. Demarchi, and D. D. Venuto, "A digital processor architecture for combined EEG/EMG falling risk prediction," in *Proc. of the 2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 714–719.

[83] O. Bai, G. Kelly, D. Y. Fei, D. Murphy, J. Fox, B. Burkhardt, W. Lovegreen, and J. Soars, "A wireless, smart EEG system for volitional control of lower-limb prosthesis," in *TENCON 2015 - 2015 IEEE Region 10 Conference*, November 2015, pp. 1–6.

[84] N. Sarkany, A. Tihanyi, and P. Szolgay, "The design of a mobile multi-channel bio-signal measuring system for rehabilitation purposes," in *Proc. of the 2014 14th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, July 2014, pp. 1–2.

[85] J. Constantin, A. Y. Dogan, O. Andersson, P. Meinerzhagen, J. N. Rodrigues, D. Atienza, and A. Burg, "Tamarisc-cs: An ultra-low power application-specific processor for compressed sensing," in *Proc. of the 2012 IEEE/IFIP 20th International Conference on VLSI and System-on-Chip (VLSI-SoC)*. IEEE, October 2012, pp. 159–164.

[86] Y. Park, J. J. K. Park, and S. Mahlke, "Efficient performance scaling of future CGRAs for mobile applications," in *Field-Programmable Technology (FPT), 2012 International Conference on*, December 2012, pp. 335–342.

[87] F. Conti, A. Marongiu, and L. Benini, "Synthesis-friendly techniques for tightly-coupled integration of hardware accelerators into shared-memory multi-core clusters," in *Proc. of the 2013 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. IEEE, September 2013, pp. 1–10.

[88] N. Ozaki, Y. Yasuda, M. Izawa, Y. Saito, D. Ikebuchi, H. Amano, H. Nakamura, K. Usami, M. Namiki, and M. Kondo, "Cool Mega-Arrays: Ultralow-Power Reconfigurable Accelerator Chips," *IEEE Micro*, vol. 31, no. 6, pp. 6–18, November 2011.

[89] F. Bouwens, M. Berekovic, A. Kanstein, and G. Gaydadjiev, "Architectural exploration of the ADRES coarse-grained reconfigurable array," in *Reconfigurable Computing: Architectures, Tools and Applications*, ser. LNCS. Berlin, Germany: Springer, June 2007, vol. 4419, pp. 1–13.

[90] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Communications magazine*, vol. 48, no. 9, 2010.

[91] M. P. Singh and M. K. Jain, "Evolution of processor architecture in mobile phones," *International Journal of Computer Applications*, vol. 90, no. 4, 2014.

[92] K. T. Cheng and Y. C. Wang, "Using mobile gpu for general-purpose computing - a case study of face recognition on smartphones," in *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test*, April 2011, pp. 1–4.

[93] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 26, no. 2, pp. 203–215, February 2007.

[94] P. Garcia and K. Compton, "Kernel sharing on reconfigurable multiprocessor systems," in *Proc. of the International Conference on Electrical and Computer Engineering (ICECE)*. IEEE, December 2008, pp. 225–232.

[95] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing fpga-based accelerator design for deep convolutional neural networks," in *Proc. of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '15. ACM, 2015, pp. 161–170.

[96] M.-H. Lee, H. Singh, G. Lu, N. Bagherzadeh, F. J. Kurdahi, E. M. C. Filho, and V. C. Alves, "Design and implementation of the morphosys reconfigurable computing processor," *Journal of VLSI Signal Processing-Systems for Signal Image and Video Technology*, vol. 24, no. 2-3, pp. 147–164, March 2000.

[97] B. Mei, S. Vernalde, D. Verkest, H. D. Man, and R. Lauwereins, "Exploiting loop-level parallelism on coarse-grained reconfigurable architectures using modulo scheduling," *IEE Proceedings - Computers and Digital Techniques*, vol. 150, no. 5, pp. 255–61–, September 2003.

[98] P. Liu and A. Hemani, "A coarse grain reconfigurable architecture for sequence alignment problems in bio-informatics," in *Proc. of the 2010 IEEE 8th Symposium on Application Specific Processors (SASP)*, June 2010, pp. 50–57.

[99] A. Traber, F. Zaruba, S. Stucki, A. Pullini, G. Haugou, E. Flamand, F. Gürkaynak, and L. Benini, "Pulpino: A small single-core risc-v soc," in *3rd RISCV Workshop*, 2016.

[100] J. H.-F. Constantin, "Processor development in LISA for biomedical applications," Master's thesis, Eidgenössische Technische Hochschule Zürich (ETHZ), Switzerland, January 2011.

[101] A. Y. Dogan, J. Constantin, M. Ruggiero, A. Burg, and D. Atienza, "Multi-core architecture design for ultra-low-power wearable health monitoring systems," in *Proc. of the 2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 988–993.

[102] D. A. Patterson, "Reduced instruction set computers," *Commun. ACM*, vol. 28, no. 1, pp. 8–21, Jan. 1985.

[103] I. Al Khatib, D. Bertozzi, F. Poletti, L. Benini, A. Jantsch, M. Bechara, H. Khalifeh, M. Hajjar, R. Nabiev, and S. Jonsson, "MPSoC ECG biochip: A multiprocessor system-on-chip for real-time human heart monitoring and analysis," in *Proc. of the 3rd Conference on Computing Frontiers*. New York, NY, USA: ACM, May 2006, pp. 21–28.

[104] D. Kirk *et al.*, "Nvidia cuda software and gpu parallel computing architecture," in *ISMM*, vol. 7, 2007, pp. 103–104.

[105] B. D. Sutter, P. Raghavan, and A. Lambrechts, *Handbook of Signal Processing Systems*. Springer US, July 2010, ch. Coarse-Grained Reconfigurable Array Architectures.

[106] G. Ansaloni, P. Bonzini, and L. Pozzi, "EGRA: A coarse-grained reconfigurable architectural template," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 6, pp. 1062–1074, June 2011.

[107] B. R. Rau, "Iterative module scheduling: an algorithm for software pipelining loops," in *Proc. of the 27th Annual International Symposium on Microarchitecture (MICRO-27)*, November 1994, pp. 63–74.

[108] T. Peyret, G. Corre, M. Thevenin, K. Martin, and P. Coussy, "An automated design approach to map applications on CGRAs," in *Proc. of the 24th Edition of the Great Lakes Symposium on VLSI*, ser. GLSVLSI'14, May 2014, pp. 229–230.

[109] G. Zacharopoulos and L. Pozzi, "ClrFreqCFGPrinter: A tool for frequency annotated control flow graph generation," European LLVM Developers Meeting, Tech. Rep., March 2017.

[110] C. Lattner and V. Adve, "LLVM: A compilation framework for lifelong program analysis & transformation," in *Proc. of the 2nd International Symposium on Code Generation and Optimization (CGO)*, Palo Alto, California, March 2004, pp. 75–86.

[111] G. Ansaloni, L. Pozzi, K. Tanimura, and N. Dutt, "Slack-aware scheduling on coarse-grained reconfigurable arrays," in *Proc. of the 2011 Design, Automation Test in Europe Conference Exhibition (DATE)*. IEEE, March 2011, pp. 1–4.

[112] P. Theocharis and B. D. Sutter, "A bimodal scheduler for coarse-grained reconfigurable arrays," *ACM Transactions on Architecture and Code Optimization*, vol. 13, no. 2, pp. 15:1–15:26, June 2016.

[113] H. Park, K. Fan, S. A. Mahlke, T. Oh, H. Kim, and H.-s. Kim, "Edge-centric modulo scheduling for coarse-grained reconfigurable architectures," in *Proc. of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT)*. New York, NY, USA: ACM, 2008, pp. 166–176.

[114] J. Cong, Y. Fan, G. Han, and Z. Zhang, "Application-specific instruction generation for configurable processor architectures," in *Proc. of the 2004 ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays (FPGA)*. New York, NY, USA: ACM, February 2004, pp. 183–189.

[115] L. Pozzi, K. Atasu, and P. Ienne, "Exact and approximate algorithms for the extension of embedded processor instruction sets," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 7, pp. 1209–29, July 2006.

[116] Synopsys, "ASIP Designer," www.synopsys.com/dw/ipdir.php?ds=asip-designer, [Last accessed on July 7th 2018].

[117] United Microelectronics Corporation (UMC), "90 nm semiconductor technology," http://www.umc.com/English/pdf/90nm_DM.pdf, 2018, [Last accessed on November 2nd 2018].

[118] Global Foundries, "130 nm semiconductor technology," https://www.globalfoundries.com/technology-solutions/cmos/mainstream/130-180nm, 2018, [Last accessed on November 2nd 2018].

[119] United Microelectronics Corporation (UMC), "65 nm semiconductor technology," http://www.umc.com/English/pdf/UMC%2065nm.pdf, 2018, [Last accessed on November 2nd 2018].

[120] Synopsys, "Design Compiler," https://www.synopsys.com/support/training/rtl-synthesis/design-compiler-rtl-synthesis.html, [Last accessed on May 7th 2018].

[121] Mentor Graphics, "ModelSim," https://www.mentor.com/products/fv/modelsim/, [Last accessed on July 7th 2018].

[122] F. Rincon, J. Recas, N. Khaled, and D. Atienza, "Development and evaluation of multilead wavelet-based ECG delineation algorithms for embedded wireless sensor nodes," *IEEE Transactions on Information Technology in Biomedicine (T-ITB)*, vol. 15, no. 6, pp. 854–863, November 2011.

[123] R. Braojos, G. Ansaloni, and D. Atienza, "A methodology for embedded classification of heartbeats using random projections," in *Proc. of the 2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 899–904.

[124] L. Benini, A. Macii, E. Macii, and M. Poncino, "Increasing energy efficiency of embedded systems by application-specific memory hierarchy generation," *IEEE Design & Test of Computers*, vol. 17, no. 2, pp. 74–85, 2000.

[125] B. Egger, H. Lee, D. Kang, M. Moghaddam, Y. Cho, Y. Lee, S. Kim, S. Ha, and K. Choi, "A space- and energy-efficient code compression/decompression technique for coarse-grained reconfigurable architectures," in *Proc. of the 2017 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, February 2017, pp. 197–209.

[126] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C Applications and Reviews*, vol. 40, no. 1, pp. 1–12, October 2010.

[127] K. J. Nowka, G. D. Carpenter, E. W. MacDonald, H. C. Ngo, B. C. Brock, K. I. Ishii, T. Y. Nguyen, and J. L. Burns, "A 32-bit powerpc system-on-a-chip with support for dynamic voltage scaling and dynamic frequency scaling," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 11, pp. 1441–1447, 2002.

[128] A. Kumar, "Leakage current controlling mechanism using high-K dielectric + metal gate," *International Journal of Information Technology and Knowledge Management*, pp. 191–194, 2012.

[129] S. Hanson, B. Zhai, M. Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin *et al.*, "Exploring variability and performance in a sub-200-mv processor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 4, pp. 881–891, 2008.

[130] L. Wang, Y. Ma, J. Yan, V. Chang, and A. Y. Zomaya, "pipscloud: High performance cloud computing for remote sensing big data management and processing," *Future Generation Computer Systems*, vol. 78, pp. 353–368, 2018.

[131] A. H. Shoeb and J. V. Guttag, "Application of machine learning to epileptic seizure detection," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 975–982.

[132] I. Kononenko, "Machine learning for medical diagnosis: history, state of the art and perspective," *Artificial Intelligence in medicine*, vol. 23, no. 1, pp. 89–109, 2001.

[133] M. Ottavi, D. Gizopoulos, and S. Pontarelli, *Dependable Multicore Architectures at Nanoscale*. Springer, 2018.

[134] X. Yang and K. Mohanram, "Unequal-error-protection codes in srams for mobile multimedia applications," in *Proc. of the 2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2011, pp. 21–27.

[135] S. Ganapathy, G. Karakonstantis, R. Canal, and A. Burg, "Variability-aware design space exploration of embedded memories," in *Proc. of the 2014 IEEE 28th Convention of Electrical Electronics Engineers in Israel (IEEEI)*, December 2014, pp. 1–5.

[136] S. Ganapathy, R. Canal, D. Alexandrescu, E. Costenaro, A. Gonzalez, and A. Rubio, "Informer: An integrated framework for early-stage memory robustness analysis," in

Proc. of the 2014 Design, Automation and Test in Europe Conference Exhibition (DATE), March 2014, pp. 1–4.

[137] N. S. Kim, S. Draper, S.-T. Zhou, S. Katariya, H. Ghasemi, and T. Park, "Analyzing the impact of joint optimization of cell size, redundancy, and ecc on low-voltage sram array total area," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 12, pp. 2333–2337, Dec 2012.

[138] D. Schroder, "Negative bias temperature instability: What do we understand?" *Microelectronics Reliability*, vol. 47, p. 841–852, June 2007.

[139] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, p. 62, May 2016.

[140] M. M. Sabry, D. Atienza, and F. Catthoor, "Ocean: An optimized hw/sw reliability mitigation approach for scratchpad memories in real-time socs," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 13, no. 4s, p. 138, 2014.

[141] R. Reis, Y. Cao, and G. Wirth, *Circuit design for reliability*. Springer, 2015.

[142] R. Entner, "Modeling and Simulation of Negative Bias Temperature Instability," Ph.D. dissertation, Technische Universität Wien, Institut für Mikroelektronik, Apr. 2007. [Online]. Available: http://www.iue.tuwien.ac.at/phd/entner

[143] H. Kükner, P. Weckx, J. Franco, M. Toledano-Luque, M. Cho, B. Kaczer, P. Raghavan, D. Jang, K. Miyaguchi, M. G. Bardon, F. Catthoor, L. V. der Perre, R. Lauwereins, and G. Groeseneken, "Scaling of BTI reliability in presence of time-zero variability," in *2014 IEEE International Reliability Physics Symposium*, June 2014, pp. CA.5.1–CA.5.7.

[144] I. Agbo, M. Taouil, D. Kraak, S. Hamdioui, H. Kükner, P. Weckx, P. Raghavan, and F. Catthoor, "Integral impact of bti, pvt variation, and workload on sram sense amplifier," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1444–1454, Apr. 2017.

[145] A. Subirats, X. Garros, J. Cluzel, J. El Husseini, F. Cacho, X. Federspiel, V. Huard, M. Rafik, G. Reimbold, O. Faynot, and G. Ghibaudo, "A New Gate Pattern Measurement for Evaluating the BTI Degradation in Circuit Conditions," in *IEEE International Reliability Physics Symposium*. IEEE, Jun. 2014, pp. 5D.1.1–5D.1.5.

[146] S. Ramey, C. Prasad, M. Agostinelli, S. Pae, S. Walstra, S. Gupta, and J. Hicks, "Frequency and Recovery Effects in High-k BTI Degradation," in *IEEE International Reliability Physics Symposium*. IEEE, Apr. 2009, pp. 1023–1027.

[147] H. Kukner, P. Weckx, P. Raghavan, B. Kaczer, F. Catthoor, L. V. D. Perre, R. Lauwereins, and G. Groeseneken, "Impact of duty factor, stress stimuli, and gate drive strength on gate delay degradation with an atomistic trap-based bti model," in *2012 15th Euromicro Conference on Digital System Design*, Sept 2012, pp. 1–7.

[148] T. Grasser, *Bias temperature instability for devices and circuits*. Springer Science & Business Media, 2013.

[149] D. Rodopoulos, S. B. Mahato, V. V. de Almeida Camargo, B. Kaczer, F. Catthoor, S. Cosemans, G. Groeseneken, A. Papanikolaou, and D. Soudris, "Time and workload dependent device variability in circuit simulations," in *IC Design & Technology (ICICDT), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–4.

[150] D. Rodopoulos, P. Weckx, M. Noltsis, F. Catthoor, and D. Soudris, "Atomistic Pseudo-Transient BTI Simulation With Inherent Workload Memory," *IEEE Transactions on Device and Materials Reliability*, vol. 14, no. 2, pp. 704–714, Jun. 2014.

[151] B. Kaczer, S. Mahato, V. V. de Almeida Camargo, M. Toledano-Luque, P. J. Roussel, T. Grasser, F. Catthoor, P. Dobrovolny, P. Zuber, G. Wirth, and G. Groeseneken, "Atomistic approach to variability of bias-temperature instability in circuit simulations," in *2011 International Reliability Physics Symposium*, April 2011, pp. XT.3.1–XT.3.5.

[152] B. Kaczer, J. Franco, P. Weckx, P. J. Roussel, E. Bury, M. Cho, R. Degraeve, D. Linten, G. Groeseneken, H. Kukner, P. Raghavan, F. Catthoor, G. Rzepa, W. Goes, and T. Grasser, "The Defect-Centric Perspective of Device and Circuit Reliability—From Individual Defects to Circuits," in *45th European Solid State Device Research Conference (ESSDERC)*, Sep. 2015, pp. 218–225.

[153] E. Maricau and G. Gielen, "Transistor aging-induced degradation of analog circuits: Impact analysis and design guidelines," in *2011 Proceedings of the ESSCIRC (ESSCIRC)*, Sept 2011, pp. 243–246.

[154] H. Kükner, S. Khan, P. Weckx, P. Raghavan, S. Hamdioui, B. Kaczer, F. Catthoor, L. Van der Perre, R. Lauwereins, and G. Groeseneken, "Comparison of reaction-diffusion and atomistic trap-based bti models for logic gates," *IEEE transactions on device and materials reliability*, vol. 14, no. 1, pp. 182–193, 2014.

[155] H. Amrouch, B. Khaleghi, A. Gerstlauer, and J. Henkel, "Reliability-aware design to suppress aging," in *Proceedings of the 53rd Annual Design Automation Conference*. ACM, 2016, p. 12.

[156] D. Stamoulis, S. Corbetta, D. Rodopoulos, P. Weckx, P. Debacker, B. Meyer, B. Kaczer, P. Raghavan, D. Soudris, F. Catthoor, and Z. Zilic, "Capturing True Workload Dependency of BTI-Induced Degradation in CPU Components," in *Proceedings of the 26th Edition on Great Lakes Symposium on VLSI (GLSVLSI)*. New York, NY, USA: ACM, May 2016, pp. 373–376.

[157] T. B. Chan, W. T. J. Chan, and A. B. Kahng, "Impact of adaptive voltage scaling on aging-aware signoff," in *Proc. of the 2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 1683–1688.

[158] D. Rodopoulos, D. Stamoulis, G. Lyras, D. Soudris, and F. Catthoor, "Understanding Timing Impact of BTI/RTN with Massively Threaded Atomistic Transient Simulations," in *IEEE International Conference on IC Design Technology (ICICDT)*, May 2014, pp. 1–4.

[159] A. Sivadasan, A. Notin, V. Huard, E. Maurin, S. Mhira, F. Cacho, and L. Anghel, "Workload Dependent Reliability Timing Analysis Flow," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, Mar. 2017, pp. 736–737.

[160] P. Weckx, B. Kaczer, H. Kukner, J. Roussel, P. Raghavan, F. Catthoor, and G. Groeseneken, "Non-monte-carlo methodology for high-sigma simulations of circuits under workload-dependent bti degradation - application to 6t sram," in *2014 IEEE International Reliability Physics Symposium*, June 2014, pp. 5D.2.1–5D.2.6.

[161] C.-C. Chen, S. Cha, T. Liu, and L. Milor, "System-Level Modeling of Microprocessor Reliability Degradation Due to BTI and HCI," in *IEEE International Reliability Physics Symposium*, Jun. 2014, pp. CA.8.1–CA.8.9.

[162] Cadence, "Incisive," https://www.cadence.com/content/cadence-www/global/en_US/home/tools/system-design-and-verification/simulation-and-testbench-verification/incisive-enterprise-simulator.html, [Last accessed on October 26th 2018].

[163] MathWorks, "MATLAB," https://www.mathworks.com/products/matlab.html, [Last accessed on October 26th 2018].

[164] Synopsys , "CustomSim," https://www.synopsys.com/content/dam/synopsys/verification/datasheets/customsim-ds.pdf, [Last accessed on July 7th 2018].

[165] Taiwan Semiconductor Manufacturing Company (TSMC), "7 nm semiconductor technology," http://www.tsmc.com/english/dedicatedFoundry/technology/7nm.htm, 2018, [Last accessed on October 27th 2018].

[166] Synopsys , "NanoTime STA," https://www.synopsys.com/content/dam/synopsys/implementation&signoff/datasheets/nanotime-ds.pdf, [Last accessed on July 7th 2018].

[167] Synopsys , "HSPICE," https://www.synopsys.com/content/dam/synopsys/verification/datasheets/hspice-ds.pdf, [Last accessed on July 7th 2018].

[168] J. Bhasker and R. Chadha, *Static Timing Analysis for Nanometer Designs*. Springer US, 2009.

[169] C. R. Lefurgy, A. J. Drake, M. S. Floyd, M. S. Allen-Ware, B. Brock, J. A. Tierno, J. B. Carter, and R. W. Berry, "Active Guardband Management in Power7+ to Save Energy and Maintain Reliability," *IEEE Micro*, vol. 33, no. 4, pp. 35–45, Jul. 2013.

[170] S. S. Lobodzinski, "ECG patch monitors for assessment of cardiac rhythm abnormalities," *Progress in cardiovascular diseases*, vol. 56, no. 2, pp. 224–229, September 2013.

[171] G. Surrel, F. J. Rincón, S. Murali, and D. Atienza, "Low-power wearable system for real-time screening of obstructive sleep apnea," in *ISVLSI*, 2016, pp. 230–235.

[172] Q. U. S. of Medicine, "Normal ECG ranges," https://meds.queensu.ca/central/assets/modules/ECG/normal_ecg.html, [Last accessed on May 1st 2018].

[173] P. S. Grassi, "Elettrocardiogramma (ECG) - Materiale Didattico Fisiologia," http://www.med.unipg.it/ccl/Materiale%20Didattico/Fisiologia%20(Grassi)/ECG.pdf, [Last accessed on Aug 17th 2018].

[174] Y. Zigel, A. Cohen, and A. Katz, "The weighted diagnostic distortion (wdd) measure for ECG signal compression," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 11, pp. 1422–1430, November 2000.

[175] S. Ramey, C. Prasad, and A. Rahman, "Technology scaling implications for bti reliability," *Microelectronics Reliability*, vol. 82, pp. 42–50, 2018.

[176] F. Forooghifar, A. Aminifar, and D. Atienza, "Self-aware wearable systems in epileptic seizure detection," in *Euromicro Conference on Digital System Design (DSD)*. IEEE, 2018.

[177] G. Surrel, A. Aminifar, F. Rincón, S. Murali, and D. Atienza, "Online obstructive sleep apnea detection on medical wearable sensors," *IEEE Transactions on Biomedical Circuits and Systems*, 2018.

[178] G. A. Klutke, P. C. Kiessler, and M. A. Wortman, "A critical look at the bathtub curve," *IEEE Transactions on Reliability*, vol. 52, no. 1, pp. 125–129, March 2003.

[179] S. Khan, I. Agbo, S. Hamdioui, H. Kukner, B. Kaczer, P. Raghavan, and F. Catthoor, "Bias temperature instability analysis of FinFET based SRAM cells," in *Proceedings of the conference on Design, Automation & Test in Europe*. European Design and Automation Association, 2014, p. 31.

[180] I. Agbo, M. Taouil, S. Hamdioui, P. Weckx, S. Cosemans, P. Raghavan, F. Catthoor, and W. Dehaene, "Quantification of sense amplifier offset voltage degradation due to zero- and run-time variability," in *VLSI (ISVLSI), 2016 IEEE Computer Society Annual Symposium on*. IEEE, 2016, pp. 725–730.

[181] H. K. Alidash, A. Calimera, A. Macii, E. Macii, and M. Poncino, "On-chip nbti and pbti tracking through an all-digital aging monitor architecture," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization and Simulation*, J. L. Ayala, D. Shang, and A. Yakovlev, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 155–165.

[182] H. Kukner, M. Khatib, S. Morrison, P. Weckx, P. Raghavan, B. Kaczer, F. Catthoor, L. Van der Perre, R. Lauwereins, and G. Groeseneken, "Degradation analysis of datapath logic sub-blocks under NBTI aging in FinFET technology," in *Quality Electronic Design (ISQED), 2014 15th International Symposium on*. IEEE, 2014, pp. 473–479.

[183] M. Choudhury, V. Chandra, K. Mohanram, and R. Aitken, "Timber: Time borrowing and error relaying for online timing error resilience," in *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, March 2010, pp. 1554–1559.

[184] Z. H. Li, T. T. Zhu, Z. J. Chen, J. Y. Meng, X. Y. Xiang, and X. L. Yan, "Eliminating timing errors through collaborative design to maximize the throughput," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 2, pp. 670–682, Feb 2017.

[185] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. S. Kim, and K. Flautner, "Razor: circuit-level correction of timing errors for low-power operation," *IEEE Micro*, vol. 24, no. 6, pp. 10–20, Nov 2004.

[186] S. Das, C. Tokunaga, S. Pant, W. H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw, "Razorii: In situ error detection and correction for pvt and ser tolerance," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 32–48, Jan 2009.

[187] M. Fojtik, D. Fick, Y. Kim, N. Pinckney, D. M. Harris, D. Blaauw, and D. Sylvester, "Bubble razor: Eliminating timing margins in an arm cortex-m3 processor in 45 nm CMOS using architecturally independent error detection and correction," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 66–81, Jan 2013.

[188] I. Kwon, S. Kim, D. Fick, M. Kim, Y. P. Chen, and D. Sylvester, "Razor-lite: A light-weight register for error detection by observing virtual supply rails," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 9, pp. 2054–2066, Sept 2014.

[189] S. Mitra and E. J. McCluskey, "Which concurrent error detection scheme to choose ?" in *Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159)*, 2000, pp. 985–994.

[190] K. Wu and R. Karri, "Algorithm level re-computing with shifted operands-a register transfer level concurrent error detection technique," in *Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159)*, 2000, pp. 971–978.

[191] X. Guo and R. Karri, "Recomputing with permuted operands: A concurrent error detection approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 10, pp. 1595–1608, Oct 2013.

[192] R. E. Lyons and W. Vanderkulk, "The use of triple-modular redundancy to improve computer reliability," *IBM Journal of Research and Development*, vol. 6, no. 2, pp. 200–209, 1962.

[193] 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, Feb 2003.

[194] M. Marshall and G. Russell, "A low power information redundant concurrent error detecting asynchronous processor," in *null*. IEEE, 2007, pp. 649–656.

[195] G. Psychou, D. Rodopoulos, M. M. Sabry, T. Gemmeke, D. Atienza, T. G. Noll, and F. Catthoor, "Classification of Resilience Techniques Against Functional Errors at Higher Abstraction Layers of Digital Systems," *ACM Computing Surveys (CSUR)*, vol. 50, no. 4, p. 50, 2017.

[196] G. Russell and A. Maamar, "Check bit prediction scheme using dong's code for concurrent error detection in vlsi processors," *IEE Proceedings-Computers and Digital Techniques*, vol. 147, no. 6, pp. 467–471, 2000.

[197] S. Shah, A. J. Al-Khalili, and D. Al-Khalili, "Comparison of 32-bit multipliers for various performance measures," in *ICM 2000. Proceedings of the 12th International Conference on Microelectronics. (IEEE Cat. No.00EX453)*, 2000, pp. 75–80.

[198] C. Lin, Y.-H. Cho, and Y.-M. Yang, "Aging-aware reliable multiplier design with adaptive hold logic," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 3, pp. 544–556, 2015.

[199] S. Corbetta *et al.*, "System-wide reliability analysis on real processor and application under VDD and t stress," *Silcon Errors and Logic System Effects-SELSE 2016*, 2016.

[200] M. Nicolaidis, "Circuit-level soft-error mitigation," in *Soft errors in modern electronic systems*. Springer, 2011, pp. 203–252.

[201] V. G. Oklobdzija, *Digital design and fabrication*. CRC press, 2007.

[202] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, "The impact of technology scaling on lifetime reliability," in *Proc. of the 2004 International Conference on Dependable Systems and Networks (DSN)*, June 2004, pp. 177–186.

[203] C. Boatella-Polo, "SEE Single Event Effects - CERN," https://indico.cern.ch/event/635099/contributions/2570672/attachments/1456364/2249943/Single_Event_Effecs_Radiation_Course_May_2017_SEE_CBP.pdf, May 2017, [Last accessed on May 28th 2018].

[204] C. Dufour, P. Garnier, T. Carriere, J. Beaucour, R. Ecoffet, and M. Labrunee, "Heavy ion induced single hard errors on submicronic memories [for space application]," *Nuclear Science, IEEE Transactions on*, vol. 39, no. 6, pp. 1693–1697, Dec 1992.

[205] M. Alam and S. Mahapatra, "A comprehensive model of PMOS NBTI degradation," *Microelectronics Reliability*, vol. 45, no. 1, p. 71–81, Jan. 2005.

[206] J. Knaizuk Jr and C. R. Hartmann, "An optimal algorithm for testing stuck-at faults in random access memories," *IEEE Transactions on Computers*, vol. 26, no. 11, pp. 1141–1144, 1977.

[207] R. W. Hamming, "Error detecting and error correcting codes," *Bell Labs Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[208] D. Rossi, N. Timoncini, M. Spica, and C. Metra, "Error correcting code analysis for cache memory high reliability and performance," in *Proc. of the 2011 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2011, pp. 1–6.

[209] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in srams," in *Proc. of the 34th European Solid-State Circuits Conference (ESSCIRC 2008)*, Sept 2008, pp. 222–225.

[210] S. Schechter, G. H. Loh, K. Straus, and D. Burger, "Use ECP, Not ECC, for Hard Failures in Resistive Memories," *SIGARCH Comput. Archit. News*, vol. 38, no. 3, pp. 141–152, June 2010.

[211] V. Sridhar and M. R. Prasad, "Built-in self-repair (bisr) technique widely used to repair embedded random access memories (rams)," *International Journal of Computer Science Engineering (IJCSE) ISSN*, 2012.

[212] A. Ejlali, B. M. Al-Hashimi, and P. Eles, "A standby-sparing technique with low energy-overhead for fault-tolerant hard real-time systems," in *Proc. of the 2009 IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New York, NY, USA: ACM, October 2009, pp. 193–202.

[213] K.-C. Wu and D. Marculescu, "Power-aware soft error hardening via selective voltage scaling," in *Proc. of the IEEE International Conference on Computer Design (ICCD)*, October 2008, pp. 301–306.

[214] G. Karakonstantis, D. Mohapatra, and K. Roy, "Logic and memory design based on unequal error protection for voltage-scalable, robust and adaptive dsp systems," *Journal of Signal Processing Systems*, vol. 68, no. 3, pp. 415–431, September 2012.

[215] H. Dickhaus and H. Heinrich, "Classifying biosignals with wavelet networks [a method for noninvasive diagnosis]," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 15, no. 5, pp. 103–111, September 1996.

[216] O. Sentieys, D. Menard, K. Parashar, and D. Novo, "Fixed-point refinement, a guaranteed approach towards energy efficient computing," *Tutorial at the 2015 Design Automation and Test in Europe Conference Exhibition (DATE)*, 2015.

[217] H. Mamaghanian, G. Ansaloni, D. Atienza, and P. Vandergheynst, "Power-efficient joint compressed sensing of multi-lead ECG signals," in *Proc. of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 4409–4412.

[218] SmartCardia, "INYU - The Inner You," http://www.smartcardia.com/, [Last accessed on May 1st 2018].

[219] D. Bortolotti, C. Pinto, A. Marongiu, M. Ruggiero, and L. Benini, "Virtualsoc: A full-system simulation environment for massively parallel heterogeneous system-on-chip," in *Proc. of the 2013 IEEE 27th International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2013, pp. 2182–2187.

[220] S. Ganapathy, R. Canal, A. Gonzalez, and A. Rubio, "Effectiveness of hybrid recovery techniques on parametric failures," in *Proc. of the Quality Electronic Design (ISQED)*, March 2013, pp. 258–264.

[221] HP Laboratories, "Cacti 6.x," http://www.hpl.hp.com/research/cacti/, [Last accessed on May 1st 2018].

[222] L. Duch, S. Basu, R. Braojos, G. Ansaloni, L. Pozzi, and D. Atienza, "A multi-core reconfigurable architecture for ultra-low power bio-signal analysis," in *Proc. of the 2016 IEEE Biomedical Circuits and Systems (BioCAS)*, October 2016, pp. 1–4.

[223] L. Duch, S. Basu, M. Peón-Quirós, G. Ansaloni, L. Pozzi, and D. Atienza, "i-DPs CGRA: An Interleaved-Datapaths Reconfigurable Accelerator for Embedded Bio-signal Processing," *IEEE Embedded Systems Letters*, 2018.

[224] İ. Güler and E. D. Übeylı, "ECG beat classifier designed by combined neural network model," *Pattern recognition*, vol. 38, no. 2, pp. 199–208, 2005.

[225] J. Peng, Y. Dai, Y. Rao, and X. Zhi, "Model of cpu-intensive applications in cloud computing," in *Advanced Multimedia and Ubiquitous Engineering*. Springer, 2016, pp. 301–315.

Loris DUCH

- Swiss Federal Institute of Technology (EPFL)
  Embedded Systems Laboratory (ESL)
  ELG 132, Station 11, CH-1015
  Lausanne, Switzerland
- Phone: +41 (0) 21.693.11.33
- E-mail: loris.duch@epfl.ch

- Born in Evian-les-Bains, FR
- Single, no children
- Driver's license (8 years)

Profile

- An R&D computer architect with excellent hardware and software development skills, supported by a wide range of programming languages.
- Expertise in designing and optimizing algorithms for real-time embedded systems with performance, energy, reliability, timing and costs constraints.
- Strong interest in emerging technologies and consumer electronics.

Education

2014 → 2018 *Doctorat en Microsystèmes et Microélectronique*

- Ph.D. degree in Microsystems & Microelectronics
  at the Embedded Systems Laboratory (ESL), Swiss Federal Institute of Technology (EPFL), Lausanne, CH
- Thesis title: Hardware/Software Architectural and Technological Exploration for Energy-Efficient and Reliable Biomedical Devices
- Advisor: Prof. David Atienza

2011 → 2014 *Diplôme d'Ingénieur en Systèmes Electroniques Intégrés*

- Master's degree with distinction in Engineering, Physics, Electronics and Materials, specialized in digital integrated circuit design
  at the Grenoble Institute of Technology, Grenoble INP (Phelma - Minatec), FR

2012 *Bachelor en Sciences de l'Ingénieur*

- Bachelor's degree in Engineering Sciences
  at the Grenoble Institute of Technology, Grenoble INP (Phelma - Minatec), FR

Professional Experiences

2014 (3 months) *Embedded systems engineer* - **Embedded Systems Laboratory (EPFL - ESL)**, Lausanne, CH

- Research tools development in C, C++ and SystemC for the laboratory.
- Hardware design of a memory error injector and implementation of an object tracker and motion estimation software application for an OpenRISC and MPARM simulator.

2014 (6 months) *Hardware/Software development engineer* - **Mercury Systems (formerly CES)**, Geneva, CH

- Involvement into a collaborative project chosen by the European Commission (FP7), to study the AFDX standard viability in on-board spacecraft/satellite communication networks.
- VHDL design of an AFDX real-time network interface integrated in a Zynq device (Xilinx FPGA SoC with two ARM cores).

2013 (4 months) *Software development intern* - **General Electric (formerly Alstom)**, Grenoble, FR

- Improvement of the turbine testing software of the R&D laboratory, by developing a real-time frequency analysis application in Visual Basic 6.

2011 (3 months) *Web development intern* - **WindSolutions SA**, Lausanne, CH

- Optimization and acceleration of the debugging process carried out by the software development team, thanks to the implementation of a log parser in PHP with HTML interface and MySQL database.

Summer jobs 2010 → 2012 *Store employee* - **Carrefour, Novasanit, Léman Store Fermeture,** Thonon-Les-Bains, FR

- Packaging, order picking, phone calls to customers, wares storages.

Management & Teaching Experiences

Project management certification: PRINCE2 (obtained in 2017)

Project supervisor for Master students at EPFL:

- Project title: Optimization of real-time algorithm for bloodstream pulse transit time computation on ultra-low power embedded platforms

Teacher assistant for Bachelor courses at EPFL:

- Microprogrammed Embedded Systems — Content: Nintendo DS video game development
- Digital Systems Design — Content: C & VHDL programming on Xilinx Virtex 5 FPGA
- Laboratoires Sciences et Technologies de l'Electricité — Content: Experiments on wireless body sensor nodes

Publications

- **L. Duch**, S. Basu, M. Peon-Quiros, G. Ansaloni, L. Pozzi and D. Atienza. "i-DPs CGRA: An Interleaved-Datapaths Reconfigurable Accelerator for Embedded Bio-Signal Processing", *IEEE Embedded Systems Letters (ESL)*, 2018
- S. Basu, **L. Duch**, M. Peon-Quiros, G. Ansaloni, L. Pozzi and D. Atienza. "Heterogeneous and Inexact: Maximizing Power Efficiency of Edge Computing Sensors for Health Monitoring Applications", *ISCAS*, Firenze, Italy, 2018
- S. Basu, **L. Duch**, R. Braojos, G. Ansaloni, L. Pozzi and D. Atienza. "An Inexact Ultra-low Power Bio-Signal Processing Architecture With Lightweight Error Recovery", *CODES+ISSS*, Seoul, South-Korea, 2017
- **L. Duch**, S. Basu, R. Braojos, G. Ansaloni, L. Pozzi and D. Atienza. "HEAL-WEAR: an Ultra-Low Power Heterogeneous System for Bio-Signal Analysis", *IEEE Transactions on Circuits and Systems-I (TCAS-I)*, 2017
- **L. Duch**, S. Basu, R. Braojos, G. Ansaloni, L. Pozzi and D. Atienza. "A Multi-Core Reconfigurable Architecture for Ultra-Low Power Bio-Signal Analysis", *BioCAS*, Shanghai, China, 2016
- **L. Duch**, P. Garcia del Valle, S. Ganapathy, A. Burg and D. Atienza. "Energy vs. Reliability Trade-Offs Exploration in Biomedical Ultra-Low Power Devices", *DATE'16*, Dresden, Germany, 2016

Technical Skills

Computer skills:

Programming	C, C++, Perl, Bash, Assembly Language, MATLAB, Python, C#, LaTeX, VB6, JAVA, PHP, SQL, HTML, LabView, Grafcet
Software	Microsoft Visual Studio, Android Studio, IAR Embedded Workbench, Eclipse, EasyPHP, LabView, Unity Pro XL, Adobe Photoshop, Microsoft Office
O.S.	Windows, Linux, Mac OS

Electronics and microelectronics design skills:

Programming	VHDL, SystemC, Verilog-A, SPICE, TCL
Software	ISE (Xilinx), Quartus (Altera), ModelSim / Precision (Mentor Graphics), Design Compiler / HSPICE / TetraMAX (Synopsys), SoC Encounter / Virtuoso (Cadence), Mplab IDE (Microchip), Catapult C (Calypto), Altium Designer, Eagle PCB

Languages

French / English *Fluent* | Italian *Intermediate (B1)* | German *Notions (A1)*

Volunteer & Personal Activities

2015 → 2018 *Research participant* - **CHUV, PSA Group, MindMaze, EPFL labs & start-ups**, Lausanne, CH

- Personal involvement in clinical studies, product testing sessions, psychological, visual and cognitive experimentations.

2008 → 2018 *Online seller* - **Ebay, Amazon, Facebook, Anibis and Leboncoin sales sites**

- New and used items sales, price negotiation, customer support (pre and post sales)